Advances in Reasoning Principles

for

Contextual Equivalence and Termination

Nina Bohr

The IT University of Copenhagen

PROGRAMMING, LOGIC AND SEMANTICS research group October 2007

Abstract

In this PhD Dissertation we develop methods for proving contextual equivalence and termination.

Contextual equivalence. We present three papers on contextual equivalence. All three share a common setup: They are based on an FM-denotational model with a parameterized admissible Kripke-style logical relation on top of a universal recursive domain. The combination of expressible parameters and a recursive domain makes it non-trivial to establish the existence of the relations, and this has required some new ideas. Our method is the first proof method for contextual equivalence based on a logical relation over a denotational semantics for a language with recursive types and dynamic allocation of references of any type. The first paper gives the general setup. The second paper gives a relationally parametric interpretation of polymorphic types. This is not quite as general as we would like; we restrict the type of references so that the type must be closed. It is to our knowledge the first relationally parametric model for higher-order store and polymorphic and recursive types. The third paper is primarily concerned with refining the definition of parameters.

Termination analysis. In the last paper in the thesis we develop a sound and fully-automated algorithm to show that evaluation of a given untyped λ-expression will terminate under call-by-value. The "size-change principle" from first-order programs is extended to arbitrary untyped λ-expressions in two steps. The first step suffices to show call-by-value termination of a single, stand-alone λ-expression. The second suffices to show termination of any member of a regular set of λ-expressions, defined by a tree grammar.

This is a PhD Dissertation
presented to the Faculty of the IT University of Copenhagen
in partial fulfillment the requirements for the degree of
Doctor of Philosophy.

The author was supervised by Lars Birkedal and associated with the research group “Programming, Logic and Semantics” during the thesis work. The work on termination analysis was supervised by Neil D. Jones, University of Copenhagen.

Acknowledgements

I would like to thank my supervisor Lars Birkedal for introducing me to and involving me in this area of research and for inspiring supervision. I also thank Lars for always being open to find time to meet with me. I am grateful to Neil Jones for involving me in research on termination analysis and for generous supervision. Thanks to Nick Benton, whom I visited at Microsoft Research, for a fruitful stay. Finally I would like to thank everyone in the “Programming, Logic and Semantics” research group for a stimulating and friendly research environment.

Contents

1 Introduction

In this thesis we develop proof methods for contextual equivalence and termination. The thesis consists of two articles and two technical reports. Three of these papers are concerned with contextual equivalence and one with termination analysis. Equivalence is analyzed in a functional language extended with constructs to dynamically allocate and update higher-order store. Termination is analyzed in the untyped λ-calculus. Here in the introduction we give a short overview of each paper and briefly mention its immediate predecessors. In the individual papers there are further discussions about related and future work.

1.1 Contextual Equivalence

It is common to define two programs to be *contextually equivalent* if they have the same termination behavior in all closing contexts. We aim at giving a sound characterization that eases proofs of contextual equivalence in many cases. So we aim to be able to prove a large set of pairs of programs equivalent in a relatively easy way. Proving equivalence of programs is important for ensuring correctness of module exchanges or program transformations. When we want to prove contextual equivalence of two programs then the universal quantification over all contexts makes the reasoning difficult. There has therefore been much research activity targeted at developing accessible methods for equivalence proofs. Such methods are particularly challenging to develop for higher-order languages with recursive types, dynamic allocation and higher-order store. There are several reasons for this. Circular structures in the store are possible and so is also recursion through the store. Further, as programs may update as well as read the store, it is necessary to consider the behavior of execution in *related* stores. For higher-order store the behaviors of stored values depend themselves on the contents of the store, and relatedness of stored values is dependent in a non-trivial way of the contents of the stores. One of the consequences of this is that we need to require a weakening property expressing that related values of any type with respect to two stores s_1, s_2 will still be related also when activated in any future possibly updated stores. This again requires a precise way of expressing that two stores belong to the set of future updated stores with relation to the stores s_1, s_2. When the language allows dynamic allocation then relatedness must capture the possibility of keeping dynamically allocated locations hidden and preserving a local invariant, as well as the possibility of export of dynamically allocated locations.

Dynamic allocation in languages with store containing values of less general types has been analyzed, prior to our work, by several researchers including Pitts and Stark(39) and Benton and Leperchey(7). Here we extend this line of research to encompass higher-order store and recursive and polymorphic types. We will just briefly mention some key insights and ideas from these papers, which are important for our work. Pitts and Stark (39) present a method for reasoning about contextual equivalence for functional programs with recursion, assignment and integer references. It is possible that local references can be exported out of their original scope during evaluation. They devise a family of syntactic logical relations parameterized by state relations. To analyse program equivalence it is necessary to define relations not only between expressions, but also between continuations. First Pitts and Stark give a continuation based formulation of termination where the continuations corresponds to evaluation contexts (Felleisen). The formulation of termination via continuations makes it possible to define a termination relation by structural induction. With this formulation of termination many properties including the unwinding theorem can be proven by induction on the derivation of termination. It is also essential for the definition of the logical relations used for proving contextual equivalence and for the proof of the fundamental property. The parametric logical relation is given by simultaneous definition of three binary relations between values, continuations and expressions. The definitions make extensions of the used store explicit. The definition of a binary relation between values is by structural induction on the type. This definition uses, in an essential way, that references are of integer types such that lookup will not require another lookup and cannot involve circularity. The logical relation is then extended to a relation between open expressions. Pitts and Stark show that the fundamental property holds and at simple parameters the relation coinside with contextual equivalence and ciu-equivalence.

The most direct predecessor for our work is Nick Benton and Benjamin Leperchey: "Relational reasoning in a Nominal Semantics of Storage" (7). Benton and Leperchey give a method for proving contextual equivalence of programs with local state in a monadically typed language with references to integers and references to such. They develop the concept of accessibility maps to express disjointness properties of store areas, and differentiate in their parameters between the visible area of stores and the hidden invariants. This understanding of locality in stores, as also Reddy and Yang's (46), takes inspiration from the work in separation logic for local reasoning about storage. The proof method by Benton and Leperchey use a parameterized logical relation over a nominal denotational semantics, where the atom set is the set of store locations. The logical relation is used to prove equivalences that involve privacy of local store.

We extend their work in several ways. We have richer type systems including recursive and polymorphic types and higher-order store. We also give more refined definitions of parameters. The structure of our setup is very much inspired from their work, but the additional features in the language add some extra significant complications to the understanding of equivalence as well as to the denotational interpretation. As Benton and Leperchey, we use FM-domains where the atoms are location-names, but because we have higher-order store we give our denotational interpretation in a universal recursive domain defined by a mixed variant recursive equation. The proof that such an FM-domain exists builds on results by Mark Shinwell in his PhD thesis: The Fresh Approach: "Functional Programming with Names and Binders"(52). Shinwell transfers the results from Andrew Pitts: "Relational Properties of domains"(40) to FM domains. In the proof that our logical relations exist we also build on the work of Pitts and Shinwell. Here, however, we have also developed some new ideas, so that we can define the relation in a way that makes a version of the proof method applicable, and at the same time makes it possible to formulate very precise properties for relatedness in the parameters.

So, we analyse equivalence in a monadically typed ML-like functional language with dynamic allocation of higher-order store. The three papers on contextual equivalence share a common setup: They are based on an FM-denotational model with an admissible Kripke-style logical relation on top of a universal recursive domain. The methods give ways to express why two programs are expected to be equivalent via definitions of local parameters. Such definitions of local parameters express the intuition for why two programs are equivalent and are essentially the only non-trivial parts in a proof of equivalence. The combination of expressible parameters and a recursive domain makes it, however, non-trivial to establish the existence of the relations in the first place.

Relational Reasoning for Recursive Types and References

The first article on contextual equivalence *Relational Reasoning for Recursive Types and References* is background for the other two. It is a conference article presented at APLAS 2006, joint work with Lars Birkedal. We develop a proof method for contextual equivalence for a language with recursive types and dynamically allocated general references. As explained the proof method is based on a Kripke-style logical relation on top of a nominal denotational semantics. Our method is the first proof method for contextual equivalence based on a logical relation over a denotational semantics for a language with recursive types and dynamic allocation of references of any type. An alternative proof method based on bisimulations was developed by Koutavas and Wand (28) in parallel to ours.

In this paper we give the denotations in a four-tuple recursive domain $\mathbb{D} = (\mathbb{V}, \mathbb{K}, \mathbb{M}, \mathbb{S})$ where denotations of values belong in $\mathbb{V}$, denotations of computations in $\mathbb{M}$, denotations of continuations in $\mathbb{K}$ and denotations of stores in $\mathbb{S}$. Our parameterized logical relation then consists of a parameterized relation on each of the four domain parts. The parameters have the form Δr where Δ is a store-type, typing the finite visible area of two stores and r express invariants for local state. There is an order on the set of parameters written $\Delta' r' \rhd \Delta r$. A larger parameter corresponds to a later time in program executions where some locations may have been updated, more visible locations may have been allocated and more hidden invariants may have been built up.

We have formulated the definition of parameters so that they can be used to express properties of higher-order store. In particular it is possible to express for any type that when certain conditions hold for the stores then two locations will hold values of that type related 'at the current time' of program execution. It is not always appropriate to require that the locations hold values related at the time (expressed by a parameter) when the local parameter is initialized. Because when later updated with related values, the relatedness of these new values may demand additional properties of stores built up in the meantime and expressed as added local parameters or extra visible locations. So these new stored values may not be related under the parameter corresponding to the time where the invariant was initialized. To facilitate such an interpretation we give, in these situations, the parameters a syntactic formulation as pairs of locations together with types. There is a one way weakening property for related values, computations and continuations expressing that relatedness is preserved under larger parameters. One of the benefits of this is that when related values are stored and later fetched again from the stores, then they are related also at the time of retrieval.

The existence of the logical relation requires a separate proof in the style of Andrew Pitts and Mark Shinwell. Since we express invariants for local state in the parameters of our relation, and these will not necessarily be preserved under approximations, one cannot immediately apply Pitts' existence proof for the relation. Thus we define a four-ary parameterized relation. An element may be thought of as two pairs (d_1', d_2') and (d_1, d_2), where we require approximation in the domain theoretical sense $d_1' \sqsubseteq d_1$ and $d_2' \sqsubseteq d_2$. The pair (d_1, d_2) may be considered as having parameter status. For related states $(s_1', s_1, s_2', s_2, \Delta r)$, the invariants for local state expressed by the parameter r are required to hold for s_1, s_2 while the approximation in the existence proof is carried out on the primed places. Correspondingly we require for chains in the relation running over the primed places, with d_1, d_2 fixed, that least upper bounds will be in the relation.

For computations $(m_1', m_1, m_2', m_2, T\tau, \Delta r)$ to be related we likewise require $m_1' \sqsubseteq m_1$ and $m_2' \sqsubseteq m_2$ and additionally we require crosswise termination approximation under application from the approximated elements in one side to the non-approximated elements in the other side. That is; for any extended parameter $\Delta' r' \rhd \Delta r$, related continuations $(k_1', k_1, k_2', k_2, (x : \tau), \Delta' r')$ and related states $(s_1', s_1, s_2', s_2, \Delta' r')$ it must hold that $m_1' k_1' s_1' = \top \Rightarrow m_2 k_2 s_2 = \top$ and $m_2' k_2' s_2' = \top \Rightarrow m_1 k_1 s_1 = \top$. With these definition we can prove by the methods from Andrew Pitts and Mark Shinwell that the four-ary relation exists.

Based on the four-ary relation we can then extract a binary relation, in which we can relate denotations of open terms. This binary relation implies contextual equivalence at parameters which only give a set of visible locations expected to hold related values but no hidden invariants. An equivalence proof for two programs then requires us to show that their denotations are in the binary relation under such a simple parameter. This proof then proceeds in a modular way and it often requires that subexpressions are proven to be related under more complicated parameters for which we define local parameters.

Relational Parametricity for Recursive Types and References of Closed Types

The technical report *Relational Parametricity for Recursive Types and References of Closed Types* is joint with Lars Birkedal. Here we add impredicative polymorphism to the language but restrict the type $\tau\ ref$ of references such that τ must be closed. We then give a relational interpretation in a function space lattice. In the relational interpretation of a type $[\![\tau]\!]^t(\Xi\vec{R})(\Delta r)$ we apply to a type environment Ξ together with relations $\vec{R}$ for all the type variables in Ξ. The relations $\vec{R}$ are functions from parameters to admissible relations. A technical reason for restricting our setup to references of closed types is that we can then keep all type expressions in the parameters Δr closed. The store-types are closed and we also only need to have closed types associated with pairs of locations in the invariant r. So we do not need to apply our parameters Δr to relations $\vec{R}$ and relatedness for states does not depend on relations $\vec{R}$. This enables us to prove the identity extension lemma. We want the interpretation of $[\![\tau]\!]^t(\Xi\vec{R})$ to depend only on the relations for the free type variables that occur in the type τ also if this is a strict subset of Ξ. We express this as the two properties 'Ξ-strengthening' and 'Ξ-weakening' which identify $[\![\tau]\!]^t(fv(\tau)\ \vec{R})$ with any $[\![\tau]\!]^t(\Xi'\vec{R}')$ where $(\Xi'\vec{R}') \supseteq (fv(\tau)\ \vec{R})$. Further we want our logical relation to have the properties downwards closure, again on the primed places, and parameter weakening and admissibility. In this paper in the existence proof for our logical relation all these properties are added as requirements to the relational structure, we then prove that the properties are preserved under intersections. The relations with these properties over a domain together with set inclusion order and set intersection meets constitute a complete lattice. Further we prove that the properties are preserved under the action of the domain construction functor on relations. The existence proof for the relation is then done mainly along the lines of Pitts and Shinwell but now in the lattice of relations with these properties, so our fixed point relation has the properties.

This paper is to our knowledge the first relationally parametric model for higher-order store and polymorphic and recursive types, even if it is not as general as we would like due to the restriction on the type of references. We hope in the future to be able to model relational parametricity in full generality; this report is a step in that direction.

Relational Reasoning for Contextual Equivalence

The technical report *Relational Reasoning for Contextual Equivalence* is mostly concerned with the format for parameters. The language we analyse has recursive types, impredicative polymorphism and general dynamically allocated references. There is no restriction on the type of references but we do not give a relationally parametric interpretation of polymorphic types. For values to be related in a polymorphic type we require that the corresponding computation bodies are related as computations for any substitution of a closed type. As before we define a four-ary relation on top of a recursive domain, and then the existence proof for the relation is done mainly along the lines of Pitts and Shinwell.

Equivalence proofs for two programs are in our setup in all three papers done in a modular way, a proof will often involve sub-proofs where sub-terms are proven to behave relatedly. To analyse if two programs are related, we will expect related elements for the free variables and apply the programs' denotations to continuations and states assumed to be related. Sometimes this will be unfolded so that we get two sub-computations applied to continuations where the initial parts are generated by the original programs. For instance this may happen if the programs have a free variable of function type, and this function-variable is applied in the middle of the programs. In the technical report we refine the definition of parameters, so that they can take advantages of knowledge of initial parts of continuations. The development may seem rather technical, there is an extensive explanation in the technical report. We differentiate between parameters for computations and continuations, and the refinement also involves two different order relations among parameters. The relation is based on a more detailed understanding of the interaction between computations and continuations in the presence of higher-order store. We can express that functions may preserve more than one invariant. If the initial part of continuations change states from one such invariant to another, then (stored) related functions will still behave related. We have also added ways to express that functions may change a local invariant in an irreversible way. Further, we have added a way to express explicit divergence. A parameter may express, that one side has diverged and in the other side an invariant holds which should then eventually ensure divergence there. We find that the parameters can express in a natural way hypotheses of why we expect two programs to be equivalent. A proof of equivalence can then be a rather automatic test of hypotheses. Such a proof may though require many rather trivial steps, unfolding of denotations and of the definitions of the relation. Because the parameters are expressive we find that the method gives much help to proofs of program equivalences. In the future we intend to explore whether this method gives more help in equivalence proofs than the methods based on sets of bisimulations.

1.2 Termination

Call-by-value Termination in the Untyped λ-Calculus

The last article *Call-by-value Termination in the Untyped λ-Calculus* is joint work with Neil D. Jones. It is a journal article accepted for publication and it extends an earlier conference paper by the same authors presented by Neil Jones at RTA 2004.

The size change principle to prove termination for first order programs with a well founded order on parameter values was described by C.S. Lee, N.D. Jones and A.M. Ben-Amram (32). In the present article we develop a termination analysis based on The Size Change Principle for a single lambda-expression which we often think of as a program together with its input. We then further develop the method so that we can analyse whether a program will terminate when applied to any input from a well formed input set given by a tree grammar. The analysis can be fully automated, we have developed a simple implementation (not included in the thesis). The method is safe. As the problem in general is undecidable and the method is safe and fully automatic, so the proof method is not complete and there are terminating programs which we cannot certify to terminate.

It is not immediate to see how the ingredients from the first version of The

Size Change Principle can be found in the λ-calculus. We use an environment based version of operational semantics where we for applications instead of substitution make updates in the environment. A state has the form of a finite tree, it is an expression together with an environment binding variables to states. The initial state consists of the program together with the empty environment. Updates in the environment are performed in the operational semantics during program execution. We can then show that any occurring expression in a state will be a subexpression of the original program. This opens up for a finite approximation of the state space. We may approximate a state by forgetting the environment completely or, for instance, by only considering the parts of environments up to some fixed level in the tree. In the paper we have chosen the coarsest version and remove environments in the approximation. Further the operational semantics is extended with an explicit calls-relation so that we can find the control flow graph for the program and the call sequences that follow it. We can then trace nontermination as an infinitely long sequential state transition. An environment will have the form of a finite tree, and the well founded order in which we show decreases is based on the height of environments. Size change graphs for calls relate bindings in environments. By abstract interpretation we find an approximation to the execution of the program with its input with a finite state space, where we can still generate safe size change graphs. Based on that we can then perform a safe analysis.

When we extend the analysis to cover a program with any arbitrary input from a given input set, e.g. Church numerals, we do this by extending the λ-calculus language with nonterminals. A nonterminal represents a set of lambda expressions, and we define what free variables of a nonterminal and what a subexpression of a nonterminal should be. We prove that execution of the program with nonterminals can simulate the execution of the program with any arbitrary input from the input set, and that the generated size change graphs are appropriate for a safe termination analysis. The termination analysis in itself can then be performed very similar to before.

Relational Reasoning for Recursive Types and References

Nina Bohr and Lars Birkedal

IT University of Copenhagen (ITU)
{ninab,birkedal}@itu.dk

Abstract. We present a local relational reasoning method for reasoning about contextual equivalence of expressions in a λ-calculus with recursive types and general references. Our development builds on the work of Benton and Leperchey, who devised a nominal semantics and a local relational reasoning method for a language with simple types and simple references. Their method uses a parameterized logical relation. Here we extend their approach to recursive types and general references. For the extension, we build upon Pitts' and Shinwell's work on relational reasoning about recursive types (but no references) in nominal semantics. The extension is non-trivial because of general references (higher-order store) and makes use of some new ideas for proving the existence of the parameterized logical relation and for the choice of parameters.

1 Introduction

Proving equivalence of programs is important for verifying the correctness of compiler optimizations and other program transformations. Program equivalence is typically defined in terms of *contextual equivalence*, which expresses that two program expressions are equivalent if they have the same observable behaviour when placed in any program context C. It is generally quite hard to show directly that two program expressions are contextually equivalent because of the universal quantification over all contexts. Thus there has been an extensive research effort to find reasoning methods that are easier to use for establishing contextual equivalence, in particular to reduce the set of contexts one has to consider, see, e.g., [40, 9, 3, 33] and the references therein. For programming languages with references, it is not enough to restrict attention to fewer contexts, since one also needs to be able to reason about equivalence under *related* stores. To address this challenge, methods based on logical relations and bisimulations have been proposed, see, e.g., [42, 7, 55]. The approaches based on logical relations have so far been restricted to deal only with simple integer references (or references to such). To extend the method to general references in typed languages, one also needs to extend the method to work in the presence of recursive types. The latter is a challenge on its own, since one cannot easily establish the existence of logical relations by induction in the presence of recursive types. Thus a number of research papers have focused on relational reasoning methods for recursive types without references, e.g., [9, 3]. Recently, the bisimulation approach has been simplified and extended to work for untyped languages with general references [28, 27]. For effectiveness of the reasoning method, we seek *local* reasoning methods, which only require that we consider the accessible part of a store and which works in the presence of a separated (non-interfering) invariant that is preserved by the context. In [7], Benton and Leperchey developed a relational reasoning method for a language with simple references that does allow for local reasoning. Their approach is inspired by related work on separation logic [47, 46]. In particular, an important feature of the state relations of Benton and Leperchey is that they depend on only part of the store: that allows us to reason that related states are still related if we update them in parts on which the relation does not depend. In this paper we extend the work of Benton and Leperchey to relational reasoning about contextual equivalence of expressions in a typed programming language with general recursive types *and* general references (thus with higher-order store). We arrive at a useful reasoning method. In particular, we have used it to verify all the examples of [28]. We believe that the method is simple to use, but more work remains to compare the strengths and weaknesses of the method we present here with that of *loc.cit.*

Before giving an overview of the technical development, we now present two examples of pairs of programs that can easily be shown contextually equivalent with the method we develop. The examples are essentially equivalent to (or perhaps slightly more involved than) examples in [28]. Section 5 contains the proofs of contextual equivalence.

The programs M and N shown below both take a function as argument and returns two functions, set and get. In M, there is one hidden reference y, which set can use to store a function. The get function returns the contents of y. The program N uses three local references y_0, y_1 and p. The p reference holds a integer value. The set function updates p and depending on the value of p it stores its argument in either y_0 or y_1. The get function returns the contents of y_0 or y_1, depending on the value of p. Note that the programs store functions in the store. Intuitively, the programs M and N are contextually equivalent because they use *local storage*. The proof method we develop allows us to prove that they are contextually equivalent via local reasoning.

```
M = rec $f$ ($g$: $\tau \to T\tau'$): $T(((\tau \to T\tau') \to T\text{unit}) \times (\text{unit} \to T(\tau \to T\tau')))$ =
        let $y \Leftarrow$ ref $g$ in
        let set $\Leftarrow$ val (rec $f_{1M}(g_1 : \tau \to T\tau') : T\text{unit} = y := g_1$) in
        let get $\Leftarrow$ val (rec $f_{2M}(x : \text{unit}) : T(\tau \to T\tau') = !y$) in
           (set,get)
```

```
N = rec $f$ ($g$: $\tau \to T\tau'$): $T(((\tau \to T\tau') \to T\text{unit}) \times (\text{unit} \to T(\tau \to T\tau')))$ =
        let $y_0 \Leftarrow$ ref $g$ in
        let $y_1 \Leftarrow$ ref $g$ in
        let $p \Leftarrow$ ref 0 in
        let set $\Leftarrow$ val (rec $f_{1N}(g_1 : \tau \to T\tau') : T\text{unit}$ =
                             if iszero($!p$) then
                                ($p := 1$; $y_1 := g_1$)
```

```
                    else
                      (p := 0; y_0 := g_1)) in
let get ⇐ val (rec f_2N(x : unit) : (τ → Tτ') =
                      if iszero(!p) then !y_0 else !y_1) in
    (set,get)
```

Next consider the programs M' and N' below. They both have a free variable g of function type. In M', g is applied to a function that just returns unit and then M' returns the constant unit function. In N', g is applied to a function that updates a reference local to N', maintaining the invariant that the value of the local reference is always greater than zero. After the call to g, N' returns the constant unit function if the value of the local reference is greater than zero; otherwise it diverges (Ω stands for a diverging term). Intuitively, it is clear that M' and N' are contextually equivalent, since the local reference in N' initially is greater than zero and g can only update the local reference via the function it is given as argument and, indeed, we can use our method to prove formally that M' and N' are contextually equivalent via local reasoning.

```
M' = let f ⇐ val (rec f'(a : unit) : Tunit = val ()) in
         let w ⇐ gf in
            val f

N' = let x ⇐ ref 1 in
         let f ⇐ val (rec f'(a : unit) : Tunit) = x := !x + 1) in
         let w ⇐ gf in
            let z ⇐ if iszero(!x) then Ω else
                        val(rec f'(a : unit) : Tunit = val ()) in
               val z
```

We now give an overview of the technical development, which makes use of a couple of new ideas for proving the existence of the parameterized logical relation and for the choice of parameters.

In Section 2 we first present the language and in Section 3 we give a denotational semantics in the category of FM-cpo's. Adapting methods developed by Pitts [40] and Shinwell [52, 51] we prove the existence of a recursive domain in $(\text{FM-Cpo}_\perp)^4$, $\mathbb{D} = (\mathbb{V}, \mathbb{K}, \mathbb{M}, \mathbb{S})$, such that $i : F(\mathbb{D}, \mathbb{D}) \cong \mathbb{D}$ where F is our domain constructor. The 4-tuple of domains $\mathbb{D}$ has the minimal invariant property, that is, $id_{\mathbb{D}}$ is the least fixed point of $\delta : (\mathbb{D} \to \mathbb{D}) \to (\mathbb{D} \to \mathbb{D})$ where $\delta(e) = i \circ F(e, e) \circ i^{-1}$. Denotations of values are given in $\mathbb{V}$, continuations in $\mathbb{K}$, computations in $\mathbb{M}$ and stores in $\mathbb{S}$. We show adequacy via a logical relation, the existence of which is established much as in [52].

The denotational semantics can be used to establish simple forms of contextual equivalence qua adequacy. For stronger proofs of contextual equivalences we define a parameterized relation between pairs of denotations of values, pairs of denotations of continuations, pairs of denotations of computations, pairs of denotations of stores. We can express contextual equivalence for two computations by requiring that they have the same terminaton behaviour when placed in the same arbitrary closing contexts.

Since our denotations belong to a recursive domain, the existence of the parameterized logical relation again involves a separate proof. The proof requires that the relations are preserved under approximations. On the other hand we want the parameters to express invariants for hidden local areas of related stores, and such properties of stores will not be preserved under approximations. Therefore our relations are really given by 4-tuples, which we think of as two pairs: the 4-tuples have the form (d'_1, d_1, d'_2, d_2), where $d'_1 \sqsubseteq d_1$ and $d'_2 \sqsubseteq d_2$. We can now let the approximation be carried out over the primed domain elements d'_1, d'_2, and preserve the invariant on the non-primed elements d_1, d_2. Correspondingly, relatedness of computations is stated as a two-sided termination approximation. Termination of application of an approximated computation m'_1 to an approximated continuation k'_1 and an approximated store S'_1 implies termination in the other side of the non-approximated elements, $m'_1 k'_1 S'_1 = \top \implies m_2 k_2 S_2 = \top$, and similarly for the other direction. With this separation of approximation from the local properties that the parameters express, we can prove that the relation exists. We can then extract a binary relation, defined via reference to the 4-ary relation, such that the binary relation implies contextual equivalence.

A parameter expresses properties of two related stores; and computations are related under a parameter if they have equivalent termination behaviour when executed in stores, which preserve at least the invariants expressed by the parameter. Our parameters are designed to express relatedness of pairs in the presence of higher-order store and therefore they are somewhat more complex than the parameters used by Benton and Leperchey [7]. As we have seen in the examples above, we can prove contextual equivalence of two functions, which allocate local store in different ways, and then return functions set and get that access the hidden local storage. These local locations can be updated later by application of the exported set-functions to related arguments. In between the return of the functions and the application of the returned set-functions, there might have been built up additional local store invariants. Thus functions stored by a later call to the returned set-function may require further properties of stores in order to have equivalent behaviour, than was the case when our set and get functions were returned. To handle this possibility our parameters include pairs of locations; two stores are then related wrt. such pairs of locations if the pair of locations contain values that are related relative to the invariants that hold for the two stores.

In more detail, a parameter has the form $\Delta\{r_1, \ldots, r_n\}$. Here Δ is a store type that types a finite set of locations; these are intuitively our "visible locations." The $r_1, \ldots, r_n$ are local parameters. A local parameter r_i has its own finite area of store in each side, disjoint from the visible area and from all the other local parameters' store areas. A local parameter r_i has the form $(P_1, LL_1) \vee \cdots \vee (P_m, LL_m)$. The Ps express properties of two stores and the LLs are lists of location pairs. It is possible to decide if two states fulfill the properties expressed by the Ps by only considering the contents of r_is private areas of store. At least one P must hold and the corresponding LL must hold values related relative to the invariants that hold for the two stores (we can also think of this as related at the given time in computation). Using FM domain theory makes it posible for us to express the parameters directly by location names.

We present the definition of our relation, state its existence and the theorem that relatedness implies contextual equivalence in Section 4. In the following Section 5 we show how we prove contextual equivalence of our example programs. We hope that the proofs will convince the reader that our logical relations proof method is fairly straightforward to apply; in particular the choice of parameters is very natural. We conclude in Section 6.

For reasons of space most proofs have been omitted from this extended abstract.

2 Language

The language we consider is a call-by-value, monadically-typed λ-calculus with recursion, general recursive types, and general dynamically allocated references. Types are either *value types* τ or *computation types* $T\tau$. Values of any closed value type can be stored in the store.

$$\begin{aligned}\tau &::= \alpha \mid \text{unit} \mid \text{int} \mid \tau \times \tau \mid \tau + \tau \mid \tau\text{ref} \mid \tau \to T\tau \mid \mu\alpha.\tau \\ \gamma &::= \tau \mid T\tau\end{aligned}$$

Typing contexts, Γ, are finite maps from variables to closed value types. We assume infinite sets of variables, ranged over by x, type variables, ranged over by α, and locations, ranged over by l. We let $\mathbb{L}$ denote the set of locations. Store types Δ are finite maps from locations to value types. Terms G are either *values* V or *computations* M:

$$\begin{aligned}V &::= x \mid \underline{n} \mid \underline{l} \mid () \mid (V, V') \mid \text{in}_i V \mid \text{rec } f(x:\tau) = M \mid \text{fold } V \\ M &::= VV' \mid \text{let } x \Leftarrow M \text{ in } M' \mid \text{val } V \mid \pi_i V \mid \text{ref } V \mid !V \mid \\ &\quad\;\; V := V' \mid \text{case } V \text{ of in}_1 x_1 \Rightarrow M_1; \text{in}_2 x_2 \Rightarrow M_2 \mid \\ &\quad\;\; V = V' \mid V + V' \mid \text{iszero } V \mid \text{unfold } V \\ G &::= M \mid V.\end{aligned}$$

Continuations K take the following form:

$$K ::= \text{val } x \mid \text{let } y \Leftarrow M \text{ in } K$$

The typing judgments take the form

$$\Delta; \Gamma \vdash V : \tau \qquad \Delta; \Gamma \vdash M : T\tau \qquad \Delta; \vdash K : (x:\tau)^{\top}$$

The typing rules for values and terms are as in [7] extended with rules for recursive types, except that the type for references is not restricted. Here we just include the following three selected rules:

$$\frac{\Delta; \Gamma \vdash V : \tau}{\Delta; \Gamma \vdash \text{ref } V : T(\tau\text{ref})}$$

$$\frac{\Delta; \Gamma \vdash V : \tau[\mu\alpha.\tau/\alpha]}{\Delta; \Gamma \vdash \mathit{fold}\ V : \mu\alpha.\tau} \qquad \frac{\Delta; \Gamma \vdash V : \mu\alpha.\tau}{\Delta; \Gamma \vdash \mathit{unfold}\ V : T(\tau[\mu\alpha.\tau/\alpha])}$$

Stores Σ are finite maps from locations to closed values. A store Σ has store type Δ, written $\Sigma : \Delta$, if, for all l in the domain of Δ, $\Delta; \vdash \Sigma(l) : \Delta(l)$.

The operational semantics is defined via a termination judgment $\Sigma, \text{let } x \Leftarrow M \text{ in } K \downarrow$, where M is closed and K is a *continuation term in x*. Typed continuation terms are defined by:

$$\frac{}{\Delta; \vdash \mathit{val}\ x : (x : \tau)^{\top}} \qquad \frac{\Delta; x : \tau \vdash M : T\tau' \quad \Delta; \vdash K : (y : \tau')^{\top}}{\Delta; \vdash \mathit{let}\ y \Leftarrow M\ \mathit{in}\ K : (x : \tau)^{\top}}$$

The defining rules for the termination judgment $\Sigma, \text{let } x \Leftarrow M \text{ in } K \downarrow$ are standard given that the language is call-by-value, with left-to-right evaluation order. We just include one rule as an example:

$$\frac{\Sigma, \text{let } x \Leftarrow \text{val } V \text{ in } K \downarrow}{\Sigma, \text{let } x \Leftarrow \text{unfold}(\text{fold } V) \text{ in } K \downarrow}$$

A *context* is a computation term with a hole, and we write $C[.] : (\Delta; \Gamma \vdash \gamma) \Rightarrow (\Delta; - \vdash T\tau)$ to mean that whenever $\Delta; \Gamma \vdash G : \gamma$ then $\Delta; - \vdash C[G] : T\tau$.

The definition of contextual equivalence is standard and as in [7].

Definition 1. *If $\Delta; \Gamma \vdash G_i : \gamma$, for $i = 1, 2$ then G_1 and G_2 are* contextually equivalent*, written*

$$\Delta; \Gamma \vdash G_1 =_{ctx} G_2,$$

if, for all types τ, for all contexts $C[.] : (\Delta; \Gamma \vdash \gamma) \Rightarrow (\Delta; - \vdash T\tau)$ and for all stores $\Sigma : \Delta$,

$$\Sigma, \text{let } x \Leftarrow C[G_1] \text{ in val } x \downarrow \Longleftrightarrow \Sigma, \text{let } x \Leftarrow C[G_2] \text{ in val } x \downarrow .$$

3 Denotational Semantics

We define a denotational semantics of the language from the previous section and show that the semantics is *adequate*. The denotational semantics is defined using FM-domains [52]. The semantics and the adequacy proof, in particular the existence proof of the logical relation used to prove adequacy, builds on Shinwell's work on semantics of recursive types in FM-domains [52]. Our approach is slightly different from that of Shinwell since we make use of universal domains to model the fact that any type of value can be stored in the store, but technically it is a minor difference.

We begin by calling to mind some basic facts about FM-domains; see [52] for more details. Fix a countable set of atoms, which in our case will be the locations, $\mathbb{L}$. A *permutation* is a bijective function $\pi \in (\mathbb{L} \to \mathbb{L})$ such that the set $\{l \mid \pi(l) \neq l\}$ is finite. An FM-set X is a set equipped with a permutation action: an operation $\pi \bullet - : \mathit{perms}(\mathbb{L}) \times X \to X$ that preserves composition and identity, and such that each element $x \in X$ is finitely supported: there is a finite set $L \subset \mathbb{L}$ such that whenever π fixes each element of L, the action of π fixes x: $\pi \bullet x = x$. There is a smallest such set, which we write $\mathit{supp}(x)$. A morphism of FM-sets is a function $f : D \to D'$ between the underlying sets that is equivariant: $\forall x.\pi \bullet (fx) = f(\pi \bullet x)$. An FM-cpo is an FM-set with an equivariant partial order relation $\sqsubseteq$ and least upper bounds of all finitely-supported ω-chains. A morphism of FM-cpos is a morphism of their underlying FM-sets that is monotone and preserves lubs of finitely-supported chains. We only require the existence and preservation of lubs of finitely-supported chains, so an FM-cpo may not be a cpo in the usual sense. The sets $\mathbb{Z}$, $\mathbb{N}$, etc., are discrete FM-cpos with the trivial action. The set of locations, $\mathbb{L}$, is a discrete FM-cpo with the action $\pi \bullet l = \pi(l)$. The category of FM-cpos is bicartesian closed: we write 1 and $\times$ for the finite products, $D \Rightarrow D'$ for the internal hom and 0,+ for the coproducts. The action on products is pointwise, and on functions is given by conjugation: $\pi \bullet f = \lambda x.\pi \bullet (f(\pi^-1 \bullet x))$. The category is not well-pointed: morphisms $1 \to D$ correspond to elements of $1 \Rightarrow D$ with empty support. The lift monad, $(-)_L$, is defined as usual with the obvious action. The Kleisli category FM-Cpo$_\perp$ is the category of pointed FM-cpos (FM-cppos) and strict continuous maps, which is symmetric monoidal closed, with smash product $\otimes$ and strict function space $\multimap$. If D is a pointed FM-cpo then $\mathit{fix} : (D \Rightarrow D) \multimap D$ is defined by the lub of an ascending chain in the usual way. We write $\mathbb{O}$ for the discrete FM-cpo with elements $\perp$ and $\top$, ordered by $\perp \sqsubseteq \top$.

As detailed in [52], one may solve recursive domain equations in FM-Cpo$_\perp$. For the denotational semantics, we use minimal invariant recursive domains:

$$\begin{aligned}
\mathbb{V} &\cong 1_\perp \oplus \mathbb{Z}_\perp \oplus \mathbb{L}_\perp \oplus (\mathbb{V} \oplus \mathbb{V}) \oplus (\mathbb{V} \otimes \mathbb{V}) \oplus (\mathbb{V} \multimap \mathbb{M})_\perp \oplus \mathbb{V}\\
\mathbb{K} &\cong (\mathbb{S} \multimap (\mathbb{V} \multimap \mathbb{O}))\\
\mathbb{M} &\cong (\mathbb{K} \multimap (\mathbb{S} \multimap \mathbb{O}))\\
\mathbb{S} &\cong \mathbb{L}_\perp \multimap \mathbb{V}.
\end{aligned}$$

Formally, these are obtained as the minimal invariant solution to a locally FM-continuous functor $F : (\text{FM-Cpo}_\perp^4)^{\text{op}} \times \text{FM-Cpo}_\perp^4 \to \text{FM-Cpo}_\perp^4$. We write $\mathbb{D}$ for $(\mathbb{V}, \mathbb{K}, \mathbb{M}, \mathbb{S})$ and i for the isomorphism $i : F(\mathbb{D}, \mathbb{D}) \cong \mathbb{D}$. We will often omit the isomorphism i and the injections into the sum writing, e.g., simply (v_1, v_2) for an element of $\mathbb{V}$.

Types, τ are interpreted by $[\![\tau]\!] = \mathbb{V}$, computation types $T\tau$ are interpreted by $[\![T\tau]\!] = \mathbb{M}$, continuation types $(x : \tau)^\top$ are interpreted by $[\![(x : \tau)^\top]\!] = \mathbb{K}$, and store types Δ are interpreted by $[\![\Delta]\!] = \mathbb{S}$. Type environments $\Gamma = x_1 : \tau_1, \ldots, x_n : \tau_n$ are interpreted by $\mathbb{V}^n$.

Typing judgments are interpreted as follows:

- $[\![\Delta; \Gamma \vdash V : \tau]\!] \in ([\![\Gamma]\!] \multimap [\![\tau]\!])$
- $[\![\Delta; \Gamma \vdash M : T\tau]\!] \in ([\![\Gamma]\!] \multimap [\![T\tau]\!])$
- $[\![\Delta; \vdash K : (x : \tau)^\top]\!] \in \mathbb{K}$

The actual definition of the interpretations is quite standard, except for allocation which makes use of the properties of FM-cpo's:

$$\begin{aligned}
&[\![\Delta; \Gamma \vdash \text{ref}V : T(\tau\text{ref})]\!]\, \rho = \lambda k.\lambda S.\\
&\quad k(S([l \mapsto [\![\Delta; \Gamma \vdash V : \tau]\!]\, \rho])l\\
&\text{for some/any } l \notin \mathit{supp}(\lambda l'.k(S[l' \mapsto [\![\Delta; \Gamma \vdash V : \tau]\!]\, \rho])l')
\end{aligned}$$

The definition is much as in [7]. The use of FM-cpo's ensure that it is a good definition. As in [7], we use the monad T to combine state with continuations to get a good control over what the new location has to be fresh for.

We only include two additional cases of the semantic definition, namely the one for unfold and the one for continuations:

$$\begin{array}{l}[\![\Delta; \Gamma \vdash \text{unfold } V : T(\tau[\mu\alpha.\tau/\alpha])]\!]\,\rho = \lambda k.\lambda S. \\ \textit{case } [\![\Delta; \Gamma \vdash V : \mu\alpha.\tau]\!]\,\rho \textit{ of } i_1 \circ in_\mu(d) \textit{ then } kSd; \textit{ else}\bot,\end{array}$$

where in_μ is the appropriate injection of $\mathbb{V}$ into $1_\bot \oplus \mathbb{Z}_\bot \oplus \mathbb{L}_\bot \oplus (\mathbb{V} \oplus \mathbb{V}) \oplus (\mathbb{V} \otimes \mathbb{V}) \oplus (\mathbb{V} \multimap \mathbb{M})_\bot \oplus \mathbb{V}$ and i_1 is the isomorphism from this sum into $\mathbb{V}$.

$$\begin{array}{l}[\![\Delta; \vdash K : (x : \tau)^\top]\!] = \lambda S.\lambda d. \\ \quad [\![\Delta; x : \tau \vdash K : T\tau']\!]\{x \mapsto d\}(\lambda S'.(\lambda d'.\top)_\bot)_\bot S\end{array}$$

Theorem 1 (Soundness and Adequacy). *If $\Delta; \vdash M : T\tau$, $\Delta; \vdash K : (x : \tau)^\top$, $\Sigma : \Delta$ and $S \in [\![\Sigma : \Delta]\!]$ then*

$$\Sigma, \text{let } x \Leftarrow M \text{ in } K \downarrow \quad \textit{iff} \quad [\![\Delta; \vdash M : T\tau]\!] * [\![\Delta; \vdash K : (x : \tau)^\top]\!]\, S = \top.$$

Soundness is proved by induction and to show adequacy one defines a formal approximation relation between the denotational and the operational semantics. The existence proof of the relation is non-trivial because of the recursive types, but follows from a fairly straightforward adaptation of Shinwell's existence proof in [52] (Shinwell shows adequacy for a language with recursive types, but without references).

Corollary 1. *$[\![\Delta; \Gamma \vdash G_1 : \gamma]\!] = [\![\Delta; \Gamma \vdash G_2 : \gamma]\!]$ implies $\Delta; \Gamma \vdash G_1 =_{ctx} G_2$.*

4 A Parameterized Logical Relation

In this section we define a parameterized logical relation on $\mathbb{D}$ and $F(\mathbb{D}, \mathbb{D})$, which we can use to prove contextual equivalence. (In the following we will sometimes omit the isomorphism i, i^{-1} between $F(\mathbb{D}, \mathbb{D})$ and $\mathbb{D}$).

4.1 Accessibility maps, simple state relations and parameters

Intuitively, the parameters express properties of two related states by expressing requirements of disjoint areas of states. There is a "visible" area and a finite number of "hidden invariants." In the logical relation, computations are related under a parameter if they have corresponding termination behaviour under the assumption that they are executed in states satisfying the properties expressed by the parameter.

Definition 2. *A function $A : \mathbb{S} \to \mathcal{P}_{\text{fin}}(\mathbb{L})$ from $\mathbb{S}$ to the set of finite subsets of $\mathbb{L}$ is an* accessibility map *if*

$$\forall S_1, S_2.\ (\forall l \in A(S_1).\ S_1 l = S_2 l) \Rightarrow A(S_1) = A(S_2)$$

We let $A_\emptyset$ denote the accessibility map defined by $\forall S.A_\emptyset(S) = \emptyset$, and we let $A_{\{l_1,\ldots,l_k\}}$ denote the accessibility map defined by $\forall S.A_{\{l_1,\ldots,l_k\}}(S) = \{l_1, \ldots, l_k\}$.

Definition 3. *A* simple state relation P *is a triple* $(\hat{p}, A_{p1}, A_{p2})$ *satisfying that* A_{p1} *and* A_{p2} *are accessibility maps and* $\hat{p}$ *is a relation on* $\mathbb{S}$ *satisfying, for all states* $S_1, S_2, S_1', S_2' \in \mathbb{S}$*,*

$$\begin{aligned}&\big(\forall l_1 \in A_{p1}(S_1).S_1 l_1 = S_1' l_1 \ \wedge\ \forall l_2 \in A_{p2}(S_2).S_2 l_2 = S_2' l_2\big)\\ &\qquad \Rightarrow \big((S_1, S_2) \in \hat{p} \Leftrightarrow (S_1', S_2') \in \hat{p}\big).\end{aligned}$$

Note that a simple state relation is essentially a relation on states for which it can be decided whether a pair of states belong to the relation only on the basis of some parts of the states, defined by a pair of accessibility maps.

We denote the "always true" simple state relation $(\mathbb{S} \times \mathbb{S}, A_\emptyset, A_\emptyset)$ by T.

We now define the notion of a local parameter, which we will later use to express hidden invariants of two related states. Intuitively, a local parameter has its own private areas of the states. These areas are used for testing conditions and for storing related values. The testing condition is a disjunction of simple state relations, where to each disjunct there is an associated list of pairs of locations from the two related states. At least one condition must be satisfied and the corresponding list of locations hold related values.

Definition 4. *A* local parameter r *is a finite non-empty set of pairs* $\{(P_1, LL_1), .., (P_m, LL_m)\}$*, where each* P_i *is a simple state relation* $P_i = (\hat{p}_i, A_{pi1}, A_{pi2})$ *and each* LL_i *is a finite set of location pairs and closed value types* $LL_i = \{\ (l_{i11}, l_{i12}, \tau_{i1}), \ldots, (l_{in_i1}, l_{in_i2}, \tau_{n_i})\ \}$*.* $(n_i \geq 0)$*.*

We often write a local parameter as $r = ((P_1, LL_1) \vee \ldots \vee (P_m, LL_m))$. For a location list LL, we write L_1 resp. L_2 for the set of locations that occur as first resp. second components in the location list LL. For a local parameter r, there are associated accessibility maps A_{r1} and A_{r2} given by $\forall S.\ A_{r1}(S) = \bigcup_i A_{pi1}(S) \cup L_1$ and $\forall S.\ A_{r2}(S) = \bigcup_i A_{pi2}(S) \cup L_2$.

We denote the "always true" local parameter $\{(T, \emptyset)\}$ also simply by T. It has the associated accessibility maps $A_\emptyset, A_\emptyset$.

As explained in the introduction we have included the LL-list to be used for storing related values which may later be updated by exported updating functions. The updated values may require more invariants to hold for the stores in order to have equivalent behaviour. This interpretation of the local parameter is expressed in the definition of our invariant relation $F(\nabla, \nabla)$ below.

Definition 5. *A* parameter $\varDelta r$ *is a pair* $(\varDelta, r)$*, with* $\varDelta$ *a store type, and* $r = \{r_1, .., r_n\}$ *a finite set of local parameters such that* $T \in r$*.*

For a parameter $\varDelta r$ we associate accessibility maps A_{r1} and A_{r2}, given by $\forall S.\ A_{r1}(S) = \bigcup A_{r_i1}(S)$ and $\forall S.\ A_{r2}(S) = \bigcup A_{r_i2}(S)$.

For each store type $\varDelta$ we have a special the "always true" parameter $\varDelta id_\emptyset = \varDelta\{T\}$.

Definition 6. *For parameters $\Delta' r'$ and Δr define*
$\Delta' r' \rhd \Delta r \overset{def}{\iff} \Delta' \supseteq \Delta$ *and* $r' \supseteq r$.

The ordering relation $\rhd$ is reflexive, transitive and antisymmetric. For all parameters Δr it holds that there are only finitely many parameters $\Delta_0 r_0$ such that $\Delta r \rhd \Delta_0 r_0$. For convenience we sometimes write $\Delta r \lhd \Delta' r'$ for $\Delta' r' \rhd \Delta r$.

4.2 Parameterized relations and contextual equivalence

In this section we will define a parameterized logical relation on $\mathbb{D}$ and $F(\mathbb{D}, \mathbb{D})$. Let $D = (D_V, D_K, D_M, D_S) \in \{\mathbb{D}, F(\mathbb{D}, \mathbb{D})\}$. We define the set of relations $\mathcal{R}(D)$ on D as follows.
$\mathcal{R}(D) = \hat{R}_V \times \hat{R}_K \times \hat{R}_M \times \hat{R}_S$ where

$\hat{R}_V$ = all subsets of
$D_V^4 \times \{\tau \mid \tau$ is a closed value type$\} \times \{$parameter$\}$ that include
$\{(\bot, v_1, \bot, v_2, \tau, \Delta r) \mid v_1, v_2 \in D_V,\ \tau$ closed value type, Δr parameter$\}$
$\hat{R}_K$ = all subsets of
$D_K^4 \times \{(x : \tau)^\top \mid (x : \tau)^\top$ is a closed continuation type$\} \times \{$parameter$\}$ that include $\{(\bot, k_1, \bot, k_2, (x : \tau)^\top, \Delta r) \mid$
$k_1, k_2 \in D_K, (x : \tau)^\top$ closed continuation type, Δr parameter$\}$
$\hat{R}_M$ = all subsets of
$D_M^4 \times \{T\tau \mid T\tau$ is a closed computation type$\} \times \{$parameter$\}$ that include
$\{(\bot, m_1, \bot, m_2, T\tau, \Delta r) \mid$
$m_1, m_2 \in D_M, T\tau$ closed computation type, Δr parameter$\}$
$\hat{R}_S$ = all subsets of $D_S^4 \times \{$parameter$\}$ that include
$\{(\bot, S_1, \bot, S_2, \Delta r) \mid S_1, S_2 \in D_S, \Delta r$ parameter$\}$

A relation $(R_1, R_2, R_3, R_4) \in \mathcal{R}(D)$ is *admissible* if,
for each i, R_i is closed under least upper bounds of finitely supported chains of the form $(d_1^i, d_1, d_2^i, d_2, (type), \Delta r)_{i \in \omega}$ where $d_1, d_2, type, \Delta r$ are constant. We let $\mathcal{R}_{adm}(D)$ denote the admissible relations over D.

Theorem 2. *There exists a relational lifting of the functor F to $(\mathcal{R}(\mathbb{D})^{\mathrm{op}} \times \mathcal{R}(\mathbb{D})) \to \mathcal{R}(F(\mathbb{D}, \mathbb{D}))$ and an* admissible *relation $\nabla = (\nabla_V, \nabla_K, \nabla_M, \nabla_S) \in \mathcal{R}_{adm}(\mathbb{D})$ satisfying the equations in Figure 1 and $(i, i) : F(\nabla, \nabla) \subset \nabla \ \wedge\ (i^{-1}, i^{-1}) : \nabla \subset F(\nabla, \nabla)$.*

Proof (Theorem 2, existence of an invariant relation ∇). The proof makes use of the ideas mentioned in the Introduction in combination with a proof method inspired from Pitts [40]. We have defined a relational structure on the domains $\mathbb{D}$ and $F(\mathbb{D}, \mathbb{D}) \in$ FM-Cpo$_\bot^4$ as products of relations on each of their four domain-projections. Each of these relations is a 4-ary relation with elements $(d_1', d_1, d_2', d_2, (type), \Delta r)$ where $d_1' = d_2' = \bot$ relates to everything.

We define the action of $F(-, +)$ on relations $R^-, R^+ \in \mathbb{D}$ such that it holds that $d_1' \sqsubseteq d_1$ and $d_2' \sqsubseteq d_2$ in elements $(d_1', d_1, d_2', d_2, (type), \Delta r)$ of $F(R^-, R^+)_n$,

$$
\begin{array}{ll}
F(\nabla,\nabla)_V & = \{(\bot,\ v_1,\ \bot,\ v_2,\ \tau,\ \mathit{\Delta} r)\ \} \cup \\
& \quad \{(v_1',\ v_1,\ v_2',\ v_2, \tau, \mathit{\Delta} r) \mid \\
& \qquad v_1' \sqsubseteq v_1 \neq \bot\ \wedge\ v_2' \sqsubseteq v_2 \neq \bot\ \wedge \\
& \qquad (v_1',\ v_1,\ v_2',\ v_2, \tau, \mathit{\Delta} r) \in \Diamond\ \} \\
& \quad \text{where} \\
\Diamond & = \{(in_1*,\ in_1*,\ in_1*,\ in_1*,\ \text{unit},\ \mathit{\Delta} r)\ \} \cup \\
& \quad \{(in_{\mathbb{Z}}n,\ in_{\mathbb{Z}}n,\ in_{\mathbb{Z}}n,\ in_{\mathbb{Z}}n,\ \text{int},\ \mathit{\Delta} r) \mid n \in \mathbb{Z}\ \} \cup \\
& \quad \{(in_{\mathbb{L}}l,\ in_{\mathbb{L}}l,\ in_{\mathbb{L}}l,\ in_{\mathbb{L}}l,\ (\mathit{\Delta} l)\text{ref},\ \mathit{\Delta} r) \mid l \in \mathit{dom}(\mathit{\Delta})\ \} \cup \\
& \quad \{(in_{\oplus}in_i d_1',\ in_{\oplus}in_i d_1,\ in_{\oplus}in_i d_2',\ in_{\oplus}in_i d_2,\ \tau_1+\tau_2,\ \mathit{\Delta} r) \mid \\
& \qquad \exists \mathit{\Delta}_0 r_0 \lhd \mathit{\Delta} r.\ (d_1',\ d_1,\ d_2,\ d_2,\ \tau_i,\ \mathit{\Delta}_0 r_0) \in \nabla_V,\ i \in \{1,2\}\ \} \cup \\
& \quad \{(in_{\otimes}(d_{1a}', d_{1b}'),\ in_{\otimes}(d_{1a}, d_{1b}),\ in_{\otimes}(d_{2a}', d_{2b}'),\ in_{\otimes}(d_{2a}, d_{2b}), \\
& \qquad \tau_a \times \tau_b,\ \mathit{\Delta} r) \mid \\
& \qquad \exists \mathit{\Delta}_0 r_0 \lhd \mathit{\Delta} r.\ (d_{1a}',\ d_{1a},\ d_{2a}',\ d_{2a}',\ \tau_a,\ \mathit{\Delta}_0 r_0) \in \nabla_V \text{ and} \\
& \qquad\qquad (d_{1b}',\ d_{1b},\ d_{2b}',\ d_{2b},\ \tau_b,\ \mathit{\Delta}_0 r_0) \in \nabla_V\ \} \cup \\
& \quad \{(in_{\ominus}d_1',\ in_{\ominus}d_1,\ in_{\ominus}d_2'\ in_{\ominus}d_2,\ \tau \to \mathrm{T}\tau',\ \mathit{\Delta} r) \mid \\
& \qquad \forall \mathit{\Delta}' r' \rhd \mathit{\Delta} r,\ (v_1'\ , v_1,\ v_2',\ v_2,\ \tau,\ \mathit{\Delta}' r') \in \nabla_V. \\
& \qquad\qquad (d_1' v_1',\ d_1 v_1,\ d_2' v_2',\ d_2 v_2,\ \mathrm{T}\tau',\ \mathit{\Delta}' r') \in \nabla_M\ \} \cup \\
& \quad \{(in_{\mu}d_1',\ in_{\mu}d_1,\ in_{\mu}d_2',\ in_{\mu}d_2,\ \mu\alpha.\tau, \mathit{\Delta} r) \mid \\
& \qquad \exists \mathit{\Delta}_0 r_0 \lhd \mathit{\Delta} r.\ (d_1',\ d_1,\ d_2',\ d_2,\ \tau[\mu\alpha.\tau/\alpha],\ \mathit{\Delta}_0 r_0) \in \nabla_V\ \}
\end{array}
$$

$$
\begin{array}{ll}
F(\nabla,\nabla)_K & = \{(k_1',\ k_1,\ k_2',\ k_2,\ (x:\tau)^{\top},\ \mathit{\Delta} r) \mid \\
& \qquad k_1' \sqsubseteq k_1\ \wedge\ k_2' \sqsubseteq k_2\ \wedge\ \forall \mathit{\Delta}' r' \rhd \mathit{\Delta} r. \\
& \qquad \forall (S_1',\ S_1,\ S_2',\ S_2,\ \mathit{\Delta}' r') \in \nabla_S. \\
& \qquad \forall (v_1',\ v_1,\ v_2',\ v_2,\ \tau,\ \mathit{\Delta}' r') \in \nabla_V. \\
& \qquad\qquad (k_1' S_1' v_1' = \top \Rightarrow k_2 S_2 v_2 = \top) \wedge \\
& \qquad\qquad (k_2' S_2' v_2' = \top \Rightarrow k_1 S_1 v_1 = \top)\ \}
\end{array}
$$

$$
\begin{array}{ll}
F(\nabla,\nabla)_M & = \{(m_1',\ m_1,\ m_2',\ m_2,\ T\tau,\ \mathit{\Delta} r) \mid \\
& \qquad m_1' \sqsubseteq m_1\ \wedge\ m_2' \sqsubseteq m_2\ \wedge\ \forall \mathit{\Delta}' r' \rhd \mathit{\Delta} r. \\
& \qquad \forall (k_1',\ k_1,\ k_2',\ k_2,\ (x:\tau)^{\top},\ \mathit{\Delta}' r') \in \nabla_K. \\
& \qquad \forall (S_1',\ S_1,\ S_2',\ S_2,\ \mathit{\Delta}' r') \in \nabla_S\ . \\
& \qquad\qquad (m_1' k_1' S_1' = \top \Rightarrow m_2 k_2 S_2 = \top) \wedge \\
& \qquad\qquad (m_2' k_2' S_2' = \top \Rightarrow m_1 k_1 S_1 = \top)\ \}
\end{array}
$$

$$
\begin{array}{ll}
F(\nabla,\nabla)_S & = \{(\bot,\ S_1,\ \bot,\ S_2,\ \mathit{\Delta} r)\ \} \cup \\
& \quad \{(S_1',\ S_1,\ S_2',\ S_2,\ \mathit{\Delta} r) \mid r = \{r_1, \ldots, r_n\} \wedge \\
& \qquad S_1' \sqsubseteq S_1 \neq \bot\ \ \wedge\ S_2' \sqsubseteq S_2 \neq \bot\ \wedge\ \forall i \neq j,\ i,j \in 1,\ldots,n. \\
& \qquad A_{ri1}(S_1) \cap A_{rj1}(S_1) = \emptyset\ \wedge\ A_{ri2}(S_2) \cap A_{rj2}(S_2) = \emptyset \wedge \\
& \qquad \mathit{dom}(\mathit{\Delta}) \cap A_{r1}(S_1) = \emptyset\ \wedge\ \mathit{dom}(\mathit{\Delta}) \cap A_{r2}(S_2) = \emptyset \wedge \\
& \qquad \forall l \in \mathit{dom}(\mathit{\Delta}).(S_1' l,\ S_1 l,\ S_2' l,\ S_2 l,\ \mathit{\Delta} l, \mathit{\Delta} r) \in \nabla_V \wedge \\
& \qquad \forall r_a \in r. \exists (P_b, LL_b) \in r_a.\ (S_1, S_2) \in \hat{p}_b \wedge \\
& \qquad\qquad \forall (l_1, l_2, \tau) \in LL_b.(S_1' l_1, S_1 l_1, S_2' l_2, S_2 l_2, \tau, \mathit{\Delta} r) \in \nabla_V
\end{array}
$$

Fig. 1. Invariant Relation ∇

$n \in \{V, K, M, S\}$. In the definition of $F(R^-, R^+)_S \in \mathcal{R}(i^{-1}\mathbb{S})$ the accessibility maps and the simple state relations mentioned in a parameter Δr are only used on the non-primed elements s_1, s_2 from $(s'_1, s_1, s'_2, s_2, \Delta r)$. As explained, approximation will be carried out on the primed domain elements. Therefore, we define application of a pair of functions (f, j) to a relation only for $f \sqsubseteq j$ with j an isomorphism $j \in \{i, i^{-1}, id_{\mathbb{D}}, id_{F(\mathbb{D},\mathbb{D})}\}$. In an application $(f, j)R$ we apply f to the elements in the primed positions, and j to the elements of the non-primed positions. Then we define $(f, j) : R \subset S$ to mean that set theoretically $(f, j)R \subseteq S$. It holds that $F(R^-, R^+)$ preserves admissibility of R^+. It also holds that $R^-, R^+, S^-, S^+ \in \mathcal{R}(\mathbb{D})$ with $(f^-, id_{\mathbb{D}}) : S^- \subset R^-$ and $(f^+, id_{\mathbb{D}}) : R^+ \subset S^+$ implies $(F(f^-, f^+), id_{F(\mathbb{D},\mathbb{D})}) : F(R^-, R^+) \subset F(S^-, S^+)$. These properties are essential for the proof of existence of the invariant relation ∇.

Proposition 1 (Weakening). *For all $\Delta' r' \rhd \Delta r$,*

- $(v'_1, v_1, v'_2, v_2, \tau, \Delta r) \in \nabla_V \Rightarrow (v'_1, v_1, v'_2, v_2, \tau, \Delta' r') \in \nabla_V$,
- $(k'_1, k_1, k'_2, k_2, (x : \tau)^\top, \Delta r) \in \nabla_K \Rightarrow (k'_1, k_1, k'_2, k_2, (x : \tau)^\top, \Delta' r') \in \nabla_K$,
- $(m'_1, m_1, m'_2, m_2, T\tau, \Delta r) \in \nabla_M \Rightarrow (m'_1, m_1, m'_2, m_2, T\tau, \Delta' r') \in \nabla_M$.

Below we define a binary relation between denotations of typing judgement conclusions. This relation will be used as basis for proofs of contextual equivalence. The relation is defined by reference to the 4-ary relations from ∇. For two closed terms, two continuations, or two states the binary relation requires that their denotations d_1, d_2 are related as two pairs $(d_1, d_1, d_2, d_2, (type), parameter) \in \nabla_j$. The denotations of open value-terms with n free variables belong to $\mathbb{V}^n \multimap \mathbb{V}$, denotations of open computation terms to $\mathbb{V}^n \multimap \mathbb{M}$. They must give related elements in ∇ whenever they are applied to n-tuples of ∇-related elements form $\mathbb{V}$.

Definition 7 (Relating denotations of open expressions).

- *For all $\Gamma = x_1 : \tau_1, \ldots, x_n : \tau_n$ and $\Delta; \Gamma \vdash V_1 : \tau$ and $\Delta; \Gamma \vdash V_2 : \tau$ let $v_1 = [\![\Delta; \Gamma \vdash V_1 : \tau]\!]$ and $v_2 = [\![\Delta; \Gamma \vdash V_2 : \tau]\!]$, and define*

$$\begin{aligned}&(v_1, v_2, \tau, \Delta r) \in \nabla^\Gamma_V \overset{def}{\iff}\\ &\quad \forall \Delta' r' \rhd \Delta r. \forall i \in \{1, \ldots, n\}. \forall (v'_{1i}, v_{1i}, v'_{2i}, v_{2i}, \tau_i, \Delta' r') \in \nabla_V.\\ &\qquad (v_1(\overline{v'_{1i}}), v_1(\overline{v_{1i}}), v_2(\overline{v'_{2i}}), v_2(\overline{v_{2i}}), \tau, \Delta' r') \in \nabla_V.\end{aligned}$$

- *For all $\Gamma = x_1 : \tau_1, \ldots, x_n : \tau_n$, $\Delta; \Gamma \vdash M_1 : T\tau$ and $\Delta; \Gamma \vdash M_2 : T\tau$, let $m_1 = [\![\Delta; \Gamma \vdash M_1 : T\tau]\!]$ and $m_2 = [\![\Delta; \Gamma \vdash M_2 : T\tau]\!]$, and define*

$$\begin{aligned}&(m_1, m_2, T\tau, \Delta r) \in \nabla^\Gamma_M \overset{def}{\iff}\\ &\quad \forall \Delta' r' \rhd \Delta r. \forall i \in \{1, \ldots, n\}. \forall (v'_{1i}, v_{1i}, v'_{2i}, v_{2i}, \tau_i, \Delta' r') \in \nabla_V.\\ &\qquad (m_1(\overline{v'_{1i}}), m_1(\overline{v_{1i}}), m_2(\overline{v'_{2i}}), m_2(\overline{v_{2i}}), T\tau, \Delta' r') \in \nabla_M.\end{aligned}$$

- *For all $\Delta; \vdash K_1 : (x : \tau)^\top$ and $\Delta; \vdash K_2 : (x : \tau)^\top$, let $k_1 = [\![\Delta; \vdash K_1 : (x : \tau)^\top]\!]$ and $k_2 = [\![\Delta; \vdash K_2 : (x : \tau)^\top]\!]$, and define*

$$(k_1, k_2, (x : \tau)^\top, \Delta r) \in \nabla^\emptyset_K \overset{def}{\iff} (k_1, k_1, k_2, k_2, (x : \tau)^\top, \Delta r) \in \nabla_K.$$

– *For all $\Sigma_1 : \Delta$, $\Sigma_2 : \Delta$, let $S_1 \in [\![\Sigma_1 : \Delta]\!]$ and $S_2 \in [\![\Sigma_2 : \Delta]\!]$, and define*

$$(S_1, S_2, \Delta r) \in \nabla^{\emptyset}_{S} \overset{def}{\iff} (S_1, S_1, S_2, S_2, \Delta r) \in \nabla_S.$$

Lemma 1.
Suppose $(m_1, m_2, T\tau, \Delta r) \in \nabla^{\Gamma}_{M}$. We then have that

$$\begin{array}{l} \forall \Delta' r' \rhd \Delta r . \forall j \in \{1, \ldots, n\}. \forall (v_{1j}, v_{2j}, \tau_j, \Delta' r') \in \nabla^{\emptyset}_{V}. \\ \forall (k_1, k_2, (x : \tau)^{\top}, \Delta' r') \in \nabla^{\emptyset}_{K}. \forall (S_1, S_2, \Delta' r') \in \nabla^{\emptyset}_{S}. \\ \quad (i^{-1}(m_1(\overline{v_{1j}})))k_1 S_1 = \top \iff (i^{-1}(m_2(\overline{v_{2j}})))k_2 S_2 = \top. \end{array}$$

Theorem 3 (Fundamental Theorem). *For all parameters Δr it holds that*

– *if $\Delta; \Gamma \vdash V : \tau$ then $([\![\Delta; \Gamma \vdash V : \tau]\!], [\![\Delta; \Gamma \vdash V : \tau]\!], \tau, \Delta r) \in \nabla^{\Gamma}_{V}$,*
– *if $\Delta; \Gamma \vdash M : T\tau$ then $([\![\Delta; \Gamma \vdash M : T\tau]\!], [\![\Delta; \Gamma \vdash M : T\tau]\!], T\tau, \Delta r) \in \nabla^{\Gamma}_{M}$.*

The Fundamental Theorem is proved in the standard way by showing that all the typing rules preserve relatedness in ∇^{Γ}; weakening (Proposition 1) is used in several proof cases.

Lemma 2.

– $\forall r.\ ([\![\Delta; \vdash val\ x : (x : \tau)^{\top}]\!], [\![\Delta; \vdash val\ x : (x : \tau)^{\top}]\!], (x : \tau)^{\top}, \Delta r) \in \nabla^{\emptyset}_{K}$,
– *if $S \in [\![\Delta]\!]$ then $(S, S, \Delta id_{\emptyset}) \in \nabla^{\emptyset}_{S}$.*

The following theorem expresses that we can show two computations or two values to be contextually equivalent by showing that they are related in ∇^{Γ} under a parameter $\Delta id_{\emptyset}$, which does not require that any hidden invariants hold for states. The computations may themselves be able to build up local state invariants and a proof of relatedness will often require one to express these invariants; see the examples in the next section.

Theorem 4 (Contextual Equivalence). *Let $C[\_] : (\Delta; \Gamma \vdash \gamma) \Rightarrow (\Delta; \vdash T\tau)$ be a context. If $\Delta; \Gamma \vdash G_1 : \gamma$ and $\Delta; \Gamma \vdash G_2 : \gamma$ and*

$$([\![\Delta; \Gamma \vdash G_1 : \gamma]\!], [\![\Delta; \Gamma \vdash G_2 : \gamma]\!], \gamma, \Delta id_{\emptyset}) \in \nabla^{\Gamma}_{j},\ j \in \{V,M\}$$

then

$$\forall \Sigma : \Delta.\ (\Sigma, let\ x \Leftarrow C[G_1]\ in\ val\ x \downarrow \iff \Sigma, let\ x \Leftarrow C[G_2]\ in\ val\ x \downarrow).$$

5 Examples

Before presenting our examples, we will first sketch how a typical proof of contextual equivalence proceeds. Thus, suppose we wish to show that two computations m_1 and m_2 are contextually equivalent. We then need to show that they are related in a parameter $\Delta id_{\emptyset}$ or, equivalently, in Δr, for any r. This requires us to show, for any extended parameter $\Delta^1 r^1$, any pair[1] of continuations k_1 and k_2 related in $\Delta^1 r^1$, and any pair of states S_1 and S_2 related in $\Delta^1 r^1$, $m_1 k_1 S_1$ and $m_2 k_2 S_2$ have the same termination behaviour. The latter amounts to showing that $k_1(S_1[\ldots])v_1$ and $k_2(S_2[\ldots])v_2$ have the same termination behaviour, where $S_1[\ldots]$ and $S_2[\ldots]$ are potentially updated versions of S_1 and S_2; and v_1 and v_2 are values. Since k_1 and k_2 are assumed related in $\Delta^1 r^1$, it suffices to define a parameter $\Delta^2 r^2$ extending $\Delta^1 r^1$ and show that $S_1[\ldots]$ and $S_2[\ldots]$ are related in $\Delta^2 r^2$ and that v_1 and v_2 are related in $\Delta^2 r^2$. Typically, the definition of the parameter $\Delta^2 r^2$ essentially consists of defining one or more local parameters, which capture the intuition for why the computations are related.

In the first example below we prove that M and N from the Introduction are contextually equivalent. In this case, the only local parameter we have to define is $\tilde{r}^3 = ((P_1, LL_1) \vee (P_2, LL_2))$, where

$$\begin{aligned} P_1 &= (\{(S_1, S_2) \mid S_2 l_p = 0\}, A_\emptyset, A_{\{l_p\}}), & LL_1 &= \{(l_y, l_{y0})\}, \\ P_2 &= (\{(S_1, S_2) \mid S_2 l_p = n \neq 0\}, A_\emptyset, A_{\{l_p\}}), & LL_2 &= \{(l_y, l_{y1})\}. \end{aligned}$$

This local parameter expresses that, depending on the value of $S_2(l_p)$, either the locations (l_y, l_{y0}) or the locations (l_y, l_{y1}) contain related values.

In the first subsection below we present the proof of contextual equivalence of M and N in detail. Formally, there are several cases to consider, but do note that the proof follows the outline given above and is almost automatic except for the definition of the local parameter shown above.

5.1 Example 1

Consider the programs M and N from the Introduction.

We want to show that M and N are related in any parameter Δr, that is $\forall \Delta r.\ (⟦\emptyset; \vdash M : \sigma⟧, ⟦\emptyset; \vdash N : \sigma⟧, \sigma, \Delta r) \in \nabla_V^\emptyset$. Here $\sigma = (\tau \to T\tau') \to T(\sigma_1 \times \sigma_2)$, and $\sigma_1 = (\tau \to T\tau') \to T\mathrm{unit}$ and $\sigma_2 = \mathrm{unit} \to (\tau \to T\tau')$. As M and N are values of function type, their denotations have the forms $in_{\multimap} d_M$ and $in_{\multimap} d_N$. We need to show $\forall \Delta^1 r^1 \rhd \Delta r. \forall (v_1', v_1, v_2', v_2, \tau \to T\tau', \Delta^1 r^1) \in \nabla_V.\ (d_M v_1', d_M v_1, d_N v_2', d_N v_2, T(\sigma_1 \times \sigma_2), \Delta^1 r^1) \in \nabla_M$.

It suffices to show that $\forall \Delta^2 r^2 \rhd \Delta^1 r^1. \forall (k_1', k_1, k_2', k_2, (x : \sigma_1 \times \sigma_2)^\top, \Delta^2 r^2) \in \nabla_K$. $\forall (S_1', S_1, S_2', S_2, \Delta^2 r^2) \in \nabla_S$ it holds that $(d_M v_1')k_1' S_1' = \top \implies (d_N v_2) k_2 S_2 = \top$ and $(d_N v_2')k_2' S_2' = \top \implies (d_M v_1) k_1 S_1 = \top$.

Now,$(d_M v_1) k_1 S_1 = k_1(S_1[l_y \mapsto v_1])(⟦\emptyset; y \vdash recf_{1M}⟧(y \mapsto l_y), ⟦\emptyset; y \vdash recf_{2M}⟧(y \mapsto l_y))$,

where l_y is a location that is fresh wrt. the store S_1 in combination with the parameter $\Delta^2 r^2$, i.e.,

$$l_y \notin dom(\Delta^2) \cup A_{r^2 1}(S_1). \tag{1}$$

The value of $(d_M v_1')k_1' S_1'$ is similar.
Moreover,

$$\begin{aligned} (d_N v_2) k_2 S_2 = k_2\,&(S_2[l_p \mapsto in_{\mathbb{Z}} 0, l_{y0} \mapsto v_2, l_{y1} \mapsto v_2]) \\ &(⟦\emptyset; p, y_0, y_1 \vdash recf_{1N}⟧(p \mapsto l_p, y_0 \mapsto l_{y0}, y_1 \mapsto l_{y1}), \\ &\ ⟦\emptyset; p, y_0, y_1 \vdash recf_{2N}⟧(p \mapsto l_p, y_0 \mapsto l_{y0}, y_1 \mapsto l_{y1})), \end{aligned}$$

[1] Formally, we consider 4-tuples.

where l_p, l_{y0}, l_{y1} are locations that are fresh wrt. the store S_2 in combination with the parameter $\Delta^2 r^2$, i.e

$$l_p, l_{y0}, l_{y1} \notin dom(\Delta^2) \cup A_{r^2 2}(S_2). \tag{2}$$

The value of $(d_N v_2')k_2' S_2'$ is similar.

Since the continuations are related in the parameter $\Delta^2 r^2$ it suffices to show that, if $S_1' \neq \bot \vee S_2' \neq \bot$ then we can give an extended parameter $\Delta^3 r^3 \rhd \Delta^2 r^2$ such that the updated states and the values (pairs of (set,get)) are related in the extended parameter $\Delta^3 r^3$.

We let $\Delta^3 r^3 = \Delta^2(r^2 \cup \{\tilde{r}^3\})$, where $\tilde{r}^3 = ((P_1, LL_1) \vee (P_2, LL_2))$, and

$$\begin{array}{ll} P_1 = (\{(S_1,S_2) \mid S_2 l_p = 0\}, A_\emptyset, A_{\{l_p\}}), & LL_1 = \{(l_y, l_{y0}, \tau \to T\tau')\}, \\ P_2 = (\{(S_1,S_2) \mid S_2 l_p = n \neq 0\}, A_\emptyset, A_{\{l_p\}}), & LL_2 = \{(l_y, l_{y1}, \tau \to T\tau')\}. \end{array}$$

Recall $\forall S.\ A_\emptyset(S) = \emptyset \ \wedge\ \forall S.\ A_{\{l_p\}}(S) = \{l_p\}$.
Then it holds that the accessibility maps associated with the local parameter $\tilde{r}^3$, are given by $\forall S. A_{\tilde{r}^3_1}(S) = \{l_y\}$ and $\forall S. A_{\tilde{r}^3_2}(S) = \{l_p, l_{y0}, l_{y1}\}$.

We now verify that

$$\begin{array}{l} (\,S_1'[l_y \mapsto v_1'], S_1[l_y \mapsto v_1], S_2'[l_p \mapsto in_{\mathbb{Z}} 0, l_{y0} \mapsto v_2', l_{y1} \mapsto v_2'], \\ \quad S_2[l_p \mapsto in_{\mathbb{Z}} 0, l_{y0} \mapsto v_2, l_{y1} \mapsto v_2], \Delta^3 r^3) \in \nabla_S. \end{array} \tag{3}$$

By (1) and (2), all locations viewed by the local parameter $\tilde{r}^3$ are disjoint from $dom(\Delta^2)$ and from all local areas viewed by r^2. The stores have only been changed in locations viewed by $\tilde{r}^3$. Since values related in a parameter are also related in any extending parameter (weakening) every requirement from $\Delta^2 r^2$ still holds. Finally, since $S_2[l_p \mapsto in_{\mathbb{Z}} 0, l_{y0} \mapsto v_2, l_{y1} \mapsto v_2](l_p) = 0$ and the values stored in locations l_y and l_{y0} in the updated stores, namely v_1', v_1, v_2', v_2, are related in $\Delta^1 r^1$ and then by weakening also in $\Delta^3 r^3$, the first disjunct of $\tilde{r}^3$ is satisfied, and hence (3) holds.

It remains to show

<u>A:</u> $([\![\emptyset; y \vdash recf_{1M}]\!](y \mapsto l_y), [\![\emptyset; y \vdash recf_{1M}]\!](y \mapsto l_y), [\![\emptyset; p, y_0, y_1 \vdash f_{1N}]\!](p \mapsto l_p, y_0 \mapsto l_{y0}, y_1 \mapsto l_{y1}), [\![\emptyset; p, y_0, y_1 \vdash recf_{1N}]\!](p \mapsto l_p, y_0 \mapsto l_{y0}, y_1 \mapsto l_{y1}), (\tau \to T\tau') \to T\text{unit}, \Delta^3 r^3) \in \nabla_V$ and

<u>B:</u> $([\![\emptyset; y \vdash recf_{2M}]\!](y \mapsto l_y), [\![\emptyset; y \vdash recf_{2M}]\!](y \mapsto l_y), [\![\emptyset; p, y_0, y_1 \vdash recf_{2N}]\!](p \mapsto l_p, y_0 \mapsto l_{y0}, y_1 \mapsto l_{y1})[\![\emptyset; p, y_0, y_1 \vdash recf_{2N}]\!](p \mapsto l_p, y_0 \mapsto l_{y0}, y_1 \mapsto l_{y1}), (\tau \to T\tau') \to T\text{unit}, \Delta^3 r^3) \in \nabla_V$

Now let $\Delta^4 r^4 \rhd \Delta^3 r^3$, $(w_1', w_1, w_2', w_2, \tau \to T\tau', \Delta^4 r^4) \in \nabla_V$, and
let $\Delta^5 r^5 \rhd \Delta^4 r^4, (K_1', K_1, K_2', K_2, (x : \tau \to T\tau')^\top \Delta^5 r^5) \in \nabla_K, (S_1', S_1, S_2', S_2, \Delta^5 r^5) \in \nabla_S$, $(c_1', c_1, c_2', c_2, (x : \text{unit})^\top, \Delta^5 r^5) \in \nabla_K$

We have denotations $[\![\emptyset; y \vdash recf_{1M}]\!](y \mapsto l_y) = in_{\multimap} d_{M1}$, $[\![\emptyset; p, y_0, y_1 \vdash recf_{1N}]\!](p \mapsto l_p, y_0 \mapsto l_{y0}, y_1 \mapsto l_{y1}) = in_{\multimap} d_{N1}$, $[\![\emptyset; y \vdash recf_{2M}]\!](y \mapsto l_y) = in_{\multimap} d_{M2}$, $[\![\emptyset; p, y_0, y_1 \vdash recf_{2N}]\!](p \mapsto l_p, y_0 \mapsto l_{y0}, y_1 \mapsto l_{y1}) = in_{\multimap} d_{N2}$.

<u>A:</u> Now we want to show relatedness of the setters. As before if $w_1' = w_2' = \bot$ or $S_1' = S_2' = \bot$ we are done. Otherwise we reason as follows.

Observe that $(d_{M1} w_1) c_1 S_1 = c_1(S_1[l_y \mapsto w_1]) in_1 *$ and similarly $(d_{M1} w_1) c_1' S_1' = c_1'(S_1'[l_y \mapsto w_1']) in_1 *$. Also, $(d_{N1} w_2) c_2 S_2 = c_2(S_2[l_p \mapsto in_{\mathbb{Z}} 0, l_{y0} \mapsto w_2]) in_1 *$, if $S_2 l_P \neq 0$, and $(d_{N1} w_2) c_2 S_2 = c_2(S_2[l_p \mapsto in_{\mathbb{Z}} 1, l_{y1} \mapsto w_2]) in_1 *$, if $S_2 l_P = 0$. Similarly for the approximation $(d_{N1} w_2') c_2' S_2'$.

Since the states are related in $\Delta^5 r^5$ which is an extension of $\Delta^3 r^3$ we know that the content of $S_2 l_p$ is $in_{\mathbb{Z}} n$ for some n. We know that the continuations c_1', c_1, c_2', c_2 are related in $\Delta^5 r^5$. $(in_1 *, in_1 *, in_1 *, in_1 *, \text{unit}, \Delta^5 r^5)$ since they are related in any parameter. So if we can show that the updated states are related in $\Delta^5 r^5$ we are done.

The states S_1', S_1, S_2', S_2 are related in $\Delta^5 r^5$. All changes are only within the store areas belonging to $\tilde{r}^3$ and the changes preserve the invariant for $\tilde{r}^3$, hence the updated states are still related in $\Delta^5 r^5$. We conclude that the setters are related in $\nabla^3 r^3$.

B: Now we want to show relatedness of the getters. As before, if the denotations are applied to related unit type values where the approximations are $\perp$ or if $S_1' = S_2' = \perp$ we are done. Otherwise we reason as follows. Note that $(d_{M2} in_1 *) K_1 S_1 = K_1 S_1 (S_1 l_y)$ and similarly $(d_{M2} in_1 *) K_1' S_1' = K_1' S_1' (S_1' l_y)$. Since the states are not $\perp$ and are related in $\Delta^5 r^5$ which is an extension of $\Delta^3 r^3$ we know that the content of $S_2 l_p$ is $in_{\mathbb{Z}} n$ for some n. We have that $(d_{N2} in_1 *) K_2 S_2 = K_2 S_2 (S_2 l_{y0})$, if $n = 0$, and $(d_{N2} in_1 *) K_2 S_2 = K_2 S_2 (S_2 l_{y1})$, if $n \neq 0$. Similarly for the approximation $(d_{N2} in_1 *) K_2' S_2'$.

We know that the continuations K_1', K_1, K_2', K_2 and the states $S_1' S_1, S_2', S_2$ are related in $\Delta^5 r^5$. So if we can show that the retrieved values are related in $\Delta^5 r^5$ we are done.

Since the states $S_1' S_1, S_2', S_2$ are related in $\Delta^5 r^5$ they satisfy the invariant of $\tilde{r}^3$. So the content of $S_2 l_p$ is $in_{\mathbb{Z}} n$ for some n. If $n = 0$ then $S_1' l_y, S_1 l_y, S_2' l_{y0}, S_2 l_{y0}$ are related in $\Delta^5 r^r$, and if $n \neq 0$ then $S_1' l_y, S_1 l_y, S_2' l_{y1}, S_2 l_{y1}$ are related in $\Delta^5 r^r$, again by the requirement from $\tilde{r}^3$. This is what we need for the retrieved values to be related. We conclude that the getters are related in $\nabla^3 r^3$.

Then we can conclude that $([\![M]\!], [\![N]\!], \sigma, \Delta r) \in \nabla_V^{\emptyset}$, and as Δr was arbitrary that they are related in any parameter. Hence the programs M and N are contextually equivalent.

5.2 Example 2

Consider the computation terms M' and N' from the Introduction. They both have a free variable g of function type. We want to show that M' and N' are related in any parameter Δr.

We need to show $\forall \Delta^1 r^1 \rhd \Delta r. \forall (g_1', g_1, g_2', g_2, \sigma, \Delta^1 r^1) \in \nabla_V$.
$\forall \Delta^2 r^2 \rhd \Delta^1 r^1. \forall (k_1', k_1, k_2', k_2, (x : \sigma_1)^\top, \Delta^2 r^2 \in \nabla_K). \forall (S_1', S_1, S_2', S_2, \Delta^2 r^2) \in \nabla_S$.
$[\![\emptyset; g : \sigma \vdash M' : T\sigma_1]\!](g \mapsto g_1') k_1' S_1' = \top \implies [\![\emptyset; g : \sigma \vdash N' : T\sigma_1]\!](g \mapsto g_2) k_2 S_2 = \top$ and $[\![\emptyset; g : \sigma \vdash N' : T\sigma_1]\!](g \mapsto g_2') k_2' S_2' = \top \implies [\![\emptyset; g : \sigma \vdash M' : T\sigma_1]\!](g \mapsto g_1) k_1 S_1 = \top$.
Here $\sigma = \sigma_1 \to T\text{unit}$, and $\sigma_1 = \text{unit} \to T\text{unit}$.

For the proof of this we define a local parameter $\tilde{r}^3 = (P^3, \emptyset)$ for $P^3 = (\{(S_a, S_b) | S_b l_x = in_{\mathbb{Z}} n > 0)\}, A_\emptyset, A_{\{l_x\}})$, where l_x is fresh for $dom(\Delta^2) \cup A_{r^2 2}(S_2)$. Then we have a parameter $\Delta^3 r^3$ where $\Delta^3 = \Delta^2$ and $r^3 = r^2 \cup \{\tilde{r}^3\}$ which we use in the proof.

6 Conclusion

We have presented a local relational proof method for establishing contextual equivalence of expressions in a language with recursive types and general references, building on earlier work of Benton and Leperchey [7]. The proof of existence of the logical relation is fairly intricate because of the interplay between recursive types and local parameters for reasoning about higher-order store. However, the method is easy to use on examples: the only non-trivial steps are to guess the right local parameters — but since the local parameters express the intuitive reason for contextual equivalence, the non-trivial steps are really fairly straightforward. It is possible to extend our method to a language also with impredicative polymorphism; we will report on that on another occasion.

Relational Parametricity for Recursive Types and References of Closed Types

Lars Birkedal
Nina Bohr

Abstract

We present a relationally parametric model of a language with impredicative polymorphism, general recursive types and general references of closed types. The model provides a *local* relational reasoning method for reasoning about parametricity (representation independence). Our development builds on the work of Bohr and Birkedal who, based on earlier work by Benton and Leperchey, devised a nominal semantics and a local relational reasoning method for reasoning about contextual equivalence for a language with recursive types and general references. Here we extend the ideas to a language with impredicative polymorphism for which we devise a relationally parametric model.

1 Introduction

Relational parametricity was proposed by Reynolds to reason about polymorphic programs, in particular, to show equivalence of polymorphic programs and to show representation independence for abstract data types.

The theory of relational parametricity was originally proposed in the setting of the second-order lambda calculus. That setting is by now fairly well-understood, see, e.g., (45; 10) But, of course, we would like to use relational parametricity for real programs with recursion and other effects. There has been a lot of research towards this goal — the efforts can be grouped roughly into two categories: *equational type theories with effects* and *programming languages with effects.*

Work in former category was initiated by Plotkin (44), who suggested a second-order linear type theory to combine polymorphism with recursion. That approach was further investigated in (11). One of the remarkable features of this calculus is that it allows one to encode a wide range of data types, including recursive types, with the desired universal properties following from parametricity. Hasegawa studied the combination of polymorphism and another effect, namely control operators (19). Recently, this line of work was extended by Møgelberg and Simpson (36), who proposed a general polymorphic type theory for effects, as captured by computational monads. The general framework has been specialized to control effects in (37).

Work in the latter category focuses on programming languages defined using an operational semantics, specifying evaluation order, etc., and was initiated by Wadler (58). Relational parametricity is concerned with program equivalence which is here typically defined as *contextual equivalence*: two program expressions are equivalent if they have the same observable behaviour when placed in any program context C. It is generally quite hard to show directly that two program expressions are contextually equivalent because of the universal quantification over all contexts. Thus there has been an extensive research effort to find reasoning methods that are easier to use for establishing contextual equivalence (see, e.g., (40) for a fairly recent overview), and the work on parametricity for programming languages with effects has been closely related to the research on reasoning methods for contextual equivalence. Relationally parametric models have been developed for languages with recursion and inductive / coinductive types (38; 21) and, recently, also for langauges with recursive types (34; 3; 15).

The two categories of work are of course related in that the type theories can be used to give semantics to programming languages. This has, e.g., been done by Møgelberg (35), who showed how to give a parametric model of the programming language FPC extended with polymorphism (i.e., a language with recursion, recursive types and polymorphism). Using a model of the type theory, adequacy wrt. the operational semantics of the programming language was proved, allowing Mogelberg to prove results about contextual equivalence using the reasoning principles of the type theory.

Thus most of the earlier work towards parametricity for languages with effects has focused on recursion (recursive functions and recursive types). Here we provide what appears to be the first relationally parametric model of a programming langauge with impredicative polymorphism, recursive types and general references of closed type. The challenge is not just to give any old model, but to give a reasonably useful one, which allows to prove representation independence results for programs that use *local state*. To achieve that, we leverage recent work on reasoning about contextual equivalence for programming languages with references. For such langauges, it is not enough to restrict attention to fewer contexts, since one also needs to reason about equivalence under *related* stores. To address this challenge, methods based on logical relations and bisimulations have been proposed, see, e.g., (42; 7; 55). Recently, the bisimulation approach has been simplified and extended to work for untyped languages with general references (28; 27). For effectiveness of the reasoning methods, we seek *local* reasoning methods, which only require that we consider the accessible part of a store and which works in the presence of a separated (non-interfering) invariant that is preserved by the context. In (7), Benton and Leperchey developed a relational reasoning method for a language with simple references that does allow for local reasoning. Their approach is inspired by related work on separation logic (47; 46). In particular, an important feature of the state relations of Benton and Leperchey is that they depend on only part of the store: that allows us to reason that related states are still related if we update them in parts on which the relation does not depend. In (13) we extended the work of Benton and Leperchey to relational reasoning about contextual equivalence for a language with general recursive types and general references (but without polymorphism). We arrived at a useful reasoning method, which, in particular, could be used to verify examples of (28).

Here we extend our earlier work in (13) to provide a relationally parametric model for a language with recursive types and general references. For technical reasons, we restrict reference types $\tau\mathsf{ref}$ somewhat, by requiring that τ must be a *closed* type.

2 Language

The language we consider is a call-by-value, monadically-typed λ-calculus with recursion, general recursive types, polymorphic types, and general dynamically allocated references. Types are either *value types* τ or *computation types* $T\tau$. We use both τ's and σ's to range over value types. Values of any closed value type can be stored in the store. For $\tau\mathsf{ref}$ to be a well formed type we require that τ is closed. $\mathit{TypeVar}$ denotes the set of type variables, $\mathit{ValueType}$ denotes the set of all value types, $\mathit{ComputationType}$ denotes the set of all computation types.

$$\begin{array}{rcl} \tau & ::= & \alpha \mid 1 \mid \mathsf{int} \mid \tau \times \tau \mid \tau + \tau \mid \tau\mathsf{ref} \mid \tau \to T\tau \mid \\ & & \mu\alpha.\tau \mid \forall\alpha.T\tau \\ \gamma & ::= & \tau \mid T\tau \end{array}$$

We assume infinite sets of variables, ranged over by x, type variables, ranged over by α, and locations, ranged over by l. We let Loc denote the set of locations. Type variable contexts, Ξ, are finite sequences of distinct type variables, $\Xi = \alpha_1, \ldots, \alpha_n$. Typing contexts, Γ, are finite maps from variables to value types. Store types Δ are finite maps from locations to closed value types. Terms G are either *values* V or *computations* M:

$$\begin{array}{rcl} V & ::= & x \mid \underline{n} \mid \underline{l} \mid () \mid (V, V') \mid \mathsf{in}_i V \mid \mathsf{rec}\, f(x:\tau) = M \mid \\ & & \mathsf{fold}\, V \mid \Lambda\alpha.\, M \\ M & ::= & VV' \mid \mathsf{let}\, x \Leftarrow M \,\mathsf{in}\, M' \mid \mathsf{val}\, V \mid \pi_i V \mid \mathsf{ref}\, V \mid !V \mid \\ & & V := V' \mid \mathsf{case}\, V \,\mathsf{of}\, \mathsf{in}_1 x_1 \Rightarrow M_1; \mathsf{in}_2 x_2 \Rightarrow M_2 \mid \\ & & V = V' \mid V + V' \mid \mathsf{iszero}\, V \mid \mathsf{unfold}\, V \mid V\tau \\ G & ::= & M \mid V. \end{array}$$

Continuations K take the following form:

$$K \quad ::= \quad \mathsf{val}\, x \mid \mathsf{let}\, y \Leftarrow M \;\mathsf{in}\; K$$

The typing judgments take the form

$$\Delta; \Xi; \Gamma \vdash V : \tau \qquad \Delta; \Xi; \Gamma \vdash M : T\tau \qquad \Delta; \Xi \vdash K : (x : \tau)^{\top}$$

$ContinuationType$ denotes the set of all continuation types.

The typing rules for values and terms are as in (7) extended with rules for recursive and polymorphic types, except that the only restriction for references is that the type must be closed. This restriction gives limitations to the storings that polymorfic values can perform. Here we just include the following five selected rules:

$$\frac{\Delta; \Xi; \Gamma \vdash V : \tau}{\Delta; \Xi; \Gamma \vdash \mathsf{ref} V : T(\tau \mathsf{ref}\,)} \quad (- \vdash \tau : type)$$

$$\frac{\Delta; \Xi; \Gamma \vdash V : \tau[\mu\alpha.\tau/\alpha]}{\Delta; \Xi; \Gamma \vdash \mathsf{fold}\; V : \mu\alpha.\tau}$$

$$\frac{\Delta; \Xi; \Gamma \vdash V : \mu\alpha.\tau}{\Delta; \Xi; \Gamma \vdash \mathsf{unfold}\; V : T(\tau[\mu\alpha.\tau/\alpha])}$$

$$\frac{\Delta; \Xi, \alpha; \Gamma \vdash M : T\tau \qquad \Xi \vdash \Gamma}{\Delta; \Xi; \Gamma \vdash \Lambda\alpha.\; M : \forall\alpha.T\tau}$$

$$\frac{\Delta; \Xi; \Gamma \vdash V : \forall\alpha.T\tau \qquad \Xi \vdash \sigma}{\Delta; \Xi; \Gamma \vdash V\sigma : T\tau[\sigma/\alpha]}$$

Stores Σ are finite maps from locations to closed values. A store Σ has store type Δ, written $\Sigma : \Delta$, if, for all l in the domain of Δ, $\Delta; ; \vdash \Sigma(l) : \Delta(l)$.

The operational semantics is defined via a termination judgment $\Sigma, \mathsf{let}\; x \Leftarrow M \;\mathsf{in}\; K \downarrow$, where M is closed and K is a *continuation term in* x. Typed continuation terms are defined by:

$$\frac{}{\Delta; \Xi \vdash \mathsf{val}\, x : (x : \tau)^{\top}}$$

$$\frac{\Delta; \Xi; x : \tau \vdash M : T\tau' \quad \Delta; \Xi \vdash K : (y : \tau')^{\top}}{\Delta; \Xi \vdash \mathsf{let}\, y \Leftarrow M \;\mathsf{in}\; K : (x : \tau)^{\top}}$$

The defining rules for the termination judgment $\Sigma, \mathsf{let}\, x \Leftarrow M \;\mathsf{in}\; K \downarrow$ are standard given that the language is call-by-value, with left-to-right evaluation order. We just include one rule as an example:

$$\frac{\Sigma, \mathsf{let}\, x \Leftarrow \mathsf{val}\; V \;\mathsf{in}\; K \downarrow}{\Sigma, \mathsf{let}\, x \Leftarrow \mathsf{unfold}(\mathsf{fold}\; V) \;\mathsf{in}\; K \downarrow}$$

A *context* is a computation term with a hole, and we write $C[.] : (\Delta; \Xi; \Gamma \vdash \gamma) \Rightarrow (\Delta; ; \vdash T\tau)$ to mean that whenever $\Delta; \Xi; \Gamma \vdash G : \gamma$ then $\Delta; ; \vdash C[G] : T\tau$.

The definition of contextual equivalence is standard and much as in (7).

Definition 1
If $\Delta; \Xi; \Gamma \vdash G_i : \gamma$, *for* $i = 1, 2$ *then* G_1 *and* G_2 *are* contextually equivalent, *written*

$$\Delta; \Xi; \Gamma \vdash G_1 =_{ctx} G_2,$$

if, for all types τ, *for all contexts* $C[.] : (\Delta; \Xi; \Gamma \vdash \gamma) \Rightarrow (\Delta; ; \vdash T\tau)$ *and for all stores* $\Sigma : \Delta$,

$$\Sigma, \textit{let}\; x \Leftarrow C[G_1] \;\textit{in val}\; x \downarrow \Longleftrightarrow \Sigma, \textit{let}\; x \Leftarrow C[G_2] \;\textit{in val}\; x \downarrow .$$

3 Denotational Semantics

We define a denotational semantics of the language from the previous section and show that the semantics is *adequate*. The denotational semantics is defined using FM-domains (52). The semantics and the adequacy proof, in particular the existence proof of the logical relation used to prove adequacy, builds on Shinwell's work on semantics of recursive types in FM-domains (52). Our approach is slightly different from that of Shinwell since we make use of universal domains to model impredicative polymorphism and the fact that any type of value can be stored in the store, but technically it is not that big a difference.

We begin by calling to mind some basic facts about FM-domains; see (52) for more details. Fix a countable set of atoms, which in our case will be the locations, Loc. A *permutation* is a bijective function $\pi \in (\mathit{Loc} \to \mathit{Loc})$ such that the set $\{l \mid \pi(l) \neq l\}$ is finite. An FM-set X is a set equipped with a permutation action: an operation $\pi \bullet - : \mathit{perms}(\mathit{Loc}) \times X \to X$ that preserves composition and identity, and such that each element $x \in X$ is finitely supported: there is a finite set $L \subset \mathit{Loc}$ such that whenever π fixes each element of L, the action of π fixes x: $\pi \bullet x = x$. There is a smallest such set, which we write $\mathit{supp}(x)$. A morphism of FM-sets is a function $f : D \to D'$ between the underlying sets that is equivariant: $\forall x.\pi \bullet (fx) = f(\pi \bullet x)$. An FM-cpo is an FM-set with an equivariant partial order relation $\sqsubseteq$ and least upper bounds of all finitely-supported ω-chains. A morphism of FM-cpos is a morphism of their underlying FM-sets that is monotone and preserves lubs of finitely-supported chains. We only require the existence and preservation of lubs of finitely-supported chains, so an FM-cpo may not be a cpo in the usual sense. The sets Z, N, etc., are discrete FM-cpos with the trivial action. The set of locations, Loc, is a discrete FM-cpo with the action $\pi \bullet l = \pi(l)$. The category of FM-cpos is bicartesian closed: we write 1 and $\times$ for the finite products, $D \Rightarrow D'$ for the internal hom and 0,$+$ for the coproducts. The action on products is pointwise, and on functions is given by conjugation: $\pi \bullet f = \lambda x.\pi \bullet (f(\pi^{-1} \bullet x))$. The category is not well-pointed: morphisms $1 \to D$ correspond to elements of $1 \Rightarrow D$ with empty support. The lift monad, $(-)_\perp$, is defined as usual with the obvious action. The Kleisli category $\mathbf{FM\text{–}Cppo}_\perp$ is the category of pointed FM-cpos (FM-cppos) and strict continuous maps, which is symmetric monoidal closed, with smash product $\otimes$ and strict function space $\multimap$. If D is a pointed FM-cpo then $\mathit{fix} : (D \Rightarrow D) \multimap D$ is defined by the lub of an ascending chain in the usual way. We write O for the discrete FM-cpo with elements $\perp$ and $\top$, ordered by $\perp \sqsubseteq \top$.

As detailed in (52), one may solve recursive domain equations in $\mathbf{FM\text{–}Cppo}_\perp$. For the denotational semantics, we use a minimal invariant recursive domain V satisfying:

$$\begin{array}{rcl} V & \cong & F(V,V) \\ F(V,V) & = & 1_\perp \oplus Z_\perp \oplus \mathit{Loc}_\perp \oplus (V \otimes V) \oplus \\ & & (V \oplus V) \oplus (V \multimap TV)_\perp \ \oplus (TV)_\perp \oplus V, \end{array}$$

where

$$\begin{array}{rcl} TV & = & (KV) \multimap (S \multimap O) \\ KV & = & S \multimap (V \multimap O) \\ S & = & \mathit{Loc}_\perp \multimap V. \end{array}$$

Formally, V is obtained as the minimal invariant solution to a locally FM-continuous functor $F : (\mathbf{FM\text{–}Cppo}_\perp)^{\mathrm{op}} \times \mathbf{FM\text{–}Cppo}_\perp \to \mathbf{FM\text{–}Cppo}_\perp$. We write i for the isomorphism $i : F(V, V) \cong V$. The injections into the sum are denoted as follows:

$$\begin{array}{l} \mathit{in}_1 : 1_\perp \multimap F(V, V) \\ \mathit{in}_{\mathsf{int}} : Z_\perp \multimap F(V, V) \\ \mathit{in}_{\mathsf{ref}} : \mathit{Loc}_\perp \multimap F(V, V) \\ \mathit{in}_\times : (V \otimes V) \multimap F(V, V) \\ \mathit{in}_+ : (V \oplus V) \multimap F(V, V) \\ \mathit{in}_\to : (V \multimap TV)_\perp \multimap F(V, V) \\ \mathit{in}_\forall : (TV)_\perp \multimap F(V, V) \\ \mathit{in}_\mu : V \multimap F(V, V), \end{array}$$

but we will often omit the isomorphism i and the injections into the sum writing, e.g., simply (v_1, v_2) for an element of $\mathbb{V}$.

Types, $\Xi \vdash \tau$ are interpreted by $[\![\Xi \vdash \tau]\!] = V$, computation types $\Xi \vdash T\tau$ are interpreted by $[\![\Xi \vdash T\tau]\!] = T\,V$, continuation types $\Xi \vdash (x : \tau)^\top$ are interpreted by $[\![\Xi \vdash (x : \tau)^\top]\!] = K\,V$, and store types Δ are interpreted by $[\![\Delta]\!] = S$. Type environments in context $\Xi \mid \Gamma = x_1 : \tau_1, \ldots, x_n : \tau_n$ are interpreted by $\bigotimes_{i\in\{1,\ldots,n\}} V$.

Typing judgments are interpreted as follows:

- $[\![\Delta; \Xi; \Gamma \vdash V : \tau]\!] \in ([\![\Xi \vdash \Gamma]\!] \multimap [\![\Xi \vdash \tau]\!])$
- $[\![\Delta; \Xi; \Gamma \vdash M : T\tau]\!] \in ([\![\Xi \mid \Gamma]\!] \multimap [\![\Xi \vdash T\tau]\!])$
- $[\![\Delta; \Xi \vdash K : (x : \tau)^\top]\!] \in K\,V$

The actual definition of the interpretations is quite standard, except for allocation which makes use of the properties of FM-cpo's:

$$\begin{array}{l}[\![\Delta; \Xi; \Gamma \vdash \mathsf{ref} V : T(\tau\mathsf{ref})]\!]\,\rho = \lambda k.\lambda S.\\ \quad k(S([l \mapsto [\![\Delta; \Xi; \Gamma \vdash V : \tau]\!]\,\rho])l)\\ \text{for some/any } l \notin \mathit{supp}(\lambda l'.k(S[l' \mapsto [\![\Delta; \Xi; \Gamma \vdash V : \tau]\!]\,\rho])l')\end{array}$$

The typing rule for $\mathsf{ref} V$ requires that τ is closed. The definition is much as in (7). The use of FM-cpo's ensures that it is a good definition. As in (7), we use the monad T to combine state with continuations to get a good control over what the new location has to be fresh for.

We only include a couple of additional cases of the semantic definition, namely the one for unfold, for continuations, and for type abstraction and type application:

$$\begin{array}{l}[\![\Delta; \Xi; \Gamma \vdash \mathsf{unfold}\ V : T(\tau[\mu\alpha.\tau/\alpha])]\!]\,\rho = \lambda k.\lambda S.\\ \mathit{case}\ [\![\Delta; \Xi; \Gamma \vdash V : \mu\alpha.\tau]\!]\,\rho\ \mathit{of}\ i(\mathit{in}_\mu(d))\ \mathit{then}\ kSd;\ \mathit{else}\bot,\end{array}$$

$$\begin{array}{l}[\![\Delta; \Xi; \vdash K : (x : \tau)^\top]\!] = \lambda S.\lambda d.\\ \quad [\![\Delta; \Xi; x : \tau \vdash K : T\tau']\!]\{x \mapsto d\}(\lambda S'.(\lambda d'.\top)_\bot)_\bot S\end{array}$$

$$\begin{array}{l}[\![\Delta; \Xi; \Gamma \vdash \Lambda\alpha.M : \forall\alpha.\,T\tau]\!]\,\rho = \mathit{in}_\forall\lfloor[\![\Delta; \Xi, \alpha; \Gamma \vdash M : T\tau]\!]\,\rho\rfloor\\ [\![\Delta; \Xi; \Gamma \vdash V(\tau') : T\tau[\tau'/\alpha]]\!]\,\rho =\\ \mathit{case}\ [\![\Delta; \Xi; \Gamma \vdash V : \forall\alpha.T\tau]\!]\,\rho\ \mathit{of}\,\mathit{in}_\forall\lfloor d\rfloor\,\mathit{then}\ d;\ \mathit{else}\bot,\end{array}$$

Theorem 2 (Soundness and Adequacy)
If $\Delta; ; \vdash M : T\tau$, $\Delta; \vdash K : (x : \tau)^\top$, $\Sigma : \Delta$ *and* $S \in [\![\Sigma : \Delta]\!]$ *then* Σ, $\mathsf{let}\ x \Leftarrow M\ \mathsf{in}\ K \downarrow$ *iff*

$$[\![\Delta; ; \vdash M : T\tau]\!] * [\![\Delta; \vdash K : (x : \tau)^\top]\!]\,S = \top.$$

Soundness is proved by induction and to show adequacy one defines a formal approximation relation between the denotational and the operational semantics. The existence proof of the relation is non-trivial because of the recursive types, but follows from a fairly straightforward adaptation of Shinwell's existence proof in (52) (Shinwell shows adequacy for a language with recursive types, but without references and impredicative polymorphism).

Corollary 3
$[\![\Delta; \Xi; \Gamma \vdash G_1 : \gamma]\!] = [\![\Delta; \Xi; \Gamma \vdash G_2 : \gamma]\!]$ *implies* $\Delta; \Xi; \Gamma \vdash G_1 =_{ctx} G_2$.

4 A Parameterized Logical Relation

In this section we define a parameterized logical relation on V, which we can use to prove contextual equivalence. The relation is parameterized both on interpretations of the free type variables as needed for parametricity and on parameters that accomodate local reasoning about contextual equivalence in the presence of general references. The latter kind of parameters are introduced in the next subsection, they are essentially as in (13).

4.1 Accessibility maps, simple state relations and parameters

Intuitively, the parameters express properties of two related states by expressing requirements of disjoint areas of states. There is a “visible” area and a finite number of “hidden invariants.” In the logical relation, computations are related under a parameter if they have corresponding termination behaviour under the assumption that they are executed in states satisfying the properties expressed by the parameter.

Definition 4
A function $A : S \to P_{fin}(Loc)$ from S to the set of finite subsets of Loc is an accessibility map *if*

1. *A is continuous, and*
2. *$\forall S_1, S_2 \in S.\ (\forall l \in A(S_1).\ S_1 l = S_2 l) \Rightarrow A(S_1) = A(S_2)$.*

We let $A_\emptyset$ denote the accessibility map defined by $\forall S. A_\emptyset(S) = \emptyset$, and we let $A_{\{l_1,\ldots,l_k\}}$ denote the accessibility map defined by $\forall S. A_{\{l_1,\ldots,l_k\}}(S) = \{l_1, \ldots, l_k\}$.

Definition 5
A simple state relation *P is a triple $(\hat{p}, A_{p1}, A_{p2})$ satisfying that A_{p1} and A_{p2} are accessibility maps and $\hat{p}$ is an admissible relation on S satisfying, for all states $S_1, S_2, S'_1, S'_2 \in S$,*

$$\begin{aligned}&\big(\forall l_1 \in A_{p1}(S_1). S_1 l_1 = S'_1 l_1 \ \wedge\ \forall l_2 \in A_{p2}(S_2). S_2 l_2 = S'_2 l_2\big)\\ &\qquad \Rightarrow \big((S_1, S_2) \in \hat{p} \Leftrightarrow (S'_1, S'_2) \in \hat{p}\big).\end{aligned}$$

Note that a simple state relation is essentially a relation on states for which it can be decided whether a pair of states belong to the relation only on the basis of some parts of the states, defined by a pair of accessibility maps.

We denote the “always true” simple state relation $(S \times S, A_\emptyset, A_\emptyset)$ by T.

We now define the notion of a local parameter, which we will later use to express hidden invariants of two related states. Intuitively, a local parameter has its own private areas of the states. These areas are used for testing conditions and for storing related values. The testing condition is a disjunction of simple state relations, where to each disjunct there is an associated list of pairs of locations from the two related states (it is possible that the location list is empty). At least one condition must be satisfied and the corresponding list of locations hold related values.

Definition 6
A local parameter is a finite non-empty set of pairs
$r = \{(P_1, LL_1), .., (P_m, LL_m)\}$, where each P_i is a simple state relation $P_i = (\hat{p}_i, A_{pi1}, A_{pi2})$ and each LL_i is a finite, possibly empty, set of location pairs and closed value types
$LL_i = \{\ (l_{i11}, l_{i12}, \tau_{i1}), \ldots, (l_{in_i1}, l_{in_i2}, \tau_{n_i})\ \}$, ($n_i \geq 0$).

We often write a local parameter as $r = ((P_1, LL_1) \veebar \ldots \veebar (P_m, LL_m))$. For a location list LL, we write L_1 resp. L_2 for the set of locations that occur as first resp. second components in the location list LL. For a local parameter r, there are associated accessibility maps A_{r1} and A_{r2} given by $\forall S.\ A_{r1}(S) = \bigcup_i A_{pi1}(S) \cup L_1$ and $\forall S.\ A_{r2}(S) = \bigcup_i A_{pi2}(S) \cup L_2$.

We denote the “always true” local parameter $\{(T, \emptyset)\}$ also simply by T. It has the associated accessibility maps $A_\emptyset, A_\emptyset$.

We have included the LL-list to be used for storing related values which may later be updated by exported updating functions. The updated values may require more invariants to hold for the stores in order to have equivalent behaviour. This interpretation of the local parameter is expressed in the definition of our invariant family of relations below.

Definition 7
A basic parameter *is a pair* Δr*, where* Δ *is a store type, and* $r = \{r_1, .., r_n\}$ *a finite set of local parameters such that* $T \in r$*.*

bPar *denotes the set of basic parameters.*

For a basic parameter Δr we associate accessibility maps A_{r1} and A_{r2}, given by $\forall S.\ A_{r1}(S) = \bigcup A_{r_i 1}(S)$ and $\forall S.\ A_{r2}(S) = \bigcup A_{r_i 2}(S)$.

For each store type Δ we have a special basic parameter $\Delta\{T\}$.

Definition 8
For basic parameters $\Delta' r'$ *and* Δr *define* $(\Delta' r') \rhd (\Delta r) \overset{def}{\Longleftrightarrow} \Delta' \supseteq \Delta \ and\ r' \supseteq r.$

The ordering relation $\rhd$ is reflexive, transitive and antisymmetric. For all basic parameters Δr it holds that there are only finitely many basic parameters $\Delta_0 r_0$ such that $\Delta r \rhd \Delta_0 r_0$. For convenience we sometimes write $\Delta r \lhd \Delta' r'$ for $\Delta' r' \rhd \Delta r$.

4.2 Parameterized relations and contextual equivalence

For $D \in FMcpo_\perp$ we define the set of parameters as:

$$\mathrm{Par}(D) = \{\Delta r(d_1, d_2) \mid d_1, d_2 \in D, \Delta r \textit{ basic parameter}\}$$

$\mathbf{AdmRel}(D)$ denotes the binary admissible relations on $D \in \mathrm{FMcpo}_\perp$.

$\mathbf{ParAdmRel}(D)$:
We define a set of parameterized admissible relations on $D \in \mathrm{FMcpo}_\perp$ denoted $\mathrm{ParAdmRel}(D)$. Elements are functions $R : \mathrm{bPar} \to D^2 \to \mathrm{AdmRel}(D)$ such that $\forall \Delta r.\ \forall d_1, d_2.\ R(\Delta r)(d_1, d_2)$ is an admissible relation on D. We use the notation $(d_1', d_2', d_1, d_2) \in R(\Delta r)$ when $(d_1', d_2') \in R(\Delta r)(d_1, d_2)$.

$\mathbf{bParAdmRel}(D)$:
We define a set of parameterized admissible relations on $D \in \mathrm{FMcpo}_\perp$ denoted $\mathrm{bParAdmRel}(D)$. Elements are functions $R : \mathrm{bPar} \to D^2 \to \mathrm{AdmRel}(D)$ satisfying the properties below. For notational convenience we let
$\mathrm{bParAdmRel} = \mathrm{bParAdmRel}(F(V, V))$.

admissibility $\forall \Delta r.\ \forall d_1, d_2.$
$R(\Delta r)(d_1, d_2)$ is an admissible relation on D.

parameter weakening $\forall \Delta r.\ \forall \Delta' r'.\ \forall d_1', d_2', d_1, d_2.$

$$(d_1', d_2', d_1, d_2) \in R(\Delta r) \ \wedge\ \Delta' r' \rhd \Delta r \ \Rightarrow\ (d_1', d_2', d_1, d_2) \in R(\Delta' r')$$

approximation $\forall \Delta r.\ \forall d_1', d_2', d_1, d_2.$

$$(d_1', d_2', d_1, d_2) \in R(\Delta r) \ \Rightarrow\ (d_1' \sqsubseteq d_1 \wedge d_2' \sqsubseteq d_2)$$

definedness $\forall \Delta r.\ \forall d_1', d_2', d_1, d_2.$

$$(d_1', d_2', d_1, d_2) \in R(\Delta r) \ \Rightarrow\ (d_1' = d_2' = \perp) \vee (d_1 \neq \perp \wedge d_2 \neq \perp)$$

downwards closure $\forall \Delta r.\ \forall d_1'', d_2'', d_1', d_2', d_1, d_2.$

$$d_1'' \sqsubseteq d_1' \wedge d_2'' \sqsubseteq d_2' \wedge (d_1', d_2', d_1, d_2) \in R(\Delta r) \ \Rightarrow\ (d_1'', d_2'', d_1, d_2) \in R(\Delta r)$$

Let $D \in \mathrm{FMcpo}_\perp$.
Let $\mathcal{H}_D : \mathit{ValueType} \to \Pi\{\alpha_1, \ldots, \alpha_n\} \in \mathcal{P}_{fin}(\mathit{TypeVar}) \to$
$(\mathrm{bParAdmRel}(F(V, V)))^n \to \mathrm{ParAdmRel}(D)$.

We define some properties $\mathcal{H}_D$ may have (it is admissible by definition):

Admissibility $\forall(\tau).\forall(\Xi\vec{R}).\forall\Delta r.\forall d_1, d_2.$

$$(\perp, \perp, d_1, d_2) \in \mathcal{H}_D(\tau)(\Xi\vec{R})(\Delta r)\ and$$

$$\begin{aligned}&\forall i.\ (d_1^i \sqsubseteq d_1^{i+1}) \wedge (d_2^i \sqsubseteq d_2^{i+1}) \wedge (d_1^i, d_2^i, d_1, d_2) \in \mathcal{H}_D(\tau)(\Xi\vec{R})(\Delta r) \implies \\ &(\bigsqcup d_1^i, \bigsqcup d_2^i, d_1, d_2) \in \mathcal{H}_D(\tau)(\Xi\vec{R})(\Delta r)\end{aligned}$$

Approximation $\forall(\tau).\forall(\Xi\vec{R}).\forall\Delta r.\forall d_1', d_2', d_1, d_2.$

$$(d_1', d_2', d_1, d_2) \in \mathcal{H}_D(\tau)(\Xi\vec{R})(\Delta r) \implies (d_1' \sqsubseteq d_1 \wedge d_2' \sqsubseteq d_2)$$

Downwards closure $\forall(\tau).\forall(\Xi\vec{R}).\forall\Delta r.\forall d_1'', d_2'', d_1', d_2', d_1, d_2.$

$$\begin{aligned}&(d_1'' \sqsubseteq d_1' \wedge d_2'' \sqsubseteq d_2' \wedge (d_1', d_2', d_1, d_2) \in \mathcal{H}_D(\tau)(\Xi\vec{R})(\Delta r) \implies \\ &(d_1'', d_2'', d_1, d_2) \in \mathcal{H}_D(\tau)(\Xi\vec{R})(\Delta r)\end{aligned}$$

Definedness $\forall(\tau).\forall(\Xi\vec{R}).\forall\Delta r.\forall d_1', d_2', d_1, d_2.$

$$(d_1', d_2', d_1, d_2) \in \mathcal{H}_D(\tau)(\Xi\vec{R})(\Delta r) \implies (d_1' = d_2' = \perp\ \vee\ (d_1 \neq \perp \wedge d_2 \neq \perp))$$

Ξ-strengthening $\forall\Xi \vdash \tau.\ \forall(\Xi\vec{R}).\ \forall(\Xi'\vec{R}') \supseteq (\Xi\vec{R}).\ \forall(\Delta r).$

$$(d_1', d_2', d_1, d_2) \in \mathcal{H}_D(\tau)(\Xi'\vec{R}')(\Delta r) \Rightarrow (d_1', d_2', d_1, d_2) \in \mathcal{H}_D(\tau)(\Xi\vec{R})(\Delta r)$$

Ξ-weakening $\forall(\Xi\vec{R}).\ \forall(\Xi'\vec{R}') \supseteq (\Xi\vec{R}).\ \forall(\Delta r).$

$$(d_1', d_2', d_1, d_2) \in \mathcal{H}_D(\tau)(\Xi\vec{R})(\Delta r) \Rightarrow (d_1', d_2', d_1, d_2) \in \mathcal{H}_D(\tau)(\Xi'\vec{R}')(\Delta r)$$

Parameter weakening $\forall(\tau).\forall(\Xi\vec{R}).\forall\Delta r(d_1, d_2).\ \forall\Delta' r' \rhd \Delta r.$

$$\mathcal{H}(\tau)(\Xi\vec{R})(\Delta r(d_1, d_2)) \subseteq \mathcal{H}(\tau)(\Xi\vec{R})(\Delta' r'(d_1, d_2))$$

We define:

Definition 9

$$\begin{aligned}\mathcal{L}_V \subseteq\ &\mathit{ValueType} \to \Pi\{\alpha_1, \ldots, \alpha_n\} \in \mathcal{P}_{fin}(\mathit{TypeVar}) \to \\ &(\mathrm{bParAdmRel})^n \to \mathrm{ParAdmRel}(V)\big)\end{aligned}$$

$$\begin{aligned}\mathcal{L}_{F(V,V)} \subseteq\ &\mathit{ValueType} \to \Pi\{\alpha_1, \ldots, \alpha_n\} \in \mathcal{P}_{fin}(\mathit{TypeVar}) \to \\ &(\mathrm{bParAdmRel})^n \to \mathrm{ParAdmRel}(F(V, V))\big)\end{aligned}$$

For $D \in \{V, F(V, V)\}$ by definition $\mathcal{L}_D$ is the subset with the properties admissibility, downwards closure, Ξ-strengthening, Ξ-weakening *and* parameter weakening , *and with the pointwise ordering. If $\mathcal{R}, \mathcal{S} \in \mathcal{L}_D$, then*

$$\mathcal{R} \leq \mathcal{S} \iff$$
$$\big(\forall(\tau).\ \forall(\Xi\vec{R}).\ \forall\Delta r.\ \forall v_1', v_2', v_1, v_2 \in D.$$
$$(v_1', v_2', v_1, v_2) \in \mathcal{R}(\tau)(\Xi\vec{R})(\Delta r) \Rightarrow (v_1', v_2', v_1, v_2) \in \mathcal{S}(\tau)(\Xi\vec{R})(\Delta r)\big)$$

Comment: The relation we are aiming at will also by definition have the properties approximation and (for values) definedness.

Let $\mathbb{S} \subseteq \mathcal{L}_D$. Then $\bigcap \mathbb{S} \in \mathit{ValueType} \to \Pi\{\alpha_1, \ldots, \alpha_n\} \in \mathcal{P}_{fin}(\mathit{TypeVar}) \to (\text{bParAdmRel})^n \to$ bPar $\to D^2 \to \mathit{Rel}(D)$ such that
$(v_1', v_2') \in \bigcap \mathbb{S}(\tau)(\Xi\vec{R})(\Delta r(v_1, v_2)) \iff (\forall \mathcal{S} \in \mathbb{S}.\ (v_1', v_2') \in \mathcal{S}(\tau)(\Xi\vec{R})(\Delta r(v_1, v_2))$.

Lemma 10
$\mathcal{L}_D$ is a complete lattice. If $\mathbb{S}$ is a set of relations in $\mathcal{L}_D$, then $\bigcap \mathbb{S}$ is a relation in $\mathcal{L}_D$, $D \in \{V, F(V, V)\}$.

We need to prove, that $\bigcap \mathbb{S}$ has the properties admissibility, downwards closure, Ξ-strengthening, Ξ-weakening and parameter weakening.

Admissibility. Any chain in $(\bigcap \mathbb{S})(\tau)(\Xi\vec{R})(\Delta r(v_1, v_2))$ is a chain in each $\mathcal{S}(\tau)(\Xi\vec{R})(\Delta r(v_1, v_2))$ where $\mathcal{S} \in \bigcap \mathbb{S}$ and each of these is admissibile.

Downwards closure. Assume $v_1'' \sqsubseteq v_1' \wedge v_2'' \sqsubseteq v_2' \wedge (v_1', v_2') \in (\bigcap \mathbb{S})(\tau)(\Xi\vec{R})(\Delta r(v_1, v_2))$. Then $\forall \mathcal{S} \in \mathbb{S}.\ (v_1', v_2') \in \mathcal{S}(\tau)(\Xi\vec{R})\ (\Delta r(v_1, v_2))$. Each $\mathcal{S}$ is downwards closed, so $\forall \mathcal{S} \in (\bigcap \mathbb{S})$. $(v_1'', v_2'') \in \mathcal{S}(\tau)(\Xi\vec{R})\ (\Delta r(v_1, v_2))$. It follows that $(v_1'', v_2'') \in (\bigcap \mathbb{S})(\tau)(\Xi\vec{R})(\Delta r(v_1, v_2))$.

Ξ-strengthening. Assume $\Xi \vdash \tau$ and $(\Xi'\vec{R}') \supseteq (\Xi\vec{R})$ and $(d_1', d_2', d_1, d_2) \in (\bigcap \mathbb{S})(\tau)(\Xi'\vec{R}')(\Delta r)$. Then $\forall S \in \mathbb{S}.\ (d_1', d_2', d_1, d_2) \in S(\tau)(\Xi'\vec{R}')(\Delta r)$ and since each S has Ξ-strengthening then $\forall S \in \mathbb{S}.\ (d_1', d_2', d_1, d_2) \in S(\tau)(\Xi\vec{R})(\Delta r)$. So $(d_1', d_2', d_1, d_2) \in (\bigcap \mathbb{S})(\tau)(\Xi\vec{R})(\Delta r)$.

Ξ-weakening. Assume $(\Xi'\vec{R}') \supseteq (\Xi\vec{R})$ and $(d_1', d_2', d_1, d_2) \in (\bigcap \mathbb{S})(\tau)(\Xi\vec{R})(\Delta r)$. Then $\forall S \in \mathbb{S}.\ (d_1', d_2', d_1, d_2) \in S(\tau)(\Xi\vec{R})(\Delta r)$ and since each S has Ξ-weakening then $\forall S \in \mathbb{S}.\ (d_1', d_2', d_1, d_2) \in S(\tau)(\Xi'\vec{R}')(\Delta r)$. So $(d_1', d_2', d_1, d_2) \in (\bigcap \mathbb{S})(\tau)(\Xi'\vec{R}')(\Delta r)$.

Parameter weakening. Assume $(v_1', v_2') \in (\bigcap \mathbb{S})(\tau)(\Xi\vec{R})(\Delta r(v_1, v_2))$ so $(\forall \mathcal{S} \in \mathbb{S}.\ (v_1', v_2') \in \mathcal{S}(\tau)(\Xi\vec{R})\ (\Delta r(v_1, v_2))$. Let $\Delta' r' \rhd \Delta r$. Since each $\mathcal{S}$ is parameter weakened, then $(\forall \mathcal{S} \in \mathbb{S}.\ (v_1', v_2') \in \mathcal{S}(\tau)(\Xi\vec{R})(\Delta' r'(v_1, v_2))$. Hence $(v_1', v_2') \in \bigcap \mathbb{S}(\tau)(\Xi\vec{R})(\Delta' r'(v_1, v_2))$.

We now define monotone functions

$$\begin{aligned}
&\Psi_S : \mathcal{L}_V^{\text{op}} \times \mathcal{L}_V \to \text{ParAdmRel}(S)\\
&\Psi_K : \mathcal{L}_V^{\text{op}} \times \mathcal{L}_V \to \mathit{ContinuationType} \to \Pi\{\alpha_1 \ldots \alpha_n\} \to\\
&\qquad \text{bParAdmRel}^n \to \text{ParAdmRel}(K\,V)\\
&\Psi_T : \mathcal{L}_V^{\text{op}} \times \mathcal{L}_V \to \mathit{ComputationType} \to \Pi\{\alpha_1 \ldots \alpha_n\}\\
&\qquad \to \text{bParAdmRel}^n \to \text{ParAdmRel}(T\,V)\\
&\Psi : \mathcal{L}_V^{\text{op}} \times \mathcal{L}_V \to \mathcal{L}_{F(V,V)}
\end{aligned}$$

The functions are defined in the order shown above. The function Ψ_S is defined without referring to any of the other functions. The definition of the function Ψ_K makes use of Ψ_S. The definition of the function Ψ_T makes use of Ψ_S and Ψ_K. The function $\Psi(\mathcal{R}, \mathcal{S}) \in \mathcal{L}_{F(V,V)}$ is defined *by case analysis* over the structure of its argument τ (we do *not* use induction over τ) and makes use of the definition of the function Ψ_T. Here we make use of the fact that $\mathcal{L}_V$ and $\mathcal{L}_{F(V,V)}$ are complete lattices of functions. Technically it is an important idea since it makes it easy for us to deal with nested recursive types.

The actual definitions of the above functions are given in Figure 1.

Theorem 11
$\Psi_S,\ \Psi_K,\ \Psi_T,\ \Psi$ are well-defined.

We will also show that $\forall \mathcal{R} \in \mathcal{L}^{op}, \mathcal{S} \in \mathcal{L}$ (with the natural definition of the involved properties)

> $\Psi_S(\mathcal{R}, \mathcal{S})$ has the properties *admissibility* and *downwards closure*. (But not parameter weakening).
>
> $\Psi_K(\mathcal{R}, \mathcal{S})$ has the properties *admissibility, downwards closure, Ξ-strengthening, Ξ-weakening* and *parameter weakening*.
>
> $\Psi_T(\mathcal{R}, \mathcal{S})$ has the properties *admissibility, downwards closure, Ξ-strengthening, Ξ-weakening* and *parameter weakening*.

Lemma 12
The generated relations are downwards closed.
$\forall \mathcal{R} \in \mathcal{L}_V^{\mathrm{op}}, \mathcal{S} \in \mathcal{L}_V.\ \forall \Xi.\ \forall \vec{R}.\ \forall \Delta r.$
$\forall s_1'', s_2'', s_1', s_2', s_1, s_2, k_1'', k_2'', k_1', k_2', k_1, k_2,$
$m_1'', m_2'', m_1', m_2', m_1, m_2, v_1'', v_2'', v_1', v_2', v_1, v_2.$

$S\colon$ $(s_1', s_2', s_1, s_2) \in \Psi_S(\mathcal{R}, \mathcal{S})(\Delta r) \wedge s_1'' \sqsubseteq s_1' \wedge s_2'' \sqsubseteq s_1' \Rightarrow$
$(s_1'', s_2'', s_1, s_2) \in \Psi_S(\mathcal{R}, \mathcal{S})(\Delta r)$

$KV\colon$ $(k_1', k_2', k_1, k_2) \in \Psi_K(\mathcal{R}, \mathcal{S})(\tau^\top)(\Xi\vec{R})(\Delta r) \wedge k_1'' \sqsubseteq k_1' \wedge k_2'' \sqsubseteq k_1' \Rightarrow$
$(k_1'', k_2'', k_1, k_2) \in \Psi_K(\mathcal{R}, \mathcal{S})(\tau^\top)(\Xi\vec{R})(\Delta r)$

$TV\colon$ $(m_1', m_2', m_1, m_2) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau)(\Xi\vec{R})(\Delta r) \wedge m_1'' \sqsubseteq m_1' \wedge m_2'' \sqsubseteq m_1' \Rightarrow$
$(m_1'', m_2'', m_1, m_2) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau)(\Xi\vec{R})(\Delta r)$

$V\colon$ $(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\tau)(\Xi\vec{R})(\Delta r) \wedge v_1'' \sqsubseteq v_1' \wedge v_2'' \sqsubseteq v_1' \Rightarrow$
$(v_1'', v_2'', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\tau)(\Xi\vec{R})(\Delta r)$

Proof:

- For Ψ_S the lemma follows from the property for $\mathcal{S} \in \mathcal{L}$. All requirements of disjointness and membership in simple state relations are only on $s_1,\ s_2$. Requirements of relatedness of stored values are in $\mathcal{S} \in \mathcal{L}$ and $\mathcal{S}$ is downwards closed.
- For Ψ_K and Ψ_T the lemma follows from weaker termination properties under applications.
- For Ψ the proof is by case analysis over the constructor in the argument τ
  α_j: follows from downwards closure of $R_j \in \mathrm{bParAdmRel}$. unit, int, $\tau\,\mathsf{ref}$: immidiate from the definition.
  $(\tau_1 + \tau_2)$,$(\tau_1 \times \tau_2)$,$(\mu\alpha.\tau)$: Follows from the property for $\mathcal{S} \in \mathcal{L}$.
  $\tau \to T\tau'$ and $\forall\alpha.T\tau$: follows downwards closure of $\mathcal{R}, \mathcal{S} \in \mathcal{L}$ and Ψ_T preserves this property.

■

Lemma 13
The generated relations are admissible
$\forall \mathcal{R} \in \mathcal{L}_V^{\mathrm{op}}, \mathcal{S} \in \mathcal{L}_V.\ \forall (\Xi\vec{R}).\ \forall \Delta r.$
$\forall S_1, S_2, k_1, k_2, m_1, m_2, v_1, v_2.$

$S\colon$ $(\bot, \bot, S_1, S_2) \in \Psi_S(\mathcal{R}, \mathcal{S})(\Delta r) \wedge$
$\big(\forall i \in \omega.\ S_1^i \sqsubseteq S_1^{i+1} \wedge S_2^i \sqsubseteq S_2^{i+1} \wedge (S_1^i, S_2^i, S_1, S_2) \in \Psi_S(\mathcal{R}, \mathcal{S})(\Delta r) \Rightarrow$
$(\bigsqcup S_1^i, \bigsqcup S_2^i, S_1, S_2) \in \Psi_S(\mathcal{R}, \mathcal{S})(\Delta r)$

$KV\colon$ $(\bot, \bot, k_1, k_2) \in \Psi_K(\mathcal{R}, \mathcal{S})(\tau)(\Xi\vec{R})(\Delta r) \wedge$
$\big(\forall i \in \omega.\ k_1^i \sqsubseteq k_1^{i+1} \wedge k_2^i \sqsubseteq k_2^{i+1} \wedge (k_1^i, k_2^i, k_1, k_2) \in \Psi_K(\mathcal{R}, \mathcal{S})(\tau^\top)(\Xi\vec{R})(\Delta r) \Rightarrow$
$(\bigsqcup k_1^i, \bigsqcup k_2^i, k_1, k_2) \in \Psi_S(\mathcal{R}, \mathcal{S})(\tau^\top)(\Xi\vec{R})(\Delta r)$

TV: $(\bot,\bot,m_1,m_2) \in \Psi_T(\mathcal{R},\mathcal{S})(T\tau)(\Xi\vec{R})(\Delta r) \wedge$
$\big(\forall i \in \omega.\ m_1^i \sqsubseteq m_1^{i+1} \wedge m_2^i \sqsubseteq m_2^{i+1} \wedge (m_1^i, m_2^i, m_1, m_2) \in \Psi_T(\mathcal{R},\mathcal{S})(T\tau)(\Xi\vec{R})(\Delta r) \Rightarrow$
$(\bigsqcup m_1^i, \bigsqcup m_2^i, m_1, m_2) \in \Psi_S(\mathcal{R},\mathcal{S})(T\tau)(\Xi\vec{R})(\Delta r)$

V: $(\bot,\bot,v_1,v_2) \in \Psi_T(\mathcal{R},\mathcal{S})(\tau)(\Xi\vec{R})(\Delta r) \wedge$
$\big(\forall i \in \omega.\ v_1^i \sqsubseteq v_1^{i+1} \wedge v_2^i \sqsubseteq v_2^{i+1} \wedge (v_1^i, v_2^i, v_1, v_2) \in \Psi(\mathcal{R},\mathcal{S})(\tau)(\Xi\vec{R})(\Delta r) \Rightarrow$
$(\bigsqcup v_1^i, \bigsqcup v_2^i, v_1, v_2) \in \Psi_S(\mathcal{R},\mathcal{S})(\tau)(\Xi\vec{R})(\Delta r)$

Proof: $(\bot,\bot)$ is always in the relations, this is immidiate from the definition. Proof for chains which are not constantly $\bot,\bot$:

1. Assume $\forall i \in \omega.\ (S_1^i, S_2^i, S_1, S_2) \in \Psi_S(\mathcal{R},\mathcal{S})(\Delta r)$, and this is a chain. Then $\forall i.\ S_1^i \sqsubseteq S_1$ and $\forall i.\ S_2^i \sqsubseteq S_2$. This implies $\bigsqcup S_1^i \sqsubseteq S_1$ and $\bigsqcup S_2^i \sqsubseteq S_2$. Requirements of disjointness of areas given by accessibility maps and membership in state relations are only on S_1, S_2, so they are still fulfilled, and the same LL's will be required to hold related values. By assumption for each such LL it holds that $\forall i \in \omega.\ \forall (l_1, l_2, - \vdash \tau) \in LL.\ (S_1^i l_1, S_2^i l_2, S_1 l_1, S_2 l_2) \in \mathcal{S}(\tau)()(\Delta r)$. Since $\mathcal{S}$ is admissible then also $(\bigsqcup_i (S_1^i l_1), \bigsqcup_i (S_2^i l_2),\ S_1 l_1, S_2 l_2) = ((\bigsqcup_i S_1^i) l_1, (\bigsqcup_i S_2^i) l_2, S_1 l_1, S_2 l_2) \in \mathcal{S}(\tau)()(\Delta r)$. Similarly for $(l \mapsto - \vdash \tau) \in \Delta$ we see that $((\bigsqcup_i S_1^i) l, (\bigsqcup_i S_2^i) l, S_1 l, S_2 l) \in \mathcal{S}(\tau)()(\Delta r)$. We conclude that $(\bigsqcup_i S_1^i, \bigsqcup_i S_2^i, S_1, S_2) \in \Psi_S(\mathcal{R},\mathcal{S})(\Delta r)$.

2. Assume a chain $(k_1^i, k_2^i, k_1, k_2 \in \Psi_K(\mathcal{R},\mathcal{S})(\tau^\top)(\Xi\vec{R})(\Delta r)$. Then $\forall i.\ k_1^i \sqsubseteq k_1 \wedge k_2^i \sqsubseteq k_2$, so $\bigsqcup k_1^i \sqsubseteq k_1 \wedge \bigsqcup k_2^i \sqsubseteq k_2$. Let $\Delta' r' \rhd \Delta r$, $(S_1', S_2', S_1, S_2) \in \Psi_S(\mathcal{S},\mathcal{R})(\Delta' r')$ and $(v_1', v_2', v_1, v_2) \in \mathcal{R}(\tau)(\Xi\vec{R})(\Delta' r')$. $k_1^i S_1' v_1'$ is a chain in O. If the chain is constantly $\bot$ then $\bigsqcup k_1^i S_1' v_1' = \bot$ and the implication $(\bigsqcup k_1^i S_1' v_1' = \top \Rightarrow k_2 S_2 v_2 = \top)$ holds trivially. Else $\exists j \in \omega.\ \forall i \geq j.\ k^i S_1' v_1' = \top$. Then also $\bigsqcup k^i S_1' v_1' = \top \wedge k_2 S_2 v_2 = \top$. The other direction is similar. We conclude $(\bigsqcup k_1^i, \bigsqcup k_2^i, k_1, k_2) \in \Psi_K(\mathcal{R},\mathcal{S})(\tau^\top)(\Xi\vec{R})(\Delta r)$.

3. The proof is similar to the previous for 2), so we omit it.

4. The proof is by case analysis of the constructor of the argument τ.
   - Assume a chain $(v_1^i, v_2^i, v_1, v_2) \in \Psi(\mathcal{R},\mathcal{S})(\alpha_j)(\Xi\vec{R})(\Delta r)$. Then (in the non bottom cases) $\alpha_j \in \Xi$ and $\forall i.\ v_1^i \sqsubseteq v_1 \wedge v_2^i \sqsubseteq v_2$. Then also $\bigsqcup v_1^i \sqsubseteq v_1$ and $\bigsqcup v_2^i \sqsubseteq v_2$. Also $\forall i \in \omega.\ (v_1^i, v_2^i, v_1, v_2) \in R_j(\Delta r)$ and since R_j is admissible then also $(\bigsqcup_i v_1^i, \bigsqcup_i v_2^i, v_1, v_2) \in R_j(\Delta r)$. So $(\bigsqcup_i v_1^i, \bigsqcup_i v_2^i, v_1, v_2) \in \Psi(\mathcal{R},\mathcal{S})(\alpha_j)(\Xi\vec{R})(\Delta r)$.
   - Chains with type-argument $unit, int$ or $\tau\mathsf{ref}$ will be constant from some point onwards.
   - For chains with type argument $\tau_1 + \tau_2,\ \tau_1 \times \tau_2$ and $\mu\alpha.\ \tau$ admissibility follows from admissibility of relations in $\mathcal{L}$ and $\mathcal{S} \in \mathcal{L}$.
   - Assume a chain $(v_1^i, v_2^i, v_1, v_2) \in \Psi(\mathcal{R},\mathcal{S})(\tau \to T\tau')(\Xi\vec{R})\ (\Delta r)$. Then $\forall i.\ (v_1^i \sqsubseteq v_1 \wedge v_2^i \sqsubseteq v_2 \wedge \exists d_1^i, d_2^i, d_1, d_2 : V \multimap TV.\ ((v_1^i = \bot \wedge d_1^i = \bot) \vee v_1^i = in_\to\lfloor d_1^i \rfloor) \wedge ((v_2^i = \bot \wedge d_2^i = \bot) \vee v_2^i = in_\to\lfloor d_2^i \rfloor) \wedge v_1 = in_\to\lfloor d_1 \rfloor \wedge v_2 = in_\to\lfloor d_2 \rfloor \wedge \forall \Delta' r' \rhd \Delta r.\ \forall v_{11}', v_{22}', v_{11}, v_{22} : V.\ (v_{11}', v_{22}', v_{11}, v_{22}) \in \mathcal{R}(\tau)(\Xi\vec{R})(\Delta' r') \Rightarrow (d_1^i v_{11}',\ d_2^i v_{22}', d_1 v_{11}, d_2 v_{22}) \in (\Psi_T(\mathcal{R},\mathcal{S})$ $(T\tau')(\Xi\vec{R})(\Delta' r')$. As $\bigsqcup(in_\to\lfloor d^i \rfloor) = in_\to\lfloor \bigsqcup d^i \rfloor$ and $\bigsqcup(d^i v) = (\bigsqcup d^i)v$ and by 3) then $\forall \Delta' r' \rhd \Delta r.\ (v_{11}', v_{22}', v_{11}, v_{22}) \in \mathcal{R}(\tau)(\Xi\vec{R})(\Delta' r') \Rightarrow (\bigsqcup d_1^i v_{11}', \bigsqcup d_2^i v_{22}', d_1 v_{11}, d_2 v_{22}) \in (\Psi_T(\mathcal{R},\mathcal{S})(T\tau')\ (\Xi\vec{R})(\Delta' r')$.
     So $(\bigsqcup v_1^i, \bigsqcup v_2^i, v_1, v_2) \in \Psi(\mathcal{R},\mathcal{S})(\tau \to T\tau')(\Xi\vec{R})\ (\Delta r)$.
   - Assume a chain $(v_1^i, v_2^i, v_1, v_2) \in \Psi(\mathcal{R},\mathcal{S})(\forall\alpha.T\tau)(\Xi\vec{R})(\Delta r)$. Then $\forall i.\ v_1^i \sqsubseteq v_1$ and $\forall i.\ v_2^i \sqsubseteq v_2$. This implies $\bigsqcup v_1^i \sqsubseteq v_1$ and $\bigsqcup v_2^i \sqsubseteq v_2$. It holds that $\forall i.\ \exists d_1^i, d_2^i : TV.\ (v_1^i = d_1^i = \bot \vee v_1^i = in_\forall\lfloor d_1^i \rfloor) \wedge (v_2^i = d_2^i = \bot \vee v_2^i = in_\forall\lfloor d_2^i \rfloor) \wedge \exists d_1, d_2 : TV.\ v_1 = in_\forall\lfloor d_1 \rfloor \wedge v_2 = in_\forall\lfloor d_2 \rfloor \wedge \forall i.\ \forall R_\alpha :$ bParAdmRel. $(d_1^i, d_2^i, d_1, d_2) \in \Psi_T(\mathcal{R},\mathcal{S})(T\tau)(\Xi\alpha\ \vec{R}R_\alpha)(\Delta r)$. This is a chain, and then by 3) also $\forall R_\alpha :$ bParAdmRel. $(\bigsqcup d_1^i, \bigsqcup d_2^i, d_1, d_2) \in \Psi_T(\mathcal{R},\mathcal{S})(\tau)(\Xi\alpha\ \vec{R}\ R_\alpha)(\Delta r)$. As $in_\forall\lfloor \bigsqcup d_1^i \rfloor = \bigsqcup in_\forall\lfloor d_1^i \rfloor$ and $\bigsqcup(\bot - chain) = \bot$ and possibly using downwards closure we have $(\bigsqcup v_1^i, \bigsqcup v_2^i v_1, v_2) \in \Psi(\mathcal{R},\mathcal{S})(\forall\alpha.T\tau)(\Xi\vec{R})(\Delta r)$.

■

Lemma 14
The generated relations have parameter weakening.
$\forall \mathcal{R} \in \mathcal{L}_V^{\text{op}}, \mathcal{S} \in \mathcal{L}_V.$
$\forall(\Xi\vec{R}).\ \forall\Delta r.\ \forall\Delta' r' \rhd \Delta r.$
$\forall k_1', k_2', k_1, k_2, m_1', m_2', m_1, m_2, v_1', v_2', v_1, v_2.$

KV: $(k_1', k_2', k_1, k_2) \in \Psi_K(\mathcal{R}, \mathcal{S})(\tau^\top)(\Xi\vec{R})(\Delta r) \Rightarrow$
$(k_1', k_2', k_1, k_2) \in \Psi_K(\mathcal{R}, \mathcal{S})(\tau^\top)(\Xi\vec{R})(\Delta' r')$

TV: $(m_1', m_2', m_1, m_2) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau)(\Xi\vec{R})(\Delta r) \Rightarrow$
$(m_1', m_2', m_1, m_2) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau)(\Xi\vec{R})(\Delta' r')$

V: $(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\tau)(\Xi\vec{R})(\Delta r) \Rightarrow$
$(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\tau)(\Xi\vec{R})(\Delta' r')$

Proof: For KV and TV the lemma follows from transitivity $\Delta'' r'' \rhd \Delta' r'$ and $\Delta' r' \rhd \Delta r \Rightarrow \Delta'' r'' \rhd \Delta r$.

For V the proof is by case analysis over the constructor of the argument τ

- α_j: Assume $(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\alpha_j)(\Xi\vec{R})(\Delta r)$ and $v_1' \neq \bot \vee v_2' \neq \bot$. Then $\alpha_j \in \Xi$ and $(v_1', v_2', v_1, v_2) \in R_j(\Delta r)$. Since $R_j \in \text{bParAdmRel}$ is parameter weakened then also $(v_1', v_2', v_1, v_2) \in R_j(\Delta' r')$. So $(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\alpha_j)(\Xi\vec{R})(\Delta' r')$
- $unit$, int: immidiate from the definition
- $\tau\,\mathsf{ref}$: Follows from $\Delta' r' \rhd \Delta r \Rightarrow \Delta' \supseteq \Delta$.
- $\tau_1 + \tau_2,\ \tau_1 \times \tau_2,\ \mu\alpha.\tau$: Follows from parameter weakening for relations in $\mathcal{L}$.
- $\tau \to T\tau'$: Follows from transitivity $\Delta'' r'' \rhd \Delta' r' \wedge \Delta' r' \rhd \Delta r \Rightarrow \Delta'' r'' \rhd \Delta r$.
- $\forall\alpha.T\tau$: Follows from the lemma for TV. Assume $(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\forall\alpha.T\tau)(\Xi\vec{R})(\Delta r)$. This implies $(v_1 = v_2 = \bot)\ \vee \Xi \vdash \forall\alpha.T\tau \wedge \alpha \notin \Xi \wedge (v_1' \sqsubseteq v_1 \wedge v_2' \sqsubseteq v_2 \wedge \exists d_1', d_2' : TV.\ ((v_1' = \bot \wedge d_1' = \bot) \vee v_1' = in_\forall\lfloor d_1' \rfloor) \wedge ((v_2' = \bot \wedge d_2' = \bot) \vee v_2' = in_\forall\lfloor d_2' \rfloor) \wedge \exists d_1, d_2 : TV.\ v_1 = in_\forall\lfloor d_1 \rfloor \wedge v_2 = in_\forall\lfloor d_2 \rfloor \wedge \forall R_\alpha : \text{bParAdmRel}.\ (d_1', d_2', d_1, d_2) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau)(\Xi\alpha\ \vec{R}R_\alpha)(\Delta r)$. The $(\bot, \bot)$ case is immidiate. Else by the lemma for TV it holds that $(d_1', d_2', d_1, d_2) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau)$ $(\Xi\alpha\ \vec{R}R_\alpha)(\Delta' r')$. So $(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\forall\alpha.T\tau)(\Xi\vec{R})(\Delta' r')$.

■

Lemma 15
The generated relations have Ξ-weakening and Ξ-strengthening.
$\forall \mathcal{R} \in \mathcal{L}_V^{\text{op}}, \mathcal{S} \in \mathcal{L}_V.$
$\forall\tau.\ \forall(\Xi\vec{R}).\ \forall(\Xi'\vec{R}') \supseteq (\Xi\vec{R}).\ \forall\Delta r.$
$\forall k_1', k_2', k_1, k_2, m_1', m_2' m_1, m_2, v_1', v_2', v_1, v_2.$

KV: *1)* $(k_1', k_2', k_1, k_2) \in \Psi_K(\mathcal{R}, \mathcal{S})(\tau^\top)(\Xi\vec{R})(\Delta r) \Rightarrow$
$(k_1', k_2', k_1, k_2) \in \Psi_K(\mathcal{R}, \mathcal{S})(\tau^\top)(\Xi'\vec{R}')(\Delta r)$
2) $\Xi \vdash \tau \wedge (k_1', k_2', k_1, k_2) \in \Psi_K(\mathcal{R}, \mathcal{S})(\tau^\top)(\Xi'\vec{R}')(\Delta r) \Rightarrow$
$(k_1', k_2', k_1, k_2) \in \Psi_K(\mathcal{R}, \mathcal{S})(\tau^\top)(\Xi\vec{R})(\Delta r)$

TV: 1) $(m_1', m_2', m_1, m_2) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau)(\Xi\vec{R})(\Delta r) \Rightarrow$
$(m_1', m_2', m_1, m_2) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau)(\Xi'\vec{R}')(\Delta r)$
2) $\Xi \vdash \tau \wedge (m_1', m_2', m_1, m_2) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau)(\Xi'\vec{R}')(\Delta r) \Rightarrow$
$(m_1', m_2', m_1, m_2) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau)(\Xi\vec{R})(\Delta r)$

V: 1) $(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\tau)(\Xi\vec{R})(\Delta r) \Rightarrow$
$(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\tau)(\Xi'\vec{R}')(\Delta r)$
2) $\Xi \vdash \tau \wedge (v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\tau)(\Xi'\vec{R}')(\Delta r) \Rightarrow$
$(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\tau)(\Xi\vec{R})(\Delta r)$

Proof:

KV: 1_K) Assume $(k_1', k_2', k_1, k_2) \in \Psi_K(\mathcal{R}, \mathcal{S})(\tau^\top)(\Xi\vec{R})(\Delta r)$ and $(\Xi'\vec{R}') \supseteq (\Xi\vec{R})$. To show $(k_1', k_2', k_1, k_2) \in \Psi_K(\mathcal{R}, \mathcal{S})(\tau^\top)$ $(\Xi'\vec{R}')(\Delta r)$. Let $\Delta^0 r^0 \triangleright \Delta r$ and let $(S_1', S_2', S_1, S_2) \in \Psi_S(\mathcal{S}, \mathcal{R})(\Delta^0 r^0)$ and $(v_1', v_2', v_1, v_2) \in \mathcal{R}(\tau)(\Xi'\vec{R}')(\Delta^0 r^0)$. By the definition of the action of F the assumption implies that either $k_1' = k_2' = \bot$ or $\Xi \vdash \tau$. The first case is trivial. In the second case, since $\mathcal{R} \in \mathcal{L}_V$ it has Ξ-strengthening, so $(v_1', v_2', v_1, v_2) \in \mathcal{R}(\tau)(\Xi\vec{R})(\Delta^0 r^0)$. By the assumptions on the k's we then get the required termination approximation.

2_K) Assume $(k_1', k_2', k_1, k_2) \in \Psi_K(\mathcal{R}, \mathcal{S})(\tau^\top)(\Xi'\vec{R}')(\Delta r) \wedge \Xi \vdash \tau$ and $(\Xi'\vec{R}') \supseteq (\Xi\vec{R})$. To show $(k_1', k_2', k_1, k_2) \in \Psi_K(\mathcal{R}, \mathcal{S})(\tau^\top)(\Xi\vec{R})(\Delta r)$. Let $\Delta^0 r^0 \triangleright \Delta r$ and let $(S_1', S_2', S_1, S_2) \in \Psi_S(\mathcal{S}, \mathcal{R})(\Delta^0 r^0)$ and $(v_1', v_2', v_1, v_2) \in \mathcal{R}(\tau)(\Xi\vec{R})$ $(\Delta^0 r^0)$. The case $k_1' = k_2' = \bot$ is trivial. Else since $\mathcal{R} \in \mathcal{L}_V$ it has Ξ-weakening, so $(v_1', v_2', v_1, v_2) \in \mathcal{R}(\tau)(\Xi'\vec{R}')(\Delta^0 r^0)$. By the assumptions on the k's we then get the required termination approximation.

TV: 1_M) Assume $(m_1', m_2', m_1, m_2) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau)(\Xi\vec{R})(\Delta r)$ and $(\Xi'\vec{R}') \supseteq (\Xi\vec{R})$. To show $(m_1', m_2', m_1, m_2) \in \Psi_T(\mathcal{R}, \mathcal{S})$ $(T\tau)(\Xi'\vec{R}')(\Delta r)$. Let $\Delta^0 r^0 \triangleright \Delta r$ and let $(S_1', S_2', S_1, S_2) \in \Psi_S(\mathcal{S}, \mathcal{R})$ $(\Delta^0 r^0)$ and $(k_1', k_2', k_1, k_2) \in \Psi_K(\mathcal{S}, \mathcal{R})(\tau^\top)(\Xi'\vec{R}')$ $(\Delta^0 r^0)$. By the definition of the action of F the assumption implies that either $m_1' = m_2' = \bot$ or $\Xi \vdash \tau$. The first case is trivial. In the second case, since $\mathcal{S}, \mathcal{R} \in \mathcal{L}_V$ so by the previous 2_K) it holds that $(k_1', k_2', k_1, k_2) \in \Psi_K(\mathcal{S}, \mathcal{R})(\tau^\top)(\Xi\vec{R})$ $(\Delta^0 r^0)$. By the assumptions on the m's we then get the required termination approximation.

2_M) Assume $(m_1', m_2', m_1, m_2) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau^\top)(\Xi'\vec{R}')$ $(\Delta r) \wedge \Xi \vdash \tau$ and $(\Xi'\vec{R}') \supseteq (\Xi\vec{R})$. To show $(m_1', m_2', m_1, m_2) \in \Psi_T(\mathcal{R}, \mathcal{S})(\tau^\top)(\Xi\vec{R})(\Delta r)$. Let $\Delta^0 r^0 \triangleright \Delta r$ and let $(S_1', S_2', S_1, S_2) \in \Psi_S(\mathcal{S}, \mathcal{R})(\Delta^0 r^0)$ and $(k_1', k_2', k_1, k_2) \in \Psi_K(\mathcal{S}, \mathcal{R})$ $(\tau^\top)(\Xi\vec{R})(\Delta^0 r^0)$. The case $m_1' = m_2' = \bot$ is trivial. Else since $\mathcal{R} \in \mathcal{L}_V$ by the previous 1_K) it holds that $(k_1', k_2', k_1, k_2) \in \Psi_K(\mathcal{S}, \mathcal{R})(\tau^\top)(\Xi'\vec{R}')$ $(\Delta^0 r^0)$. By the assumptions on the m's we then get the required termination approximation.

V: 1_V) Assume $(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\tau)(\Xi\vec{R})(\Delta r)$ and $(\Xi'\vec{R}') \supseteq (\Xi\vec{R})$. To show $(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})$ $(T\tau)(\Xi'\vec{R}')$ (Δr).

2_V) Assume $(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\tau)(\Xi'\vec{R}')(\Delta r)$ and $\Xi \vdash \tau$ and $(\Xi'\vec{R}') \supseteq (\Xi\vec{R})$. To show $(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})$ $(T\tau)(\Xi\vec{R})(\Delta r)$.

The proof is by cases over the structure of the argument τ. In all cases $v_1' = v_2' = \bot$ is trivial. Else:

(α_j): In both directions it follows from the assumptions that $\alpha_j \in \Xi \subseteq \Xi'$ and as $(\Xi'\vec{R}') \supseteq (\Xi\vec{R})$ in both cases it is required that $(v_1', v_2', v_1, v_2) \in R_j(\Delta r)$.

(unit),(int): both directions are immidiate from the definition.

(τ ref): Follows from that the argument Δr is the same. In both directions it follows from the assumption that τ is closed.

($\tau_1 + \tau_2$),($\tau_1 \times \tau_2$),($\mu\alpha.\tau$): Follows from $\mathcal{S} \in \mathcal{L}_V$ as so $\mathcal{S}$ has Ξ-weakening and Ξ-strengthening. We show the $\mu\alpha.\tau$ case, the others are similar. 1) Assume $(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\mu\alpha.\tau)(\Xi\vec{R})(\Delta r)$.

So $\exists d'_1, d'_2, d_1, d_2 \in V.\ ((v'_1 = \bot \wedge d'_1 = \bot) \vee v'_1 = in_\mu(d'_1)) \wedge ((v'_2 = \bot \wedge d'_2 = \bot) \vee v'_2 = in_\mu(d'_2)) \wedge v_1 = in_\mu(d_1) \neq \bot \wedge v_2 = in_\mu(d_2) \neq \bot \wedge (d'_1, d'_2, d_1, d_2) \in \mathcal{S}(\tau[\mu\alpha.\tau/\alpha])(\Xi\vec{R})(\Delta r)$. Since $\Xi \vdash \mu\alpha.\tau$ iff $\Xi \vdash \tau[\mu\alpha.\tau/\alpha]$ and $\mathcal{S}$ has Ξ-wakening then $(d'_1, d'_2, d_1, d_2) \in \mathcal{S}(\tau[\mu\alpha.\tau/\alpha])(\Xi'\vec{R}')(\Delta r)$. So $(v'_1, v'_2, v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\mu\alpha.\tau)(\Xi'\vec{R}')(\Delta r)$.
2) Assume $(v'_1, v'_2, v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\mu\alpha.\tau)(\Xi'\vec{R}')(\Delta r)$ and $\Xi \vdash \mu\alpha.\tau$. So $\exists d'_1, d'_2, d_1, d_2 \in V.\ ((v'_1 = \bot \wedge d'_1 = \bot) \vee v'_1 = in_\mu(d'_1)) \wedge ((v'_2 = \bot \wedge d'_2 = \bot) \vee v'_2 = in_\mu(d'_2)) \wedge v_1 = in_\mu(d_1) \neq \bot \wedge v_2 = in_\mu(d_2) \neq \bot \wedge (d'_1, d'_2, d_1, d_2) \in \mathcal{S}(\tau[\mu\alpha.\tau/\alpha])(\Xi'\vec{R}')(\Delta r)$. Since $\Xi \vdash \mu\alpha.\tau$ implies $\Xi \vdash \tau[\mu\alpha.\tau/\alpha]$ and $\mathcal{S}$ has Ξ-strengthening then $(d'_1, d'_2, d_1, d_2) \in \mathcal{S}(\tau[\mu\alpha.\tau/\alpha])(\Xi\vec{R})(\Delta r)$. So $(v'_1, v'_2, v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})\ (\mu\alpha.\tau)(\Xi\vec{R})(\Delta r)$.

$(\forall\alpha.T\tau)$: Follows from 1_M) and 2_M) above, we may use α conversion. 1) Assume $(v'_1, v'_2, v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})\ (\forall\alpha.T\tau)(\Xi\vec{R})\ (\Delta r)$. We may additionally assume $\alpha \notin \Xi'$ else we α-convert $\forall\alpha.T\tau$. By assumption so $\exists d'_1, d'_2, d_1, d_2 \in TV.\ ((v'_1 = \bot \wedge d'_1 = \bot) \vee v'_1 = in_\forall\lfloor d'_1\rfloor) \wedge ((v'_2 = \bot \wedge d'_2 = \bot) \vee v'_2 = in_\forall\lfloor d'_2\rfloor) \wedge v_1 = in_\forall\lfloor d_1\rfloor \wedge v_2 = in_\forall\lfloor d_2\rfloor \wedge \forall R_\alpha \in \text{bParAdmRel}.\ (d'_1, d'_2, d_1, d_2) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau)(\Xi\alpha\ \vec{R}R_\alpha)\ (\Delta r)$. Since $\Xi \vdash \forall\alpha.T\tau$ implies $\Xi, \alpha \vdash T\tau$ and by assumption $\alpha \notin \Xi' \supseteq \Xi$ and using 1_M) we see $\forall R_\alpha \in \text{bParAdmRel}.\ (d'_1, d'_2, d_1, d_2) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau)(\Xi'\alpha\ \vec{R}'R_\alpha)$. So $(v'_1, v'_2, v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\forall\alpha.T\tau)(\Xi'\vec{R}')\ (\Delta r)$.
2) Assume $(v'_1, v'_2, v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\forall\alpha.T\tau)(\Xi'\vec{R}')\ (\Delta r)$ and $\Xi \vdash \forall\alpha.T\tau$. So $\alpha \notin \Xi' \supseteq \Xi$ and $\exists d'_1, d'_2, d_1, d_2 \in TV.\ ((v'_1 = \bot \wedge d'_1 = \bot) \vee v'_1 = in_\forall\lfloor d'_1\rfloor) \wedge ((v'_2 = \bot \wedge d'_2 = \bot) \vee v'_2 = in_\forall\lfloor d'_2\rfloor) \wedge v_1 = in_\forall\lfloor d_1\rfloor \wedge v_2 = in_\forall\lfloor d_2\rfloor \wedge \forall R_\alpha \in \text{bParAdmRel}.\ (d'_1, d'_2, d_1, d_2) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau)(\Xi'\alpha\ \vec{R}'R_\alpha)\ (\Delta r)$. Since $\Xi \vdash \forall\alpha.T\tau$ implies $\Xi, \alpha \vdash T\tau$ and by 2_M) it holds that $\forall R_\alpha \in \text{bParAdmRel}.\ (d'_1, d'_2, d_1, d_2) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau)(\Xi\alpha\ \vec{R}R_\alpha)\ (\Delta r)$. So $(v'_1, v'_2, v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\forall\alpha.T\tau)(\Xi\vec{R})\ (\Delta r)$.

$(\tau \to T\tau')$: Follows from Follows from $\mathcal{R} \in \mathcal{L}_V$ as so $\mathcal{R}$ has Ξ-weakening and Ξ-strengthening together with 1_M) and 2_M). 1) Assume $(v'_1, v'_2, v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\tau \to T\tau')(\Xi\vec{R})\ (\Delta r)$. By assumption so $\Xi \vdash \tau \to T\tau'$ and $\exists d'_1, d'_2, d_1, d_2 \in (V \multimap TV).\ ((v'_1 = \bot \wedge d'_1 = \bot) \vee v'_1 = in_\to\lfloor d'_1\rfloor) \wedge ((v'_2 = \bot \wedge d'_2 = \bot) \vee v'_2 = in_\to\lfloor d'_2\rfloor) \wedge v_1 = in_\to\lfloor d_1\rfloor \wedge v_2 = in_\to\lfloor d_2\rfloor \wedge \forall\Delta^0 r^0 \rhd \Delta r.\ \forall(w'_1, w'_2, w_1, w_2) \in \mathcal{R}(\tau)(\Xi\vec{R})(\Delta^0 r^0).\ (d'_1 w'_1, d'_2 w'_2, d_1 w_1, d_2 w_2) \in \Psi_M(\mathcal{R}, \mathcal{S})(T\tau')(\Xi\vec{R})(\Delta^0 r^0)$. We must show $\forall\Delta^0 r^0 \rhd \Delta r.\ \forall(w'_1, w'_2, w_1, w_2) \in \mathcal{R}(\tau)(\Xi'\vec{R}')\ (\Delta^0 r^0).\ (d'_1 w'_1, d'_2 w'_2, d_1 w_1, d_2 w_2) \in \Psi_M(\mathcal{R}, \mathcal{S})(T\tau')(\Xi'\vec{R}')\ (\Delta^0 r^0)$. Let $(w'_1, w'_2, w_1, w_2) \in \mathcal{R}(\tau)(\Xi'\vec{R}')(\Delta^0 r^0)$. Since $\Xi \vdash \tau$ and $\mathcal{R}$ has Ξ-strengthening then $(w'_1, w'_2, w_1, w_2) \in \mathcal{R}(\tau)(\Xi\vec{R})(\Delta^0 r^0)$. So it follows from the assumption that $(d'_1 w'_1, d'_2 w'_2, d_1 w_1, d_2 w_2) \in \Psi_M(\mathcal{R}, \mathcal{S})\ (T\tau')(\Xi\vec{R})(\Delta^0 r^0)$. By 1_M) then also $(d'_1 w'_1, d'_2 w'_2, d_1 w_1, d_2 w_2) \in \Psi_M(\mathcal{R}, \mathcal{S})\ (T\tau')(\Xi'\vec{R}')(\Delta^0 r^0)$. So $(v'_1, v'_2, v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\tau \to T\tau')(\Xi'\vec{R}')\ (\Delta r)$.
2) Assume $(v'_1, v'_2, v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\tau \to T\tau')(\Xi'\vec{R}')(\Delta r)$ and $\Xi \vdash \tau \to T\tau'$. So $\exists d'_1, d'_2, d_1, d_2 \in (V \multimap TV).\ ((v'_1 = \bot \wedge d'_1 = \bot) \vee v'_1 = in_\to\lfloor d'_1\rfloor) \wedge ((v'_2 = \bot \wedge d'_2 = \bot) \vee v'_2 = in_\to\lfloor d'_2\rfloor) \wedge v_1 = in_\to\lfloor d_1\rfloor \wedge v_2 = in_\to\lfloor d_2\rfloor \wedge \forall\Delta^0 r^0 \rhd \Delta r.\ \forall(w'_1, w'_2, w_1, w_2) \in \mathcal{R}(\tau)(\Xi'\vec{R}')(\Delta^0 r^0).\ (d'_1 w'_1, d'_2 w'_2, d_1 w_1, d_2 w_2) \in \Psi_M(\mathcal{R}, \mathcal{S})(T\tau')(\Xi'\vec{R}')(\Delta^0 r^0)$. We must show $\forall\Delta^0 r^0 \rhd \Delta r.\ \forall(w'_1, w'_2, w_1, w_2) \in \mathcal{R}(\tau)(\Xi\vec{R})(\Delta^0 r^0).\ (d'_1 w'_1, d'_2 w'_2, d_1 w_1, d_2 w_2) \in \Psi_M(\mathcal{R}, \mathcal{S})(T\tau')(\Xi\vec{R})(\Delta^0 r^0)$. Let $(w'_1, w'_2, w_1, w_2) \in \mathcal{R}(\tau)(\Xi\vec{R})(\Delta^0 r^0)$. Since $\mathcal{R}$ has Ξ-weakening then $(w'_1, w'_2, w_1, w_2) \in \mathcal{R}(\tau)(\Xi'\vec{R}')(\Delta^0 r^0)$. So it follows from the assumption that $(d'_1 w'_1, d'_2 w'_2, d_1 w_1, d_2 w_2) \in \Psi_M(\mathcal{R}, \mathcal{S})(T\tau')(\Xi'\vec{R}')(\Delta^0 r^0)$. By 2_M) since $\Xi \vdash T\tau'$ then also $(d'_1 w'_1, d'_2 w'_2, d_1 w_1, d_2 w_2) \in \Psi_M(\mathcal{R}, \mathcal{S})(T\tau')(\Xi\vec{R})(\Delta^0 r^0)$. So $(v'_1, v'_2, v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\tau \to T\tau')(\Xi\vec{R})\ (\Delta r)$.

■

Definition 16

- *Let* $D, E \in \{V, F(V,V)\}$, $\mathcal{R} \in \mathcal{L}_D,\ \mathcal{S} \in \mathcal{L}_E,\ f : D \multimap E,\ g : D \cong E,\ f \sqsubseteq g$. *Define*
  $(f,g) : \mathcal{R} \subset \mathcal{S} \overset{def}{\iff} \forall\tau.\ \forall(\Xi\vec{R}).\ \forall(\Delta r).\ \forall v_1', v_2', v_1, v_2 \in D.$
  $(v_1', v_2', v_1, v_2) \in \mathcal{R}(\tau)(\Xi\vec{R})(\Delta r) \Rightarrow (fv_1', fv_2', gv_1, gv_2) \in \mathcal{S}(\tau)(\Xi\vec{R})(\Delta r).$

- *Let* $D \in \{V, F(V,V)\}$, $\mathcal{R}, \mathcal{S} \in \mathcal{L}_D,\ f : D \multimap D,\ f \sqsubseteq id_D$. *Define*
  $f : \mathcal{R} \subset \mathcal{S} \overset{def}{\iff} (f, id_D) : \mathcal{R} \subset \mathcal{S}$

We also define $\forall f, g : V \multimap V$.

- $\forall S \in S.\ (\Psi_S(f,g))S = \lambda l.g(Sl)$
- $\forall k \in KV.\ (\Psi_K(f,g))k = \lambda s.\lambda v.k(\Psi_S(g,f)s)(fv)$
- $\forall m \in TV.\ (\Psi_T(f,g))m = \lambda k.\lambda s.m(\Psi_K(g,f)k)(\Psi_S(g,f)s)$

Lemma 17
$\forall \mathcal{R}, \mathcal{R}', \mathcal{S}, \mathcal{S}' \in \mathcal{L}_V.\forall f, g : V \multimap V,\ f, g \sqsubseteq id_V$ *it holds that*
If $f : \mathcal{R}' \subset \mathcal{R}$ *and* $g : \mathcal{S} \subset \mathcal{S}'$ *then*

1. $\forall\Delta r.\ \forall S_1', S_2', S_1, S_2.$
   $(S_1', S_2', S_1, S_2) \in \Psi_S(\mathcal{R}, \mathcal{S})(\Delta r) \Rightarrow$
   $(\Psi_S(f,g)S_1', \Psi_S(f,g)S_2', S_1, S_2) \in \Psi_S(\mathcal{R}', \mathcal{S}')(\Delta r).$

2. $\forall\tau^\top.\ \forall(\Xi\vec{R}).\ \forall\Delta r.\ \forall k_1', k_2', k_1, k_2.$
   $(k_1', k_2', k_1, k_2) \in \Psi_K(\mathcal{R}, \mathcal{S})(\tau^\top)(\Xi\vec{R})(\Delta r) \Rightarrow$
   $(\Psi_K(f,g)k_1', \Psi_K(f,g)k_2', k_1, k_2) \in \Psi_K(\mathcal{R}', \mathcal{S}')(\tau^\top)(\Xi\vec{R})(\Delta r).$

3. $\forall T\tau.\ \forall(\Xi\vec{R}).\ \forall\Delta r.\ \forall m_1', m_2', m_1, m_2.$
   $(m_1', m_2', m_1, m_2) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau)(\Xi\vec{R})(\Delta r) \Rightarrow$
   $(\Psi_T(f,g)m_1', \Psi_T(f,g)m_2', m_1, m_2) \in \Psi_T(\mathcal{R}', \mathcal{S}')(T\tau)(\Xi\vec{R})(\Delta r).$

4. $F(f,g) : \Psi(\mathcal{R}, \mathcal{S}) \subset \Psi(\mathcal{R}', \mathcal{S}')$

Corollary 18
Monotonicity of Ψ *follows from the lemma, with* $f = g = id_V$.

Proof: We prove 1,2,3,4 in this order. Assume $f : \mathcal{R}' \subset \mathcal{R}$ and $g : \mathcal{S} \subset \mathcal{S}'$. So $f \sqsubseteq id_V,\ g \sqsubseteq id_V$ and then $\forall S \in S.\ \Psi_S(f,g)S \sqsubseteq S$, $\forall k \in KV.\ \Psi_K(f,g)k \sqsubseteq k$, $\forall m \in TV.\ \Psi_T(f,g)m \sqsubseteq m$.

1. Assume $(S_1', S_2', S_1, S_2) \in \Psi_S(\mathcal{R}, \mathcal{S})(\Delta r)$. To show
   $(\Psi_S(f,g)S_1', \Psi_S(f,g)S_2', S_1, S_2) \in \Psi_S(\mathcal{R}', \mathcal{S}')(\Delta r)$. If $S_1' = S_2' = \bot$ then $\Psi_S(f,g)S_1' = \Psi_S(f,g)S_2' = \bot$ and we are done. Else it holds that $S_1 \neq \bot \wedge S_2 \neq \bot$, and we reason as follows. We must show $(\lambda l.g(S_1'l), \lambda l.g(S_2'l), S_1, S_2) \in \Psi_S(\mathcal{R}', \mathcal{S}')(\Delta r)$. S_1, S_2 have not been changed. Requirements of disjointness of areas given by accessibility maps and membership of state relations are only on S_1, S_2, so these properties are preserved. Since by assumption $g : \mathcal{S} \subset \mathcal{S}'$ and $\forall l \in dom(\Delta).\ (S_1'l, S_2'l, S_1l, S_2l) \in \mathcal{S}(\Delta(l))()(\Delta r)$ then $\forall l \in dom(\Delta).\ (g(S_1'l), g(S_2'l), S_1l, S_2l) \in \mathcal{S}'(\Delta(l))()(\Delta r)$. Since all requirements of membership in state relations are preserved, the LL's required to hold related values will be the same. For each such LL, since by assumption $g : \mathcal{S} \subset \mathcal{S}'$ and $\forall(l_1, l_2, \tau) \in LL.\ (S_1'l_1, S_2'l_2, S_1l_1, S_2l_2) \in \mathcal{S}(\tau)()(\Delta r)$ then also $\forall(l_1, l_2, \tau) \in LL.\ (g(S_1'l_1), g(S_2'l_2), S_1l_1, S_2l_2) \in \mathcal{S}'(\tau)()(\Delta r)$. So we can conclude
   $(\Psi_S(f,g)S_1', \Psi_S(f,g)S_2', S_1, S_2) \in \Psi_S(\mathcal{R}', \mathcal{S}')(\Delta r).$

2. Assume $(k'_1, k'_2, k_1, k_2) \in \Psi_K(\mathcal{R}, \mathcal{S})(\tau^\top)(\Xi\vec{R})(\Delta r)$. To show $(\Psi_K(f,g)k'_1, \Psi_K(f,g)k'_2, k_1, k_2) \in \Psi_K(\mathcal{R}', \mathcal{S}')(\tau^\top)\ (\Xi\vec{R})(\Delta r)$. If $k'_1 = k'_2 = \bot$ then $\Psi_K(f,g)k'_1 = \Psi_K(f,g)k'_2 = \bot$ and we are done. Else we must show $(\lambda s.\lambda v.k'_1(\Psi_S(g,f)s)(fv), \lambda s.\lambda v.k'_2(\Psi_S(g,f)s)(fv), k_1, k_2) \in \Psi_K(\mathcal{R}', \mathcal{S}')(\tau^\top)\ (\Xi\vec{R})(\Delta r)$. Let $\Delta' r' \rhd \Delta r$ and assume $(S'_1, S'_2, S_1, S_2) \in \Psi_S(\mathcal{S}', \mathcal{R}')(\Delta' r')$ and $(v'_1, v'_2, v_1, v_2) \in \mathcal{R}'(\tau)(\Xi\vec{R})(\Delta' r')$. Since by assumption $f : \mathcal{R}' \subset \mathcal{R},\ g : \mathcal{S} \subset \mathcal{S}'$ and by 1) it follows that $(\Psi_S(g,f)S'_1, \Psi_S(g,f)S'_2, S_1, S_2) \in \Psi_S(\mathcal{S}, \mathcal{R})(\Delta' r')$ and $(fv'_1, fv'_2, v_1, v_2) \in \mathcal{R}(\tau)(\Xi\vec{R})(\Delta' r')$. Then it follows from the assumptions that
$(\Psi_K(f,g)k'_1)S'_1 v'_1 = k'_1(\Psi_S(g,f)S'_1)(fv'_1) = \top \Longrightarrow k_2 S_2 v_2 = \top$ and
$(\Psi_K(f,g)k'_2)S'_2 v'_2 = k'_2(\Psi_S(g,f)S'_2)(fv'_2) = \top \Longrightarrow k_1 S_1 v_1 = \top$.
We conclude $((\Psi_K(f,g)k'_1), (\Psi_K(f,g)S'_2), k_1, k_2) \in \Psi_K(\mathcal{R}', \mathcal{S}')(\tau^\top)(\Xi\vec{R})(\Delta r)$.

3. Assume $(m'_1, m'_2, m_1, m_2) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau)(\Xi\vec{R})(\Delta r)$. To show $(\Psi_T(f,g)m'_1, \Psi_T(f,g)m'_2, m_1, m_2) \in \Psi_T(\mathcal{R}', \mathcal{S}')(T\tau)\ (\Xi\vec{R})(\Delta r)$. If $m'_1 = m'_2 = \bot$ then $\Psi_T(f,g)m'_1 = \Psi_T(f,g)m'_2 = \bot$ and we are done. Else we must show $(\lambda k.\lambda s.m'_1(\Psi_K(g,f)k)(\Psi_S(g,f)s)$,
$\lambda s.\lambda v.m'_2(\Psi_K(g,f)k)(\Psi_S(g,f)s), m_1, m_2) \in \Psi_T(\mathcal{R}', \mathcal{S}')(T\tau)\ (\Xi\vec{R})(\Delta r)$. Let $\Delta' r' \rhd \Delta r$ and assume $(k'_1, k'_2, k_1, k_2) \in \Psi_K(\mathcal{S}', \mathcal{R}')(\tau^\top)(\Xi\vec{R})(\Delta' r')$ and $(S'_1, S'_2, S_1, S_2) \in \Psi_S(\mathcal{S}', \mathcal{R}')(\Delta' r')$. Since by assumption $f : \mathcal{R}' \subset \mathcal{R},\ g : \mathcal{S} \subset \mathcal{S}'$ and by 1) and 2) it follows that $(\Psi_K(g,f)k'_1$, $\Psi_K(g,f)k'_2, k_1, k_2) \in \Psi_K(\mathcal{S}, \mathcal{R})(\tau^\top)(\Xi\vec{R})(\Delta' r')$ and $(\Psi_S(g,f)S'_1, \Psi_S(g,f)S'_2, S_1, S_2) \in \Psi_S(\mathcal{S}, \mathcal{R})(\Delta' r')$. Then it follows from the assumptions that
$(\Psi_T(f,g)m'_1)k'_1 S'_1 = m'_1(\Psi_K(g,f)k'_1)(\Psi_S(g,f)S'_1) = \top \Longrightarrow m_2 k_2 S_2 = \top$ and
$(\Psi_T(f,g)m'_2)k'_2 S'_2 = m'_2(\Psi_K(g,f)k'_2)(\Psi_S(g,f)S'_2) = \top \Longrightarrow m_1 k_1 S_1 = \top$.
We conclude $((\Psi_T(f,g)m'_1), (\Psi_T(f,g)m'_2), m_1, m_2) \in \Psi_T(\mathcal{R}', \mathcal{S}')(T\tau)(\Xi\vec{R})(\Delta r)$.

4. The proof is by case analysis over the structure of the argument τ.
   - Assume $(v'_1, v'_2, v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\alpha_i)(\Xi\vec{R})(\Delta r)$. To show $(F(f,g)v'_1, F(f,g)v'_2, v_1, v_2) \in \Psi(\mathcal{R}', \mathcal{S}')\ (\alpha_i)(\Xi\vec{R})\ (\Delta r)$. If $v'_1 = v'_2 = \bot$ then $F(f,g)(v'_1) = F(f,g)(v'_2) = \bot$ and we are done. Else it holds that $\alpha_i \in \Xi$ and $v'_1 \sqsubseteq v_1 \neq \bot \wedge v'_2 \sqsubseteq v_2 \neq \bot \wedge (v'_1, v'_2, v_1, v_2) \in R_i(\Delta r)$ (independant of $\mathcal{R}, \mathcal{S}$). To show $F(f,g)v'_1 \sqsubseteq v_1 \wedge F(f,g)v'_2 \sqsubseteq v_2 \wedge (F(f,g)v'_1, F(f,g)v'_2, v_1, v_2) \in R_i(\Delta r)$. This follows from assumptions: since $f, g \sqsubseteq id_V$ then $F(f,g) \sqsubseteq id_{F(V,V)}$, and R_i is downwards closed.
   - Assume $(v'_1, v'_2, v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(1)(\Xi\vec{R})(\Delta r)$. This implies $(v'_1 = v'_2 = \bot)\ \vee v'_1 \sqsubseteq v_1 = in_1(*) \wedge v'_2 \sqsubseteq v_2 = in_1(*)$, independant of $\mathcal{R}, \mathcal{S}$. $F(f,g)\bot = \bot$ and $F(f,g)in_1(*) = in_1(*)$. So we conclude $(F(f,g)v'_1,\ F(f,g)v'_2, v_1, v_2) \in \Psi(\mathcal{R}', \mathcal{S}')(1)(\Xi\vec{R})(\Delta r)$.
   - Assume $(v'_1, v'_2, v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(int)(\Xi\vec{R})(\Delta r)$. This implies $(v'_1 = v'_2 = \bot)\ \vee \exists n : Z.\ v'_1 \sqsubseteq v_1 = in_{\mathsf{int}}(n) \wedge v'_2 \sqsubseteq v_2 = in_{\mathsf{int}}(n)$, independant of $\mathcal{R}, \mathcal{S}$. $F(f,g)\bot = \bot$ and $F(f,g)in_{\mathsf{int}}(n) = in_{\mathsf{int}}(n)$. So we conclude $(F(f,g)v'_1,\ F(f,g)v'_2,\ v_1, v_2) \in \Psi(\mathcal{R}', \mathcal{S}')(int)$ $(\Xi\vec{R})(\Delta r)$.
   - Assume $(v'_1, v'_2, v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\tau\mathsf{ref})(\Xi\vec{R})(\Delta r)$. This implies $(v'_1 = v'_2 = \bot)\ \vee - \vdash \tau\ \wedge \exists l : Loc.\ v'_1 \sqsubseteq v_1 = in_{\mathsf{ref}}(l) \wedge v'_2 \sqsubseteq v_2 = in_{\mathsf{ref}}(l) \wedge l \in \mathrm{dom}(\Delta) \wedge \Delta(l) = \tau)$, independant of $\mathcal{R}, \mathcal{S}$. $F(f,g)\bot = \bot$ and $F(f,g)in_{\mathsf{ref}}(l) = in_{\mathsf{ref}}(l)$. So we conclude $(F(f,g)v'_1,\ F(f,g)v'_2$, $v_1, v_2) \in \Psi(\mathcal{R}', \mathcal{S}')(\tau\mathsf{ref})(\Xi\vec{R})(\Delta r)$.
   - Assume $(v'_1, v'_2, v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\tau_1 + \tau_2)(\Xi\vec{R})(\Delta r)$. This implies $(v'_1 = v'_2 = \bot)\ \vee\ \Xi \vdash \tau_1 + \tau_2 \wedge v'_1 \sqsubseteq v_1 \neq \bot \wedge v'_2 \sqsubseteq v_2 \neq \bot \wedge \exists d'_1, d'_2 : V, d_1, d_2 : V_\downarrow.\ ((v'_1 = \bot \wedge d'_1 = \bot) \vee (v'_1 = in_+(in_i(d'_1) \neq \bot) \wedge ((v'_2 = \bot \wedge d'_2 = \bot) \vee (v'_2 = in_+(in_i(d'_2) \neq \bot) \wedge v_1 = in_+(in_i(d_1)) \wedge v_2 = in_+(in_i(d_2)) \wedge (d'_1, d'_2, d_1, d_2) \in \mathcal{S}(\tau_i)(\Xi\vec{R})(\Delta r) \wedge i \in \{1, 2\})$. Since $g : \mathcal{S} \subset \mathcal{S}'$ then $(gd'_1, gd'_2, d_1, d_2) \in \mathcal{S}'(\tau_i)(\Xi\vec{R})(\Delta r)$. $F(f,g)\bot = \bot$ and $F(f,g)in_+(in_i(d'_1)) = in_+(in_i(gd'_1))$ and $F(f,g)\ in_+(in_i(d'_2)) = in_+(in_i(gd'_2))$). We conclude $(F(f,g)v'_1,\ F(f,g)v'_2, v_1, v_2) \in \Psi(\mathcal{R}', \mathcal{S}')(\tau_1 + \tau_2)(\Xi\vec{R})(\Delta r)$.
   - Assume $(v'_1, v'_2, v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\tau_1 \times \tau_2)(\Xi\vec{R})(\Delta r)$. The proof is similar to the proof for $\tau_1 + \tau_2$ so we omit it.

- Assume $(v'_1, v'_2, v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\tau \to T\tau')(\Xi\vec{R})(\Delta r)$. To prove $(F(f,g)v'_1, F(f,g)v'_2, v_1, v_2) \in \Psi(\mathcal{R}', \mathcal{S}')(\tau \to T\tau')(\Xi\vec{R})(\Delta r)$. If $v'_1 = v'_2 = \bot$ this follows by strictness of $F(f,g)$. Else it follows from the assumption that $(\Xi \vdash \tau \to T\tau') \wedge v'_1 \sqsubseteq v_1 \neq \bot \wedge v'_2 \sqsubseteq v_2 \neq \bot \wedge \exists d'_1, d'_2, d_1, d_2 : V \multimap TV.\ ((v'_1 = \bot \wedge d'_1 = \bot) \vee v'_1 = in_{\to}\lfloor d'_1 \rfloor) \wedge ((v'_2 = \bot \wedge d'_2 = \bot) \vee v'_2 = in_{\to}\lfloor d'_2 \rfloor) \wedge v_1 = in_{\to}\lfloor d_1 \rfloor \wedge v_2 = in_{\to}\lfloor d_2 \rfloor \wedge \forall \Delta' r' \rhd \Delta r.\ \forall v'_{11}, v'_{22}, v_{11}, v_{22} : V.\ (v'_{11}, v'_{22}, v_{11}, v_{22}) \in \mathcal{R}(\tau)(\Xi\vec{R})(\Delta' r') \Rightarrow (d'_1 v'_{11}, d'_2 v'_{22}, d_1 v_{11}, d_2 v_{22}) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau')$ $(\Xi\vec{R})(\Delta' r')$. $F(f,g)v'_1 \sqsubseteq v_1 \wedge F(f,g)v'_2 \sqsubseteq v_2$ follows from $F(f,g) \sqsubseteq id_{F(V,V)} \wedge v'_1 \sqsubseteq v_1 \wedge v'_2 \sqsubseteq v_2$. $F(f,g)v'_1 \sqsubseteq in_{\to}\lfloor \lambda v.\Psi_T(f,g)(d'_1(fv)) \rfloor \wedge F(f,g)v'_2 \sqsubseteq in_{\to}\lfloor \lambda v.\Psi_T(f,g)(d'_2(fv)) \rfloor$. Let $\Delta' r' \rhd \Delta r$ and suppose $(v'_{11}, v'_{22}, v_{11}, v_{22}) \in \mathcal{R}'(\tau)\ (\Xi\vec{R})(\Delta' r')$. Since $f : \mathcal{R}' \subset \mathcal{R}$ then $(fv'_{11}, fv'_{22}, v_{11}, v_{22}) \in \mathcal{R}(\tau)(\Xi\vec{R})(\Delta' r')$. Then by assumptions $(d'_1(fv'_{11}),\ d'_2(fv'_{22}),$ $d_1(v_{11}), d_2(v_{22})) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau')(\Xi\vec{R})(\Delta' r')$. From 3) follows $((\Psi_T(f,g)\ (d'_1(fv'_{11}))),$ $(\Psi_T(f,g)(d'_2\ (fv'_{22}))), d_1 v_{11}, d_2 v_{22}) \in \Psi_T(\mathcal{R}', \mathcal{S}')(T\tau')(\Xi\vec{R})(\Delta' r')$.
  So $(F(f,g)v'_1, F(f,g)v'_2, v_1, v_2) \in \Psi(\mathcal{R}', \mathcal{S}')(\tau \to T\tau')(\Xi\vec{R})(\Delta r)$.
- Assume $(v'_1, v'_2, v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\mu\alpha.\tau)(\Xi\vec{R})(\Delta r)$. This implies $(v'_1 = v'_2 = \bot)\ \vee \Xi \vdash \mu\alpha.\tau \wedge v'_1 \sqsubseteq v_1 \neq \bot \wedge v'_2 \sqsubseteq v_2 \neq \bot \wedge \exists d'_1, d'_2 : V, d_1, d_2 : V_{\downarrow}.\ ((v'_1 = \bot \wedge d'_1 = \bot) \vee (v'_1 = in_{\mu}(d'_1) \neq \bot) \wedge ((v'_2 = \bot \wedge d'_2 = \bot) \vee (v'_2 = in_{\mu}(d'_2) \neq \bot) \wedge v_1 = in_{\mu}(d_1) \wedge v_2 = in_{\mu}(d_2) \wedge (d'_1, d'_2, d_1, d_2) \in \mathcal{S}(\tau[\mu\alpha.\tau/\alpha])(\Xi\vec{R})(\Delta r)$. Since $g : \mathcal{S} \subset \mathcal{S}'$ then $(gd'_1, gd'_2, d_1, d_2) \in \mathcal{S}'(\tau[\mu\alpha.\tau/\alpha])(\Xi\vec{R})(\Delta r)$. $F(f,g)\bot = \bot$ and $F(f,g)in_{\mu}(d'_1) = in_{\mu}(gd'_1)$ and $F(f,g)in_{\mu}(d'_2) = in_{\mu}(gd'_2)$. We conclude
  $(F(f,g)v'_1, F(f,g)v'_2, v_1, v_2) \in \Psi(\mathcal{R}', \mathcal{S}')(\mu\alpha.\tau)(\Xi\vec{R})(\Delta r)$.
- Assume $(v'_1, v'_2, v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\forall\alpha.\ T\tau)(\Xi\vec{R})(\Delta r)$.
  To prove $(F(f,g)v'_1, F(f,g)v'_2, v_1, v_2) \in \Psi(\mathcal{R}', \mathcal{S}')\ (\forall\alpha.\ T\tau)\ (\Xi\vec{R})(\Delta r)$. If $v'_1 = v'_2 = \bot$ this follows by strictness of $F(f,g)$. Else it follows from the assumption that $\Xi \vdash \forall\alpha.T\tau \wedge \alpha \notin \Xi \wedge v'_1 \sqsubseteq v_1 \neq \bot \wedge v'_2 \sqsubseteq v_2 \neq \bot \wedge \exists d'_1, d'_2, d_1, d_2 : TV.\ ((v'_1 = \bot \wedge d'_1 = \bot) \vee v'_1 = in_{\forall}\lfloor d'_1 \rfloor) \wedge ((v'_2 = \bot \wedge d'_2 = \bot) \vee v'_2 = in_{\forall}\lfloor d'_2 \rfloor) \wedge v_1 = in_{\forall}\lfloor d_1 \rfloor \wedge v_2 = in_{\forall}\lfloor d_2 \rfloor \wedge \forall R_{\alpha}.\ (d'_1, d'_2, d_1, d_2) \in (\Psi_T(\mathcal{R}, \mathcal{S})(T\tau)(\Xi\alpha\vec{R}R_{\alpha})(\Delta r)$. By 3) then $\forall R_{\alpha}.\ (\Psi_T(f,g)d'_1, \Psi_T(f,g)d'_2, d_1, d_2) \in \Psi_T(\mathcal{R}', \mathcal{S}')(T\tau)(\Xi\alpha\ \vec{R}R_{\alpha})(\Delta r)$. $\big(v'_1 = \bot \Rightarrow F(f,g)v'_1 = \bot \wedge v'_1 = in_{\forall}\lfloor d'_1 \rfloor \Rightarrow F(f,g)v'_1 = in_{\forall}\lfloor \lambda k.\lambda s.(d'_1)(\Psi_K(g,f)k)(\Psi_S(g,f)s) \rfloor = in_{\forall}\lfloor \Psi_T(f,g)(d'_1) \rfloor\big) \wedge \big(v'_2 = \bot \Rightarrow F(f,g)v'_2 = \bot \wedge v'_2 = in_{\forall}\lfloor d'_2 \rfloor \Rightarrow F(f,g)v'_2 = in_{\forall}\lfloor \Psi_T(f,g)(d'_2) \rfloor\big)$. We let $m'_1 = \Psi_T(f,g)(d'_1)$ and $m'_2 = \Psi_T(f,g)(d'_2)$. So we have $\exists m'_1, m'_2, d_1, d_2 : TV.\ ((F(f,g)v'_1 = \bot \wedge m'_1 = \bot) \vee F(f,g)v'_1 = in_{\forall}\lfloor m'_1 \rfloor) \wedge ((F(f,g)v'_2 = \bot \wedge m'_2 = \bot) \vee F(f,g)v'_2 = in_{\forall}\lfloor m'_2 \rfloor) \wedge v_1 = in_{\forall}\lfloor d_1 \rfloor \wedge v_2 = in_{\forall}\lfloor d_2 \rfloor \wedge \forall R_{\alpha}.\ (m'_1, m'_2, m_1, m_2) \in \Psi_T(\mathcal{R}', \mathcal{S}')(T\tau)(\Xi\alpha\vec{R}R_{\alpha})(\Delta r)$. We conclude $(F(f,g)v'_1,\ F(f,g)v'_2, v_1, v_2) \in \Psi(\mathcal{R}', \mathcal{S}')(\forall\alpha.\ T\tau)(\Xi\vec{R})(\Delta r)$.

■

Let $\mathcal{A} \in \mathcal{L}_{F(V,V)}$, we define $(i^{-1})^*\mathcal{A}$ to be the largest relation $\mathcal{B} \in \mathcal{L}_V$ such that $(i^{-1}, i^{-1}) : \mathcal{B} \subset \mathcal{A}$. Then $\mathcal{R} = (i^{-1})^*\Psi(\mathcal{R}, \mathcal{R}) \iff \big((i,i) : \Psi(\mathcal{R}, \mathcal{R}) \subset \mathcal{R} \wedge (i^{-1}, i^{-1}) : \mathcal{R} \subset \Psi(\mathcal{R}, \mathcal{R})\big)$.

We now proceed by defining
$$\Psi^{\S} : \mathcal{L}_V^{\mathrm{op}} \times \mathcal{L}_V \to \mathcal{L}_V^{\mathrm{op}} \times \mathcal{L}_V$$
by
$$\Psi^{\S}(\mathcal{R}, \mathcal{S}) = ((i^{-1})^*\Psi(\mathcal{S}, \mathcal{R}), (i^{-1})^*\Psi(\mathcal{R}, \mathcal{S})).$$
Since $\Psi^{\S}$ is monotone it has a least fixed point which is also the least prefixed point
$$(\nabla^-, \nabla^+) = \Psi^{\S}(\nabla^-, \nabla^+)$$
satisfying
$$\nabla^- = (i^{-1})^*\Psi(\nabla^+, \nabla^-) \qquad \nabla^+ = (i^{-1})^*\Psi(\nabla^-, \nabla^+)$$

We have by definition
$\Psi^{\S}(\nabla^+, \nabla^-) = ((i^{-1})^*\Psi(\nabla^-, \nabla^+), (i^{-1})^*\Psi(\nabla^+, \nabla^-))$.
And so $\Psi^{\S}(\nabla^+, \nabla^-) = (\nabla^+, \nabla^-)$. Since (∇^-, ∇^+) is the least (pre)fixed point it follows, as with Pitts, that

$$\nabla^+ \leq \nabla^-.$$

We now want to show that $\nabla^- \leq \nabla^+$, that is, that

$$id_V \in \Phi = \{\, e \mid e : \nabla^- \subset \nabla^+ \}$$

Recall $e : \nabla^- \subset \nabla^+ \Leftrightarrow (e, id_V) : \nabla^- \subset \nabla^+ \Leftrightarrow$
$\forall \tau \in \mathit{ValueType}.\forall(\Xi\vec{R}) \in (\mathit{TypeVar}^n \times \text{bParAdmRel}^n).\forall\Delta r.\forall(v_1', v_2', v_1, v_2)$.
$(v_1', v_2', v_1, v_2) \in \nabla^-(\tau)(\Xi\vec{R})(\Delta r) \Rightarrow (ev_1', ev_2', v_1, v_2) \in \nabla^+(\tau)(\Xi\vec{R})(\Delta r)$.
Since the ordering on functions is pointwise and ∇^+ is admissible, so it holds that if e^i is a chain s.t. $\forall i.\ e^i : \nabla^- \subset \nabla^+$ then also $\bigsqcup e^i : \nabla^- \subset \nabla^+$. We have that $\bot \in \Phi$. By the minimal invariant property of V it holds that $id_V = \bigsqcup \delta^n(\bot)$, where $\delta(e) = i \circ \Psi(e, e) \circ i^{-1}$. Hence it suffices to show that Φ is closed under $\delta : (V \multimap V) \to (V \multimap V)$

Assume $e : \nabla^- \subset \nabla^+$. Then by lemma 17 it holds that $\Psi(e, e) : \Psi(\nabla^+, \nabla^-) \subset \Psi(\nabla^-, \nabla^+)$. We also have from the definition and the fixed point property, that $\nabla^- = (i^{-1})^*\Psi(\nabla^+, \nabla^-)$ and $\nabla^+ = (i^{-1})^*\Psi(\nabla^-, \nabla^+)$. This gives $(i^{-1}, i^{-1}) : \nabla^- \subset \Psi(\nabla^+, \nabla^-)$ and $(i^{-1}, i^{-1}) : \nabla^+ \subset \Psi(\nabla^-, \nabla^+)$. Now $\forall \Xi \vdash \tau, \vec{R}, \Delta r(v_1, v_2)$it holds that if $(v_1', v_2') \in \Psi(\nabla^-, \nabla^+)(\tau)(\Xi\vec{R})\ (\Delta r(v_1, v_2))$ then $(i^{-1} \circ i)v_1', (i^{-1}\circ i)v_2') \in \Psi(\nabla^-, \nabla^+)(\tau)(\Xi\vec{R})\ (\Delta r((i^{-1}\circ i)v_1, (i^{-1}\circ i)v_2))$. So $(i(v_1'), i(v_2')) \in ((i^{-1})^*\Psi(\nabla^-, \nabla^+)(\tau)(\Xi\vec{R})\ (\Delta r(i(v_1), i(v_2))))$, that is $(i(v_1'), i(v_2')) \in (\nabla^+(\tau)\ (\Xi\vec{R})(\Delta r(i(v_1), i(v_2))))$, i.e. $(i, i) : \Psi(\nabla^-, \nabla^+) \subset \nabla^+$.

Assuming $e : \nabla^- \subset \nabla^+$, that is $(e, id_V) : \nabla^- \subset \nabla^+$, and combining $(i^{-1}, i^{-1}) : \nabla^- \subset \Psi(\nabla^+, \nabla^-)$ and $\Psi(e, e) : \Psi(\nabla^+, \nabla^-) \subset \Psi(\nabla^-, \nabla^+)$ and $(i, i) : \Psi(\nabla^-, \nabla^+) \subset \nabla^+$ we have $\delta(e) : \nabla^- \subset \nabla^+$. And then it follows that $id_V : \nabla^- \subset \nabla^+$. So $\nabla^- = \nabla^+$.
Let $\nabla = \nabla^- = \nabla^+$.

Finally, $[\![\tau]\!]^t$ is defined to be $\nabla(\tau)$.

Lemma 19

$\forall \Xi, \alpha \notin \Xi.\ \forall \vec{R} \in \text{bParAdmRel}^{|\Xi|}$.
$\forall(\Xi, \alpha \vdash \tau).\ \forall \Xi \vdash \sigma.\ \forall \Delta r.$

$$[\![\tau]\!]^t(\Xi\alpha\ \vec{R}\ ([\![\sigma]\!]^t(\Xi\ \vec{R}))(\Delta r) =$$
$$[\![\tau[\sigma/\alpha]]\!]^t(\Xi\ \vec{R})(\Delta r).$$

Comment: Recall that we require for $\alpha_1 \ldots \alpha_k \vdash \tau'\ ref$ that τ' is closed. Therefore it does not hold that $\Xi \vdash \tau[\sigma/\alpha]$ is a derivable typing judgement implies that $\Xi, \alpha \vdash \tau$ is derivable. In the lemma we require $\Xi, \alpha \vdash \tau$ and so this will not give any problems.

Note the use of fixed point induction in the proof below to establish a property of our recursively defined relation.

$$
\begin{aligned}
&(S_1', S_2', S_1, S_2) \in \Psi_S(\mathcal{R}, \mathcal{S})(\Delta r) \iff \exists r_1, \dots r_n.\ r = \{r_1, \dots r_n\} \wedge \\
&\quad \big(S_1' = S_2' = \bot\big)\ \vee \big(S_1' \sqsubseteq S_1 \neq \bot \wedge S_2' \sqsubseteq S_2 \neq \bot \wedge \\
&\quad \forall i \neq j, i, j \in \{1, \dots, n\}.\ A_{ri1}(S_1) \cap A_{rj1}(S_1) = \emptyset \wedge A_{ri2}(S_2) \cap A_{rj2}(S_2) = \emptyset \wedge \\
&\quad \mathrm{dom}(\Delta) \cap A_{r1}(S_1) = \emptyset \wedge \mathrm{dom}(\Delta) \cap A_{r2}(S_2) = \emptyset \wedge \\
&\quad \forall l \in \mathrm{dom}(\Delta).\ (S_1' l, S_2' l, S_1 l, S_2 l) \in (\mathcal{S}(\Delta(l))()(\Delta r) \wedge \\
&\quad \forall r_a \in r.\ \exists (P_b, LL_b) \in r_a.\ \ (S_1, S_2) \in \hat{p}_b \wedge \forall (l_1, l_2, \tau) \in LL_b.\ (S_1' l_1, S_2' l_2, S_1 l_1, S_2 l_2) \in \mathcal{S}(\tau)()(\Delta r)\big)
\end{aligned}
$$

$$
\begin{aligned}
&(k_1', k_2', k_1, k_2) \in \Psi_K(\mathcal{R}, \mathcal{S})(\tau^\top)(\Xi \vec{R})(\Delta r) \iff \\
&\quad (k_1' = k_2' = \bot)\ \vee (\Xi \vdash \tau \wedge k_1' \sqsubseteq k_1 \wedge k_2' \sqsubseteq k_2 \wedge \\
&\quad \forall \Delta' r' \rhd \Delta r.\ \forall S_1', S_2', S_1, S_2.\ \forall v_1', v_2', v_1, v_2. \\
&\qquad \big((S_1', S_2', S_1, S_2) \in \Psi_S(\mathcal{S}, \mathcal{R})(\Delta' r') \wedge (v_1', v_2', v_1, v_2) \in \mathcal{R}(\tau)(\Xi \vec{R})(\Delta' r')\big) \Rightarrow \\
&\qquad\quad \big((k_1' S_1' v_1' = \top \Rightarrow k_2 S_2 v_2 = \top) \wedge (k_2' S_2' v_2' = \top \Rightarrow k_1 S_1 v_1 = \top)\big)
\end{aligned}
$$

$$
\begin{aligned}
&(m_1', m_2', m_1, m_2) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau)(\Xi \vec{R})(\Delta r) \iff \\
&\quad (m_1' = m_2' = \bot)\ \vee (\Xi \vdash \tau \wedge m_1' \sqsubseteq m_1 \wedge m_2' \sqsubseteq m_2 \wedge \\
&\quad \forall \Delta' r' \rhd \Delta r.\ \forall k_1', k_2', k_1, k_2.\ \forall S_1', S_2', S_1, S_2. \\
&\qquad \big((k_1', k_2', k_1, k_2) \in \Psi_K(\mathcal{S}, \mathcal{R})(\tau^\top)(\Xi \vec{R})(\Delta' r') \wedge (S_1', S_2', S_1, S_2) \in \Psi_S(\mathcal{S}, \mathcal{R})(\Delta' r')\big) \Rightarrow \\
&\qquad\quad \big((m_1' k_1' S_1' = \top \Rightarrow m_2 k_2 S_2 = \top) \wedge (m_2' k_2' S_2' = \top \Rightarrow m_1 k_1 S_1 = \top)\big)
\end{aligned}
$$

$$
\begin{aligned}
&(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\alpha_i)(\Xi \vec{R})(\Delta r) \iff (v_1' = v_2' = \bot)\ \vee\ (\alpha_i \in \Xi \wedge (v_1', v_2', v_1, v_2) \in R_i(\Delta r) \\
&(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(1)(\Xi \vec{R})(\Delta r) \iff (v_1' = v_2' = \bot)\ \vee (v_1' \sqsubseteq v_1 = in_1 \lfloor * \rfloor \wedge v_2' \sqsubseteq v_2 = in_1 \lfloor * \rfloor) \\
&(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\mathsf{int})(\Xi \vec{R})(\Delta r) \iff (v_1' = v_2' = \bot)\ \vee (\exists n : Z.\ v_1' \sqsubseteq v_1 = in_{\mathsf{int}} \lfloor n \rfloor \wedge v_2' \sqsubseteq v_2 = in_{\mathsf{int}} \lfloor n \rfloor) \\
&(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\tau \mathsf{ref}\,)(\Xi \vec{R})(\Delta r) \iff \\
&\quad (v_1' = v_2' = \bot)\ \vee (- \vdash \tau : \mathit{type} \wedge \exists\, l : \mathit{Loc}.\ v_1' \sqsubseteq v_1 = in_{\mathsf{ref}}\, \lfloor l \rfloor \wedge v_2' \sqsubseteq v_2 = in_{\mathsf{ref}}\, \lfloor l \rfloor \wedge l \in \mathrm{dom}(\Delta) \wedge \Delta(l) = \tau) \\
&(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\tau_1 + \tau_2)(\Xi \vec{R})(\Delta r) \iff \\
&\quad (v_1' = v_2' = \bot)\ \vee (\Xi \vdash \tau_1 + \tau_2 \wedge v_1' \sqsubseteq v_1 \neq \bot \wedge v_2' \sqsubseteq v_2 \neq \bot \wedge \\
&\quad \exists d_1', d_2' \in V.\ ((v_1' = \bot \wedge d_1' = \bot) \vee v_1' = in_+(in_i(d_1')) \neq \bot) \wedge ((v_2' = \bot \wedge d_2' = \bot) \vee v_2' = in_+(in_i(d_2')) \neq \bot) \wedge \\
&\quad \exists d_1, d_2 : V\downarrow.\ v_1 = in_+(in_i(d_1)) \wedge v_2 = in_+(in_i(d_2)) \wedge \\
&\qquad (d_1', d_2', d_1, d_2) \in (\mathcal{S}(\tau_i)(\Xi \vec{R})(\Delta r) \wedge i \in \{1, 2\}) \\
&(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\tau_1 \times \tau_2)(\Xi \vec{R})(\Delta r) \iff \\
&\quad (v_1' = v_2' = \bot)\ \vee (\Xi \vdash \tau_1 \times \tau_2 \wedge v_1' \sqsubseteq v_1 \neq \bot \wedge v_2' \sqsubseteq v_2 \neq \bot \wedge \\
&\quad \exists d_{11}', d_{12}', d_{21}', d_{22}' : V. \\
&\qquad ((v_1' = \bot \wedge (d_{11}' = \bot \vee d_{12}' = \bot)) \vee v_1' = in_\times((d_{11}', d_{12}')) \neq \bot) \wedge \\
&\qquad ((v_2' = \bot \wedge (d_{21}' = \bot \vee d_{22}' = \bot)) \vee v_2' = in_\times((d_{21}', d_{22}')) \neq \bot) \wedge \\
&\quad \exists d_{11}, d_{12}, d_{21}, d_{22} : V\downarrow.\ v_1 = in_\times((d_{11}, d_{12})) \wedge v_2 = in_\times((d_{21}, d_{22})) \wedge \\
&\qquad (d_{11}', d_{21}', d_{11}, d_{21}) \in (\mathcal{S}(\tau_1)(\Xi \vec{R})(\Delta r) \wedge (d_{12}', d_{22}', d_{12}, d_{22}) \in (\mathcal{S}(\tau_2)(\Xi \vec{R})(\Delta r) \\
&(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\mu\alpha.\ \tau)(\Xi \vec{R})(\Delta r) \iff \\
&\quad (v_1' = v_2' = \bot)\ \vee (\Xi \vdash \mu\alpha.\ \tau \wedge v_1' \sqsubseteq v_1 \neq \bot \wedge v_2' \sqsubseteq v_2 \neq \bot \wedge \\
&\quad \exists d_1', d_2' : V.\ (v_1' = d_1' = \bot \vee v_1' = in_\mu(d_1') \neq \bot) \wedge (v_2' = d_2' = \bot \vee v_2' = in_\mu(d_2') \neq \bot) \wedge \\
&\quad \exists d_1, d_2 : V\downarrow.\ v_1 = in_\mu(d_1) \wedge v_2 = in_\mu(d_2) \wedge \\
&\qquad (d_1', d_2', d_1, d_2) \in \mathcal{S}(\tau[\mu\alpha.\ \tau/\alpha])(\Xi \vec{R})(\Delta r) \\
&(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\tau \to T\tau')(\Xi \vec{R})(\Delta r) \iff \\
&\quad (v_1' = v_2' = \bot)\ \vee\ (\Xi \vdash \tau \to T\tau' \wedge v_1' \sqsubseteq v_1 \neq \bot \wedge v_2' \sqsubseteq v_2 \neq \bot \wedge \\
&\quad \exists d_1', d_2', d_1, d_2 : V \multimap TV.\ v_1 = in_\to \lfloor d_1 \rfloor \wedge v_2 = in_\to \lfloor d_2 \rfloor \wedge \\
&\quad ((v_1' = \bot \wedge d_1' = \bot) \vee v_1' = in_\to \lfloor d_1' \rfloor) \wedge ((v_2' = \bot \wedge d_2' = \bot) \vee v_2' = in_\to \lfloor d_2' \rfloor) \wedge \\
&\qquad \forall \Delta' r' \rhd \Delta r.\ \forall v_{11}', v_{22}', v_{11}, v_{22} : V. \\
&\qquad\quad (v_{11}', v_{22}', v_{11}, v_{22}) \in \mathcal{R}(\tau)(\Xi \vec{R})(\Delta' r') \Rightarrow (d_1' v_{11}', d_2' v_{22}', d_1 v_{11}, d_2 v_{22}) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau')(\Xi \vec{R})(\Delta' r') \\
&(v_1', v_2', v_1, v_2) \in \Psi(\mathcal{R}, \mathcal{S})(\forall\alpha.\ T\tau)(\Xi \vec{R})(\Delta r) \iff \\
&\quad (v_1' = v_2' = \bot)\ \vee\ (\Xi \vdash \forall\alpha.\tau \wedge \alpha \notin \Xi \wedge v_1' \sqsubseteq v_1 \neq \bot \wedge v_2' \sqsubseteq v_2 \neq \bot \wedge \\
&\quad \exists d_1', d_2', d_1, d_2 : TV.\ v_1 = in_\forall \lfloor d_1 \rfloor \wedge v_2 = in_\forall \lfloor d_2 \rfloor \wedge \\
&\quad ((v_1' = \bot \wedge d_1' = \bot) \vee v_1' = in_\forall \lfloor d_1' \rfloor) \wedge ((v_2' = \bot \wedge d_2' = \bot) \vee v_2' = in_\forall \lfloor d_2' \rfloor) \wedge \\
&\qquad \forall R_\alpha : \mathrm{bParAdmRel}.\ (d_1', d_2', d_1, d_2) \in \Psi_T(\mathcal{R}, \mathcal{S})(T\tau)(\Xi\alpha\ \vec{R} R_\alpha)(\Delta r)
\end{aligned}
$$

Figure 1: Functions used in proof of existence of typed interpretation

$$\begin{array}{l}
(S'_1, S'_2, S_1, S_2) \in \nabla_S(\Delta r) \iff \exists r_1, \ldots r_n.\ r = \{r_1, \ldots r_n\} \wedge \\
\quad \big(S'_1 = S'_2 = \bot\big) \ \vee \big(S'_1 \sqsubseteq S_1 \neq \bot \wedge S'_2 \sqsubseteq S_2 \neq \bot \wedge \\
\quad \forall i \neq j, i, j \in \{1, \ldots, n\}.\ A_{ri1}(S_1) \cap A_{rj1}(S_1) = \emptyset \wedge A_{ri2}(S_2) \cap A_{rj2}(S_2) = \emptyset \wedge \\
\quad \mathrm{dom}(\Delta) \cap A_{r1}(S_1) = \emptyset \wedge \mathrm{dom}(\Delta) \cap A_{r2}(S_2) = \emptyset \wedge \\
\quad \forall l \in \mathrm{dom}(\Delta).\ (S'_1 l, S'_2 l, S_1 l, S_2 l) \in \llbracket \Delta(l) \rrbracket^t ()(\Delta r) \wedge \\
\quad \forall r_a \in r.\ \exists (P_b, LL_b) \in r_a.\ (S_1, S_2) \in \hat{p}_b \wedge \forall (l_1, l_2, \tau) \in LL_b.\ (S'_1 l_1, S_2 l_2, S_1 l_1, S_2 l_2) \in \llbracket \tau \rrbracket^t ()(\Delta r)\big)
\end{array}$$

$$\begin{array}{l}
(k'_1, k'_2, k_1, k_2) \in \llbracket \tau^\top \rrbracket^K (\Xi \vec{R})(\Delta r) \iff \\
\quad (k'_1 = k'_2 = \bot) \ \vee \ (\Xi \vdash \tau \wedge k'_1 \sqsubseteq k_1 \wedge k'_2 \sqsubseteq k_2 \wedge \\
\quad \forall \Delta^1 r^1 \rhd \Delta r.\ \forall S'_1, S'_2, S_1, S_2.\ \forall v'_1, v'_2, v_1, v_2. \\
\qquad \big((S'_1, S'_2, S_1, S_2) \in \nabla_S(\Delta^1 r^1) \wedge (v'_1, v'_2, v_1, v_2) \in \llbracket \tau \rrbracket^t (\Xi \vec{R})(\Delta^1 r^1) \Rightarrow \\
\qquad\quad (k'_1 S'_1 v'_1 = \top \Rightarrow k_2 S_2 v_2 = \top) \wedge (k'_2 S'_2 v'_2 = \top \Rightarrow k_1 S_1 v_1 = \top))
\end{array}$$

$$\begin{array}{l}
(m'_1, m'_2, m_1, m_2) \in \llbracket T\tau \rrbracket^T (\Xi \vec{R})(\Delta r) \iff \\
\quad (m'_1 = m'_2 = \bot) \ \vee (\Xi \vdash \tau \wedge m'_1 \sqsubseteq m_1 \wedge m'_2 \sqsubseteq m_2 \wedge \\
\quad \forall \Delta^1 r^1 \rhd \Delta r.\ \forall k'_1, k'_2, k_1, k_2.\ \forall S'_1, S'_2, S_1, S_2. \\
\qquad \big((k'_1, k'_2, k_1, k_2) \in \llbracket \tau^\top \rrbracket^K (\Xi \vec{R})(\Delta^1 r^1) \wedge (S'_1, S'_2, S_1, S_2) \in \nabla_S(\Delta^1 r^1)\big) \Rightarrow \\
\qquad\quad (m'_1 k'_1 S'_1 = \top \Rightarrow m_2 k_2 S_2 = \top) \wedge (m'_2 k'_2 S'_2 = \top \Rightarrow m_1 k_1 S_1 = \top)
\end{array}$$

$$\begin{array}{l}
(v'_1, v'_2, v_1, v_2) \in \llbracket \alpha_i \rrbracket^t (\Xi \vec{R})(\Delta r) \iff (v'_1 = v'_2 = \bot) \ \vee \ (\alpha_i \in \Xi \wedge (v'_1, v'_2, v_1, v_2) \in R_i(\Delta r) \\
(v'_1, v'_2, v_1, v_2) \in \llbracket 1 \rrbracket^t (\Xi \vec{R})(\Delta r) \iff (v'_1 = v'_2 = \bot) \ \vee (v'_1 \sqsubseteq v_1 = in_1 \lfloor * \rfloor \wedge v'_2 \sqsubseteq v_2 = in_1 \lfloor * \rfloor) \\
(v'_1, v'_2, v_1, v_2) \in \llbracket \mathsf{int} \rrbracket^t (\Xi \vec{R})(\Delta r) \iff (v'_1 = v'_2 = \bot) \ \vee (\exists n : Z.\ v'_1 \sqsubseteq v_1 = in_{\mathsf{int}} \lfloor n \rfloor \wedge v'_2 \sqsubseteq v_2 = in_{\mathsf{int}} \lfloor n \rfloor) \\
(v'_1, v'_2, v_1, v_2) \in \llbracket \tau \mathsf{ref} \rrbracket^t (\Xi \vec{R})(\Delta r) \iff \\
\quad (v'_1 = v'_2 = \bot) \ \vee (- \vdash \tau : type \wedge \exists\, l : Loc.\ v'_1 \sqsubseteq v_1 = in_{\mathsf{ref}} \lfloor l \rfloor \wedge v'_2 \sqsubseteq v_2 = in_{\mathsf{ref}} \lfloor l \rfloor \wedge l \in \mathrm{dom}(\Delta) \wedge \Delta(l) = \tau) \\
(v'_1, v'_2, v_1, v_2) \in \llbracket \tau_1 + \tau_2 \rrbracket^t (\Xi \vec{R})(\Delta r) \iff \\
\quad (v'_1 = v'_2 = \bot) \ \vee (\Xi \vdash \tau_1 + \tau_2 \wedge v'_1 \sqsubseteq v_1 \neq \bot \wedge v'_2 \sqsubseteq v_2 \neq \bot \wedge \\
\quad \exists d'_1, d'_2 \in V.\ ((v'_1 = d'_1 = \bot) \vee v'_1 = in_+(in_i(d'_1)) \neq \bot) \wedge ((v'_2 = d'_2 = \bot) \vee v'_2 = in_+(in_i(d'_2)) \neq \bot) \wedge \\
\quad \exists d_1, d_2 : V \downarrow .\ v_1 = in_+(in_i(d_1)) \wedge v_2 = in_+(in_i(d_2)) \wedge \\
\qquad (d'_1, d'_2, d_1, d_2) \in \llbracket \tau_i \rrbracket^t (\Xi \vec{R})(\Delta r) \wedge i \in \{1, 2\}) \\
(v'_1, v'_2, v_1, v_2) \in \llbracket \tau_1 \times \tau_2 \rrbracket^t (\Xi \vec{R})(\Delta r) \iff \\
\quad (v'_1 = v'_2 = \bot) \ \vee (\Xi \vdash \tau_1 \times \tau_2 \wedge v'_1 \sqsubseteq v_1 \neq \bot \wedge v'_2 \sqsubseteq v_2 \neq \bot \wedge \\
\quad \exists d'_{11}, d'_{12}, d'_{21}, d'_{22} : V. \\
\qquad ((v'_1 = \bot \wedge (d'_{11} = \bot \vee d'_{12} = \bot)) \vee v'_1 = in_\times((d'_{11}, d'_{12})) \neq \bot) \wedge \\
\qquad ((v'_2 = \bot \wedge (d'_{21} = \bot \vee d'_{22} = \bot)) \vee v'_2 = in_\times((d'_{21}, d'_{22})) \neq \bot) \wedge \\
\quad \exists d_{11}, d_{12}, d_{21}, d_{22} : V \downarrow .\ v_1 = in_\times((d_{11}, d_{12})) \wedge v_2 = in_\times((d_{21}, d_{22})) \wedge \\
\qquad (d'_{11}, d'_{21}, d_{11}, d_{21}) \in \llbracket \tau_1 \rrbracket^t (\Xi \vec{R})(\Delta r) \wedge (d'_{12}, d'_{22}, d_{12}, d_{22}) \in \llbracket \tau_2 \rrbracket^t (\Xi \vec{R})(\Delta r) \\
(v'_1, v'_2, v_1, v_2) \in \llbracket \mu\alpha.\ \tau \rrbracket^t (\Xi \vec{R})(\Delta r) \iff \\
\quad (v'_1 = v'_2 = \bot) \ \vee (\Xi \vdash \mu\alpha.\ \tau \wedge v'_1 \sqsubseteq v_1 \neq \bot \wedge v'_2 \sqsubseteq v_2 \neq \bot \ \wedge \\
\quad \exists d'_1, d'_2 : V.\ (v'_1 = d'_1 = \bot \vee v'_1 = in_\mu(d'_1) \neq \bot) \wedge (v'_2 = d'_2 = \bot \vee v'_2 = in_\mu(d'_2) \neq \bot) \wedge \\
\quad \exists d_1, d_2 : V \downarrow .\ v_1 = in_\mu(d_1) \wedge v_2 = in_\mu(d_2) \wedge \\
\qquad (d'_1, d'_2, d_1, d_2) \in \llbracket \tau[\mu\alpha.\ \tau/\alpha] \rrbracket^t (\Xi \vec{R})(\Delta r) \\
(v'_1, v'_2, v_1, v_2) \in \llbracket \tau \to T\tau' \rrbracket^t (\Xi \vec{R})(\Delta r) \iff \\
\quad (v'_1 = v'_2 = \bot) \ \vee \ (\Xi \vdash \tau \to T\tau' \wedge v'_1 \sqsubseteq v_1 \neq \bot \wedge v'_2 \sqsubseteq v_2 \neq \bot \ \wedge \\
\quad \exists d'_1, d'_2, d_1, d_2 : V \multimap TV.\ v_1 = in_\to \lfloor d_1 \rfloor \wedge v_2 = in_\to \lfloor d_2 \rfloor \wedge \\
\quad ((v'_1 = \bot \wedge d'_1 = \bot) \vee v'_1 = in_\to \lfloor d'_1 \rfloor) \wedge ((v'_2 = \bot \wedge d'_2 = \bot) \vee v'_2 = in_\to \lfloor d'_2 \rfloor) \wedge \\
\qquad \forall \Delta^1 r^1 \rhd \Delta r.\ \forall v'_{11}, v'_{22}, v_{11}, v_{22} : V. \\
\qquad\quad (v'_{11}, v'_{22}, v_{11}, v_{22}) \in \llbracket \tau \rrbracket^t (\Xi \vec{R})(\Delta^1 r^1) \Rightarrow (d'_1 v'_{11}, d'_2 v'_{22}, d_1 v_{11}, d_2 v_{22}) \in \llbracket T\tau' \rrbracket^T (\Xi \vec{R})(\Delta^1 r^1) \\
(v'_1, v'_2, v_1, v_2) \in \llbracket \Xi \vdash \forall\alpha.\ T\tau \rrbracket^t (\Xi \vec{R})(\Delta r) \iff \\
\quad (v'_1 = v'_2 = \bot) \ \vee \ (\Xi \vdash \forall\alpha.\ T\tau \wedge \alpha \notin \Xi \wedge v'_1 \sqsubseteq v_1 \neq \bot \wedge v'_2 \sqsubseteq v_2 \neq \bot \ \wedge \\
\quad \exists d'_1, d'_2 : TV.\ ((v'_1 = \bot \wedge d'_1 = \bot) \vee v'_1 = in_\forall \lfloor d'_1 \rfloor) \wedge ((v'_2 = \bot \wedge d'_2 = \bot) \vee v'_2 = in_\forall \lfloor d'_2 \rfloor) \wedge \\
\quad \exists d_1, d_2 : TV.\ v_1 = in_\forall \lfloor d_1 \rfloor \wedge v_2 = in_\forall \lfloor d_2 \rfloor \wedge \\
\qquad \forall R_\alpha : \mathrm{bParAdmRel}.\ (d'_1, d'_2, d_1, d_2) \in \llbracket T\tau \rrbracket^T (\Xi, \alpha\ \vec{R} R_\alpha)(\Delta r)
\end{array}$$

Figure 2: Interpretation of Types

Proof: It suffices to show that id_V is in the set

$$\begin{array}{l} E = \{\, e : V \multimap V \mid \forall \Xi, \alpha \notin \Xi. \\ \forall \Xi \vdash \sigma.\ \forall \vec{R} \in \mathrm{bParAdmRel}^{|\Xi|}.\ \forall \Xi, \alpha \vdash \tau.\ \forall \Delta r. \\ \forall (v_1', v_2', v_1, v_2), (w_1', w_2', w_1, w_2). \\ \quad (v_1', v_2', v_1, v_2) \in [\![\tau]\!]^t (\Xi\alpha\ \vec{R}([\![\sigma]\!]^t(\Xi\vec{R})))(\Delta r) \Rightarrow \\ \qquad (ev_1', ev_2', v_1, v_2) \in [\![\tau[\sigma/\alpha]]\!]^t(\Xi\vec{R})(\Delta r) \\ \quad \wedge \\ \quad (w_1', w_2', w_1, w_2) \in [\![\tau[\sigma/\alpha]]\!]^t(\Xi\vec{R})(\Delta r) \Rightarrow \\ \qquad (ew_1', ew_2', w_1, w_2) \in [\![\tau]\!]^t(\Xi\alpha\ \vec{R}\ ([\![\sigma]\!]^t(\Xi\vec{R}))(\Delta r)\} \end{array}$$

We prove that by fixed-point induction. Clearly, $\perp_V$ is in E as the relations involved are admissible. Hence it suffices to show that E is closed under $\delta : (V \multimap V) \to (V \multimap V)$, whose least fixed point by minimal invariance is id_V. This is done by case analysis over the structure of τ and makes use of the fact that $(i, i) : \Psi(\nabla, \nabla) \subset \nabla$ and $(i^{-1}, i^{-1}) : \nabla \subset \Psi(\nabla, \nabla)$. Recall the notation $\nabla(\tau) = [\![\tau]\!]^t$. Let

$$\begin{array}{l} G = \{\, g : F(V, V) \multimap F(V, V) \mid \forall \Xi, \alpha \notin \Xi. \\ \forall \Xi \vdash \sigma.\ \forall \vec{R} \in \mathrm{bParAdmRel}^{|\Xi|}.\ \forall \Xi, \alpha \vdash \tau.\ \forall \Delta r. \\ \forall (v_1', v_2', v_1, v_2), (w_1', w_2', w_1, w_2). \\ \quad (v_1', v_2', v_1, v_2) \in \Psi(\nabla, \nabla)(\tau)(\Xi\alpha\ \vec{R}([\![\sigma]\!]^t(\Xi\vec{R}))(\Delta r) \Rightarrow \\ \quad (gv_1', gv_2', v_1, v_2) \in \Psi(\nabla, \nabla)(\tau[\sigma/\alpha])(\Xi\vec{R})(\Delta r) \\ \quad \wedge \\ \quad (w_1', w_2', w_1, w_2) \in \Psi(\nabla, \nabla)(\tau[\sigma/\alpha])(\Xi\vec{R})(\Delta r) \Rightarrow \\ \quad (gw_1', gw_2', w_1, w_2) \in \Psi(\nabla, \nabla)(\tau)(\Xi\alpha\ \vec{R}\ ([\![\sigma]\!]^t(\Xi\vec{R}))(\Delta r)\} \end{array}$$

So it suffices to show that $e \in E \Rightarrow F(e, e) \in G$. We also show that the relations are carried over by Ψ_S, Ψ_K, Ψ_M. ■

Assume $e \in E$. To show $\delta(e) \in E$ or $F(e, e) \in G$.
It holds that $e \sqsubseteq id_V \Rightarrow \Psi_S(e, e) \sqsubseteq id_S \wedge \Psi_K(e, e) \sqsubseteq id_{KV} \wedge \Psi_T(e, e) \sqsubseteq id_{TV} \wedge F(e, e) \sqsubseteq id_{F(V,V)} \wedge \delta(e) \sqsubseteq id_V$.

If $v_1' = v_2' = \perp$ then $\delta(e)v_1' = \delta(e)v_2' = \perp$.

We often omit the isomorphism i, i^{-1} in the proof below.
S) Assume $(S_1', S_2', S_1, S_2) \in \Psi_S(\nabla, \nabla)(\Delta r)$.
To show $(\Psi_S(e, e)S_1',\ \Psi_S(e, e)S_2',\ S_1,\ S_2) \in \Psi_S(\nabla, \nabla)(\Delta r)$.
This follows by downwards closure of $\Psi_S(\nabla, \nabla)$.

1_K) Assume $(k_1', k_2', k_1, k_2) \in \Psi_K(\nabla, \nabla)(\tau^\top)(\Xi\alpha\ \ \vec{R}([\![\sigma]\!]^t(\Xi\vec{R}))(\Delta r)$.
To show $(\Psi_K(e, e)k_1',\ \Psi_K(e, e)k_2',\ k_1,\ k_2) \in \Psi_K(\nabla, \nabla)(\tau[\sigma/\alpha]^\top)(\Xi\vec{R})(\Delta r)$. If $k_1' = k_2' = \perp$ then $\Psi_K(e, e)k_1' = \Psi_K(e, e)k_2' = \perp$ and we are done. Else we reason as follows. We must show $((\lambda s.\lambda v.k_1'(\Psi_S(e, e)s)(ev)), (\lambda s.\lambda v.k_2'(\Psi_S(e, e)s)(ev)), k_1, k_2) \in \Psi_K(\nabla, \nabla)(\tau[\sigma/\alpha]^\top)(\Xi\vec{R})(\Delta r)$. Let $\Delta^1 r^1 \rhd \Delta r$. Assume $(S_1', S_2', S_1, S_2) \in \Psi_S(\nabla, \nabla)(\Delta^1 r^1)$ and $(v_1', v_2', v_1, v_2) \in ([\![\tau[\sigma/\alpha]]\!]^t(\Xi\vec{R})(\Delta^1 r^1)$. Since by assumption $e \in E$ and by S) it follows that $((\Psi_S(e, e)S_1'), (\Psi_S(e, e)S_2'), S_1, S_2) \in \Psi_S(\nabla, \nabla)$ $(\Delta^1 r^1)$ and $(ev_1', ev_2', v_1, v_2) \in ([\![\tau]\!]^t(\Xi\alpha\ \vec{R}([\![\sigma]\!]^t(\Xi\vec{R}))(\Delta^1 r^1)$. Then it follows from the assumptions that
$(\Psi_K(e, e)k_1')S_1'v_1' = k_1'(\Psi_S(e, e)S_1')(ev_1') = \top \Longrightarrow k_2 S_2 v_2 = \top$ and
$(\Psi_K(e, e)k_2')S_2'v_2' = k_2'(\Psi_S(e, e)S_2')(ev_2') = \top \Longrightarrow k_1 S_1 v_1 = \top$.
We conclude $(\Psi_K(e, e)k_1', (\Psi_K(e, e)k_2'), k_1, k_2) \in \Psi_K(\nabla, \nabla)(\tau[\sigma/\alpha]^\top)(\Xi\vec{R})(\Delta r)$.

2_K) Assume $(k'_1, k'_2, k_1, k_2) \in \Psi(\nabla, \nabla)(\tau[\sigma/\alpha]^\top)(\Xi\vec{R})(\Delta r)$, $\alpha \notin \Xi \wedge \Xi \vdash \sigma$. To show $(\Psi_K(e,e)k'_1,$ $\Psi_K(e,e)k'_2, k_1,\ k_2) \in \Psi_K(\nabla,\nabla)(\tau^\top)(\Xi\alpha\ \vec{R}([\![\sigma]\!]^t(\Xi\vec{R}))(\Delta r)$. If $k'_1 = k'_2 = \bot$ then $\Psi_K(e,e)k'_1 = \Psi_K(e,e)k'_2 = \bot$ and we are done. Else we reason as follows. We must show $((\lambda s.\lambda v.k'_1(\Psi_S(e,e)s)(ev)),$ $(\lambda s.\lambda v.k'_2(\Psi_S(e,e)s)(ev)), k_1, k_2,) \in \Psi_K(\nabla,\nabla)(\tau^\top)(\Xi\alpha\ \vec{R}([\![\sigma]\!]^t(\Xi\vec{R}))(\Delta r)$.
Let $\Delta^1 r^1 \rhd \Delta r$ and assume $(S'_1, S'_2 S_1, S_2) \in \nabla_S(\Delta^1 r^1)$ and $(v'_1, v'_2, v_1, v_2) \in ([\![\tau]\!]^t(\Xi\alpha\ \vec{R}([\![\sigma]\!]^t(\Xi\vec{R})$ $)(\Delta^1 r^1)$. Since by assumption $e \in E$ and by S) it follows that $(\Psi_S(e,e)S'_1, \Psi_S(e,e)S'_2, S_1, S_2) \in$ $(\nabla_S(\Delta^1 r^1)$ and $(ev'_1, ev'_2,\ v_1, v_2) \in ([\![\tau[\sigma/\alpha]]\!]^t(\Xi\vec{R})(\Delta^1 r^1)$. Then it follows from the assumptions on k'_1, k'_2, k_1, k_2 that $k'_1(\Psi_S(e,e)S'_1)(ev'_1) = \top \Rightarrow k_2 s_2 w_2 = \top$ and $k'_2(\Psi_S(e,e)S'_2)(ev'_2) = \top \Rightarrow$ $k_1 s_1 w_1 = \top$. We conclude $((\Psi_K(e,e)k'_1), (\Psi_K(e,e)k'_2), k_1, k_2) \in ([\![\tau^\top]\!]^K(\Xi\alpha\ \vec{R}([\![\sigma]\!]^t(\Xi\vec{R}))(\Delta r)$.

1_M) Assume $(m'_1, m'_2, m_1, m_2) \in ([\![T\tau]\!]^T(\Xi\alpha\ \ \vec{R}([\![\sigma]\!]^t(\Xi\vec{R})))(\Delta r)$.
To show $((\Psi_T(e,e)m'_1),\ (\Psi_T(e,e)m'_2),\ m_1,\ m_2) \in ([\![T\tau[\sigma/\alpha]]\!]^T(\Xi\vec{R})(\Delta r)$. If $m'_1 = m'_2 = \bot$ then $\Psi_T(e,e)m'_1 = \Psi_T(e,e)m'_2 = \bot$ and we are done. Else we reason as follows. We must show $((\lambda k.\lambda s.m'_1(\Psi_K(e,e)k)(\Psi_S(e,e)s)),\ (\lambda s.\lambda v.m'_2(\Psi_K(e,e)k)(\Psi_S(e,e)s),\ m_1,\ m_2) \in ([\![T\tau[\sigma/\alpha]]\!]^T$ $(\Xi\vec{R})(\Delta r))$. Let $\Delta^1 r^1 \rhd \Delta r$ Assume $(S'_1, S'_2, S_1, S_2) \in (\nabla_S(\Delta^1 r^1)$ and $(k'_1, k'_2, k_1, k_2) \in ([\![\tau[\sigma/\alpha]^\top]\!]^K$ $(\Xi\vec{R})(\Delta^1 r^1)$. Since by assumption $e \in E$ and by S) and 2_K) it follows that $(\Psi_S(e,e)S'_1, \Psi_S(e,e)S'_2,$ $S_1, S_2) \in (\nabla_S(\Delta^1 r^1)$ and $(\Psi_K(e,e)k'_1, \Psi_K(e,e)k'_2, k_1, k_2) \in [\![\tau^\top]\!]^K(\Xi\alpha\ \vec{R}([\![\sigma]\!]^t(\Xi\vec{R})))(\Delta^1 r^1)$. Then it follows from the assumptions on m'_1, m'_2, m_1, m_2 that
$(\Psi_T(e,e)m'_1)S'_1 v'_1 = m'_1(\Psi_K(e,e)k'_1)(\Psi_S(e,e)S'_1) = \top \Longrightarrow k_2 S_2 v_2 = \top$ and
$(\Psi_T(e,e)m'_2)S'_2 v'_2 = m'_2(\Psi_K(e,e)k'_2)(\Psi_S(e,e)S'_1) = \top \Longrightarrow k_1 S_1 v_1 = \top$.
We conclude $((\Psi_T(e,e)m'_1), (\Psi_T(e,e)m'_2), m_1, m_2)([\![T\tau[\sigma/\alpha]]\!]^T(\Xi\vec{R})(\Delta r)$.

2_M) Assume $(m'_1,\ m'_2,\ m_1,\ m_2) \in ([\![T\tau[\sigma/\alpha]]\!]^T(\Xi\vec{R})(\Delta r) \wedge \alpha \notin \Xi \wedge \Xi \vdash \sigma$. To show $((\Psi_T(e,e)m'_1),$ $(\Psi_T(e,e)m'_2),\ m_1, m_2) \in ([\![T\tau]\!]^T(\Xi\alpha\ \vec{R}([\![\sigma]\!]^t(\Xi\vec{R})))(\Delta r)$. If $m'_1 = m'_2 = \bot$ then $\Psi_T(e,e)m'_1 = \Psi_T(e,e)m'_2 = \bot$ and we are done. Else we reason as follows. We must show
$((\lambda k.\lambda S.m'_1(\Psi_K(e,e)k)(\Psi_S(e,e)s)),\ (\lambda k.\lambda s.m'_2(\Psi_K(e,e)k)(\Psi_S(e,e)s)),\ m_1,\ m_2) \in ([\![T\tau]\!]^T(\Xi\alpha$ $\vec{R}([\![\sigma]\!]^t(\Xi\vec{R})))(\Delta r)$. Let $\Delta^1 r^1 \rhd \Delta r$, and assume $(k'_1, k'_2, k_1, k_2) \in ([\![\tau^\top]\!]^K(\Xi\alpha\ \vec{R}\vec{R}'([\![\sigma]\!]^t(\Xi\vec{R})))(\Delta^1 r^1)$ and $(S'_1, S'_2, S_1, S_2) \in (\nabla_S(\Delta^1 r^1)$.
Since by assumption $e \in E$ and by S) and 1_K) it follows that $((\Psi_S(e,e)S'_1),\ (\Psi_S(e,e)S'_2), S_1, S_2) \in$ $(\nabla_S(\Delta^1 r^1)$ and $((\Psi_K(e,e)k'_1),\ (\Psi_K(e,e)k'_2), k_1, k_2) \in ([\![\tau[\sigma/\alpha]^\top]\!]^K\ (\Xi\vec{R})(\Delta^1 r^1)$. Then it follows from the assumptions on m'_1, m'_2, m_1, m_2 that $m'_1(\Psi_K(e,e)k'_1)(\Psi_S(e,e)S'_1) = \top \Rightarrow k_2 s_2 w_2 = \top$ and $m'_2(\Psi_K(e,e)k'_2)(\Psi_S(e,e)S'_2) = \top \Rightarrow k_1 s_1 w_1 = \top$.
We conclude $((\Psi_T(e,e)m'_1), (\Psi_T(e,e)m'_2), m_1, m_2) \in ([\![T\tau]\!]^T(\Xi\alpha\ \vec{R}([\![\sigma]\!]^t(\Xi\vec{R}))(\Delta r)$.

V) The $v'_1 = v'_2 = \bot$ cases are imidiate. Assume in all cases in the following proof that $(v'_1 \neq \bot \vee v'_2 \neq \bot)$.
• Assume $(v'_1, v'_2, v_1, v_2) \in ([\![\alpha_i]\!]^t(\Xi\alpha\ \vec{R}([\![\sigma]\!]^t(\Xi\vec{R}))(\Delta r) \wedge \alpha_i \neq \alpha$. Then $\alpha_i[\sigma/\alpha] = \alpha_i$ and $(v'_1, v'_2,$ $v_1, v_2) \in (R_i(\Delta r)$. By downwards closure of R_i it follows that $(F(e,e)v'_1, F(e,e)v'_2, v_1,\ v_2) \in R_i(\Delta r)$. So $(F(e,e)v'_1,\ F(e,e)v'_2, v_1, v_2) \in ([\![\alpha_i]\!]^t(\Xi\vec{R})(\Delta r)$ and $(F(e,e)v'_1, F(e,e)v'_2, v_1, v_2) \in ([\![\alpha_i[\sigma/\alpha]]\!]^t$ $(\Xi\vec{R})(\Delta r)$.

Assume $(v'_1, v'_2, v_1, v_2) \in ([\![\alpha_i[\sigma/\alpha]]\!]^t(\Xi\vec{R})(\Delta r) \wedge \alpha \notin \Xi \wedge \alpha_i \neq \alpha$. Then $\alpha_i[\sigma/\alpha] = \alpha_i$ and $(v'_1, v'_2, v_1, v_2) \in (R_i(\Delta r)$. By downwards closure of R_i it follows that $(F(e,e)v'_1, F(e,e)v'_2, v_1,\ v_2) \in$ $(R_i(\Delta r)$. So $(F(e,e)v'_1,\ F(e,e)v'_2, v_1, v_2) \in ([\![\alpha_i]\!]^t(\Xi\alpha\ \ \vec{R}\ ([\![\Xi \vdash \sigma]\!]^t(\vec{R}))(\Delta r)$.

Assume $(v'_1, v'_2, v_1, v_2) \in ([\![\alpha]\!]^t(\Xi\alpha\ \vec{R}\ ([\![\sigma]\!]^t(\Xi\vec{R}))(\Delta r)$. Then $(v'_1, v'_2, v_1, v_2) \in ([\![\sigma]\!]^t(\Xi\vec{R})(\Delta r)$. To show $(F(e,e)v'_1,\ F(e,e)v'_2, v_1, v_2) \in ([\![\alpha[\sigma/\alpha]]\!]^t(\Xi\vec{R})(\Delta r)$, that is $(F(e,e)v'_1,\ F(e,e)v'_2, v_1, v_2) \in$ $([\![\sigma]\!]^t(\Xi\vec{R})(\Delta r)$. This follows by downwards closure.

Assume $\alpha \notin \Xi$ and $(v_1', v_2', v_1, v_2) \in (\llbracket \alpha[\sigma/\alpha] \rrbracket^t (\Xi \vec{R})(\Delta r)$, that is $(v_1', v_2', v_1, v_2) \in (\llbracket \sigma \rrbracket^t (\Xi \vec{R})(\Delta r)$. To show $(F(e,e)v_1',\ F(e,e)v_2', v_1, v_2) \in (\llbracket \alpha \rrbracket^t (\Xi\alpha\ \vec{R}\ (\llbracket\sigma\rrbracket^t(\Xi\vec{R})))(\Delta r)$. This holds when $(F(e,e)v_1',\ F(e,e)v_2', v_1, v_2) \in (\llbracket\sigma\rrbracket^t(\Xi\vec{R})(\Delta r)$. This follows by downwards closure.

• Assume $(v_1', v_2', v_1, v_2) \in \llbracket \tau \to T\tau' \rrbracket^t (\Xi\alpha\ \vec{R}(\llbracket\sigma\rrbracket^t(\Xi\vec{R})))(\Delta r)$. This implies $\alpha \notin \Xi$ and $v_1' \sqsubseteq v_1 \wedge v_2' \sqsubseteq v_2 \wedge \exists d_1', d_2', d_1, d_2 : V \multimap TV.\ ((v_1' = \bot \wedge d_1' = \bot) \vee v_1' = in_{\to}\lfloor d_1' \rfloor) \wedge ((v_2' = \bot \wedge d_2' = \bot) \vee v_2' = in_{\to}\lfloor d_2' \rfloor) \wedge v_1 = in_{\to}\lfloor d_1 \rfloor \wedge v_2 = in_{\to}\lfloor d_2 \rfloor \wedge \forall \Delta^1 r^1 \rhd \Delta r.\ \forall w_{11}', w_{22}', w_{11}, w_{22} : V.\ (w_{11}', w_{22}', w_{11}, w_{22}) \in \llbracket \tau \rrbracket^t (\Xi\alpha\ \vec{R}(\llbracket\sigma\rrbracket^t(\Xi\vec{R})))(\Delta^1 r^1) \Rightarrow (d_1'w_{11}', d_2'w_{22}', d_1 w_{11}, d_2 w_{22}) \in (\llbracket T\tau' \rrbracket^T (\Xi\alpha\ \vec{R}(\llbracket\sigma\rrbracket^t(\Xi\vec{R})))(\Delta^1 r^1)$. To prove $(F(e,e)v_1', F(e,\ e)v_2', v_1, v_2) \in \llbracket (\tau \to T\tau')[\sigma/\alpha] \rrbracket^t\ (\Xi\vec{R})(\Delta r)$. $F(e,e)v_1' \sqsubseteq v_1' \sqsubseteq v_1 \wedge F(e,e)v_2' \sqsubseteq v_2' \sqsubseteq v_2$ follows from $F(e,e) \sqsubseteq id_{F(V,V)} \wedge v_1' \sqsubseteq v_1 \wedge v_2' \sqsubseteq v_2$. $F(e,e)\lfloor d_1' \rfloor = in_{\to}\lfloor \lambda w.\Psi_T(e,e)(d_1'(ew)) \rfloor$ and $F(e,e)\lfloor d_2' \rfloor = in_{\to}\lfloor \lambda w.\Psi_T(e,e)(d_2'(ew)) \rfloor$ and $F(e,e)\bot = \bot$. Let $\Delta^1 r^1 \rhd \Delta r$ and suppose $(w_{11}',\ w_{22}', w_{11}, w_{22}) \in (\llbracket \tau[\sigma/\alpha] \rrbracket^t\ (\Xi\vec{R})(\Delta^1 r^1)$. By assumptions on e then $(ew_{11}', ew_{22}', w_{11}, w_{22}) \in (\llbracket \tau \rrbracket^t (\Xi\alpha\ \vec{R}(\llbracket\sigma\rrbracket^t(\Xi\vec{R})))(\Delta^1 r^1)$. And so by assumptions on v_1', v_2', v_1, v_2 it holds that $(d_1'w_{11}',\ d_2'w_{22}', d_1 w_{11}, d_2 w_{22}) \in (\llbracket T\tau' \rrbracket^T (\Xi\alpha\ \vec{R}(\llbracket\sigma\rrbracket^t(\Xi\vec{R})))(\Delta^1 r^1)$. By assumptions on e together with 1_M) then $(\Psi_T(e,e)(d_1'w_{11}'),\ \Psi_T(e,e)(d_2'w_{22}'), d_1 w_{11}, d_2 w_{22}) \in (\llbracket \tau'[\sigma/\alpha] \rrbracket^T (\Xi\vec{R})\ (\Delta^1 r^1)$. We conclude $(F(e,e)v_1', F(e,e)v_2', v_1, v_2) \in \llbracket (\tau \to T\tau')[\sigma/\alpha] \rrbracket^t\ (\Xi\vec{R})(\Delta r)$.

Assume $(v_1', v_2', v_1, v_2) \in \llbracket (\tau \to T\tau')[\sigma/\alpha] \rrbracket^t (\Xi\vec{R})(\Delta r)$, $\alpha \notin \Xi$ and $\Xi \vdash \sigma$. This implies $v_1' \sqsubseteq v_1 \wedge v_2' \sqsubseteq v_2 \wedge \exists d_1', d_2', d_1, d_2 : V \multimap TV.\ ((v_1' = \bot \wedge d_1' = \bot) \vee v_1' = in_{\to}\lfloor d_1' \rfloor) \wedge ((v_2' = \bot \wedge d_2' = \bot) \vee v_2' = in_{\to}\lfloor d_2' \rfloor) \wedge v_1 = in_{\to}\lfloor d_1 \rfloor \wedge v_2 = in_{\to}\lfloor d_2 \rfloor \wedge \forall \Delta^1 r^1 \rhd \Delta r.\ \forall w_{11}', w_{22}', w_{11}, w_{22} : V.\ (w_{11}', w_{22}', w_{11}, w_{22}) \in (\llbracket \tau[\sigma/\alpha] \rrbracket^t (\Xi\vec{R})(\Delta^1 r^1) \Rightarrow (d_1'w_{11}', d_2'w_{22}', d_1 w_{11}, d_2 w_{22}) \in (\llbracket T\tau'[\sigma/\alpha] \rrbracket^T (\Xi\vec{R})(\Delta^1 r^1)$. To prove $(F(e,e)v_1', F(e,e)v_2', v_1, v_2) \in \llbracket \tau \to T\tau' \rrbracket^t)(\Xi\alpha\ \ \vec{R}(\llbracket\sigma\rrbracket^t(\Xi\vec{R})))(\Delta r)$.
$F(e,e)v_1' \sqsubseteq v_1' \sqsubseteq v_1 \wedge F(e,e)v_2' \sqsubseteq v_2' \sqsubseteq v_2$ follows from $F(e,e) \sqsubseteq id_{F(V,V)} \wedge v_1' \sqsubseteq v_1 \wedge v_2' \sqsubseteq v_2$. $F(e,e)v_1' \sqsubseteq in_{\to}\lfloor \lambda w.\Psi_T(e,e)(d_1'(ew)) \rfloor \wedge F(f,g)v_2' \sqsubseteq in_{\to}\lfloor \lambda w.\Psi_T(e,e)(d_2'(ew)) \rfloor$. Let $\Delta^1 r^1 \rhd \Delta r$ and suppose $(w_{11}',\ w_{22}', w_{11}, w_{22}) \in (\llbracket \tau \rrbracket^t (\Xi\alpha\ \vec{R}(\llbracket\sigma\rrbracket^t(\Xi\vec{R})))(\Delta^1 r^1)$. By assumptions on e then $(ew_{11}', ew_{22}', w_{11}, w_{22}) \in (\llbracket \tau[\sigma/\alpha] \rrbracket^t (\Xi\vec{R})(\Delta^1 r^1)$. And so by assumptions on $v_1', v_2', v_1,\ v_2$ it holds that $(d_1'w_{11}', d_2'w_{22}',\ d_1 w_{11}, d_2 w_{22}) \in (\llbracket T\tau'[\sigma/\alpha] \rrbracket^T (\Xi\vec{R})(\Delta^1 r^1)$. By assumptions on e together with 2_M) then $(\Psi_T(e,e)(d_1'w_{11}'), \Psi_T(e,e)(d_2'w_{22}'),\ d_1 w_{11}, d_2 w_{22}) \in (\llbracket T\tau' \rrbracket^T (\Xi, \alpha\ \vec{R}(\llbracket\sigma\rrbracket^t(\Xi\vec{R})))(\Delta^1 r^1)$. We conclude $(F(e,e)v_1', F(e,e)v_2', v_1, v_2) \in \llbracket (\tau \to T\tau') \rrbracket^t)(\Xi\alpha\ \vec{R}\ (\llbracket\sigma\rrbracket^t(\Xi\vec{R})))(\Delta r)$.

• Assume $(v_1', v_2', v_1, v_2) \in \llbracket \forall\beta.\ T\tau \rrbracket^t (\Xi\alpha\ \vec{R}(\llbracket\sigma\rrbracket^t(\Xi\vec{R})))(\Delta r)$. This implies $\alpha \notin \Xi$, $\beta \notin (\Xi, \alpha)$ and $v_1' \sqsubseteq v_1 \wedge v_2' \sqsubseteq v_2 \wedge \exists d_1', d_2', d_1, d_2 : TV.\ ((v_1' = \bot \wedge d_1' = \bot) \vee v_1' = in_{\forall}\lfloor d_1' \rfloor) \wedge ((v_2' = \bot \wedge d_2' = \bot) \vee v_2' = in_{\forall}\lfloor d_2' \rfloor) \wedge v_1 = in_{\forall}\lfloor d_1 \rfloor \wedge v_2 = in_{\forall}\lfloor d_2 \rfloor \wedge \forall R_\beta :$ bParAdmRel. $(d_1', d_2', d_1, d_2) \in (\llbracket T\tau \rrbracket^T (\Xi\alpha\beta\ \vec{R}(\llbracket\sigma\rrbracket^t(\Xi\vec{R}))R_\beta)(\Delta r)$. To prove $(F(e,e)v_1',\ F(e,e)v_2', v_1, v_2) \in \llbracket \forall\beta.\ T\tau[\sigma/\alpha] \rrbracket^t)(\Xi\vec{R})$ (Δr). $F(e,e)v_1' \sqsubseteq v_1' \sqsubseteq v_1 \wedge F(e,e)v_2' \sqsubseteq v_2' \sqsubseteq v_2$ follows from $F(e,e) \sqsubseteq id_{F(V,V)} \wedge v_1' \sqsubseteq v_1 \wedge v_2' \sqsubseteq v_2$. $v_1' = \bot \Rightarrow F(e,e)v_1' = \bot \wedge v_1' = in_{\forall}\lfloor d_1' \rfloor \Rightarrow F(e,e)v_1' = in_{\forall}\lfloor \Psi_T(e,e)(d_1') \rfloor \wedge v_2' = \bot \Rightarrow F(e,e)v_2' = \bot \wedge v_2' = in_{\forall}\lfloor d_2' \rfloor \Rightarrow F(e,e)v_2' = in_{\forall}\lfloor \Psi_T(e,e)(d_2') \rfloor$. By assumptions on (v_1', v_2', v_1, v_2) and e and by 2_M) $\forall R_\beta.\ (\Psi_T(e,e)(d_1'), \Psi_T(e,e)(d_2'), d_1, d_2) \in (\llbracket T\tau[\sigma/\alpha] \rrbracket^T\ (\Xi\beta\ \vec{R}R_\beta)(\Delta r)$. So we conclude $(F(e,e)v_1', F(e,e)v_2', v_1, v_2) \in \llbracket \forall\beta.\ T\tau[\sigma/\alpha] \rrbracket^t)(\Xi\vec{R})(\Delta r)$.

Assume $(v_1', v_2', v_1, v_2) \in \llbracket \forall\beta.\ T\tau[\sigma/\alpha] \rrbracket^t (\Xi\vec{R})(\Delta r)$, $\alpha \notin \Xi$. This implies $v_1' \sqsubseteq v_1 \wedge v_2' \sqsubseteq v_2 \wedge \exists d_1', d_2', d_1, d_2 : TV.\ ((v_1' = \bot \wedge d_1' = \bot) \vee v_1' = in_{\forall}\lfloor d_1' \rfloor) \wedge ((v_2' = \bot \wedge d_2' = \bot) \vee v_2' = in_{\forall}\lfloor d_2' \rfloor) \wedge v_1 = in_{\forall}\lfloor d_1 \rfloor \wedge v_2 = in_{\forall}\lfloor d_2 \rfloor \wedge \forall R_\beta :$ bParAdmRel. $(d_1', d_2', d_1, d_2 \in (\llbracket T\tau[\sigma/\alpha] \rrbracket^T (\Xi\beta\ \vec{R}R_\beta)(\Delta r)$. To prove $(F(e,e)v_1', F(e,e)v_2', v_1, v_2) \in \llbracket \forall\beta.\ T\tau \rrbracket^t)(\Xi, \alpha\ \vec{R}\ (\llbracket\sigma\rrbracket^t(\Xi\vec{R})))(\Delta r)$. $F(e,e)v_1' \sqsubseteq v_1' \sqsubseteq v_1 \wedge F(e,e)v_2' \sqsubseteq v_2' \sqsubseteq v_2$ follows from $F(e,e) \sqsubseteq id_{F(V,V)} \wedge v_1' \sqsubseteq v_1 \wedge v_2' \sqsubseteq v_2$. $v_1' = \bot \Rightarrow F(e,e)v_1' = \bot \wedge v_1' = in_{\forall}\lfloor d_1' \rfloor) \Rightarrow F(e,e)v_1' = in_{\forall}\lfloor \Psi_T(e,e)(d_1') \rfloor \wedge v_2' = \bot \Rightarrow F(e,e)v_2' = \bot \wedge v_2' = in_{\forall}\lfloor d_2' \rfloor) \Rightarrow F(e,e)v_2' = in_{\forall}\lfloor \Psi_T(e,e)(d_2') \rfloor$. By assumptions on (v_1', v_2', v_1, v_2) and e and by 1_M) $\forall R_\beta.\ (\Psi_T(e,e)(d_1'), \Psi_T(e,e)(d_2'), d_1, d_2) \in (\llbracket T\tau \rrbracket^T (\Xi\alpha\beta\ \vec{R}(\llbracket\sigma\rrbracket^t(\Xi\vec{R}))R_\beta)(\Delta r)$. So we conclude $(F(e,e)v_1', F(e,e)v_2', v_1, v_2) \in \llbracket \forall\beta.\ T\tau \rrbracket^t)(\Xi\alpha\ \vec{R}(\llbracket\sigma\rrbracket^t(\Xi\ \vec{R})))(\Delta r)$.

• Assume $(v'_1, v'_2, v_1, v_2) \in [\![\mu\beta.\ \tau]\!]^t(\Xi\alpha\ \vec{R}([\![\sigma]\!]^t(\Xi\vec{R})))(\Delta r)$. We may assume $\beta \notin (\Xi, \alpha)$. The assumption implies $\alpha \notin \Xi$ and $v'_1 \sqsubseteq v_1 \wedge v'_2 \sqsubseteq v_2 \wedge \exists d'_1, d'_2, d_1, d_2 : V.\ ((v'_1 = \bot \wedge d'_1 = \bot) \vee v'_1 = in_\mu(d'_1) \neq \bot) \wedge ((v'_2 = \bot \wedge d'_2 = \bot) \vee v'_2 = in_\mu(d'_2) \neq \bot) \wedge v_1 = in_\mu(d_1) \neq \bot \wedge v_2 = in_\mu(d_2) \neq \bot \wedge (d'_1, d'_2, d_1, d_2) \in ([\![\tau[\mu\beta.\tau/\beta]]\!]^t(\Xi\alpha\ \vec{R}([\![\sigma]\!]^t(\Xi\vec{R})))(\Delta r)$. To prove $(F(e,e)v'_1, F(e,e)v'_2, v_1, v_2) \in [\![(\mu\beta.\ \tau)[\sigma/\alpha]]\!]^t)(\Xi\vec{R})(\Delta r)$. $F(e,e)v'_1 \sqsubseteq v'_1 \sqsubseteq v_1 \wedge F(e,e)v'_2 \sqsubseteq v'_2 \sqsubseteq v_2$ follows from $F(e,e) \sqsubseteq id_{F(V,V)} \wedge v'_1 \sqsubseteq v_1 \wedge v'_2 \sqsubseteq v_2$. $v'_1 = \bot \Rightarrow F(e,e)v'_1 = \bot \wedge v'_1 = in_\mu(d'_1)) \Rightarrow F(e,e)v'_1 = in_\mu(ed'_1) \wedge v'_2 = \bot \Rightarrow F(e,e)v'_2 = \bot \wedge v'_2 = in_\mu(d'_2)) \Rightarrow F(e,e)v'_2 = in_\mu(ed'_2)$. By assumptions on (v'_1, v'_2, v_1, v_2) and e so $(ed'_1, ed'_2, d_1, d_2) \in ([\![(\tau[\mu\beta.\tau/\beta])[\sigma/\alpha]]\!]^t(\Xi\vec{R})(\Delta r)$. As $\alpha \neq \beta$ so $(\tau[\mu\beta.\tau/\beta])[\sigma/\alpha] = (\tau[\sigma/\alpha])[\mu\beta.\tau[\sigma/\alpha]/\beta])$ and $(\mu\beta.\tau)[\sigma/\alpha] = \mu\beta.(\tau[\sigma/\alpha])$ So we conclude $(F(e,e)v'_1, F(e,e)v'_2, v_1, v_2) \in [\![(\mu\beta.\tau)[\sigma/\alpha]]\!]^t)(\Xi\vec{R})(\Delta r)$.

• Assume $(v'_1, v'_2, v_1, v_2) \in [\![(\mu\beta.\tau)[\sigma/\alpha]]\!]^t(\Xi\vec{R})(\Delta r)$, $\alpha \notin \Xi$. Then $\Xi \vdash (\mu\beta.\tau)[\sigma/\alpha]$. We may assume $\beta \notin (\Xi, \alpha)$ and so $(\mu\beta.\tau)[\sigma/\alpha] = \mu\beta.(\tau[\sigma/\alpha])$ as $\Xi \vdash \sigma$. The assumption implies $v'_1 \sqsubseteq v_1 \wedge v'_2 \sqsubseteq v_2 \wedge \exists d'_1, d'_2, d_1, d_2 : V.\ ((v'_1 = \bot \wedge d'_1 = \bot) \vee v'_1 = in_\mu(d'_1) \neq \bot) \wedge ((v'_2 = \bot \wedge d'_2 = \bot) \vee v'_2 = in_\mu(d'_2) \neq \bot) \wedge v_1 = in_\mu(d_1) \neq \bot \wedge v_2 = in_\mu(d_2) \neq \bot \wedge (d'_1, d'_2, d_1, d_2) \in ([\![(\tau[\sigma/\alpha])[\mu\beta.(\tau[\sigma/\alpha])/\beta]]\!]^t(\Xi\vec{R})(\Delta r)$. To prove $(F(e,e)v'_1, F(e,e)v'_2, v_1, v_2) \in [\![\mu\beta.\tau]\!]^t)(\Xi\alpha\ \vec{R}([\![\sigma]\!]^t$ $(\Xi\vec{R})))(\Delta r)$. $F(e,e)v'_1 \sqsubseteq v'_1 \sqsubseteq v_1 \wedge F(e,e)v'_2 \sqsubseteq v'_2 \sqsubseteq v_2$ follows from $F(e,e) \sqsubseteq id_{F(V,V)} \wedge v'_1 \sqsubseteq v_1 \wedge v'_2 \sqsubseteq v_2$. $v'_1 = \bot \Rightarrow F(e,e)v'_1 = \bot \wedge v'_1 = in_\mu(d'_1)) \Rightarrow F(e,e)v'_1 = in_\mu(ed'_1) \wedge v'_2 = \bot \Rightarrow F(e,e)v'_2 = \bot \wedge v'_2 = in_\mu(d'_2) \Rightarrow F(e,e)v'_2 = in_\mu(ed'_2)$. As $\alpha \neq \beta$ then $(\tau[\sigma/\alpha])[\mu\beta.(\tau[\sigma/\alpha])/\beta] = (\tau[\mu\beta.\tau)/\beta])[\sigma/\alpha]$ and so by assumptions on (v'_1, v'_2, v_1, v_2) and e then $(ed'_1, ed'_2, d_1, d_2) \in ([\![\tau[\mu\beta.\tau/\beta]]\!]^t$ $(\Xi\alpha\ \vec{R}([\![\sigma]\!]^t(\Xi\vec{R}))R_\beta)(\Delta r)$. So we conclude $(F(e,e)v'_1, F(e,e)v'_2, v_1, v_2) \in [\![\mu\beta.\tau]\!]^t)(\Xi\alpha\ \vec{R}([\![\sigma]\!]^t(\Xi$ $\vec{R})))(\Delta r)$.

• The cases for $+$ and $\times$ types hold similarly by assumptions on e.

• Assume $(v'_1, v'_2, v_1, v_2) \in [\![\tau \mathsf{ref}]\!]^t(\Xi\alpha\ \vec{R}([\![\sigma]\!]^t(\Xi\vec{R})))(\Delta r)$. This implies either $v'_1 = v'_2 = \bot$ or $\exists l \in dom(\Delta).\ v'_1 \sqsubseteq v_1 = in_L(l) \wedge v'_2 \sqsubseteq v_2 = in_L(l) \wedge \Delta(l) = \tau \wedge - \vdash \tau : type$. To prove $(F(e,e)v'_1, F(e,e)v'_2, v_1, v_2) \in [\![\tau \mathsf{ref}\,[\sigma/\alpha]]\!]^t(\Xi\vec{R})(\Delta r)$, that is since τ is closed $(F(e,e)v'_1, F(e,e)v'_2, v_1,$ $v_2) \in [\![\tau \mathsf{ref}]\!]^t(\Xi\vec{R})\ (\Delta r)$. $F(e,e)v'_1 \sqsubseteq v'_1 \sqsubseteq v_1 = in_L(l) \wedge F(e,e)v'_2 \sqsubseteq v'_2 \sqsubseteq v_2 = in_L(l)$ follows from $F(e,e) \sqsubseteq id_{F(V,V)} \wedge v'_1 \sqsubseteq v_1 = in_L(l) \wedge v'_2 \sqsubseteq v_2 = in_L(l)$. So we conclude $(F(e,e)v'_1, F(e,e)v'_2, v_1,$ $v_2) \in [\![\tau \mathsf{ref}\,[\sigma/\alpha]]\!]^t)(\Xi\vec{R})(\Delta r)$.

The other direction holds similarly from τ closed and downwards closure.

• Also the cases for 1 and int hold by downwards closure.

Definition 20 (Related environments)
We extend the relational interpretation of types to environments. For environment $\Xi \vdash \Gamma$, *with* $\Gamma = x_1 : \tau_1, \ldots, x_n : \tau_n,\ n \geq 1$, *let* $\rho'_1, \rho'_2, \rho_1, \rho_2 \in \bigotimes_{i \in \{1,\ldots,n\}} V$. *Then*
$(\rho'_1, \rho'_2, \rho_1, \rho_2) \in [\![\Gamma]\!]^t(\Xi\vec{R})(\Delta r)$ *iff*
$\forall i \in \{1 \ldots n\}.\ \exists(v'_{1i}, v'_{2i}, v_{1i}, v_{2i}) \in [\![\Gamma(x_i)]\!]^t(\Xi\vec{R})(\Delta r)$ *and*
$\rho'_1 = \bigotimes v'_{1i},\ \rho'_2 = \bigotimes v'_{2i},\ \rho_1 = \bigotimes v_{1i},\ \rho_2 = \bigotimes v_{2i}$

For $\Gamma = \{\}$ define $(\rho'_1, \rho'_2, \rho_1, \rho_2) \in [\![\Gamma]\!]^t(\Xi\vec{R})(\Delta r)$ iff
$\rho'_1 = \rho'_2 = \rho_1 = \rho_2 = \lfloor * \rfloor$

It follows from the definition that
$(\rho'_1, \rho'_2, \rho_1, \rho_2) \in [\![\Gamma]\!]^t(\Xi\vec{R})(\Delta r) \Rightarrow$
$((\rho'_1 = \rho'_2 = \bot) \vee \rho'_1 \sqsubseteq \rho_1 \neq \bot \wedge \rho'_2 \sqsubseteq \rho_2 \neq \bot))$.

Definition 21 (Relating denotations of open expressions)

- *For all* $\Xi = \alpha_1, \ldots, \alpha_k$*, for all* $\Gamma = x_1 : \tau_1, \ldots, x_n : \tau_n$ *and* $\Delta; \Xi; \Gamma \vdash V_1 : \tau$ *and* $\Delta; \Xi; \Gamma \vdash V_2 : \tau$*, for all* Δr*, let* $v_1 = [\![\Delta; \Xi; \Gamma \vdash V_1 : \tau]\!]$ *and* $v_2 = [\![\Delta; \Xi; \Gamma \vdash V_2 : \tau]\!]$*, and define*

$$\begin{array}{l}(v_1, v_2) \in ([\![\tau]\!]^t_{\Xi\Gamma})(\Delta r) \overset{def}{\Longleftrightarrow} \\ \quad \forall \Delta' r' \rhd \Delta r.\ \forall \vec{R} : \mathrm{bParAdmRel}^k.\ \forall \rho_1', \rho_2', \rho_1, \rho_2. \\ \qquad (\rho_1', \rho_2', \rho_1, \rho_2) \in [\![\Gamma]\!]^t(\Xi\vec{R})(\Delta' r') \Rightarrow \\ \qquad ((v_1 \rho_1'), (v_2 \rho_2'), (v_1 \rho_1), (v_2 \rho_2)) \in [\![\tau]\!]^t(\Xi\vec{R})(\Delta' r')\end{array}$$

- *For all* $\Xi = \alpha_1, \ldots, \alpha_k$*, for all* $\Gamma = x_1 : \tau_1, \ldots, x_n : \tau_n$ *and* $\Delta; \Xi; \Gamma \vdash M_1 : T\tau$ *and* $\Delta; \Xi; \Gamma \vdash M_2 : T\tau$*, for all* Δr*, let* $m_1 = [\![\Delta; \Xi; \Gamma \vdash M_1 : T\tau]\!]$ *and* $m_2 = [\![\Delta; \Xi; \Gamma \vdash M_2 : T\tau]\!]$*, and define*

$$\begin{array}{l}(m_1, m_2) \in ([\![T\tau]\!]^t_{\Xi\Gamma})(\Delta r) \overset{def}{\Longleftrightarrow} \\ \quad \forall \Delta' r' \rhd \Delta r.\ \forall \vec{R} : \mathrm{bParAdmRel}^k.\ \forall \rho_1', \rho_2', \rho_1, \rho_2. \\ \qquad (\rho_1', \rho_2', \rho_1, \rho_2) \in [\![\Gamma]\!]^t(\Xi\vec{R})(\Delta' r') \Rightarrow \\ \qquad ((v_1 \rho_1'), (v_2 \rho_2'), (v_1 \rho_1), (v_2 \rho_2)) \in [\![T\tau]\!]^T(\Xi\vec{R})(\Delta' r')\end{array}$$

Lemma 22
The typing rules preserve relatedness in $([\![\cdot]\!]^t_{\cdot\,\cdot})(\cdot\ r)$*, for all* r*.*

For typing rules with no premisses it holds that

if $\dfrac{}{\Delta; \Xi; \Gamma \vdash G : \gamma}$ *and* $d = [\![\Delta; \Xi; \Gamma \vdash G : \gamma]\!]$

then $\forall r.\ (d, d) \in ([\![\gamma]\!]^t_{\Xi\Gamma})(\Delta r)$*.*

For typing rules with j *premisses it holds that*

if $\dfrac{\Delta; \Xi_1; \Gamma_1 \vdash G_{11} : \gamma_1\ \ldots\ldots\ \Delta; \Xi_j; \Gamma_j \vdash G_{j1} : \gamma_j}{\Delta; \Xi'; \Gamma' \vdash G_1' : \gamma'}$ *and*

$\dfrac{\Delta; \Xi_1; \Gamma_1 \vdash G_{12} : \gamma_1 \ldots\ldots \Delta; \Xi_j; \Gamma_j \vdash G_{j2} : \gamma_j}{\Delta; \Xi'; \Gamma' \vdash G_2' : \gamma'}$ *and*

for the well typed terms
$d_{11} = [\![\Delta; \Xi_1; \Gamma_1 \vdash G_{11} : \gamma_1]\!],\ d_{12} = [\![\Delta; \Xi_1; \Gamma_1 \vdash G_{12} : \gamma_1]\!], \ldots,$
$d_{j1} = [\![\Delta; \Xi_j; \Gamma_j \vdash G_{j1} : \gamma_j]\!],\ d_{j2} = [\![\Delta; \Xi_j; \Gamma_j \vdash G_{j2} : \gamma_j]\!]$*. And* $d_1' = [\![\Delta; \Xi'; \Gamma' \vdash G_1' : \gamma']\!],\ d_2' = [\![\Delta; \Xi'; \Gamma' \vdash G_2' : \gamma']\!]$*.*

And $\forall i \in 1 \ldots j.\ (d_{i1}, d_{i2}) \in ([\![\gamma_i]\!]^t_{\Xi_i\Gamma_i})(\Delta r)$*.*

then it holds that $(d_1', d_2') \in ([\![\gamma']\!]^t_{\Xi'\Gamma'})(\Delta r)$*.*

Proof: The proof is by induction over the typing relation and parameter weakening is used in several proof cases.

- $\dfrac{}{\Delta; \Xi; \Gamma \vdash x_j : \tau_j}$

  Let $v = [\![\Delta; \Xi; \Gamma \vdash x_j : \tau_j]\!]$

Let Δr be a basic parameter. To prove $(v, v) \in ([\![\tau_j]\!]^t_{\Xi\Gamma})(\Delta r)$. Let $\vec{R} \in \text{bParAdmRel}^k$ and let $\Delta' r' \rhd \Delta r$.
Assume $\forall i \in \{1 \dots n\}.\ (v'_{1i}, v'_{2i}, v_{1i}, v_{2i}) \in [\![\Gamma(x_i)]\!]^t(\Xi\vec{R})(\Delta' r')$ and $\rho'_1 = \bigotimes v'_{1i},\ \rho'_2 = \bigotimes v'_{2i}$, $\rho_1 = \bigotimes v_{1i},\ \rho_2 = \bigotimes v_{2i}$. It then holds that $(\rho'_1, \rho'_2, \rho_1, \rho_2) \in [\![\Gamma]\!]^t(\Xi\vec{R})(\Delta' r')$ and $\rho'_1 = \rho'_2 = \bot \vee (\rho_1 \neq \bot \wedge \rho_2 \neq \bot)$. We need to show $(v_1\rho'_1, v_2\rho'_2, v_1\rho_1, v_2\rho_2) \in [\![\tau_j]\!]^t(\Xi\vec{R})(\Delta' r')$. $\rho'_1 = \rho'_2 = \bot \Rightarrow v_1\rho'_1 = v_2\rho'_2 = \bot$ and we are done. $(\rho_1 \neq \bot \wedge \rho_2 \neq \bot) \Rightarrow \big((v_1\rho_1 = \pi_j(\rho_1) = v_{j1}) \wedge (v_2\rho_2 = \pi_j(\rho_2) = v_{j2}) \wedge (v'_1\rho'_1 \in \{\bot, v'_{j1}\}) \wedge (v'_2\rho'_2 \in \{\bot, v'_{j2}\})\big)$. By downwards closure and assumption $(v'_{1j}, v'_{2j}, v_{1j}, v_{2j}) \in [\![\Gamma(x_j)]\!]^t(\Xi\vec{R})(\Delta' r')$ it holds that $(v_1\rho'_1, v_2\rho'_2, v_1\rho_1, v_2\rho_2) \in [\![\tau_j]\!]^t(\Xi\vec{R})(\Delta' r')$. So we have $(v, v) \in [\![\tau_j]\!]^t_{\Xi\Gamma})(\Delta r)$.

- $$\frac{\Delta; \Xi; \Gamma \vdash V : \tau}{\Delta; \Xi; \Gamma \vdash \mathit{val}\ V : T\tau}$$

Let $v_1 = [\![\Delta; \Xi; \Gamma \vdash V_1 : \tau]\!]$ and $v_2 = [\![\Delta; \Xi; \Gamma \vdash V_2 : \tau]\!]$. Assume $(v_1, v_2) \in [\![\tau]\!]^t_{\Xi\Gamma}(\Delta r)$.
Let $m_1 = [\![\Delta; \Xi; \Gamma \vdash \mathit{val}\ V_1 : T\tau]\!]$ and $m_2 = [\![\Delta; \Xi; \Gamma \vdash \mathit{val}\ V_2 : T\tau]\!]$. To show, $(m_1, m_2) \in [\![T\tau]\!]^T_{\Xi\Gamma}(\Delta r)$.
This requires $\forall \vec{R}.\ \forall \Delta' r' \rhd \Delta r.\ \forall \rho'_1, \rho'_2, \rho_1, \rho_2$.
$(\rho'_1, \rho'_2, \rho_1, \rho_2) \in [\![\Gamma]\!]^t(\Xi\vec{R})(\Delta' r') \Rightarrow (m_1(\rho'_1), m_2(\rho'_2), m_1(\rho_1), m_2(\rho_2)) \in [\![T\tau]\!]^t(\Xi\vec{R})(\Delta' r')$.
If $\rho'_1 = \rho'_2 = \bot$ this holds, else $\rho_1 \neq \bot \wedge \rho_2 \neq \bot$. $\rho'_1 = \bot \Rightarrow v_1\rho'_1 = \bot \wedge m_1\rho'_1 = \bot$ and $\rho'_2 = \bot \Rightarrow v_2\rho'_2 = \bot \wedge m_2\rho'_2 = \bot$.
Let $\Delta'' r'' \rhd \Delta' r'$ and suppose $(k'_1, k'_2, k_1, k_2) \in [\![\tau^\top]\!]^K(\Xi\vec{R})(\Delta'' r'')$ and $(s'_1, s'_2, s_1, s_2) \in \nabla_S(\Delta'' r'')$.
$(m_1\rho_1)k_1s_1 = k_1s_1(v_1\rho_1), (m_1\rho'_1)k'_1s'_1 = k'_1s'_1(v_1\rho'_1)$,
$(m_2\rho_2)k_2s_2 = k_2s_2(v_2\rho_2), (m_2\rho'_2)k'_2s'_2 = k'_2s'_2(v_2\rho'_2)$.
The continuations and states are related under the basic parameter $\Delta'' r''$ by assumption, the values are also related under $\Delta'' r''$ by assumption and parameter weakening. Hence we get the required termination approximation, so $(m_1(\rho'_1), m_2(\rho'_2), m_1(\rho_1), m_2(\rho_2)) \in [\![T\tau]\!]^T(\Xi\vec{R})(\Delta' r')$.
We conclude $(m_1, m_2) \in [\![T\tau]\!]^T_{\Xi\Gamma}(\Delta r)$.

- $$\frac{\Delta; \Xi; \Gamma \vdash V : \tau\mathsf{ref}}{\Delta; \Xi; \Gamma \vdash !V : T\tau}$$

$\Delta; \Xi; \Gamma \vdash V : \tau\mathsf{ref}$ implies $- \vdash \tau$. Let
$v_1 = [\![\Delta; \Xi; \Gamma \vdash V_1 : \tau\mathsf{ref}]\!]$ and $v_2 = [\![\Delta; \Xi; \Gamma \vdash V_2 : \tau\mathsf{ref}]\!]$. Assume $(v_1, v_2) \in [\![\tau\mathsf{ref}]\!]^t_{\Xi\Gamma}(\Delta r)$.
Let $m_1 = [\![\Delta; \Xi; \Gamma \vdash !V_1 : T\tau]\!]$ and $m_2 = [\![\Delta; \Xi; \Gamma \vdash !V_2 : T\tau]\!]$. To show, $(m_1, m_2) \in [\![T\tau]\!]^T_{\Xi\Gamma}(\Delta r)$.
This requires $\forall \vec{R}.\ \forall \Delta' r' \rhd \Delta r.\ \forall \rho'_1, \rho'_2, \rho_1, \rho_2$.
$(\rho'_1, \rho'_2, \rho_1, \rho_2) \in [\![\Gamma]\!]^t(\Xi\vec{R})(\Delta' r') \Rightarrow (m_1(\rho'_1), m_2(\rho'_2), m_1(\rho_1), m_2(\rho_2)) \in [\![T\tau]\!]^t(\Xi\vec{R})(\Delta' r')$.
If $\rho'_1 = \rho'_2 = \bot$ this holds, else $\rho_1 \neq \bot \wedge \rho_2 \neq \bot$, and the assumption implies $\exists l : \tau \in \Delta'$. $v_1\rho_1 = v_2\rho_2 = \mathit{in}_{\mathsf{ref}}\, l,\ v_1\rho'_1, v_2\rho'_2 \in \{\bot, \mathit{in}_{\mathsf{ref}}\, l\}$ and it holds that τ is closed. Let $\Delta'' r'' \rhd \Delta' r'$ and suppose $(k'_1, k'_2, k_1, k_2) \in [\![\tau^\top]\!]^K(\Xi\vec{R})(\Delta'' r'')$ and $(s'_1, s'_2, s_1, s_2) \in \nabla_S(\Delta'' r'')$.
$(m_1\rho_1)k_1s_1 = k_1s_1(s_1l), (m_1\rho'_1)k'_1s'_1 \sqsubseteq k'_1s'_1(s'_1l)$,
$(m_2\rho_2)k_2s_2 = k_2s_2(s_2l), (m_2\rho'_2)k'_2s'_2 \sqsubseteq k'_2s'_2(s'_2l)$.
The continuations and states are related under the basic parameter $\Delta'' r''$ by assumption. As $l : \tau \in \Delta' \subseteq \Delta''$ then also $(s'_1l, s'_2l, s_1l, s_2l) \in [\![\tau]\!]^t()(\Delta'' r'')$ and by Ξ-weakening so $(s'_1l, s'_2l, s_1l, s_2l) \in [\![\tau]\!]^t(\Xi\vec{R})(\Delta'' r'')$. Hence we get the required termination approximation, so $(m_1(\rho'_1), m_2(\rho'_2), m_1(\rho_1), m_2(\rho_2)) \in [\![T\tau]\!]^T(\Xi\vec{R})\,(\Delta' r')$. We conclude $(m_1, m_2) \in [\![T\tau]\!]^T_{\Xi\Gamma}(\Delta r)$.

- $$\frac{\Delta; \Xi; \Gamma \vdash V_a : \tau\mathsf{ref} \qquad \Delta; \Xi; \Gamma \vdash V_b : \tau}{\Delta; \Xi; \Gamma \vdash V_a := V_b : T1}$$

Let $v_{a1} = [\![\Delta; \Xi; \Gamma \vdash V_{a1} : \tau\mathsf{ref}\,]\!]$ and $v_{a2} = [\![\Delta; \Xi; \Gamma \vdash V_{a2} : \tau\mathsf{ref}\,]\!]$.
Let $v_{b1} = [\![\Delta; \Xi; \Gamma \vdash V_{b1} : \tau]\!]$ and $v_{b2} = [\![\Delta; \Xi; \Gamma \vdash V_{b2} : \tau]\!]$.
Assume $(v_{a1}, v_{a2}) \in [\![\tau\mathsf{ref}\,]\!]^t_{\Xi\Gamma}(\Delta r)$, and $(v_{b1}, v_{b2}) \in [\![\tau]\!]^t_{\Xi\Gamma}(\Delta r)$.
Let $m_1 = [\![\Delta; \Xi; \Gamma \vdash V_{a1} := V_{b1} : T1]\!]$ and $m_2 = [\![\Delta; \Xi; \Gamma \vdash V_{a2} := V_{b2} : T1]\!]$.
To show, $(m_1, m_2) \in [\![T1]\!]^t_{\Xi\Gamma}(\Delta r)$.
This requires $\forall \vec{R}.\ \forall \Delta' r' \rhd \Delta r.\ \forall \rho_1', \rho_2', \rho_1, \rho_2.$
$(\rho_1', \rho_2', \rho_1, \rho_2) \in [\![\Gamma]\!]^t(\Xi\vec{R})(\Delta' r') \Rightarrow (m_1(\rho_1'), m_2(\rho_2'), m_1(\rho_1), m_2(\rho_2)) \in [\![T1]\!]^t(\Xi\vec{R})(\Delta' r')$.
If $\rho_1' = \rho_2' = \bot$ this holds, else $\rho_1 \neq \bot \wedge \rho_2 \neq \bot$, and the assumption implies $- \vdash \tau$ and $\exists l : \tau \in \Delta'.\ v_{a1}\rho_1 = v_{a2}\rho_2 = in_{\mathsf{ref}}\, l,\ v_{a1}\rho_1', v_{a2}\rho_2' \in \{\bot, in_{\mathsf{ref}}\, l\}$
Let $\Delta'' r'' \rhd \Delta' r'$ and suppose $(k_1', k_2', k_1, k_2) \in [\![1^\top]\!]^K(\Xi\vec{R})\,(\Delta'' r'')$ and $(s_1', s_2', s_1, s_2) \in \nabla_S(\Delta'' r'')$.
$(m_1\rho_1)k_1 s_1 = k_1(s_1[l \mapsto v_{b1}\rho_1])in_1{*},\ (m_2\rho_2)k_2 s_2 = k_2(s_2[l \mapsto v_{b2}\rho_2])in_1{*},$
$(m_1\rho_1')k_1' s_1' \sqsubseteq k_1'(s_1'[l \mapsto v_{b1}\rho_1'])in_1{*},\ (m_2\rho_2')k_2' s_2' \sqsubseteq k_2'(s_2'[l \mapsto v_{b2}\rho_2'])in_1{*}.$
The continuations and the original states are related under the basic parameter $\Delta'' r''$ by assumption. It follows from the assumptions that the values stored are related $(v_{b1}\rho_1', v_{b2}\rho_2', v_{b1}\rho_1,\ v_{b2}\rho_2) \in [\![\tau]\!]^t(\Xi\vec{R})(\Delta' r')$. Since τ is closed then by the Ξ-strengthening property and parameter weakening for ∇ also $(v_{b1}\rho_1', v_{b2}\rho_2', v_{b1}\rho_1, v_{b2}\rho_2) \in [\![\tau]\!]^t()(\Delta'' r'')$. As $l : \tau \in \Delta'' \supseteq \Delta'$ then the updated states are still related under $\Delta'' r''$. The $in_1{*}$ values are related under any parameter. Hence we get the required termination approximation, so $(m_1(\rho_1'), m_2(\rho_2'), m_1(\rho_1), m_2(\rho_2)) \in [\![T1]\!]^t(\Xi\vec{R})(\Delta' r')$.
We conclude $(m_1, m_2) \in [\![T1]\!]^t_{\Xi\Gamma}(\Delta r)$.

- $$\frac{\Delta; \Xi, \alpha; \Gamma \vdash M : T\tau \qquad \Xi \vdash \Gamma}{\Delta; \Xi; \Gamma \vdash \Lambda\alpha.\ M : \forall\alpha.T\tau}$$

Let $m_1 = [\![\Delta; \Xi, \alpha; \Gamma \vdash M_1 : T\tau]\!]$ and $m_2 = [\![\Delta; \Xi, \alpha; \Gamma \vdash M_2 : T\tau]\!]$.
Let $v_1 = [\![\Delta; \Xi; \Gamma \vdash \Lambda\alpha.\ M_1 : \forall\alpha.T\tau]\!]$ and $v_2 = [\![\Delta; \Xi; \Gamma \vdash \Lambda\alpha.\ M_2 : \forall\alpha.T\tau]\!]$.
Assume $(m_1, m_2) \in [\![T\tau]\!]^t_{\Xi\alpha\Gamma}(\Delta r)$ and assume $\Xi \vdash \Gamma$.
The assumption implies $\forall \vec{R}, R_\alpha.\ \forall \Delta' r' \rhd \Delta r.\ (\rho_1', \rho_2', \rho_1, \rho_2) \in [\![\Gamma]\!]^t(\Xi\alpha\,\vec{R}' R_\alpha)(\Delta' r') \Rightarrow (m_1(\rho_1'),$ $m_2(\rho_2'), m_1(\rho_1), m_2(\rho_2)) \in [\![T\tau]\!]^T(\Xi, \alpha\ \vec{R}R_\alpha)(\Delta' r')$. When $\rho \neq \bot$ then $v_1\rho = in_\forall\lfloor m_1\rho\rfloor$ and $v_2\rho = in_\forall\lfloor m_2\rho\rfloor$. To show $(v_1, v_2) \in ([\![\forall\alpha.T\tau]\!]^t_{\Xi\Gamma})(\Delta r)$.
This requires $\forall \vec{R}.\ \forall \Delta' r' \rhd \Delta r.\ \forall \rho_1', \rho_2', \rho_1, \rho_2.$
$(\rho_1', \rho_2', \rho_1, \rho_2) \in [\![\Gamma]\!]^t(\Xi\vec{R})(\Delta' r') \Rightarrow (v_1(\rho_1'), v_2(\rho_2'), v_1(\rho_1), v_2(\rho_2)) \in [\![\forall\alpha.T\tau]\!]^t(\Xi\vec{R})(\Delta' r')$.
This again requires $((v_1(\rho_1')) = (v_2(\rho_2')) = \bot)\ \vee\ \big(\Xi \vdash \forall\alpha.T\tau \wedge \alpha \notin \Xi \wedge (v_1(\rho_1') \sqsubseteq v_1(\rho_1) \neq \bot \wedge v_2(\rho_2') \sqsubseteq v_2(\rho_2) \neq \bot \wedge (\exists d_1', d_1, d_2', d_2 : TV.\ ((v_1(\rho_1') = \bot \wedge d_1' = \bot) \vee v_1(\rho_1')) = in_\forall\lfloor d_1'\rfloor) \wedge (v_1(\rho_1) = in_\forall\lfloor d_1\rfloor) \wedge ((v_2(\rho_2') = \bot \wedge d_2' = \bot) \vee v_2(\rho_2')) = in_\forall\lfloor d_2'\rfloor) \wedge (v_2(\rho_2) = in_\forall\lfloor d_2\rfloor) \wedge \forall R_\alpha \in \text{bParAdmRel}.\ (d_1', d_2', d_1, d_2) \in [\![T\tau]\!]^T(\Xi\alpha\ \vec{R}R_\alpha)(\Delta' r')$.
This follows from the assumptions. Then we can conclude $(v_1, v_2) \in [\![\forall\alpha.T\tau]\!]^t_{\Xi\Gamma}(\Delta r)$.

- $$\frac{\Delta; \Xi; \Gamma \vdash V : \forall\alpha.T\tau \qquad \Xi \vdash \sigma}{\Delta; \Xi; \Gamma \vdash V\sigma : T\tau[\sigma/\alpha])}$$

Let $v_1 = [\![\Delta; \Xi; \Gamma \vdash V_1 : \forall\alpha.T\tau]\!]$ and $v_2 = [\![\Delta; \Xi; \Gamma \vdash V_2 : \forall\alpha.T\tau]\!]$.
Let $m_1 = [\![\Delta; \Xi; \Gamma \vdash V_1\sigma : T\tau[\sigma/\alpha])]\!]$ and $m_2 = [\![\Delta; \Xi; \Gamma \vdash V_2\sigma : T\tau[\sigma/\alpha])]\!]$.
Assume $(v_1, v_2) \in [\![\forall\alpha.T\tau]\!]^t_{\Xi\Gamma}(\Delta r)$ and assume $\Xi \vdash \sigma$.
To show $(m_1, m_2) \in [\![T\tau[\sigma/\alpha])]\!]^T_{\Xi\Gamma})(\Delta r)$. This requires $\forall \vec{R}.\ \forall \Delta' r' \rhd \Delta r.\ (\rho_1', \rho_2', \rho_1, \rho_2) \in [\![\Gamma]\!]^t(\Xi\vec{R})(\Delta' r') \Rightarrow (m_1(\rho_1'), m_2(\rho_2'), m_1(\rho_1), m_2(\rho_)) \in [\![T\tau[\sigma/\alpha]]\!]^t(\Xi\vec{R})(\Delta' r')$.
The assumption $(v_1, v_2) \in [\![\forall\alpha.T\tau]\!]^t_{\Xi\Gamma}(\Delta r)$ implies $\forall \vec{R}.\ \forall \Delta' r' \rhd \Delta r.\ (\rho_1', \rho_2', \rho_1, \rho_2) \in [\![\Gamma]\!]^t(\Xi\vec{R})$ $(\Delta' r') \Rightarrow (v_1(\rho_1'), v_2(\rho_2'), v_1(\rho_1), v_2(\rho_2)) \in [\![\forall\alpha.T\tau]\!]^t(\Xi\vec{R})(\Delta' r')$. This again implies $(v_1(\rho_1') =$

$v_2(\rho_2') = \bot) \vee \alpha \notin \Xi \wedge \big((v_1(\rho_1') \sqsubseteq v_1(\rho_1) \neq \bot) \wedge (v_2(\rho_2') \sqsubseteq v_2(\rho_2) \neq \bot) \ \wedge \exists d_1', d_1, d_2', d_2 : TV.\ ((v_1(\rho_1') = \bot \wedge d_1' = \bot) \vee v_1(\rho_1') = in_\forall\lfloor d_1' \rfloor) \wedge ((v_2(\rho_2') = \bot \wedge d_2' = \bot) \vee v_2(\rho_2')) = in_\forall\lfloor d_2' \rfloor) \wedge v_1(\rho_1) = in_\forall\lfloor d_1 \rfloor \wedge v_2(\rho_2) = in_\forall\lfloor d_2 \rfloor \wedge \forall R_\alpha \in \text{bParAdmRel}.\ (d_1', d_2', d_1, d_2) \in [\![T\tau]\!]^T(\vec{R}R_\alpha)(\Delta' r')\big)$.

Also by definition of the denotational semantics
$(v_1(\rho_1') = \bot \Rightarrow m_1(\rho_1') = \bot = d_1') \wedge v_1(\rho_1') = in_\forall\lfloor d_1' \rfloor \Rightarrow m_1(\rho_1') = d_1'$,
$(v_2(\rho_2') = \bot \Rightarrow m_2(\rho_2') = \bot = d_2') \wedge v_2(\rho_2') = in_\forall\lfloor d_2' \rfloor \Rightarrow m_2(\rho_2') = d_2'$,
$(v_1(\rho_1) = \bot \Rightarrow m_1(\rho_1) = \bot \wedge v_1(\rho_1) = in_\forall\lfloor d_1 \rfloor \Rightarrow m_1(\rho_1) = d_1$,
$(v_2(\rho_2') = \bot \Rightarrow m_2(\rho_2') = \bot \wedge v_2(\rho_2') = in_\forall\lfloor d_2' \rfloor \Rightarrow m_2(\rho_2') = d_2'$.
If $(\bot = (v_1(\rho_1')) = (v_2(\rho_2')) = (m_1(\rho_1')) = (m_2(\rho_2')))$ we are done. Else by assumption $\forall R_\alpha \in \text{bParAdmRel}.\ (d_1', d_2', d_1,\ d_2) \in [\![T\tau]\!]^T(\Xi\alpha\ \vec{R}R_\alpha)(\Delta' r')$. We choose R_α as $[\![\sigma]\!]^t(\Xi\vec{R})$, so by assumptions it holds that $(d_1', d_2', d_1, d_2) \in [\![T\tau]\!]^T(\Xi\alpha\ \vec{R}([\![\sigma]\!]^t(\Xi\vec{R}))(\Delta' r')$. By lemma 19 this implies $(d_1', d_2', d_1, d_2) \in [\![T\tau[\sigma/\alpha']]\!]^T(\Xi\vec{R})(\Delta' r')$. So $(m_1(\rho_1'), m_2(\rho_2'), m_1(\rho_1), m_2(\rho_2) \in [\![T\tau[\sigma/\alpha]]\!]^T(\Xi\vec{R})(\Delta' r')$. We conclude $(m_1, m_2) \in [\![T\tau[\sigma/\alpha]]\!]^T_{\Xi\Gamma}(\Delta r)$.

- $$\frac{\Delta; \Xi; \Gamma \vdash V : \tau}{\Delta; \Xi; \Gamma \vdash refV : T(\tau\mathsf{ref})} \quad (- \vdash \tau)$$

  Let $v_1 = [\![\Delta; \Xi; \Gamma \vdash V_1 : \tau]\!]$ and $v_2 = [\![\Delta; \Xi; \Gamma \vdash V_2 : \tau]\!]$. Assume $(v_1, v_2) \in [\![\tau]\!]^t_{\Xi\Gamma})(\Delta r)$ and $- \vdash \tau$.
  Let $m_1 = [\![\Delta; \Xi; \Gamma \vdash refV_1 : T(\tau\mathsf{ref})]\!]$ and $m_2 = [\![\Delta; \Xi; \Gamma \vdash refV_2 : T(\tau\mathsf{ref})]\!]$.
  To show: $(m_1, m_2) \in [\![T(\tau\mathsf{ref})]\!]^T_{\Xi\Gamma}(\Delta r)$.
  $(m_1, m_2) \in [\![T(\tau\mathsf{ref})]\!]^T_{\Xi\Gamma}(\Delta r)$ requires $\forall\vec{R}.\ \forall\Delta' r' \rhd \Delta r.\ (\rho_1', \rho_2', \rho_1, \rho_2) \in [\![\Gamma]\!]^t(\Xi\vec{R})(\Delta' r') \Rightarrow (m_1(\rho_1'), m_2(\rho_2'), m_1(\rho_1), m_2(\rho_2)) \in [\![T\tau]\!]^T(\Xi\vec{R})(\Delta' r')$. This again requires $(m_1(\rho_1')) \sqsubseteq (m_1(\rho_1)) \wedge (m_2(\rho_2')) \sqsubseteq (m_2(\rho_2)) \ \wedge \forall\Delta'' r'' \rhd \Delta' r'.\ \forall k_1', k_1, k_2', k_2.\ \forall S_1', S_1, S_2', S_2.$
  $\big((k_1', k_2', k_1, k_2) \in [\![\tau^\top]\!]^K(\Xi\vec{R})(\Delta'' r'') \wedge (S_1', S_2', S_1, S_2) \in \nabla_S(\Delta'' r'') \Rightarrow$
  $\big(((m_1(\rho_1'))k_1'S_1' = \top \Rightarrow (m_2(\rho_2))k_2S_2 = \top) \wedge$
  $((m_2(\rho_2'))k_2'S_2' = \top \Rightarrow (m_1(\rho_1))k_1S_1 = \top)\big)$.
  $(m_1(\rho_1')) \sqsubseteq (m_1(\rho_1)) \wedge (m_2(\rho_2')) \sqsubseteq (m_2(\rho_2))$ follows from assuptions on $\rho_1', \rho_1, \rho_2', \rho_2$. For the remaining proof, assume $(k_1', k_2', k_1, k_2) \in [\![\tau^\top]\!]^K(\Xi\vec{R})(\Delta'' r'') \wedge (S_1', S_2', S_1, S_2) \in \nabla_S(\Delta'' r'')$.
  By the denotational semantics and the theory of FM-domains
  $(m_1(\rho_1'))k_1'S_1' = k_1'(S_1'[l \mapsto (v_1(\rho_1'))])(in_{\mathsf{ref}}\, l)$ $(m_1(\rho_1))k_1S_1 = k_1(S_1[l \mapsto (v_1(\rho_1))])(in_{\mathsf{ref}}\, l)$
  $(m_2(\rho_2'))k_2'S_2' = k_2'(S_2'[l \mapsto (v_2(\rho_2'))])(in_{\mathsf{ref}}\, l)$ $(m_2(\rho_2))k_2S_2 = k_2(S_2[l \mapsto (v_2(\rho_2))])(in_{\mathsf{ref}}\, l)$ for $l \notin supp(\lambda l'.k_1(S_1[l \mapsto (v_1(\rho_1))])(in_{\mathsf{ref}}\, l')) \cup supp(\lambda l'.k_2(S_2[l \mapsto (v_2(\rho_2))])(in_{\mathsf{ref}}\, l')) \cup dom(\Delta'') \cup A_{r''1}(S_1) \cup A_{r''2}(S_2)$. We define an extended parameter $\Delta^3 r'' = (\Delta'' \uplus l \mapsto \tau)r'' \rhd \Delta'' r''$. By assumption $(v_1(\rho_1'), v_2(\rho_2'),\ v_1(\rho_1), v_2(\rho_1)) \in [\![\tau]\!]^t(\Xi\vec{R})(\Delta' r')$ and $- \vdash \tau$. So $(in_{\mathsf{ref}}\, l,\ in_{\mathsf{ref}}\, l, in_{\mathsf{ref}}\, l, in_{\mathsf{ref}}\, l) \in [\![\tau\mathsf{ref}]\!]^t(\Xi\vec{R})(\Delta^3 r'')$, and by Ξ-strengthening and parameter weakening $(v_1(\rho_1'), v_2(\rho_2'), v_1(\rho_1),\ v_2(\rho_1)) \in [\![\tau]\!]^t()(\Delta^3 r'')$. The original states were related under $\Delta'' r''$. The domain of Δ'' as well as the areas wiewed by accessibility maps have not been changed, so by parameter weakening $(S_1'[l \mapsto (v_1(\rho_1'))], S_2'[l \mapsto (v_2(\rho_2'))],\ S_1[l \mapsto (v_1(\rho_1))], S_2[l \mapsto (v_2(\rho_2))]) \in \nabla_S(\Delta^3 r'')$. So we have continuations related under $\Delta'' r''$ applied to states and values related under the extended parameter $\Delta^3 r''$, and then we get the required termination approximation. So $(m_1(\rho_1'), m_2(\rho_2'),\ m_1(\rho_1), m_2(\rho_2)) \in [\![T\tau]\!]^T(\Xi\vec{R})(\Delta' r')$. We conclude $(m_1, m_2) \in [\![T(\tau\mathsf{ref})]\!]^T_{\Xi\Gamma}(\Delta r)$.

- $$\frac{\Delta; \Xi; \Gamma \vdash M_a : T\tau_a \quad \Delta; \Xi; \Gamma, x : \tau_a \vdash M_b : T\tau_b}{\Delta; \Xi; \Gamma \vdash let\ x \Leftarrow M_a\ in\ M_b : T\tau_b}$$

  Let $m_{1a} = [\![\Delta; \Xi; \Gamma \vdash M_{1a} : T\tau_a]\!]$, $m_{2a} = [\![\Delta; \Xi; \Gamma \vdash M_{2a} : T\tau_a]\!]$.
  Let $m_{1b} = [\![\Delta; \Xi; \Gamma, x : \tau_a \vdash M_{1b} : T\tau_b]\!]$, $m_{2b} = [\![\Delta; \Xi; \Gamma, x : \tau_a \vdash M_{2b} : T\tau_b]\!]$.

Assume $(m_{1a}, m_{2a}) \in ([\![T\tau_a]\!]^T_{\Xi\Gamma})(\Delta r)$ and $(m_{1b}, m_{2b}) \in ([\![T\tau_b]\!]^T_{\Xi\Gamma,x:\tau_a})(\Delta r)$. Then $\forall \vec{R}.\ \forall \Delta' r' \rhd \Delta r.\ \forall (\rho_1', \rho_2', \rho_1, \rho_2) \in ([\![\Gamma]\!]^t(\Xi\vec{R})(\Delta' r').\ (m_{1a}(\rho_1'), m_{2a}(\rho_2'), m_{1a}(\rho_1), m_{2a}(\rho_2)) \in ([\![T\tau_a]\!]^T(\Xi\vec{R})(\Delta' r')$, and further $\forall (v_{1x}', v_{2x}', v_{1x}, v_{2x}) \in [\![\tau_a]\!]^t(\Xi\vec{R})(\Delta' r').\ (m_{1b}(\rho_1' \otimes v_{1x}'), m_{2b}(\rho_2' \otimes v_{2x}'), m_{1b}(\rho_1 \otimes v_{1x}), m_{2b}(\rho_2 \otimes v_{2x})) \in [\![T\tau_b]\!]^T(\Xi\vec{R})(\Delta' r')$.

Let $m_1 = [\![\Delta; \Xi; \Gamma \vdash let\ x \Leftarrow M_{1a}\ in\ M_{1b} : T\tau_b]\!]$ and let $m_2 = [\![\Delta; \Xi; \Gamma \vdash let\ x \Leftarrow M_{2a}\ in\ M_{2b} : T\tau_b]\!]$.

We need to show $(m_1, m_2) \in [\![T\tau_b]\!]^T_{\Xi\Gamma}(\Delta r)$ that is $\forall \vec{R}.\ \forall \Delta' r' \rhd \Delta r.\ \forall \rho_1', \rho_2', \rho_1, \rho_2.\ \big((\rho_1', \rho_2', \rho_1, \rho_2) \in [\![\Gamma]\!]^t(\Xi\vec{R})(\Delta' r') \Rightarrow (m_1(\rho_1'), m_2(\rho_2'), m_1(\rho_1), m_2(\rho_2)) \in [\![T\tau_b]\!]^T(\Xi\vec{R})(\Delta' r')\big)$

$m_1(\rho_1') = \lambda k.\lambda S.\ (m_{1a}(\rho_1'))(\lambda S_0.\lambda d_x.((m_{1b}(\rho_1' \otimes d_x))kS_0)S,$

$m_2(\rho_2') = \lambda k.\lambda S.\ (m_{2a}(\rho_2'))(\lambda S_0.\lambda d_x.((m_{2b}(\rho_2' \otimes d_x))kS_0)S,$

$m_1(\rho_1) = \lambda k.\lambda S.\ (m_{1a}(\rho_1))(\lambda S_0.\lambda d_x.((m_{1b}(\rho_1 \otimes d_x))kS_0)S,$

$m_2(\rho_2) = \lambda k.\lambda S.\ (m_{2a}(\rho_2))(\lambda S_0.\lambda d_x.((m_{2b}(\rho_2 \otimes d_x))kS_0)S.$

Let $\vec{R}, \Delta' r' \rhd \Delta r$ be given and assume $(\rho_1', \rho_2', \rho_1, \rho_2) \in [\![\Gamma]\!]^t(\Xi\vec{R})(\Delta' r')$. Let $\Delta'' r'' \rhd \Delta' r'$ and assume $(k_1', k_2', k_1, k_2) \in [\![\tau_b^\top]\!]^K(\Xi\vec{R})(\Delta'' r'')$ and $(S_1', S_2', S_1, S_2) \in \nabla_S(\Delta'' r'')$.

$(m_1(\rho_1'))k_1'S_1' = (m_{1a}(\rho_1'))(\lambda S_0.\lambda d_x.((m_{1b}(\rho_1' \otimes d_x))k_1'S_0)S_1',$

$(m_1(\rho_1))k_1S_1 = (m_{1a}(\rho_1))(\lambda S_0.\lambda d_x.((m_{1b}(\rho_1 \otimes d_x))k_1S_0)S_1,$

$(m_2(\rho_2'))k_2'S_2' = (m_{2a}(\rho_2'))(\lambda S_0.\lambda d_x.((m_{2b}(\rho_2' \otimes d_x))k_2'S_0)S_2',$

$(m_2(\rho_2))k_2S_2 = (m_{2a}(\rho_2))(\lambda S_0.\lambda d_x.((m_{2b}(\rho_2 \otimes d_x))k_2S_0)S_2.$

By assupmtion $(m_{1a}(\rho_1'), m_{2a}(\rho_2'), m_{1a}(\rho_1), m_{2a}(\rho_2)) \in [\![T\tau_a]\!]^T(\Xi\vec{R})(\Delta' r')$ and $(S_1', S_2', S_1, S_2) \in \nabla_S(\Delta''\ r'')$. So, to prove the required termination-approximations we want to show that

$(\lambda S_0.\lambda d_x.\ ((m_{1b}(\rho_1' \otimes d_x))\ k_1'S_0),\ \lambda S_0.\lambda d_x.\ ((m_{1b}(\rho_1 \otimes d_x))k_1S_0),$

$\lambda S_0.\lambda d_x.\ ((m_{2b}(\rho_2 \otimes d_x))k_2S_0),\ \lambda S_0.\lambda d_x.\ ((m_{2b}(\rho_2' \otimes d_x))k_2'S_0)) \in [\![\tau_a^\top]\!]^K(\Xi\vec{R})(\Delta'' r'').$

Let $\Delta^3 r^3 \rhd \Delta'' r''$ and assume $(S_{10}', S_{20}', S_{10}, S_{20}) \in \nabla_S(\Delta^3 r^3)$ and $(d_1', d_2', d_1, d_2) \in [\![\tau_a]\!]^t(\Xi\vec{R})(\Delta^3 r^3)$.

$(\lambda S_0.\lambda d_x.(m_{1b}(\rho_1' \otimes d_x))k_1'S_0)S_{10}'d_1' = (m_{1b}(\rho_1' \otimes d_1'))k_1'S_{10}',$

$(\lambda S_0.\lambda d_x.(m_{2b}(\rho_2' \otimes d_x))k_2'S_0)S_{20}'d_2' = (m_{2b}(\rho_2' \otimes d_2'))k_2'S_{20}',$

$(\lambda S_0.\lambda d_x.(m_{1b}(\rho_1 \otimes d_x))k_1S_0)S_{10}d_1 = (m_{1b}(\rho_1 \otimes d_1))k_1S_{10},$

$(\lambda S_0.\lambda d_0.(m_{2b}(\rho_2 \otimes d_x))k_2S_0)S_{20}d_2 = (m_{2b}(\rho_2 \otimes d_2))k_2S_{20}.$

By assumptions on d's and ρ's and parameter weakening on ρ's it holds that $((\rho_1' \otimes d_1'), (\rho_2' \otimes d_2'), (\rho_1 \otimes d_1), (\rho_2 \otimes d_2)) \in [\![\Gamma, x : \tau_a]\!]^t(\Xi\vec{R})(\Delta^3 r^3)$. By assumptions on m_b's then $(m_{1b}(\rho_1' \otimes d_1'), m_{2b}(\rho_2' \otimes d_2'), m_{1b}(\rho_1 \otimes d_1), m_{2b}(\rho_2 \otimes d_2)) \in [\![T\tau_b]\!]^T(\Xi\vec{R})(\Delta^3 r^3)$. Since also $(S_{10}', S_{20}', S_{10}, S_{20}) \in \nabla_S(\Delta^3 r^3)$ and by weakening $(k_1', k_2', k_1, k_2) \in [\![\tau_b^\top]\!]^K(\Xi\vec{R})(\Delta^3 r^3)$ then we get the required termination approximation. We can conclude that $(m_1, m_2) \in [\![T\tau_b]\!]^T_{\Xi\Gamma}(\Delta r)$.

- $$\frac{\Delta; \Xi; \Gamma, f : \tau \to T\tau', x : \tau \vdash M : T\tau'}{\Delta; \Xi; \Gamma \vdash rec\ f(x : \tau) = M : \tau \to T\tau'}$$

  Let $m_1 = [\![\Delta; \Xi; \Gamma, f : \tau \to T\tau', x : \tau \vdash M_1 : T\tau']\!]$, and
  let $m_2 = [\![\Delta; \Xi; \Gamma, f : \tau \to T\tau', x : \tau \vdash M_2 : T\tau']\!]$,
  Assume $(m_1, m_2) \in [\![T\tau']\!]^T_{\Xi\Gamma_0}(\Delta r)$, where $\Gamma_0 = \Gamma \cup \{\ f : \tau \to T\tau', x : \tau\ \}$.
  Let $v_1 = [\![\Delta; \Xi; \Gamma \vdash rec\ f(x : \tau) = M_1 : \tau \to T\tau']\!]$ and
  let $v_2 = [\![\Delta; \Xi; \Gamma \vdash rec\ f(x : \tau) = M_2 : \tau \to T\tau']\!]$. To prove: $(v_1, v_2) \in [\![\tau \to T\tau']\!]^t_{\Xi\Gamma}(\Delta r)$. This requires $\forall \vec{R}.\ \forall \Delta' r' \rhd \Delta r.\ \forall (\rho_1', \rho_2', \rho_1, \rho_2) \in [\![\Gamma]\!]^t(\Xi\vec{R})(\Delta' r').\ (v_1(\rho_1'), v_2(\rho_2'), v_1(\rho_1), v_2(\rho_2)) \in [\![(\tau \to T\tau')]\!]^t(\Xi\vec{R})(\Delta' r') = [\![\tau \to T\tau']\!]^t(\Xi\vec{R})(\Delta' r')$.
  If $\rho_1' = \rho_2' = \bot$ then $v_1(\rho_1') = v_2(\rho_2') = \bot$. When $\rho \neq \bot$ we have
  $[\![\Delta; \Xi; \Gamma \vdash rec\ f(x : \tau) =\ M : \tau \to T\tau'\ ]\!]\,\rho =$
  $i \circ in_\to \lfloor fix(\lambda f' \in (V \multimap TV).(\lambda x' \in V.\ [\![\Delta; \Xi; \Gamma, f : \tau \to T\tau', x : \tau \vdash M : T\tau']\!](\rho \otimes f \mapsto i \circ in_\to \lfloor f' \rfloor \otimes x \mapsto x')))\rfloor = i \circ in_\to \lfloor \bigsqcup_{n \in \omega} g_n \rfloor$

where $g_n \in (V \multimap TV),\ g_0 = \bot_{V \multimap TV}$ and
$g_{n+1} = \ \lambda x_0 \in V. \llbracket \Delta; \Xi; \Gamma, f : \tau \to T\tau', x : \tau \vdash M : T\tau' \rrbracket(\rho \otimes f \mapsto i \circ in_{\to}\lfloor g_n \rfloor \otimes x \mapsto x_0)$.
so we have that when $\rho_1' \neq \bot \vee \rho_2' \neq \bot$ then
$v_1(\rho_1) = i \circ in_{\to}\lfloor \bigsqcup_{n\in\omega} g_n^1 \rfloor, v_2(\rho_2) = i \circ in_{\to}\lfloor \bigsqcup_{n\in\omega} g_n^2 \rfloor$.
$v_1(\rho_1') \sqsubseteq i \circ in_{\to}\lfloor \bigsqcup_{n\in\omega} g_n^{1'} \rfloor, v_2(\rho_2') \sqsubseteq i \circ in_{\to}\lfloor \bigsqcup_{n\in\omega} g_n^{2'} \rfloor$ where
$g_0^1 = g_0^2 = g_0^{1'} = g_0^{2'} = \bot_{\mathbb{V} \multimap \mathbb{M}}$ and
$g_{n+1}^1 = \lambda x_0.m_1(\rho_1 \otimes i \circ in_{\to}\lfloor g_n^1 \rfloor \otimes x_0), g_{n+1}^2 = \lambda x_0.m_1(\rho_2 \otimes i \circ in_{\to}\lfloor g_n^2 \rfloor \otimes x_0)$,
$g_{n+1}^{1'} = \lambda x_0.m_1(\rho_1' \otimes i \circ in_{\to}\lfloor g_n^{1'} \rfloor \otimes x_0), g_{n+1}^{2'} = \lambda x_0.m_1(\rho_2' \otimes i \circ in_{\to}\lfloor g_n^{2'} \rfloor \otimes x_0)$
It holds that
$\bigsqcup_{n\in\omega} g_n^1 = (\lambda x' \in V.m_1(\rho_1 \otimes i \circ in_{\to}\lfloor(\bigsqcup_{n\in\omega} g_n^1)\rfloor \otimes x'))$ and
$\bigsqcup_{n\in\omega} g_n^2 = (\lambda x' \in \mathbb{V}.m_2(\rho_2 \otimes i \circ in_{\multimap}\lfloor(\bigsqcup_{n\in\omega} g_n^2)\rfloor \otimes x'))$.
We have $g_0^{1'} \sqsubseteq g_0^1 \sqsubseteq \bigsqcup g_n^1 \wedge g_0^{2'} \sqsubseteq g_0^2 \sqsubseteq \bigsqcup g_n^2$ and from the definition and the assumptions $\rho_1' \sqsubseteq \rho_1 \wedge \rho_2' \sqsubseteq \rho_2$ then by induction $\forall n. g_n^{1'} \sqsubseteq g_n^1 \sqsubseteq \bigsqcup g_n^1 \wedge g_n^{2'} \sqsubseteq g_n^2 \sqsubseteq \bigsqcup g_n^2$.
Also $\forall d_1, d_2.\ \bot_{\mathbb{V}\multimap\mathbb{M}}(d_1) = \bot_{\mathbb{V}\multimap\mathbb{M}}(d_2) = \bot_{\mathbb{M}}$. Since $\forall \Delta r. \forall \vec{R}.\ \forall d_1, d_2 \in V.\ (\bot, \bot, \bigsqcup g_n^1(d_1), \bigsqcup g_n^1(d_2)) \in \llbracket T\tau' \rrbracket^T(\Xi\vec{R})(\Delta r)$, so $(i \circ in_{\to}\lfloor g_0^1 \rfloor, i \circ in_{\to}\lfloor g_0^2 \rfloor, v_1(\rho_1), v_2(\rho_2)) \in \llbracket \tau \to T\tau' \rrbracket^t(\Xi\vec{R})(\Delta' r')$. We will show by induction on n that
$\forall n \in \omega.\ (i \circ in_{\to}\lfloor g_n^{1'} \rfloor, i \circ in_{\to}\lfloor g_n^{2'} \rfloor \mid v_1(\rho_1), v_2(\rho_2)) \in \llbracket \tau \to T\tau' \rrbracket^t(\Xi\vec{R})(\Delta' r')$.
Assume $(i \circ in_{\to}\lfloor g_n^{1'} \rfloor, i \circ in_{\to}\lfloor g_n^{2'} \rfloor \mid v_1(\rho_1), v_2(\rho_2)) \in \llbracket \tau \to T\tau' \rrbracket^t(\Xi\vec{R})(\Delta' r')$.
We have $v_1(\rho_1) = i \circ in_{\to}\lfloor \bigsqcup_{n\in\omega} g_n^1 \rfloor$ and $v_2(\rho_2) = i \circ in_{\to}\lfloor \bigsqcup_{n\in\omega} g_n^2 \rfloor$.
To show that $(i \circ in_{\to}\lfloor g_{n+1}^{1'} \rfloor, i \circ in_{\to}\lfloor g_{n+1}^{2'} \rfloor \mid v_1(\rho_1), v_2(\rho_2)) \in \llbracket \tau \to T\tau' \rrbracket^t(\Xi\vec{R})(\Delta' r')$ we must have $\forall \Delta'' r'' \rhd \Delta' r'. \forall (d_1', d_2', d_1, d_2) \in \llbracket \tau \rrbracket^t(\Xi\vec{R})(\Delta'' r'')$. $(\lambda x_0.m_1(\rho_1' \otimes i \circ in_{\to}\lfloor g_n^{1'} \rfloor \otimes x_0)d_1'$, $\lambda x_0.m_2(\rho_{2'} \otimes i \circ in_{\to}\lfloor g_n^{2'} \rfloor \otimes x_0)d_2', \bigsqcup g_n^1(d_1), \bigsqcup g_n^2(d_2)) \in \llbracket T\tau' \rrbracket^T(\Xi\vec{R})\ (\Delta'' r'')$. That is $(m_1(\rho_1' \otimes i \circ in_{\multimap}\lfloor g_n^{1'} \rfloor \otimes d_1'),\ m_2(\rho_2' \otimes i \circ in_{\multimap}\lfloor g_n^{2'} \rfloor \otimes d_2'), m_1(\rho_1 \otimes i \circ in_{\multimap}\lfloor(\bigsqcup_{n\in\omega} g_n^1)\rfloor \otimes d_1, m_2(\rho_2 \otimes i \circ in_{\multimap}\lfloor(\bigsqcup_{n\in\omega} g_n^2)\rfloor \otimes d_2, T\tau',\ p'') \in \nabla_M$
Since by assumption $(m_1, m_2) \in \llbracket T\tau' \rrbracket^T_{\Xi\Gamma_0}(\Delta r)$ and we have $(i \circ in_{\to}\lfloor g_n^{1'} \rfloor, i \circ in_{\to}\lfloor g_n^{2'} \rfloor, i \circ in_{\to}\lfloor \bigsqcup_{n\in\omega} g_n^1 \rfloor, i \circ in_{\to}\lfloor \bigsqcup_{n\in\omega} g_n^2 \rfloor) \in \llbracket \tau \to T\tau' \rrbracket^t(\Xi\vec{R})(\Delta' r')$ and $(d_1', d_2', d_1, d_2) \in \llbracket \tau \rrbracket^t\ (\Xi\vec{R})$ $(\Delta'' r'')$ it holds that $(m_1(\rho_1' \otimes i \circ in_{\to}\lfloor g_n^{1'} \rfloor \otimes d_1'),\ m_2(\rho_2' \otimes i \circ in_{\to}\lfloor g_n^{2'} \rfloor \otimes d_2'),\ m_1(\rho_1 \otimes i \circ in_{\to}\lfloor \bigsqcup_{n\in\omega} g_n^1 \rfloor \otimes d_1),\ m_2(\rho_2 \otimes i \circ in_{\to}\lfloor \bigsqcup_{n\in\omega} g_n^2 \rfloor \otimes d_2)) \in \llbracket T\tau' \rrbracket^T(\Xi\vec{R})(\Delta'' r'')$. So $(i \circ in_{\to}\lfloor g_{n+1}^{1'} \rfloor, i \circ in_{\to}\lfloor g_{n+1}^{2'} \rfloor, i \circ in_{\to}\lfloor \bigsqcup_{n\in\omega} g_n^1 \rfloor, i \circ in_{\to}\lfloor \bigsqcup_{n\in\omega} g_n^2 \rfloor) = (i \circ in_{\to}\lfloor g_{n+1}^{1'} \rfloor, i \circ in_{\to}\lfloor g_{n+1}^{2'} \rfloor$, $v_1(\rho_1), v_2(\rho_2)) \in \llbracket \tau \to T\tau' \rrbracket^t(\Xi\vec{R})(\Delta' r')$.
We have shown that $\forall n \in \omega.\ (i \circ in_{\multimap}\lfloor g_n^{1'} \rfloor, i \circ in_{\multimap}\lfloor g_n^{2'} \rfloor, v_1(\rho_1), v_2(\rho_2)) \in \llbracket \tau \to T\tau' \rrbracket^t(\Xi\vec{R})(\Delta' r')$.
Then since ∇_V is admissible also $(\bigsqcup i \circ in_{\multimap}\lfloor g_n^{1'} \rfloor, \bigsqcup i \circ in_{\multimap}\lfloor g_n^{2'} \rfloor, v_1(\rho_1), v_2(\rho_2)) \in \llbracket \tau \to T\tau' \rrbracket^t$ $(\Xi\vec{R})(\Delta' r')$. So $(v_1(\rho_1'), v_2(\rho_2'), v_1(\rho_1), v_2(\rho_2)) \in \llbracket \tau \to T\tau' \rrbracket^t(\Xi\vec{R})(\Delta' r')$,
hence $(v_1, v_2) \in \llbracket \tau \to T\tau' \rrbracket^t_{\Xi\Gamma}(\Delta r)$.

- $$\frac{\Delta; \Xi; \Gamma \vdash V_a : \tau \to T\tau' \quad \Delta; \Xi; \Gamma \vdash V_b : \tau}{\Delta; \Xi; \Gamma \vdash V_a V_b : T\tau'}$$

Let $v_{1a} = \llbracket \Delta; \Xi; \Gamma \vdash V_{1a} : \tau \to T\tau' \rrbracket$, $v_{2a} = \llbracket \Delta; \Xi; \Gamma \vdash V_{2a} : \tau \to T\tau' \rrbracket$.
Let $v_{1b} = \llbracket \Delta; \Xi; \Gamma \vdash V_{1b} : \tau \rrbracket$, $v_{2b} = \llbracket \Delta; \Xi; \Gamma \vdash V_{2b} : \tau \rrbracket$.
Assume $(v_{1a}, v_{2a}) \in \llbracket \tau \to T\tau' \rrbracket^t_{\Xi\Gamma}(\Delta r)$ and $(v_{1b}, v_{2b}) \in \llbracket \tau \rrbracket^t_{\Xi\Gamma}(\Delta r)$.
Let $m_1 = \llbracket \Delta; \Xi; \Gamma \vdash V_{1a}V_{1b} : T\tau' \rrbracket$ and $m_2 = \llbracket \Delta; \Xi; \Gamma \vdash V_{2a}V_{2b} : T\tau' \rrbracket$.
We aim to show $(m_1, m_2) \in \llbracket T\tau' \rrbracket^T_{\Xi\Gamma}(\Delta r)$. Let $\vec{R} \in \text{bParAdmRel}^n, \Delta' r' \rhd \Delta r$ and assume $(\rho_1', \rho_2', \rho_1, \rho_2) \in \llbracket \Gamma \rrbracket^t(\Xi\vec{R})(\Delta' r')$. We want to show $(m_1(\rho_1'), m_2(\rho_2'), m_1(\rho_1), m_2(\rho_2)) \in (\llbracket T\tau' \rrbracket^T$ $(\Xi\vec{R})(\Delta' r')$. The assumption $(v_{1a}, v_{2a}) \in \llbracket \tau \to T\tau' \rrbracket^t_{\Xi\Gamma}(\Delta r)$ implies either $(v_{1a}(\rho_1') = v_{2a}(\rho_2') =$

$\bot$) or $\exists f_1', f_2', f_1, f_2.((v_{1a}(\rho_1')) = \bot \wedge f_1' = \bot) \vee (v_{1a}(\rho_1')) = in_{\rightarrow}\lfloor f_1' \rfloor) \wedge ((v_{2a}(\rho_2')) = \bot \wedge f_2' = \bot) \vee (v_{2a}(\rho_2')) = in_{\rightarrow}\lfloor f_2' \rfloor \wedge (v_{1a}(\rho_1)) = in_{\rightarrow}\lfloor f_1 \rfloor \wedge (v_{2a}(\rho_2)) = in_{\rightarrow}\lfloor f_2 \rfloor \wedge \forall \Delta'' r'' \rhd \Delta' r'.\forall (d_1', d_2', d_1, d_2) \in [\![\tau]\!]^t(\Xi\vec{R})(\Delta'' r'').\ ((f_1'd_1', f_2'd_2', f_1 d_1, f_2 d_2) \in [\![T\tau']\!]^T(\Xi\vec{R})(\Delta'' r'')$.

In the first case $m_1(\rho_1') = m_2(\rho_2') = \bot$, and so $(m_1(\rho_1'), m_2(\rho_2'), (m_1(\rho_1), m_2(\rho_2)) \in [\![T\tau']\!]^T(\Xi\vec{R})$ $(\Delta' r')$. In the second case we have $m_1(\rho_1') = f_1'(v_{1b}(\rho_1'))$, $m_2(\rho_2') = f_2'(v_{2b}(\rho_2'))$, $m_1(\rho_1) = f_1(v_{1b}(\rho_1))$, $m_2(\rho_2) = f_2(v_{0b}(\rho_2))$ and it follows from the assumption $(v_{1b}, v_{2b}) \in [\![\tau]\!]^t_{\Xi\gamma}(\Delta r)$ that $(v_{1b}(\rho_1'), v_{2b}(\rho_2'),\ v_{1b}(\rho_1), v_{2b}(\rho_2)) \in [\![\tau]\!]^t(\Xi\vec{R})(\Delta' r')$. And so $(f_1'(v_{1b}(\rho_1')), f_2'(v_{2b}(\rho_2')), f_1(v_{1b}(\rho_1)), f_2(v_{2b}(\rho_2))) \in [\![T\tau']\!]^T\ (\Xi\vec{R})(\Delta' r')$. That is $(m_1(\rho_1'), m_2(\rho_2'), m_1(\rho_1), m_2(\rho_2)) \in [\![T\tau']\!]^T(\Xi\vec{R})(\Delta' r')$. We conclude $(m_1, m_2) \in [\![T\tau']\!]^T_{\Xi\Gamma}(\Delta r)$.

- $$\frac{}{\Delta; \Xi; \Gamma \vdash l : \tau\,\mathsf{ref}}\ (\Delta l : \tau)$$

  Let $v = [\![\Delta; \Xi; \Gamma \vdash l : \tau\,\mathsf{ref}]\!]$, and assume $\Delta l : \tau$.
  To show: $(v, v) \in [\![\tau\,\mathsf{ref}]\!]^t_{\Xi\Gamma}(\Delta r)$, this requires $\forall \vec{R}.\ \forall \Delta' r' \rhd \Delta r.\ \forall \rho_1', \rho_2', \rho_1, \rho_2$.
  $(\rho_1', \rho_2', \rho_1, \rho_2) \in [\![\Gamma]\!]^t(\Xi\vec{R})(\Delta' r') \Rightarrow (v_1(\rho_1'), v_2(\rho_2'), v_1(\rho_1), v_2(\rho_2)) \in [\![\tau\,\mathsf{ref}]\!]^t(\Xi\vec{R})(\Delta' r')$.
  $\Delta l : \tau$ implies that τ is closed.
  If $\rho_1' = \rho_2' = \bot \Rightarrow (v_1(\rho_1')) = (v_2(\rho_2')) = \bot$ and we are done. Else $\rho_1' \sqsubseteq \rho_1 \neq \bot \wedge \rho_2' \sqsubseteq \rho_2 \neq \bot$, then $v_1(\rho_1')) \sqsubseteq (v_1(\rho_1)) \wedge v_2(\rho_2')) \sqsubseteq (v_2(\rho_2)) \wedge v_1(\rho_1) = v_2(\rho_2) = in_{\mathsf{ref}}\, l$. Then since $\Delta' r' \rhd \Delta r$ so $\Delta' l : \tau$ and it holds that $(v_1(\rho_1'), v_2(\rho_2'), v_1(\rho_1), v_2(\rho_2)) \in [\![\tau\,\mathsf{ref}]\!]^t(\Xi\vec{R})(\Delta' r')$. So $(v, v) \in [\![\tau\,\mathsf{ref}]\!]^t_{\Xi\Gamma}(\Delta r)$.

■

Theorem 23 (Fundamental Theorem)
For all parameters Δr it holds that

- *if* $\Delta; \Xi; \Gamma \vdash V : \tau$ *then*
  $([\![\Delta; \Xi; \Gamma \vdash V : \tau]\!], [\![\Delta; \Xi; \Gamma \vdash V : \tau]\!]) \in [\![\tau]\!]^t_{\Xi\Gamma}(\Delta r)$,
- *if* $\Delta; \Xi; \Gamma \vdash M : T\tau$ *then*
  $([\![\Delta; \Xi; \Gamma \vdash M : T\tau]\!], [\![\Delta; \Xi; \Gamma \vdash M : T\tau]\!]) \in [\![T\tau]\!]^T_{\Xi\Gamma}(\Delta r)$.

The Fundamental Theorem follows from Lemma 22.

Lemma 24
For all Δr and $k = [\![\Delta; \vdash val\ x : (x : \tau)^\top]\!]$, we have that $(k, k, k, k) \in [\![\tau^\top]\!]^K()(\Delta r)$. Moreover, if $S \in [\![\Delta \vdash \Sigma]\!]$ then $(S, S, S, S) \in \nabla_S(\Delta T)$.

Proof:

- Let $k = [\![\Delta; \vdash val\ x : (x : \tau)^\top]\!]^K$, then $k =$
  $\lambda S.\lambda d.\,[\![\Delta;\ ; x : \tau \vdash val\ x : T\tau]\!]\{x \mapsto d\}(\lambda S'.(\lambda d'.\top)_\bot)_\bot S =$
  $\lambda S.\lambda d.(\lambda S'.(\lambda d'.\top)_\bot)_\bot S d$
  Let Δr be a basic parameter. Let $\Delta' r' \rhd \Delta r$. Assume $(s_1', s_2', s_1, s_2) \in \nabla_S(\Delta' r')$ and $(v_1', v_2', v_1, v_2) \in [\![\tau]\!]^t()(\Delta' r')$. We need to prove
  $ks_1'v_1' = \top \Rightarrow ks_2 v_2 = \top$ and $ks_2'v_2' = \top \Rightarrow ks_1 v_1 = \top$.
  It holds that $ks_1'v_1' \sqsubseteq ks_1 v_1$ and $ks_2'v_2' \sqsubseteq ks_2 v_2$. It follows from the assumptions that $s_1' = s_2' = \bot \vee (s_1 \neq \bot \wedge s_2 \neq \bot)$ and $v_1' = v_2' = \bot \vee (v_1 \neq \bot \wedge v_2 \neq \bot)$
  If $s_1' = s_2' = \bot$ or $v_1' = v_2' = \bot$ then $ks_1'v_1' = ks_2'v_2' = \bot$ and the implications holds trivilally. If this is not the case then it holds that $s_1 \neq \bot \wedge s_2 \neq \bot \wedge v_1 \neq \bot \wedge v_1 \neq \bot$. So $ks_1 v_1 = (\lambda S.\lambda d.(\lambda S'.(\lambda d'.\top)_\bot)_\bot S d) s_1 v_1 = \top \wedge ks_2 v_2 = (\lambda S.\lambda d.(\lambda S'.(\lambda d'.\top)_\bot)_\bot S d) s_2 v_2 = \top$. Again the implications holds. So we conclude $(k, k, k, k) \in [\![\tau^\top]\!]^K()(\Delta r)$.

- Let $\Delta \vdash \Sigma$ be a typing judgement for a store Σ. Let $S \in [\![\Delta \vdash \Sigma]\!] = \{S' \mid \forall (l : \tau) \in \Delta.\ S'l = [\![\Delta;;\vdash \Sigma l : \tau]\!]\}$. We want to show $(S, S, S, S) \in \nabla_S(\Delta T)$. This requires $S = \bot$ or $S \neq \bot \wedge \forall (l : \tau) \in \Delta.\ (Sl, Sl, Sl, Sl) \in [\![\tau]\!]^t()(\Delta T)$. This follows by fundamental lemma from the assumptions $Sl = [\![\Delta;;\vdash \Sigma l : \tau]\!]$.

■

The following theorem expresses that we can show two computations or two values to be contextually equivalent by showing that their denotations are related under a parameter ΔT, which does not require that any hidden invariants hold for states. The computations may themselves be able to build up local state invariants and a proof of relatedness will often require one to express these invariants.

Theorem 25 (Contextual Equivalence)
Let $C[\_] : (\Delta; \Xi; \Gamma \vdash \gamma) \Rightarrow (\Delta;;\vdash T\tau)$ *be a context. If* $\Delta; \Xi; \Gamma \vdash G_1 : \gamma$ *and* $\Delta; \Xi; \Gamma \vdash G_2 : \gamma$ *and*

$$([\![\Delta; \Xi; \Gamma \vdash G_1 : \gamma]\!], [\![\Delta; \Xi; \Gamma \vdash G_2 : \gamma]\!]) \in [\![\gamma]\!]^t_{\Xi\Gamma}(\Delta T)$$

then, for all $\Sigma : \Delta$,

$$\Sigma, let\ x \Leftarrow C[G_1]\ in\ val\ x \downarrow \Longleftrightarrow \Sigma, let\ x \Leftarrow C[G_2]\ in\ val\ x \downarrow .$$

Proof: Let $C[\_] : (\Delta; \Xi; \Gamma \vdash \gamma) \Rightarrow (\Delta;;\vdash T\tau)$ be a context, and assume $([\![\Delta; \Xi; \Gamma \vdash G_1 : \gamma]\!], [\![\Delta; \Xi; \Gamma \vdash G_2 : \gamma]\!]) \in [\![\gamma]\!]^t_{\Xi\Gamma}(\Delta T)$. By induction over the structure of $C[\_]$ and by lemma 22 it holds that $([\![\Delta;;\vdash C[G_1] : T\tau]\!], [\![\Delta;;\vdash C[G_2] : T\tau]\!]) \in [\![T\tau]\!]^T_{\{\}}(\Delta T)$. Let $k = [\![\Delta;\vdash val\ x : (x : \tau)^\top]\!]$. By lemma 24 $(k, k, k, k) \in [\![\tau^\top]\!]^K()(\Delta T)$ and $\forall S \in [\![\Delta \vdash \Sigma]\!].\ (S, S, S, S) \in \nabla_S(\Delta T)$. So $[\![\Delta;;\vdash C[G_1] : T\tau]\!]\,[\![\Delta;\vdash val\ x : (x : \tau)^\top]\!]\,S = \top \Leftrightarrow [\![\Delta;;\vdash C[G_2] : T\tau]\!]\,[\![\Delta;\vdash val\ x : (x : \tau)^\top]\!]\,S = \top$. The lemma then follows by soundness and adequacy of the denotational semantics. ■

5 Example

In the following example we consider a polymorphically typed client c, which uses an abstract stack of integers. The client is polymorphic in the type representing the stack and thus we expect that the client produces the same result for two different, but related, implementations of a stack. And, indeed, we are able to prove so using our proof method established in the previous section. In one case, the client is given a stack, which is implemented using ML-style lists and in the other case, the client is given a stack, which is implemented using mutable lists.

$c : \forall\alpha.T\big(\big((1 \to T\alpha) \times (int \times \alpha \to T\alpha) \times (\alpha \to T(1 + (int \times \alpha)))\big) \to T(1 + int)\big) = \forall\alpha.T(\tau \to T(1 + int))$

$c = \Lambda\alpha.val\ rec\ f(empty, push, pop) = \big($
$let\ s_0 \Leftarrow empty()\ in$
$let\ s_1 \Leftarrow push(1, s_0)\ in$
$let\ x \Leftarrow pop(s_1)\ in$
$case\ x\ of\ in_1 x_1 \Rightarrow val\ in_1();\ in_2 x_2 \Rightarrow in_2(\pi_1 x_2)\ \big)$

$\alpha_1 = \mu\beta.1 + (int \times \beta)$
$\alpha_2 = \big(\mu\beta.1 + (int \times \beta\ ref)\big)\ ref$

$e_1 = rec\ E_1(a : 1) = val\ (fold\ in_1())$
$e_2 = rec\ E_2(a : 1) = let\ z \Leftarrow val\ (fold\ in_1())\ in\ ref\ z$

$push_1 = rec\, Push_1(x : int \times \alpha) = val\ (fold\, in_2 x)$
$push_2 = rec\, Push_2(x : int \times \alpha) = ref\ (fold\, in_2 x)$

$pop_1 = rec\, Pop_1(x : \alpha) = let\ z \Leftarrow unfold\ x\ in\ z$
$pop_2 = rec\, Pop_2(x : \alpha) = let\ y \Leftarrow !x\ in\ letz \Leftarrow unfold\ y\ in\ z$

$p_1 = let\ x \Leftarrow c(\alpha_1)\ in\ x(empty_1, push_1, pop_1) \qquad p_1 : T(1 + int)$
$p_2 = let\ x \Leftarrow c(\alpha_2)\ in\ x(empty_2, push_2, pop_2) \qquad p_2 : T(1 + int)$

We define some local parameters: For each $l \in \mathbb{L}$ define local parameter $r_l : (S_2 l = in_\mu in_\oplus in_1 in_{\mathbf{1}} *),\ A_\emptyset, A_{\{l\}}$.
For each l, l' with $l \neq l' \in \mathbb{L}$ for each $n \in Z$ define $r_{ll'n} : (S_2 l = in_\mu in_\oplus in_2 in_\otimes(in_Z n, in_\mathbb{L} l')),\ A_\emptyset, A_{\{l\}}$.

Further define $\hat{R} \in$ bParAdmRel: $(d_1', d_2', d_1, d_2) \in \hat{R}(\Delta r) \iff$
$d_1' = d_2' = \perp \vee$
$\big(d_1' \sqsubseteq d_1 \wedge d_2' \sqsubseteq d_2 \wedge$
$\big(\exists l.\ r_l \in r \wedge (d_1, d_2) = (in_\mu in_\oplus in_1 in_{\mathbf{1}} *, in_\mathbb{L} l)\ \vee$
$\exists l_0, l_1$ with $l_0 \neq l_1$ and $\exists n_1 \in Z \wedge r \supseteq \{r_{l_0}, r_{l_1 l_0 n_1}\} \wedge$
$\quad (d_1, d_2) = (in_\mu in_\oplus in_2 in_\otimes(in_Z n_k, in_\mu in_\oplus in_1 in_{\mathbf{1}} *)),\ in_\mathbb{L} l_1) \vee$
$\exists k > 1 \wedge \exists l_0 \ldots l_k \in \mathbb{L}$ pairwise different $\wedge \exists n_1 \ldots n_k \in Z \wedge r \supseteq \{r_{l_0}, r_{l_1 l_0 n_1}, r_{l_2 l_1 n_2} \ldots r_{l_k l_{k-1} n_k}\}\ \wedge$
$(d_1, d_2) = (in_\mu in_\oplus in_2 in_\otimes(in_Z n_k, in_\mu in_\oplus in_2 in_\otimes(in_Z n_{k-1}, \ldots\ldots in_\mu in_\oplus in_1 in_{\mathbf{1}} *) \ldots),\ in_\mathbb{L} l_k)\big)\big)$

(*) It follows from the definition of $\hat{R}$ that if $(d_1', d_2', d_1, d_2) \in \hat{R}(\Delta^0 r^0)$ and $d_1 \neq in_\mu in_\oplus in_1 in_{\mathbf{1}} *$ then $\exists l_0 \ldots l_k \in \mathbb{L}$ pairwise different s.t. $d_2 = in_\mathbb{L} l_k \wedge \exists n_1 \ldots n_k \in Z \wedge r \supseteq \{r_{l_0}, r_{l_1 l_0 n_1}, r_{l_2 l_0 n_2} \ldots r_{l_k l_{k-1} n_k}\}$ and then $(in_\mu in_\oplus in_2 in_\otimes(in_Z n_j, in_\mu in_\oplus \ldots\ldots in_\mu in_\oplus in_1 in_{\mathbf{1}} *) \ldots),\ in_\mathbb{L} l_j) \in \hat{R}(\Delta^0 r^0)$ for all $j \in 1 \ldots k$, and
$((in_\mu in_\oplus in_1 in_{\mathbf{1}} *, in_\mathbb{L} l_0) \in \hat{R}(\Delta^0 r^0)$.

To show: $(\llbracket ;; \vdash p_1 \rrbracket, \llbracket ;; \vdash p_2 \rrbracket) \in \llbracket T(1 + int) \rrbracket^t_{\emptyset\emptyset}(\Delta_\emptyset T)$
Let $\Delta' r' \rhd \Delta_\emptyset T$ and assume $(k_1', k_2', k_1, k_2) \in \llbracket (1 + int)^\top \rrbracket^t()(\Delta' r')$ and $(s_1', s_2', s_1, s_2) \in \nabla_S(\Delta' r')$.

$\llbracket ;; \vdash p_1 \rrbracket\, k_1 s_1 = \llbracket ;; \vdash c(\alpha_1) \rrbracket(\lambda S^0 \lambda d^0.\ \llbracket ; \alpha; x \vdash x(empty_1, push_1, pop_1) \rrbracket\{x \mapsto d^0\} k_1 S^0) s_1 =$
$\llbracket ; \alpha; \vdash val\ rec\ f \rrbracket(\lambda S^0 \lambda d^0.\ \llbracket ; \alpha; x \vdash x(empty_1, push_1, pop_1) \rrbracket\{x \mapsto d^0\} k_1 S^0) s_1$

$\llbracket ;; \vdash p_2 \rrbracket\, k_2 s_2 = \llbracket ;; \vdash c(\alpha_2) \rrbracket(\lambda S^0 \lambda d^0.\ \llbracket ; \alpha; x \vdash x(empty_2, push_2, pop_2) \rrbracket\{x \mapsto d^0\} k_2 S^0) s_2 =$
$\llbracket ; \alpha; \vdash val\ rec\ f \rrbracket(\lambda S^0 \lambda d^0.\ \llbracket ; \alpha; x \vdash x(empty_2, push_2, pop_2) \rrbracket\{x \mapsto d^0\} k_2 S^0) s_2$

By assumption $(s_1', s_2', s_1, s_2) \in \nabla_S(\Delta' r')$ and by fundamental lemma
$(\llbracket ; \alpha; \vdash val\ rec\ f \rrbracket, \llbracket ; \alpha; \vdash val\ rec\ f \rrbracket) \in \llbracket \tau \to T(1 + int) \rrbracket^t_{\alpha\emptyset}(\Delta_\emptyset r')$, and it follows that
$\forall R \in$ bParAdmRel. $(\llbracket ; \alpha; \vdash val\ rec\ f \rrbracket, \llbracket ; \alpha; \vdash val\ rec\ f \rrbracket,\ \llbracket ; \alpha; \vdash val\ rec\ f \rrbracket, \llbracket ; \alpha; \vdash val\ rec\ f \rrbracket) \in \llbracket \tau \to T(1 + int) \rrbracket^t(\alpha R)(\Delta' r')$.
So also for $R = \hat{R}$, and we get the required termination approximation if
$((\lambda S^0 \lambda d^0.\ \llbracket ; \alpha; x \vdash x(empty_1, push_1, pop_1) \rrbracket\{x \mapsto d^0\} k_1 S^0),$
$(\lambda S^0 \lambda d^0.\ \llbracket ; \alpha; x \vdash x(empty_2, push_2, pop_2) \rrbracket\{x \mapsto d^0\} k_2 S^0),$
$(\lambda S^0 \lambda d^0.\ \llbracket ; \alpha; x \vdash x(empty_1, push_1, pop_1) \rrbracket\{x \mapsto d^0\} k_1 S^0),$
$(\lambda S^0 \lambda d^0.\ \llbracket ; \alpha; x \vdash x(empty_2, push_2, pop_2) \rrbracket\{x \mapsto d^0\} k_2 S^0)) \in \llbracket (\tau \to T(1 + int))^\top \rrbracket^t(\alpha\, \hat{R})(\Delta' r')$.

To show this let $\Delta^2 r^2 \rhd \Delta' r'$ and assume $(S_1', S_2', S_1, S_2) \in \nabla_S(\Delta^2 r^2)$ and
$(d_1', d_2', d_1, d_2) \in \llbracket \tau \to T(1 + int) \rrbracket^t\ (\alpha\, \hat{R})(\Delta^2 r^2)$. Then $d_1' = d_2' = \perp$ or $\exists g_1', g_2', g_1, g_2.\ ((d_1' = \perp \wedge g_1' = \perp) \vee d_1' = in_{\multimap} \lfloor g_1' \rfloor) \wedge ((d_2' = \perp \wedge g_2' = \perp) \vee d_2' = in_{\multimap} \lfloor g_2' \rfloor) \wedge (d_1 = in_{\multimap} \lfloor g_1' \rfloor) \wedge (d_2 = in_{\multimap} \lfloor d_2 \rfloor) \wedge \forall \Delta^3 r^3 \rhd \Delta^2 r^2.\ \forall (v_1', v_2', v_1, v_2) \in \llbracket \tau \rrbracket^t(\alpha\, \hat{R})(\Delta^3 r^3).\ (g_1' v_1', g_2' v_2', g_1 v_1, g_2 v_2) \in \llbracket T(1 + int) \rrbracket^t(\alpha\, \hat{R})(\Delta^3 r^3)$

$(\lambda S^0 \lambda d^0.\, \llbracket ;\alpha; x \vdash x(\mathit{empty}_1, \mathit{push}_1, \mathit{pop}_1)\rrbracket\{x \mapsto d^0\}k_1 S^0)S_1 d_1 =$
$\llbracket ;\alpha; x \vdash x(\mathit{empty}_1, \mathit{push}_1, \mathit{pop}_1)\rrbracket\{x \mapsto d_1\}k_1 S_1 = g_1(\llbracket ;\alpha; \vdash (\mathit{empty}_1, \mathit{push}_1, \mathit{pop}_1)\rrbracket)k_1 S_1$
$(\lambda S^0 \lambda d^0.\, \llbracket ;\alpha; x \vdash x(\mathit{empty}_2, \mathit{push}_2, \mathit{pop}_2)\rrbracket\{x \mapsto d^0\}k_1 S^0)S_2 d_2 =$
$\llbracket ;\alpha; x \vdash x(\mathit{empty}_2, \mathit{push}_2, \mathit{pop}_2)\rrbracket\{x \mapsto d_2\}k_2 S_2 = g_2(\llbracket ;\alpha; \vdash (\mathit{empty}_2, \mathit{push}_2, \mathit{pop}_2)\rrbracket)k_2 S_2$.
So by parameter weakening for the continuations it suffices to show that
$(\llbracket ;\alpha; \vdash (\mathit{empty}_1, \mathit{push}_1, \mathit{pop}_1)\rrbracket, \llbracket ;\alpha; \vdash (\mathit{empty}_2, \mathit{push}_2, \mathit{pop}_2)\rrbracket,$
$\llbracket ;\alpha; \vdash (\mathit{empty}_1, \mathit{push}_1, \mathit{pop}_1)\rrbracket, \llbracket ;\alpha; \vdash (\mathit{empty}_2, \mathit{push}_2, \mathit{pop}_2)\rrbracket) \in \llbracket \tau \rrbracket^t(\alpha\hat{R})(\Delta^2 r^2)$. This requires that we can show that it holds that each of $\mathit{empty}'s, \mathit{push}'s$ and $\mathit{pop}'s$ are related under $(\alpha\hat{R})(\Delta^2 r^2)$.

$\mathit{empty}'s$: Let $(\Delta^3 r^3) \rhd (\Delta^2 r^2)$ and assume $(K_1', K_2', K_1, K_2) \in \llbracket \alpha \rrbracket^t(\alpha\hat{R})(\Delta^3 r^3)$ and $(S_1', S_2', S_1, S_2) \in \nabla_S(\Delta^3 r^3)$.
$\llbracket ;\alpha; \vdash \mathit{empty}_1\rrbracket(in_{\mathbf{1}}*)K_1 S_1 = K_1 S_1(in_\mu in_\oplus in_1(in_{\mathbf{1}}*))$
$\llbracket ;\alpha; \vdash \mathit{empty}_2\rrbracket(in_{\mathbf{1}}*)K_2 S_2 = K_2(S_2[l \mapsto in_\mu in_\oplus in_1 in_{\mathbf{1}}*])(in_{\mathbb{L}} l)$ where l is fresh.
Let $\Delta^4 r^4 = \Delta^3(r^3 \cup \{r_l\})$. Since by assumption $(S_1', S_2', S_1, S_2) \in \nabla_S(\Delta^3 r^3)$ and l is fresh then $(S_1', S_2'[l \mapsto in_\mu in_\oplus in_1 in_{\mathbf{1}}*], S_1, S_2[l \mapsto in_\mu in_\oplus in_1 in_{\mathbf{1}}*]) \in \nabla_S(\Delta^4 r^4)$.
Also $(in_\mu in_\oplus in_1 in_{\mathbf{1}}*,\ in_{\mathbb{L}} l,\ in_\mu in_\oplus in_1 in_{\mathbf{1}}*,\ in_{\mathbb{L}} l) \in \llbracket \alpha \rrbracket^t(\alpha\hat{R})(\Delta^4 r^4)$. So we have related continuations applied to states and values related in an extension, so $(\mathit{empty}_1, \mathit{empty}_2, \mathit{empty}_1, \mathit{empty}_2) \in \llbracket 1 \to T\alpha \rrbracket^t(\alpha\hat{R})(\Delta^2 r^2)$.

$\mathit{push}'s$: Let $(\Delta^4 r^4) \rhd (\Delta^3 r^3) \rhd (\Delta^2 r^2)$ and assume $(K_1', K_2', K_1, K_2) \in \llbracket \alpha \rrbracket^t(\alpha\hat{R})(\Delta^4 r^4)$ and $(S_1', S_2', S_1, S_2) \in \nabla_S(\Delta^4 r^4)$. Assume $(V_1', V_2', V_1, V_2) \in \llbracket \mathit{int} \times \alpha \rrbracket^t(\alpha\hat{R})(\Delta^3 r^3)$.
It must hold that $V_1' = V_2' = \bot$ or $\exists n, d_1', d_2', d_1, d_2.\ V_1' \sqsubseteq (in_Z n, d_1') \sqsubseteq V_1 = (in_Z n, d_1) \neq \bot \wedge V_2' \sqsubseteq (in_Z n, d_2') \sqsubseteq V_2 = (in_Z n, d_2) \neq \bot \wedge (d_1', d_2', d_1, d_2) \in \llbracket \alpha \rrbracket^t(\alpha\hat{R})(\Delta^3 r^3)$. Further by the definition of $\hat{R}$ it holds that if $V_1' \neq \bot \vee V_2' \neq \bot$ then $\exists k \geq 0, l_0 \ldots l_k, n_1 \ldots n_k.\ r^3 \supseteq \{r_{l_0}, r_{l_1 l_0 n_1} \ldots r_{l_k l_0 n_k}\}$ and if $k = 0$ then $d_1 = in_\mu in_\oplus in_1 in_{\mathbf{1}}*$ and $d_2 = in_{\mathbb{L}} l_0$, if $k > 0$ then $d_1 = in_\mu in_\oplus in_2 in_\otimes(in_Z n_k, in_\mu \ldots)$ and $d_2 = in_{\mathbb{L}} l_k$.
$\llbracket ;\alpha; \vdash \mathit{push}_1\rrbracket\, V_1 K_1 S_1 = K_1 S_1(in_\mu in_\oplus in_2 in_\otimes V_1)$.
$\llbracket ;\alpha; \vdash \mathit{push}_2\rrbracket\, V_2 K_2 S_2 = K_2 S_2[l' \mapsto in_\mu in_\oplus in_2 V_2](in_{\mathbb{L}} l')$ where l' is fresh.
Let $\Delta^5 r^5 = \Delta^4(r^4 \cup r_{l' l_k n})$. Then since l' is fresh and by assumptions on the $V's$ and by definition of $\hat{R}$ it holds that $(in_\mu in_\oplus in_2 in_\otimes V_1,\ in_{\mathbb{L}} l',\ in_\mu in_\oplus in_2 in_\otimes V_1,\ in_{\mathbb{L}} l') \in \hat{R}(\Delta^5 r^5)$, and also the updated states are related $(S_1', S_2'[l' \mapsto in_\mu in_\oplus in_2 V_2], S_1, S_2[l' \mapsto in_\mu in_\oplus in_2 V_2]) \in \nabla_S(\Delta^5 r^5)$. So we have related continuations applied to states and values related in an extension, so $(\mathit{push}_1, \mathit{push}_2, \mathit{push}_1, \mathit{push}_2) \in \llbracket (\mathit{int} \times \alpha) \to T\alpha \rrbracket^t(\alpha\hat{R})(\Delta^2 r^2)$.

$\mathit{pop}'s$: Let $(\Delta^4 r^4) \rhd (\Delta^3 r^3) \rhd (\Delta^2 r^2)$ and assume $(K_1', K_2', K_1, K_2) \in \llbracket \alpha \rrbracket^t(\alpha\hat{R})(\Delta^4 r^4)$ and $(S_1', S_2', S_1, S_2) \in \nabla_S(\Delta^4 r^4)$. Assume $(V_1', V_2', V_1, V_2) \in \llbracket \alpha \rrbracket^t(\alpha\hat{R})(\Delta^3 r^3)$.
It must hold that $V_1' = V_2' = \bot$ or $\exists k \geq 0, l_0 \ldots l_k, n_1 \ldots n_k.\ r^3 \supseteq \{r_{l_0}, r_{l_1 l_0 n_1} \ldots r_{l_k l_0 n_k}\}$ and if $k = 0$ then $V_1' \sqsubseteq V_1 = in_\mu in_\oplus in_1 in_{\mathbf{1}}*$ and $d_2' \sqsubseteq d_2 = in_{\mathbb{L}} l_0$, if $k > 0$ then $V_1' \sqsubseteq V_1 = in_\mu in_\oplus in_2 in_\otimes(in_Z n_k, in_\mu \ldots)$ and $V_1' \sqsubseteq V_2 = in_{\mathbb{L}} l_k$.

If $k = 0$ then $\llbracket ;\alpha; \vdash \mathit{pop}_1\rrbracket\, V_1 K_1 S_1 = K_1 S_1(in_\oplus in_1 in_{\mathbf{1}}*)$ and since the states are related in $\Delta^4 r^4 \rhd \Delta^3 r^3$ then $S_2 l_0 = in_\mu in_\oplus in_1 in_{\mathbf{1}}*$ and $\llbracket ;\alpha; \vdash \mathit{pop}_2\rrbracket\, V_2 K_2 S_2 = K_2 S_2(in_\oplus in_1 in_{\mathbf{1}}*)$. By the definition of ∇ it holds that $(in_\oplus in_1 in_{\mathbf{1}}*, in_\oplus in_1 in_{\mathbf{1}}*, in_\oplus in_1 in_{\mathbf{1}}*, in_\oplus in_1 in_{\mathbf{1}}*) \in \llbracket 1 + (\mathit{int} \times \alpha) \rrbracket^t(\alpha\hat{R})(\Delta^4 r^4)$.

If $k = 1$ then $\llbracket ;\alpha; \vdash \mathit{pop}_1\rrbracket\, V_1 K_1 S_1 = K_1 S_1 in_\oplus in_2 in_\otimes(in_Z n_k, in_\mu in_\oplus in_1 in_{\mathbf{1}}*)$ and since the states are related in $\Delta^4 r^4 \rhd \Delta^3 r^3$ then $S_2 l_0 = in_\mu in_\oplus in_2 in_\otimes(in_Z n_k, in_{\mathbb{L}} l_0)$ and
$\llbracket ;\alpha; \vdash \mathit{pop}_2\rrbracket\, V_2 K_2 S_2 = K_2 S_2(in_\oplus in_2 in_\otimes(in_Z n_k, in_{\mathbb{L}} l_0))$.
As $r_{l_0} \in r^4$ so $(in_\mu in_\oplus in_1 in_{\mathbf{1}}*,\ in_{\mathbb{L}} l_0,\ in_\mu in_\oplus in_1 in_{\mathbf{1}}*,\ in_{\mathbb{L}} l_0) \in \llbracket \alpha \rrbracket^t(\alpha\hat{R})(\Delta^4 r^4)$.

If $k > 1$ then $V_1 = in_\mu in_\oplus in_\otimes(in_Z n_k, d_1)$ and $\llbracket ;\alpha; \vdash \mathit{pop}_1\rrbracket\, V_1 K_1 S_1 = K_1 S_1 in_\oplus in_2 in_\otimes(inZn_k, d_1)$

since the states are related in $\Delta^4 r^4 \rhd \Delta^3 r^3$ then $S_2 l_k = in_\mu in_\oplus in_2 in_\otimes(in_Z n_k, in_\mathbb{L} l_{k-1})$ and $[\![; \alpha; \vdash pop_2]\!]\, V_2 K_2 S_2 \; = \; K_2 S_2(in_\oplus in_2 in_\otimes(in_Z n_k, in_\mathbb{L} l_{k-1}))$. By the definition of $\hat{r}$ and since $r^4 \supseteq \{r_{l_0}, r_{l_1 l_0 n_1} \ldots r_{l_k l_0 n_k}\}$ then by (*) $(d_1, in_\mathbb{L} l_{k-1}, d_1, in_\mathbb{L} l_{k-1}) \in [\![\alpha]\!]^t(\alpha \hat{R}))(\Delta^4 r^4)$.

In all cases we get related continuations applied to related states and related values, so we get the required termination approximation.
We conclude $([\![;;\vdash p_1]\!], [\![;;\vdash p_2]\!]) \in [\![T(1 + int)]\!]^t_{\emptyset\emptyset}(\Delta_\emptyset T)$.

6 Conclusion and Future Work

We have given what appears to be the first relationally parametric model of a langauge with recursive and polymorphic types and references to values of closed type. We are currently working on lifting the restriction that reference types have to be closed.

Relational Reasoning for Contextual Equivalence

Contents

Relational Reasoning for Contextual Equivalence

Nina Bohr

Abstract. We device a proof method for showing contextual equivalences of programs in a monadically typed language with recursive types, impredicative polymorphism and general references. This method extends the proof method presented at APLAS 06 (Nina Bohr and Lars Birkedal: Relational Reasoning for Recursive Types and References [13]). The definition of parameters has been refined so that we can use our method to prove more programs equivalent. The refined parameters make it possible to express that two programs are temporarily at inequal steps but will later come back to equality, or that two programs both diverge but not at the same 'step of computations'. It is also possible to express that programs make irreversible changes within hidden areas as the "Awkward Example" from Pitts and Stark [39]. We think that a closer comparison between this method and the bisimulaition based method devised by Kuotavas and Wand [28] will show that our method makes proofs quite accessible. We have added polymorphic types, but we do not treat relational parametricity here. In the previous paper (Relational Parametricity for Recursive Types and References of Closed Types) we discuss relational parametricity.

1 Introduction

In our paper from APLAS 2006 [13] we presented a proof method for contextual equivalence, which could be used to prove equivalence of several interesting programs. We want however to make the method more helpful in proving program equivalences. To that end we define several extensions in this report. Most of the extensions require a more refined definition of parameters. However, in actual examples we find that the definition of the local parameters we use in proofs, express in a natural way the intuition of why we believe the programs to be equivalent. So proofs of contextual equivalence of programs become a reasonable straightforward test of hypotheses of equivalences. When the local parameters are defined, the rest of the proof is almost automatic. So the complications in the definition of the format for parameters arise from the attempt to make the method reasonably easy to use on program examples.

To motivate the definitions in the following sections, we will first informally discuss some properties of a pair of programs together with example programs. We come back to the examples again in the Example Sections 7 and 10, where we will formally define the parameters we use. We hope that this informal presentation will make the definitions more accessible for the reader. Hongseok Yang proposed several example programs, which we gratefully acknowledge. In the informal presentation below, we assume that the reader is familiar with the definition of parameters in the article [13] which this report extends.

1.1 Irreversible change of states

First, we want to handle irreversible changes of local state. A program may have the possibility of changing state in a local area in an irreversible way, and the equivalence of two programs may require that we use the knowledge that the state cannot be changed back again. Recall the Awkward Example from Pitts and Stark [39]:

$$
\begin{aligned}
M\colon\quad & let\ a \Leftarrow ref\ 0\ in\\
& \quad val\ \big(rec\ f_M(g : unit \to T\tau) : Tint =\\
& \qquad let\ x \Leftarrow a := 1\ in\ let\ y \Leftarrow g()\ in\ !a)\big)
\end{aligned}
$$

$$
\begin{aligned}
N\colon\quad & val\ (rec\ f_N(g : unit \to T\tau) : Tint =\\
& \qquad let\ z \Leftarrow g()\ in\ val\ 1)
\end{aligned}
$$

M and N are closed computations. N is the computation that gives a function f_N. f_N takes as input a function g, applies $g()$ and if this terminates then f_N always returns 1. M allocates a new local location l_a and assigns 0 to it. Then it computes a function f_M, which takes as input a function g, then updates l_a to 1 and applies $g()$. If this terminates then f_M reads l_a and returns the result.

By informal reasoning, we expect the two computations to be equivalent because the only way the location l_a can be accessed is by application of the function f_M, so once l_a has been set to 1 it can never be set to any other value. Pitts and Stark use computation induction to prove these programs contextually equivalent and they conjecture that a Kripke style relation will make the programs accessible for a proof of equivalence. They suggest (in their setup) "$r_1 \supset r_2$ where $r_1 = \{(s, ())|s(l) = 0 \text{ } or \text{ } 1\}$ and $r_2 = \{()|s(l) = 1\}$". Such a local parameter is expressible in our setup and makes it possible to prove the programs equivalent (we have though preferred a local parameter that is a little different). We obtain this by defining what we call local extensions of a local parameter. Together with the definition of a local parameter, we now also include a precision of which local extensions the parameter has. We always require that related computations are related at any later time expressed by a parameter larger in the parameter ordering, and local extensions give larger parameters. Let in an informal notation a local parameter with local extension be $p = (q^0 \prec q^1) = (\ (s_1(l) = 0)\ \textit{with extension}\ (s_1(l) = 1)\ )$. Computations related under parameter p are required to behave related under each of the set of states in $p = (\ (s_1(l) = 0) \prec (s_1(l) = 1)\ )$ and the set of states in $(s_1(l) = 1)$. States related under p must have $s_1(l) = 0$ but it is also required, that the stored related values are related under p so as described also under p's local extension. Computations related under p may change the value stored in $S_1(l)$ from 0 to 1 and so extend the parameter. At a later time the parameter may have been locally extended to $q^1 : (s_1(l) = 1)$. Now states related under q^1 must have $s_1(l) = 1$, and computations related under q^1 must behave related under the set of states in $\{(s_1, s_2)|s_1(l) = 1\}$.

We aim at defining parameters, order on parameters and relatedness under parameters in such a way that computations related under a parameter will also be related under any bigger parameter. A bigger parameter is meant to describe a later step in computations. If related functions are stored and later retrieved we expect them to be related in the new contexts. We also want the parameters to be able to express such irreversible change of states which equivalent programs may perform in local areas. So, together with the definition of a local parameter, we now also include a precision of which local extensions the parameter has. A local extension uses at most the same local areas of two states as the local parameter it extends. A local parameter together with its local extensions has the form of a tree. Irreversible changes of states to local extensions corresponds to stepping down to a subtree. So for these example programs the local parameter is a two-node tree $(q^0 \prec q^1)$. The root-node says that location l_a in the M-side must hold 0. The sub-tree-node says that l_a must hold 1. With such a local parameter we can prove the programs equivalent, see the Example Section 7.1 for details.

It is necessary to define which local extensions are legal together with the local parameter. The reason for this is, that when we define a local parameter and prove programs equivalent under the parameter, then we must show that all exported functions preserve the parameter and we require the "parameter weakening property" i.e. that all bigger parameters will also be preserved. Usually this will not be the case for any arbitrary local strengthening of the parameter. In proofs we often need to do case analysis, and show that in all cases of extensions down a "local parameter tree" we get the required termination behavior.

Connected to the question of which computations we relate is the snapback-problem. We show in the Example Section 7.6 that at most types the "snapback"-function is not in our relation.

1.2 Knowing the initial steps of continuations

The parameters from [13] as well as the parameters with local extensions from above are preserved by each pair of related computations and pair of related continuations independently. We will now discuss extended parameters, for which we can use knowledge of initial parts of the continuations our computations are applied to.

When we develop a proof of equivalence of two programs we will often reach a step, where we apply a computation m which is part of the initial program to a continuation, in which the initial steps are generated from the original program. In such situations m may change states in ways which are so to say set back to an appropriate form by the known initial steps in the computation. An example of such a situation is given here.

```
M:   let w ⇐ ref 0 in
       g(rec f_M(u : unit) = (w := 1));
       val (rec get_M(u : unit) = (!w) , rec set_M(n : int) = (w := n))

N:   let x ⇐ ref 0 in    //flag: inside program N
     let y ⇐ ref 0 in    //flag: argument function f_N has not been applied inside program N
     let v ⇐ ref 0 in
       g(rec f_N(u : unit) = (if (!x = 0) then (y := 1) else (v := 1)));
       if (!y ≠ 0) then (v := 1);
       x := 1;
       val (rec get_N(u : unit) = (!v) , rec set_N(n : int) = (v := n))
```

M and N are open computations with a free variable g of type $((unit \to Tunit) \to T\tau)$.

M first allocates a new location l_w and stores 0 in l_w. Then g is applied to a function f_M. If g applies f_M then f_M's only action is to update l_w to hold 1. If $g(f_M)$ terminates then M returns a getter and a setter for l_w.

N is a little more complicated. N first allocates three new locations l_x, l_y, l_v each with the value 0. Then g is applied to a function f_N. If g applies f_N then f_N will read the value stored in l_x. If the value in l_x is 0 then f_N will update l_v to hold 0 and l_y to hold 1. If the value in l_x is not 0 then f_N will update l_v to hold 1. If $g(f_N)$ terminates then N will read the value stored in l_y and if it differs from 0 then l_v will be set to 1. Then l_x will be set to 1. Finally N returns a getter and a setter for l_v.

We expect M and N to be contextually equivalent by reasoning as follows. Inside M and N the functions g (expected to be equivalent) may store and/or apply their argument functions f_M and f_N. When g is applied inside M and N, then the only way g can access l_w and l_x, l_y and l_v respectively is by application of the argument functions f_M and f_N. The reason is that these locations are locally allocated and at this time the getters and setters for l_w and $l_v$have not been exported. Function f_N only reads l_x, so g cannot change the value stored in location l_x. l_x is updated to hold 1 just before computation N finishes. Hence l_x can be used as a flag for being inside N or after N has finished. l_y can only be changed by f_N and this will only happen if f_N is applied inside of N, hence the value of l_y can be used as a flag telling if f_N has been applied inside N. Since g's are expected to be equivalent, we expect them to have similar application and storing actions for the arguments f_M and f_N if $g(f_M)$ and $g(f_N)$ terminates. Assume f_M and f_N are applied one or more times inside $g(f_M)$ and $g(f_N)$. f_M will update l_w to hold 1, and f_N will update l_y to hold 1 but leave l_v with the value 0. So just after $g(f_M)$ and $g(f_N)$ terminate then l_w and l_v do *not* hold related values. However, this is remedied by N in the next step, where the value in l_y is read to be 1 and so l_v updated to 1. Now l_w and l_v hold the same value. N will set l_x to 1. If g has stored f_M and f_N then they may be called in the future after N has finished, and then f_M and f_N will always update l_w and l_v with the same value because l_x will always hold 1. Setters and getters also keep the values stored in l_w and l_v identical. Suppose the g's store f_M and f_N but do not apply them. After $g(f_M)$ and $g(f_N)$ terminate l_w and l_v will both hold 0 and l_y also be untouched. Now N will read the value in l_y to 0 and leave l_v untouched. We see, that under execution of the programs M and N the values in l_w and l_v may possibly have different values temporarily.

Look at the denotations for M and N and assume that they are given related functions $g_M,\ g_N$ and applied to related continuations $k_M,\ k_N$ and related states $S_M,\ S_N$.[1] In an attempt to prove M and N equivalent, we will reach a point

(*) $[\![M]\!](g \mapsto g_M)k_M S_M = g_M([\![f_M]\!][w \mapsto l_w])(\lambda S\lambda d.[\![val\ (get_M, set_N)]\!][w \mapsto l_w]k_M S)S_M[l_w \mapsto 0]$

$[\![N]\!](g \mapsto g_N)k_N S_N = g_N([\![f_N]\!][x \mapsto l_x, y \mapsto l_y, w \mapsto l_w])(\lambda S\lambda d.[\![if\ (!y \neq 0)\ then\ (v := 1); (x := 1); val\ (get_N, set_N)]\!][x \mapsto l_x, y \mapsto l_y, w \mapsto l_w]k_N S)S_N[l_x \mapsto 0, l_y \mapsto 0, l_v \mapsto 0]$

where we (when we follow our usual way of proving equivalences) would like to be able to say that: We apply related computations to related functions, so this gives us related computations. These computations are then applied to related continuations and related stores. In the end we want (also) to say that setters and getters are related, this is the case if l_w and l_v can always be expected to hold the same value. But this property is not preserved by f_M and f_N unless l_x holds 1, which is not initially the case.

In our informal reasoning for why we expect M and N to be equivalent, we benefitted from two observations: That the functions $f_M,\ f_N$ preserved two different invariants, namely either "l_x holds 0 and values stored in l_w and l_y are equal" or "l_x holds 1 and values stored in l_w and l_v are equal". Then knowing that the initial steps of the continuations which $g(f_M)$ and $g(f_N)$ are applied to would use the first invariant to establish the other invariant necessary for the set, get functions to be related, we could see that in the end things would match up.

In this report, we extend parameters from [13] to be able to benefit from knowing the initial steps of the continuations we apply to, and to allow us to employ that computations may preserve more than one invariant. The extended parameters will make it possible to make formal statements expressing:

> Computations related under a parameter that tells that they will preserve each of a set of invariants will have similar termination behavior whenever they are applied to
> states fulfilling one of the invariants and where all stored values preserve each of the set of invariants, together with continuations that give similar termination behavior, when they are executed in stores fulfilling this one invariant .

So, with these extended parameters, related continuations informally speaking "knows how to bring the situations back to equivalence, if it has temporarily leaped away in a well defined way". For continuations to behave well in this sense, they must know the invariants we are considering. For this reason the extended parameters require two different order relations on parameters: The $\rhd$ order is used for parameters with additional invariants, which are preserved by each of computations and continuations separately, so an invariant that continuations do not know about, does not matter for their related performance. The $\blacktriangleright$ order is used, when we add sets of invariants, and expect computations to be applied to continuations, that will ensure termination equivalence, knowing the invariants hold.

In the example above the parameter says that either $A : (S_1 l_w = S_2 l_y \in \{0, 1\} \land S_2 l_v = 0 \land S_2 l_x = 0)$ or $B : (S_1 l_w = S_2 l_v \land S_2 l_x = 1)$. It is a property of our relation together with the parameter definition that related computations cannot change which invariant A, B the stores satisfy. In (*), the states we apply the computations $g_M(f_M)$ and $g_N(f_N)$ to fulfill A, and the continuations we apply to can be shown to give similar termination when applied to states that fulfill A. This proof benefits from the fact that the initial part of the continuations change the states, so that B will be fulfilled. See example 7.2 for details.

1.3 Exporting hidden locations to visible

Sometimes two programs may require some locations to be local in the beginning of execution, but may later on export these locations so they become visible to the outside. We have included

[1] Our relations will be 4-tuples, but for simplicity we here look at pairs

a more refined treatment of local vs. visible state in the new definitions. The following example (cf. Example Section 7.3) is a variation of the previous example. The only difference is that in this example the computations M and N finally export the locations $l_w,\ l_v$ to visible. In the previous example the computations M and N exported setter and getter functions to $l_w,\ l_v$.

$$\begin{array}{ll} M: & let\ w \Leftarrow\ ref\ 0\ in \\ & let\ z \Leftarrow\ g(rec\ f_M(a : unit) = (w := 1))\ in \\ & \quad val\ w \end{array}$$

$$\begin{array}{ll} N: & let\ x \Leftarrow\ ref\ 0\ in \\ & let\ y \Leftarrow\ ref\ 0\ in \\ & let\ v \Leftarrow\ ref\ 0\ in \\ & let\ z_0 \Leftarrow\ g(recf_N(a : unit) = (if\ (!x = 0)\ then\ (v := 0; y := 1)\ else\ (v := 1))\ in \\ & \quad let\ z_1 \Leftarrow\ if\ (!y \neq 0)\ then\ (v := 1)\ in \\ & \quad let\ z_2 \Leftarrow x := 1\ in \\ & \quad val\ v \end{array}$$

Informally the reason for M and N to be equivalent is that when $l_w,\ l_v$ are finally exported, they will always hold the same value in states $S_1,\ S_2$, and $S_2 l_x \mapsto 1$. The computations $g_M(f_M)$ and $g_N(f_N)$ may store f_M and f_N. Because it will hold in all future that $S_2 l_x \mapsto 1$, so all future applications of $f_M,\ f_N$ will update $S_1 l_w$ and $S_2 l_v$ to hold the same value 1. It is necessary for the equivalence-proof, that $l_w,\ l_v$ are local initially. We want to be able to use the fact that $l_w,\ l_y$ can be simultaneously updated in the proof, that $g(f_M)$ and $g(f_N)$ relate.

In this report we refine the parameters from [13] to allow the possibility of export of hidden locations in certain situations.

1.4 Divergence at different steps

We here consider a special case of the previous, where we know that the initial part of continuations brings two programs back to be correlated. It might be the case that two programs both eventually diverge, but not in the same 'step'. The initial part of the continuation may be responsible for ensuring divergence of a program, when certain conditions are present in the state.

$$M: \quad g(rec\ f_M(a : unit) = diverge)$$

$$\begin{array}{ll} N: & let\ x \Leftarrow ref\ 0\ in \quad \text{//flag: inside program } N \\ & let\ y \Leftarrow ref\ 0\ in \quad \text{//flag: argument function } f_N \text{ has not been applied inside program } N \\ & \quad let\ z \Leftarrow g(rec\ f_N(a : unit) = \big(if\ (!\mathrm{x} = 0)\ then\ (y := 1)\ else(diverge)\big)\ in \\ & \qquad if\ (!y \neq 0)\ then\ (diverge)\ else\ (x := 1; y := 1) \end{array}$$

The programs M and N are open computations with one free variable g of function type $(unit \to Tunit) \to T\tau$. M applies g to a function f_M. If f_M is applied the computation will diverge. N first allocates two new locations $l_x,\ l_y$ and stores 0 in both (as in the previous example these locations function as flags). Then g is applied to a function f_N. f_N reads the value in locations l_x, if it is 0 then f_N updates l_y to hold 1 else f_N diverges. If $g(f_N)$ terminates, then N reads the value in location l_y. If the value is not 0 then N diverges, else N updates l_x and l_y both to hold 1.

We expect M and N to be contextually equivalent for the following reasons. If $g(f_M)$ applies f_M then the application of f_M will not terminate, and hence $g(f_M)$ and M will also not terminate. In the other side f_N preserves the invariant, that l_x holds 0. If $g(f_N)$ in N applies f_N then l_x holds 0 and the application of f_N will set l_y to 1 and terminate. f_N may be applied several times in $g(f_N)$ and $g(f_N)$ may terminate. But application of f_N will have caused l_y to be set to 1 and then afterwards N will diverge.

Suppose $g(f_M)$ does not cause f_M to be applied but stores f_M and then terminates. We expect g's to be related and so expect that also f_N is stored but not applied, and $g(f_N)$ terminates. g

can only access l_y by application of f_N so the value 0 stored in l_y will be the same, and then N will finish by updating l_y to 1 and l_x to 1. l_x cannot be changed after this. If f_M and f_N are later retrieved from the store, then they will both diverge.

The situation here is somewhat similar to the previous example, but here we may have divergence in different steps, where before we had values that were differently related. To handle this we have refined our definition of parameters so that it is possible to express that one side has diverged and so the other side is required to eventually diverge. If we attempt a proof of equivalence for this example, then we assume M and N are given related functions $g_1,\ g_2$, and the computations are applied to related continuations $k_1,\ k_2$ and related stores $S_1,\ S_2$, and we get to

(**) $[\![M]\!](g_1)k_1S_1 = g_1[\![f_M]\!]k_1S_1$

$[\![N]\!](g_2)k_2S_2 = g_2[\![f_N]\!][x \mapsto l_x, y \mapsto l_y](\lambda S\lambda d.([\![\mathit{if}\ (!y \neq 0)\ \mathit{then}\ (\mathit{diverge})\ \mathit{else}\ (x := 1, y := 1)]\!][x \mapsto l_x, y \mapsto l_y])k_1S)S_1[l_x \mapsto 0, l_y \mapsto 0]$

With the refined parameters we can express formally that f_M and f_N are related for both of

A:$((S_2l_x = 0 \wedge S_2l_y = 0)\ \prec$ (left hand side has diverged $\wedge S_2l_x = 0 \wedge S_2l_y \neq 0$))

B:$(S_2l_x \neq 0)$.

The updated states fulfill A. The continuations can be shown to give similar termination behavior when applied to states that fulfill A. The proof benefits from the fact that the initial part of the continuation ensures that either both sides have diverged or the states have been changed so that B is fulfilled. See example 10.1 for details.

When we attempt a proof of contextual equivalence, then we always will start out with an attempt to prove that two programs are related under a parameter, which only requires that visible locations hold related values and does not have any hidden invariants. This is a reason why local parameters which can express that one side has diverged can be used in proofs, we use them in sub-proofs. When we can derive, that one side (potentially) diverges, then we use them to show that the other side also (potentially) diverges.

1.5 Polymorphism

Aside from the refinement of parameters and order on parameters mentioned, we have also added impredicative polymorphism to the language. This is done in a simple way, where we do not give a relationally parametric interpretation. Two related values of polymorphic type are required to behave equivalent whenever they are applied to identical types. Values of polymorphic types may make full use of the store. A relational interpretation is discussed in the previous paper, which extends the APLAS paper without the refined parameters but with relational parametricity. We believe however, that it is possible to unite the refined parameters with a relational interpretation.

1.6 Outline of contents

In the following Section 2 we introduce the language and the operational semantics which will be given by a termination judgement. In Section 3 we prove the existence of a recursive domain in FM-Cpo$^4_\perp$ in which we define the denotational semantics. This is then shown to be sound and adequate. From Section 4 and in the following sections we present our main new contributions, the new parameters and the relations which we use in proofs of contextual equivalences. The relations are defined on top of the denotational semantics. To express explicit divergence adds some extra complication to the format for parameters and also extra cases to consider in equivalence proofs. We have chosen to define two logical relations, both of which can be used for equivalence proofs. The first without format for explicit divergence, the second with an extension which handles divergence.

Each relation requires a separate proof of existence. In Section 4 we introduce the first of the logical relations we will use for proofs of contextual equivalence. In Section 5 we give the formal definition of the parameters we use in our first relation. In Section 6 we define the action of the domain constructing functor on relations and prove the existence of a minimal invariant 4-ary relation ∇. There is a pairwise domain theoretical approximation similarly as in the paper [13] so for an element $(d_1', d_2' \parallel d_1, d_2, (type), p)$ then $d_1' \sqsubseteq d_1$ and $d_2' \sqsubseteq d_2$ and we may think of d_1, d_2 as a kind of parameters. As in [13] the definition of the relation as 4-ary makes it possible to carry out an existence proof in the line of Pitts [40] and at the same time let the parameters give very specific properties of states. Based on the 4-ary relation we extract a binary relation between denotations of open terms, this is then shown to imply contextual equivalence at simple parameters with no hidden invariants. In the following Section 7 we then present proofs of equivalence for a number of examples. In Section 8 we present the extension to the definition of the parameters which handles explicit divergence. Section 9 holds the definition of the new relation and prove its existence. As before it gives the basis for a binary relation, which is proven to imply contextual equivalence at simple parameters. Then the Example Section 10 gives a proof of equivalence for the example program we have seen in the introduction. In Section 11 we conclude.

2 Language

2.1 Types, terms and stores

The language we consider is a call-by-value, monadically-typed λ-calculus with recursion, general recursive types, impredicative polymorphism and general dynamically allocated references. Types are either *value types* τ or *computation types* $T\tau$. Closed values of any closed value type can be stored in the store. The language is as in [13] extended with polymorhism.

Types

We have a countable infinite set of type variables ranged over by α.

Value types $\tau, \sigma ::= \ \alpha \mid unit \mid int \mid \tau \times \tau \mid \tau + \tau \mid \tau\ ref \mid \tau \to T\tau \mid \mu\alpha.\tau \mid \forall\alpha.T\tau$

Computation types $T\tau$

Value and Computation types $\gamma ::= \ \tau \mid T\tau$

$ftv(\gamma)$ denotes the free type variables in γ.

Type contexts and well-formed types

A type context Ξ is a finite (possibly empty) set of type variables $\Xi = \alpha_1 \dots \alpha_m$.

A value type τ or computation type $T\tau$ is well-formed from a type context Ξ if $\Xi \vdash \tau : type$ can be generated by the following rules.
A value type τ or computation type $T\tau$ is well-formed and closed if $- \vdash \tau : type$ can be generated by the following rules when Ξ is the empty type context.

$$\frac{}{\Xi \vdash unit : type} \qquad \frac{}{\Xi \vdash int : type} \qquad \frac{}{\Xi, \alpha \vdash \alpha : type} \qquad \frac{\Xi \vdash \tau : type}{\Xi \vdash \tau\ ref : type}$$

$$\frac{\Xi \vdash \tau_1 : type \quad \Xi \vdash \tau_2 : type}{\Xi \vdash \tau_1 + \tau_2 : type} \qquad \frac{\Xi \vdash \tau_1 : type \quad \Xi \vdash \tau_2 : type}{\Xi \vdash \tau_1 \times \tau_2 : type} \qquad \frac{\Xi \vdash \tau_1 : type \quad \Xi \vdash \tau_2 : type}{\Xi \vdash \tau_1 \to T\tau_2 : type}$$

$$\frac{\Xi, \alpha \vdash \tau : type}{\Xi \vdash \mu\alpha.\tau : type} \qquad \frac{\Xi, \alpha \vdash \tau : type}{\Xi \vdash \forall\alpha.T\tau : type}$$

Terms

We have a countable infinite set of term variables ranged over by x.

Locations L is the countable infinite set of locations l_i.

Values $V ::= x \mid \underline{n} \mid \underline{l} \mid () \mid (V, V') \mid in_1 V \mid in_2 V \mid rec\ f(x : \tau) = M \mid fold\ V \mid \Lambda\alpha.M$

Computations $M ::=\ VV' \mid let\ x \Leftarrow M\ in\ M' \mid val\ V \mid \pi_1 V \mid \pi_2 V \mid ref\ V \mid !V \mid V := V' \mid$
$case\ V\ of\ in_1 x_1 \Rightarrow M_1;\ in_2 x_2 \Rightarrow M_2 \mid V = V' \mid V + V' \mid iszero\ V \mid unfold\ V \mid V\tau$

Phrases $G ::=\ M \mid V$

$fv(G)$ denotes the free term variables in G.

Contexts A context Γ is a finite (empty) set of typed term variables $\Gamma ::= x_1 : \tau_1, \ldots, x_n : \tau_n$
A context Γ is well-formed from a type context Ξ iff $\forall x_i : \tau_i \in \Gamma.\ \Xi \vdash \tau_i : type$. Notation $\Xi \vdash \Gamma$.

Continuation terms in x

$$\frac{}{val\ x \in Cont_x} \qquad \frac{fv(M) \subseteq \{x\} \qquad K \in Cont_y}{\text{let } y \Leftarrow M \text{ in } K \in Cont_x}$$

Stores

A store type is a finite function from locations to closed value types and store is a finite function from locations to closed values.

Store-type $\Delta \in (L \rightharpoonup_{fin} \{\text{closed Value types}\})$

Store $\Sigma \in (L \rightharpoonup_{fin} \{\text{closed Values}\})$

2.2 Typing Rules

Typing rules for terms

A typing judgement $\Delta; \Xi; \Gamma \vdash G : \tau$ or $\Delta; \Xi; \Gamma \vdash G : T\tau$ is well-formed if $\Xi \vdash \Gamma$, $fv(G) \subseteq dom(\Gamma)$ and $\Xi \vdash \tau : type$. All typing judgements in typing rules must be well-formed.

In any store type Δ all types are closed. So the (loc)-rule requires that τ is closed. The (alloc)-rule allows that τ is open and it is possible by the (id)-rule to type $x : \tau\ ref$ for open τ.

$$\text{(id)}\ \frac{}{\Delta; \Xi; \Gamma, x : \tau, \Gamma' \vdash x : \tau} \qquad \text{(unit)}\ \frac{}{\Delta; \Xi; \Gamma \vdash () : unit} \qquad \text{(int)}\ \frac{}{\Delta; \Xi; \Gamma \vdash n : int}$$

$$\text{(loc)}\ \frac{}{\Delta; \Xi; \Gamma \vdash l : \tau\ ref}\ (\Delta l = \tau) \qquad \text{(val)}\ \frac{\Delta; \Xi; \Gamma \vdash V : \tau}{\Delta; \Xi; \Gamma \vdash val\ V : T\tau}$$

$$\text{(deref)}\quad \frac{\Delta;\Xi;\Gamma \vdash V : \tau\ ref}{\Delta;\Xi;\Gamma \vdash !V : T\tau} \qquad \text{(alloc)}\quad \frac{\Delta;\Xi;\Gamma \vdash V : \tau}{\Delta;\Xi;\Gamma \vdash refV : T(\tau\ ref)}$$

$$\text{(assign)}\quad \frac{\Delta;\Xi;\Gamma \vdash V_1 : \tau\ ref \qquad \Delta;\Xi;\Gamma \vdash V_2 : \tau}{\Delta;\Xi;\Gamma \vdash V_1 := V_2 : Tunit} \qquad \text{(eq)}\quad \frac{\Delta;\Xi;\Gamma \vdash V_1 : \tau_1\ ref \qquad \Delta;\Xi;\Gamma \vdash V_2 : \tau_2\ ref}{\Delta;\Xi;\Gamma \vdash V_1 = V_2 : T(unit + unit)}$$

$$\text{(+ intro)}\quad \frac{\Delta;\Xi;\Gamma \vdash V : \tau_i}{\Delta;\Xi;\Gamma \vdash in_i V : \tau_1 + \tau_2}\ (i \in \{1,2\})$$

$$\text{(+ elim)}\quad \frac{\Delta;\Xi;\Gamma \vdash V : \tau_1 + \tau_2 \qquad \Delta;\Xi;\Gamma, x_1 : \tau_1 \vdash M_1 : T\tau' \qquad \Delta;\Xi;\Gamma, x_2 : \tau_2 \vdash M_2 : T\tau'}{\Delta;\Xi;\Gamma \vdash case\ V\ of\ in_1 x_1 \Leftarrow M_1; in_2 x_2 \Leftarrow M_2 : T\tau'}$$

$$\text{(}\times\text{ intro)}\quad \frac{\Delta;\Xi;\Gamma \vdash V_1 : \tau_1 \qquad \Delta;\Xi;\Gamma \vdash V_2 : \tau_2}{\Delta;\Xi;\Gamma \vdash (V_1, V_2) : \tau_1 \times \tau_2} \qquad \text{(}\times\text{ elim)}\quad \frac{\Delta;\Xi;\Gamma \vdash V : \tau_1 \times \tau_2}{\Delta;\Xi;\Gamma \vdash \pi_i V : T\tau_i}\ (i \in \{1,2\})$$

$$\text{(rec)}\quad \frac{\Delta;\Xi;\Gamma, x : \tau, f : \tau \to T\tau' \vdash M : T\tau'}{\Delta;\Xi;\Gamma \vdash rec\ f(x : \tau) = M : \tau \to T\tau'} \qquad \text{(app)}\quad \frac{\Delta;\Xi;\Gamma \vdash V_1 : \tau \to T\tau' \qquad \Delta;\Xi;\Gamma \vdash V_2 : \tau}{\Delta;\Xi;\Gamma \vdash V_1 V_2 : T\tau'}$$

$$\text{(let)}\quad \frac{\Delta;\Xi;\Gamma \vdash M_1 : T\tau_1 \qquad \Delta;\Xi;\Gamma, x : \tau_1 \vdash M_2 : T\tau_2}{\Delta;\Xi;\Gamma \vdash let\ x \Leftarrow M_1\ in\ M_2 : T\tau_2}$$

$$\text{(arith)}\quad \frac{\Delta;\Xi;\Gamma \vdash V_1 : int \qquad \Delta;\Xi;\Gamma \vdash V_2 : int}{\Delta;\Xi;\Gamma \vdash V_1 + V_2 : Tint} \qquad \text{(iszero)}\quad \frac{\Delta;\Xi;\Gamma \vdash V : int}{\Delta;\Xi;\Gamma \vdash iszero\ V : T(unit + unit)}$$

$$\text{(fold)}\quad \frac{\Delta;\Xi;\Gamma \vdash V : \tau[\mu\alpha.\tau/\alpha]}{\Delta;\Xi;\Gamma \vdash fold\ V : \mu\alpha.\tau} \qquad \text{(unfold)}\quad \frac{\Delta;\Xi;\Gamma \vdash V : \mu\alpha.\tau}{\Delta;\Xi;\Gamma \vdash unfold\ V : T(\tau[\mu\alpha.\tau/\alpha])}$$

$$(\Lambda)\quad \frac{\Delta;\Xi,\alpha;\Gamma \vdash M : T\tau \qquad \Xi \vdash \Gamma}{\Delta;\Xi;\Gamma \vdash \Lambda\alpha.M : \forall\alpha.T\tau} \qquad (\Lambda\text{app})\quad \frac{\Delta;\Xi;\Gamma \vdash V : \forall\alpha.T\tau \qquad \Xi \vdash \tau' : type}{\Delta;\Xi;\Gamma \vdash V\tau' : T(\tau[\tau'/\alpha])}$$

Typing rules for continuations

$$\frac{\vdash \tau : type}{\Delta \vdash val\ x : (x : \tau)^\top} \qquad \frac{\Delta;\ ; x : \tau \vdash M : T\tau' \qquad \Delta \vdash K : (y : \tau')^\top}{\Delta \vdash let\ y \Leftarrow M\ in\ K : (x : \tau)^\top}$$

Typing for stores

$\Sigma : \Delta$ holds iff $\forall l \in dom(\Delta).\ \Delta;;\vdash \Sigma l : \Delta l$

2.3 Judgement of termination

Termination judgements have the form: $\quad \Sigma$, let $x \Leftarrow M$ in $K \downarrow$
where M is closed and $K \in Cont_x$.

$$\frac{}{\Sigma, let\ x \Leftarrow val\ V\ in\ val\ x \downarrow} \qquad \frac{\Sigma, let\ y \Leftarrow M[V/x]\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow val\ V\ in\ (let\ y \Leftarrow M\ in\ K) \downarrow}$$

$$\frac{\Sigma, let\ y \Leftarrow M_1\ in\ (let\ x \Leftarrow M_2\ in\ K) \downarrow}{\Sigma, let\ x \Leftarrow (let\ y \Leftarrow M_1\ in\ M_2)\ in\ K \downarrow} \qquad \frac{\Sigma, let\ x \Leftarrow M[V/y, (rec\ f(y:\tau) = M)/f]\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow ((rec\ f(y:\tau) = M)V)\ in\ K) \downarrow}$$

$$\frac{\Sigma, let\ x \Leftarrow val\ V_1\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow \pi_1(V_1, V_2)\ in\ K \downarrow} \qquad \frac{\Sigma, let\ x \Leftarrow val\ V_2\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow \pi_2(V_1, V_2)\ in\ K \downarrow}$$

$$\frac{\Sigma, let\ x \Leftarrow M_1[V/x_1]\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow case\ in_1 V\ of\ in_1 x_1 \Rightarrow M_1; in_2 x_2 \Rightarrow M_2\ in\ K \downarrow}$$

$$\frac{\Sigma, let\ x \Leftarrow M_2[V/x_2]\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow case\ in_2 V\ of\ in_1 x_1 \Rightarrow M_1; in_2 x_2 \Rightarrow M_2\ in\ K \downarrow}$$

$$\frac{\Sigma, let\ x \Leftarrow val\ \underline{n}\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow \underline{n_1} + \underline{n_2}\ in\ K \downarrow}\ (\underline{n} = \underline{n_1} + \underline{n_2})$$

$$\frac{\Sigma, let\ x \Leftarrow val\ true\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow iszero\ \underline{0}\ in\ K \downarrow} \qquad \frac{\Sigma, let\ x \Leftarrow val\ false\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow iszero\ \underline{n}\ in\ K \downarrow} (\underline{n} \neq 0)$$

$$\frac{\Sigma, let\ x \Leftarrow val\ true\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow \underline{l} = \underline{l}\ in\ K \downarrow} \qquad \frac{\Sigma, let\ x \Leftarrow val\ false\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow \underline{l} = \underline{l'}\ in\ K \downarrow}\ (l \neq l')$$

$$\frac{\Sigma[l \mapsto V], let\ x \Leftarrow val\ ()\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow l := V\ in\ K \downarrow} \qquad \frac{\Sigma(l) = V \qquad \Sigma, let\ x \Leftarrow val\ V\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow !l\ in\ K \downarrow}$$

$$\frac{\Sigma[l \mapsto V], let\ x \Leftarrow val\ l\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow ref\ V\ in\ K \downarrow}\ (l \notin locs(\Sigma) \cup locs(K) \cup locs(V))$$

$$\frac{\Sigma, let\ x \Leftarrow val\ V\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow unfold(fold\ V)\ in\ K \downarrow}$$

$$\frac{\Sigma, let\ x \Leftarrow M\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow (\Lambda\alpha.M)\tau\ in\ K \downarrow}$$

Permutations are bijective functions $\pi \in (\mathbb{L} \to \mathbb{L})$.

Action of permutations on syntactic terms, continuations and stores

- The action of a permutation π on a term or continuation E with occurring locations $l_1, .., l_n$ is given as: $\pi \bullet E = E[(\pi(l_1))/l_1, .., (\pi(l_n))/l_n]$ with simultaneous substitution.
- The action of a permutation on a store Σ is given by $(\pi \bullet \Sigma)(l) = \pi \bullet (\Sigma(\pi^{-1} \bullet l))$.
- The action of a permutation on a store-type Δ is given by $dom(\pi \bullet \Delta) = \pi \bullet (dom(\Delta))$ and $(\pi \bullet \Delta)(l) = \Delta(\pi^{-1} \bullet l)$.

It holds that: If V is a value term then $\pi \bullet V$ is a value term, if M is a computation term then $\pi \bullet M$ is a computation term, if K is a continuation term then $\pi \bullet K$ is a continuation term. For any term or continuation E, for all permutations π_1 and π_2 it holds that $\pi_2 \bullet (\pi_1 \bullet E) = (\pi_2 \circ \pi_1) \bullet E$. For all stores Σ, for all permutations π_1 and π_2 it holds that $\pi_2 \bullet (\pi_1 \bullet \Sigma) = (\pi_2 \circ \pi_1) \bullet \Sigma$. For any store-type Δ, for all permutations π_1 and π_2 it holds that $\pi_2 \bullet (\pi_1 \bullet \Delta) = (\pi_2 \circ \pi_1) \bullet \Delta$.

Welltypedness is preserved under permutations:

- If $\Sigma : \Delta$ then $(\pi \bullet \Sigma) : (\pi \bullet \Delta)$
- If $\Delta; \Xi; \Gamma \vdash M : T\tau$ then $(\pi \bullet \Delta); \Xi; \Gamma \vdash (\pi \bullet M) : T\tau$
- If $\Delta; \Xi; \Gamma \vdash V : \tau$ then $(\pi \bullet \Delta); \Xi; \Gamma \vdash (\pi \bullet V) : \tau$
- If $\Delta \vdash K : (x : \tau)^{\top}$ then $(\pi \bullet \Delta) \vdash (\pi \bullet K) : (x : \tau)^{\top}$

Lemma 1. *For any permutation π on the set of locations, it holds that*

$$\Sigma, let\ x \Leftarrow M\ in\ K \downarrow \Longleftrightarrow (\pi \bullet \Sigma), (\pi \bullet (let\ x \Leftarrow M\ in\ K)) \downarrow$$

Proof by induction over judgement of termination.

Corollary 1. *If $l_1, l_2 \notin locs(\Sigma) \cup locs(K) \cup locs(V)$ then*

$$\Sigma[l_1 \mapsto V], let\ x \Leftarrow val\ l_1\ in\ K \downarrow \Longleftrightarrow \Sigma[l_2 \mapsto V], let\ x \Leftarrow val\ l_2\ in\ K \downarrow$$

Proof, apply the permutation $(l_1 l_2)$

Contextual equivalence

Typed Contexts $C[.] : (\Delta; \Xi; \Gamma \vdash \gamma) \Rightarrow (\Delta; ; \vdash T\tau)$
means that whenever $\Delta; \Xi; \Gamma \vdash G : \gamma$ then $\Delta; ; \vdash C[G] : T\tau$.

Definition 1. *Contextual equivalence*
If $\Delta; \Xi; \Gamma \vdash G_1 : \gamma$ and $\Delta; \Xi; \Gamma \vdash G_2 : \gamma$ then

$\Delta; \Xi; \Gamma \vdash G_1 =_{ctx} G_2$ means

$\forall \tau. \forall C[.] : (\Delta; \Xi; \Gamma \vdash \gamma) \Rightarrow (\Delta; ; \vdash T\tau). \forall \Sigma : \Delta.$

$$\Sigma, let\ x \Leftarrow C[G_1]\ in\ val\ x \downarrow \Longleftrightarrow \Sigma, let\ x \Leftarrow C[G_2]\ in\ val\ x \downarrow$$

3 Denotational Semantics

3.1 FM-Domains

$\mathbb{L}$ is the countable infinite set of locations (atoms). Permutations are bijective functions $\pi \in (\mathbb{L} \to \mathbb{L})$. We extend the action of permutations to $\mathbb{L}_\perp$ such that $\forall \pi.\ \pi(\perp) = \perp$. $\mathbb{L}_\perp$ with flat order $\in$ FM-Cpo$_\perp$. FM-Cpo$_\perp$ is a category with pointed FM-cpos as objects and strict equivariant (empty supported) FM-continous functions as morphisms. The FM-Cpo $(D \multimap D')$ consists of the strict finitely supported FM-continuous functions from D to D' with pointwise ordering.

Definition 2. *We define a functor* $\quad F : (FM\text{-}Cpo_\perp^{op})^4 \times (FM\text{-}Cpo_\perp)^4 \longrightarrow (FM\text{-}Cpo_\perp)^4$

For $D^- = (V^-, K^-, M^-, S^-)$, $D^+ = (V^+, K^+, M^+, S^+) \in (FM\text{-}Cpo_\perp)^4$ define $F(D^-, D^+)$

where

$$F(D^-, D^+)_V = \mathbb{1}_\perp \oplus \mathbb{Z}_\perp \oplus \mathbb{L}_\perp \oplus (V^+ \oplus V^+) \oplus (V^+ \otimes V^+) \oplus (V^- \multimap M^+)_\perp \oplus V^+ \oplus M^+_\perp$$

$$F(D^-, D^+)_K = (S^- \multimap (V^- \multimap \mathbb{O}))$$

$$F(D^-, D^+)_M = (K^- \multimap (S^- \multimap \mathbb{O}))$$

$$F(D^-, D^+)_S = (\mathbb{L}_\perp \multimap V^+)$$

$\mathbb{1}_\perp$	$= \{*\}_\perp$ *with trivial action of permutations.* $\mathbb{1}_\perp \in FM\text{-}Cpo_\perp$
$\mathbb{Z}_\perp$	$=$ *The flat domain of natural numbers with trivial action of permutations.* $\mathbb{Z}_\perp \in FM\text{-}Cpo_\perp$
$\mathbb{O}$	$= \{\top, \perp\}$ *where* $\perp \sqsubseteq \top$, *with trivial action of permutations.* $\mathbb{O} \in FM\text{-}Cpo_\perp$

Proposition 1. *There exists an invariant domain which is unique up to isomorphism* $\mathbb{D} = (\mathbb{V}, \mathbb{K}, \mathbb{M}, \mathbb{S}) \in$ *FM-Cpo*$^4_\perp$ *and isomorphism* $i : F(\mathbb{D}, \mathbb{D}) \cong \mathbb{D}$ *such that* $(i, \mathbb{D})$ *has the minimal invariant property:* $id_\mathbb{D}$ *is the least fixed point of* $\delta : (\mathbb{D} \multimap \mathbb{D}) \to (\mathbb{D} \multimap \mathbb{D})$ *defined by* $\delta(f) \overset{def}{=} i \circ F(f, f) \circ i^{-1}$.

We refer to Mark Shinwell's Ph.D.Thesis [52] chapter 4 as background for the proof of existens of the recursive domain $\mathbb{D}$.

Definition 3. *LFC-functor.*
A locally FM-continuous functor G : *FM-Cpo*$_\perp \longrightarrow$ *FM-Cpo*$_\perp$ *is a monotone functor which is FM-continuous (preserves least upper bounds of overall finitely supported chains) such that for all permutations* π *and functions* $f : D \to D'$ *it holds that* $\pi(F(f)) = F(\pi(f))$.

A mixed varians LCF-functor $G : (\textit{FM-CPO}^{op}_\perp)^4 \times (\textit{FM-CPO}_\perp)^4 \longrightarrow (\textit{FM-CPO}_\perp)^4$ *is contravariant in its left argument and covariant in its right argument.*

The following operations determine LCF-functors ([52]p.49):

Lifting $(+)_\perp$
Smash product $(+) \otimes (+)$
Coalesced sum $(+) \oplus (+)$
Strict function space $(-) \multimap (+)$

Corollary 2. *The functor* F *as defined above is a mixed varians LCF-functor.*

Proposition 2. *(adapted from M.Shinwell [52] p.51)*
Let $G : (\textit{FM-CPO}^{op}_\perp)^4 \times (\textit{FM-CPO}^4_\perp) \longrightarrow \textit{FM-CPO}_\perp$ *be a mixed varians LFC-functor. Then there exists a solution (j,D) satisfying the minimal invariant property with* $j : G(D, D) \cong D$ *which is unique up to isomorphism.*

Proposition 1 stating the existence of the recursive domain $\mathbb{D} = (\mathbb{V}, \mathbb{K}, \mathbb{M}, \mathbb{S})$ with $i : F(\mathbb{D}, \mathbb{D}) \cong \mathbb{D}$ follows from proposition 2. It holds that

$$\begin{aligned}
\mathbb{V} &\cong F(\mathbb{D}, \mathbb{D})_V = \mathbb{1}_\perp \oplus \mathbb{Z}_\perp \oplus \mathbb{L}_\perp \oplus (\mathbb{V} \oplus \mathbb{V}) \oplus (\mathbb{V} \otimes \mathbb{V}) \oplus (\mathbb{V} \multimap \mathbb{M})_\perp \oplus \mathbb{V} \oplus \mathbb{M}_\perp \\
\mathbb{K} &\cong F(\mathbb{D}, \mathbb{D})_K = (\mathbb{S} \multimap (\mathbb{V} \multimap \mathbb{O})) \\
\mathbb{M} &\cong F(\mathbb{D}, \mathbb{D})_M = (\mathbb{K} \multimap (\mathbb{S} \multimap \mathbb{O})) \\
\mathbb{S} &\cong F(\mathbb{D}, \mathbb{D})_S = (\mathbb{L}_\perp \multimap \mathbb{V})
\end{aligned}$$

3.2 Denotations

We have the following injections into $F(\mathbb{D}, \mathbb{D})_V$: $in_\mathbf{1}$, $in_\mathbb{Z}$, $in_\mathbb{L}$, $in_\oplus$, $in_\otimes$, $in_\multimap$, in_μ, $in_\forall$.

Denotations of types

$$[\![unit]\!] = \mathbb{V} \quad [\![int]\!] = \mathbb{V} \quad [\![\sigma ref]\!] = \mathbb{V} \quad [\![\tau + \tau']\!] = \mathbb{V}$$

$$[\![\tau \times \tau']\!] = \mathbb{V} \quad [\![\tau \multimap T\tau']\!] = \mathbb{V} \quad [\![\mu\alpha.\tau]\!] = \mathbb{V} \quad [\![\forall\alpha.T\tau]\!] = \mathbb{V}$$

$$[\![(x : \tau)^\top]\!] = \mathbb{K} \cong (\mathbb{S} \multimap (\mathbb{V} \multimap \mathbb{O}))$$

$$[\![T\tau]\!] = \mathbb{M} \cong (\mathbb{K} \multimap (\mathbb{S} \multimap \mathbb{O}))$$

$$[\![\Delta]\!] = \mathbb{S} \cong (\mathbb{L}_\perp \multimap \mathbb{V})$$

Denotations of contexts

$$\begin{array}{ll} \Gamma & = x_1 : \tau_1, \ldots, x_n : \tau_n,\ n \geq 0 \\ [\![\Gamma]\!] & = \mathbb{V} \otimes \ldots \otimes \mathbb{V} = \mathbb{V}^n \text{ where } \mathbb{V}^0 = \mathbb{1}_\perp \end{array}$$

Denotations of well typed continuations, values, computations and stores

Denotations are given only to well typed continuations, values, computations and stores. Notice the denotations are defined without reference to the actual types. The denotations are defined by cases over syntactic constructors for terms.
Recall: $i = (i_V, i_K, i_M, i_S) : F(\mathbb{D}, \mathbb{D}) \cong \mathbb{D}$

Environments ρ are elements of $[\![\Gamma]\!] = \mathbb{V}^n$.
The empty environment is written $\{\}$, this stand for the non-bottom element in $\mathbb{V}^0$.

Values,Computations: Denotations for values and computations $[\![\Delta; \Xi; \Gamma \vdash G : \gamma\ ]\!] \in ([\![\Gamma]\!] \multimap [\![\gamma]\!])$

Continuations: Denotations for continuations : $[\![\Delta \vdash K : (x : \tau)^\top\ ]\!] \in \mathbb{K}$

$[\![\Delta \vdash K : (x : \tau)^\top\ ]\!] = i_K(\lambda S \in \mathbb{S}.\lambda d \in \mathbb{V}.$
$\quad i_M^{-1}([\![\Delta; ; x : \tau \vdash K : T\tau']\!]\{x \mapsto d\}) i_K((\lambda S' \in \mathbb{S}.(\lambda d' \in \mathbb{V}.\ \top)_\perp)_\perp) S)$

Stores: Denotations of typing judgements for *stores*:
$[\![\Sigma : \Delta]\!] = \{S \in \mathbb{S} \mid S \neq \perp\ \wedge \forall l \in dom(\Delta).(i^{-1}S)(l) = [\![\Delta; ; \vdash \Sigma(l) : \Delta(l)]\!]\{\}\ \ \}$

Denotations of typing judgements in environments

By strictness it holds that $(\rho = \perp) \Rightarrow [\![\Delta; \Xi; \Gamma \vdash G : \gamma\ ]\!]\rho = \perp$,
when $\rho \neq \perp$ the denotations are given by the following rules:

Denotations are only given to well typed terms.

Values

- $[\![\Delta; \Xi; \Gamma, x : \tau, \Gamma' \vdash x : \tau\ ]\!]\rho = \rho(x)$
- $[\![\Delta; \Xi; \Gamma \vdash () : unit\ ]\!]\rho = i_V \circ in_{\mathbf{1}} *$
- $[\![\Delta; \Xi; \Gamma \vdash n : int\ ]\!]\rho = i_V \circ in_{\mathbb{Z}} n$
- $[\![\Delta; \Xi; \Gamma \vdash l : \tau\ ref\ ]\!]\rho = i_V \circ in_{\mathbb{L}} l$
- $[\![\Delta; \Xi; \Gamma \vdash in_1 V_1 : \tau_1 + \tau_2\ ]\!]\rho = i_V \circ in_\oplus(in_1 [\![\Delta; \Xi; \Gamma \vdash V_1 : \tau_1\ ]\!]\rho)$
  $[\![\Delta; \Xi; \Gamma \vdash in_2 V_2 : \tau_1 + \tau_2\ ]\!]\rho = i_V \circ in_\oplus(in_2 [\![\Delta; \Xi; \Gamma \vdash V_2 : \tau_2\ ]\!]\rho)$
- $[\![\Delta; \Xi; \Gamma \vdash (V_1, V_2) : \tau_1 \times \tau_2\ ]\!]\rho = i_V \circ in_\otimes([\![\Delta; \Xi; \Gamma \vdash V_1 : \tau_1\ ]\!]\rho, [\![\Delta; \Xi; \Gamma \vdash V_2 : \tau_2\ ]\!]\rho)$
- $[\![\Delta; \Xi; \Gamma \vdash rec\ f(x : \tau) = M : \tau \to T\tau'\ ]\!]\rho = i \circ in_\multimap \lfloor fix(\lambda f' \in (\mathbb{V} \multimap \mathbb{M}).$
  $\quad (\lambda x' \in \mathbb{V}.[\![\Delta; \Xi; \Gamma, f : \tau \to T\tau', x : \tau \vdash M : T\tau']\!](\rho \otimes f \mapsto i \circ in_\multimap \lfloor f' \rfloor \otimes x \mapsto x')))\rfloor =$
  $i \circ in_\multimap \lfloor \bigsqcup_{n \in \omega} g_n \rfloor$
  where
  $g_n \in (\mathbb{V} \multimap \mathbb{M}),\ g_0 = \perp_{\mathbb{V} \multimap \mathbb{M}}$ and
  $g_{n+1} =\ \lambda x_0 \in \mathbb{V}.[\![\Delta; \Xi; \Gamma, f : \tau \to T\tau', x : \tau \vdash M : T\tau']\!](\rho \otimes f \mapsto i \circ in_\multimap \lfloor g_n \rfloor \otimes x \mapsto x_0)$.

– $[\![\Delta;\Xi;\Gamma \vdash fold\ V : \mu\alpha.\tau]\!]\rho = i_V \circ in_\mu([\![\Delta;\Xi;\Gamma \vdash V : \tau[\mu\alpha.\tau/\alpha]]\!]\rho)$

– $[\![\Delta;\Xi;\Gamma \vdash \Lambda\alpha.M : \forall\alpha.T\tau]\!]\rho = i_V \circ in_\forall\lfloor[\![\Delta;\Xi,\alpha;\Gamma \vdash M : T\tau]\!]\rho\rfloor$

Computations

– $[\![\Delta;\Xi;\Gamma \vdash val\ V : T\tau]\!]\rho = i_M(\lambda k \in \mathbb{K}.\lambda S \in \mathbb{S}.\ (i_K^{-1}k)S([\![\Delta;\Xi;\Gamma \vdash V : \tau]\!]\rho))$

– $[\![\Delta;\Xi;\Gamma \vdash V_1V_2 : T\tau']\!]\rho =$
$case\ [\![\Delta;\Xi;\Gamma \vdash V_1 : \tau \to T\tau']\!]\rho\ of\ i_V \circ in_{\multimap}\lfloor d_1\rfloor\ then\ (d_1[\![\Delta;\Xi;\Gamma \vdash V_2 : \tau]\!]\rho);\ else\ \bot$

– $[\![\Delta;\Xi;\Gamma \vdash let\ x \Leftarrow M_1\ in\ M_2 : T\tau_2]\!]\rho = i_M(\lambda k.\lambda S.$
$i_M^{-1}([\![\Delta;\Xi;\Gamma \vdash M_1 : T\tau_1]\!]\rho)i_K(\lambda S'.\lambda d'.i_M^{-1}([\![\Delta;\Xi;\Gamma, x : \tau_1 \vdash M_2 : T\tau_2]\!]\rho[x \mapsto d'])kS')S)$

– $[\![\Delta;\Xi;\Gamma \vdash \pi_1 V : T\tau_1]\!]\rho = i_M(\lambda k.\lambda S.$
$case\ [\![\Delta;\Xi;\Gamma \vdash V : \tau_1 \times \tau_2]\!]\rho\ of\ i_V \circ in_\otimes(d_1,d_2)\ then\ (i_K^{-1}k)S(d_1);\ else\ \bot)$
$[\![\Delta;\Xi;\Gamma \vdash \pi_2 V]\!]\rho = i_M(\lambda k.\lambda S.$
$case\ [\![\Delta;\Xi;\Gamma \vdash V : \tau_1 \times \tau_2]\!]\rho\ of\ i_V \circ in_\otimes(d_1,d_2)\ then\ (i_K^{-1}k)S(d_2);\ else\ \bot)$

– $[\![\Delta;\Xi;\Gamma \vdash !V : T\tau]\!]\rho = i_M(\lambda k.\lambda S.$
$case\ [\![\Delta;\Xi;\Gamma \vdash V : \tau\ ref]\!]\rho\ of\ i_V \circ in_{\mathbb{L}} l\ then\ (i_K^{-1}k)S(((i_S^{-1}S)l));\ else\ \bot)$

– $[\![\Delta;\Xi;\Gamma \vdash refV : T\tau\ ref]\!]\rho = i_M(\lambda k.\lambda S.$
$(i_K^{-1}k)(i_S((i_S^{-1}S)[l \mapsto [\![\Delta;\Xi;\Gamma \vdash V : \tau]\!]\rho]))(i_V \circ in_{\mathbb{L}} l))$
for some/any $l \notin supp(\lambda l'.(i_K^{-1}k)(i_S((i_S^{-1}S)[l' \mapsto [\![\Delta;\Xi;\Gamma \vdash V : \tau]\!]\rho]))(i_V \circ in_{\mathbb{L}} l'))$

– $[\![\Delta;\Xi;\Gamma \vdash V_1 := V_2]\!]\rho = i_M(\lambda k.\lambda S.$
$case\ [\![\Delta;\Xi;\Gamma \vdash V_1 : \tau\ ref]\!]\rho\ of\ i_V \circ in_{\mathbb{L}} l\ then\ (i_K^{-1}k)(i_S((i_S^{-1}S)[l \mapsto [\![\Delta;\Xi;\Gamma \vdash V_2 : \tau]\!]\rho]))(i_V \circ$
$in_1 *); else \bot)$

– $[\![\Delta;\Xi;\Gamma \vdash case\ V\ of\ in_1x_1 \Rightarrow M_1;\ in_2x_2 \Rightarrow M_2 : T\tau']\!]\rho = i_M(\lambda k.\lambda S.$
$case\ [\![\Delta;\Xi;\Gamma \vdash V : \tau_1 + \tau_2]\!]\rho$
$of\ i_V \circ in_\oplus(in_1 d_1)\ then\ i_M^{-1}([\![\Delta;\Xi;\Gamma, x_1 : \tau_1 \vdash M_1 : T\tau']\!]\rho[x_1 \mapsto d_1])kS;$
$of\ \ i_V \circ in_\oplus(in_2 d_2)\ then\ i_M^{-1}([\![\Delta;\Xi;\Gamma, x_2 : \tau_2 \vdash M_2 : T\tau']\!]\rho[x_2 \mapsto d_2])kS; \qquad else\ \bot)$

– $[\![\Delta;\Xi;\Gamma \vdash V_1 = V_2 : Tunit]\!]\rho = i_M(\lambda k.\lambda S.$
$case\ [\![\Delta;\Xi;\Gamma \vdash V_1 : \tau\ ref]\!]\rho\ of\ i_V \circ in_{\mathbb{L}} l_1\ and\ [\![\Delta;\Xi;\Gamma \vdash V_2 : \tau\ ref]\!]\rho\ of\ i_V \circ in_{\mathbb{L}} l_2\ then$
$(if\ l_1 = l_2\ then\ (i_K^{-1}k)S(i_V \circ in_\oplus \circ in_1(i_V(in_1 *)))\ else\ (i_K^{-1}k)S(i_V \circ in_\oplus \circ in_2(i_V(in_1 *))));$
$else\ \bot)$

– $[\![\Delta;\Xi;\Gamma \vdash V_1 + V_2 : Tint]\!]\rho = i_M(\lambda k.\lambda S.$
$case\ [\![\Delta;\Xi;\Gamma \vdash V_1 : int]\!]\rho\ of\ i_V \circ in_{\mathbb{Z}} n_1\ and\ [\![\Delta;\Xi;\Gamma \vdash V_2 : int]\!]\rho\ of\ i_V \circ in_{\mathbb{Z}} n_2$
$then\ (i_K^{-1}k)S(i_V \circ in_{\mathbb{Z}} n)\ where\ n = n_1 + n_2;$
$else\ \bot)$

– $[\![\Delta;\Xi;\Gamma \vdash iszeroV : Tunit]\!]\rho = i_M(\lambda k.\lambda S.$
$case\ [\![\Delta;\Xi;\Gamma \vdash V : int]\!]\rho\ of\ i_V \circ in_{\mathbb{Z}} n\ then$
$(if\ n = 0\ then\ (i_K^{-1}k)S(i_V \circ in_\oplus \circ in_1(i_V(in_1 *))\ else\ (i_K^{-1}k)S(i_V \circ in_\oplus \circ in_2(i_V(in_1 *))));$
$else \bot)$

- $[\![\Delta;\Xi;\Gamma \vdash unfold\ V : T(\tau[\mu\alpha.\tau/\alpha])\ ]\!]\rho = i_M(\lambda k.\lambda S.$
  $case\ [\![\Delta;\Xi;\Gamma \vdash V : \mu\alpha.\tau\ ]\!]\rho\ of\ i_V \circ in_\mu(d)\ then\ (i_K^{-1}k)Sd;\ else\bot)$

- $[\![\Delta;\Xi;\Gamma \vdash V\tau' : T(\tau[\tau'/\alpha])]\!]\rho =$
  $case\ [\![\Delta;\Xi;\Gamma \vdash V : \forall\alpha.\tau]\!]\rho\ of\ i_V \circ in_\forall\lfloor d_m \rfloor\ then\ d_m;\ else\bot)$

Soundness and Adequacy

First we note some properties af the denotational semantics.

Lemma 2. *If* $\Delta;;\vdash V : \tau$ *then* $[\![\Delta;;\vdash V : \tau]\!]\{\} \neq \bot$

Proof by induction over the structure of V.
(the domains $(\mathbb{V} \multimap \mathbb{M})$ and $\mathbb{M}$ are lifted in the sum $\mathbb{V}$).

Lemma 3.

1. *The denotational semantics is well defined (the denotations exist and are unique, especially interesting for the fresh l case).*

   Proof by induction over term structure. The proof use FM-theory to show that the denotation of $ref\ V$ *is independent of which fresh location is chosen.*

2. $\Delta \vdash K : (x : \tau\ ref)^\top$ *and* $\Delta;;\vdash V : \tau$ *and* $S \in [\![\Sigma : \Delta]\!]$ *and* $l \notin (locs(\Sigma) \cup locs(K) \cup locs(V))$ *implies* $l \notin supp\big(\lambda l'.[\![\Delta \vdash K : (x : \tau\ ref)^\top]\!](S[l' \mapsto [\![\Delta;;\vdash V : \tau]\!]\{\}])\ l'\big)$

3. *Substitution. If* $\Delta;\Xi;\Gamma \vdash V : \tau$ *and* $\Delta;\Xi;\Gamma, x : \tau \vdash G : \gamma$ *and* $[\![\Delta;\Xi;\Gamma \vdash V : \tau\ ]\!]\rho \neq \bot$ *then* $[\![\Delta;\Xi;\Gamma \vdash G[V/x] : \gamma\ ]\!]\rho = [\![\Delta;\Xi;\Gamma, x : \tau \vdash G : \gamma\ ]\!]\rho[x \mapsto [\![\Delta;\Xi;\Gamma \vdash V : \tau\ ]\!]\rho]$

   Proof by induction over the structure of G. We assume $\Delta;\Xi;\Gamma \vdash V : \tau$ *and* $\Delta;\Xi;\Gamma, x : \tau \vdash G : \gamma$ *and* $\rho \neq \bot$.

4. *Compositionality. If* $C[\cdot] : (\Delta;\Xi;\Gamma \vdash \gamma) \Rightarrow (\Delta;;\vdash T\tau)$ *and* $\Delta;\Xi;\Gamma \vdash G_1 : \gamma$, $\Delta;\Xi;\Gamma \vdash G_2 : \gamma$ *and* $[\![\Delta;\Xi;\Gamma \vdash G_1 : \gamma\ ]\!] = [\![\Delta;\Xi;\Gamma \vdash G_2 : \gamma\ ]\!]$ *then* $[\![\Delta;;\vdash C[G_1] : T\tau\ ]\!] = [\![\Delta;\vdash C[G_2] : \tau\ ]\!]$

   Proof by induction over the structure of $C[\ ]$.

5. *Type variables. If* $\Delta;\Xi,\alpha;\Gamma \vdash G : \gamma$ *and* $\Xi \vdash \Gamma$ *then* $\forall - \vdash \tau' : type.$ $([\![\Delta;\Xi,\alpha;\Gamma \vdash G : \gamma]\!] = [\![\Delta;\Xi;\Gamma \vdash G : \gamma[\tau'/\alpha]]\!])$.

6. *Terms and continuations. If* $\Delta \vdash K : (y : \tau')^\top$ *and* $\Delta;;x : \tau \vdash M : T\tau'$, *then* $[\![\Delta \vdash let\ y \Leftarrow M\ in\ K : (x : \tau)^\top]\!] = i_K^{-1}(\lambda S\lambda d.[\![\Delta;;x : \tau \vdash M : T\tau']\!]\{x \mapsto d\})[\![\Delta \vdash K : (y : \tau')^\top]\!]S$

   Proof: $[\![\Delta \vdash let\ y \Leftarrow M\ in\ K : (x : \tau)^\top]\!] =$
   $\lambda S.\lambda d.[\![\Delta;;x : \tau \vdash let\ y \Leftarrow M\ in\ K : T\tau'']\!]\{x \mapsto d\}((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S =$
   $\lambda S.\lambda d.(\lambda k^2.\lambda S^2.[\![\Delta;;x : \tau \vdash M : T\tau'\ ]\!]\{x \mapsto d\}(\lambda S^3.\lambda d^3.[\![\Delta;;y : \tau' \vdash K : T\tau'']\!]\{y \mapsto d^3\}k^2S^3)S^2)((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S =$
   $\lambda S.\lambda d.([\![\Delta;;x : \tau \vdash M : T\tau'\ ]\!]\{x \mapsto d\}(\lambda S^3.\lambda d^3.[\![\Delta;;y : \tau' \vdash K : T\tau''\ ]\!]\{y \mapsto d^3\}((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S^3)S) =$
   $\lambda S.\lambda d.[\![\Delta;;x : \tau \vdash M : T\tau'\ ]\!]\{x \mapsto d\}[\![\Delta \vdash K : (y : \tau')^\top]\!]S$

Soundness

Lemma 4. *Soundness*

$$\text{If } \Delta;;\vdash M : T\tau, \quad \Delta \vdash K : (x:\tau)^\top, \quad \Sigma : \Delta \quad \text{and} \quad S \in [\![\Sigma : \Delta]\!] \quad \text{then}$$

$$\Sigma, let\ x \Leftarrow M\ in\ K \downarrow \quad \text{implies} \quad i_M^{-1}([\![\Delta;;\vdash M : T\tau\ ]\!]\{\})[\![\Delta \vdash K : (x:\tau)^\top]\!]S = \top$$

Proof by induction over termination judgements. The proof uses the previous lemmas.
The proof is in the appendix.

Adequacy
We want to show:

$$\text{If } \Delta;;\vdash M : T\tau, \quad \Delta \vdash K : (x:\tau)^\top, \quad \Sigma : \Delta \quad \text{and} \quad S \in [\![\Sigma : \Delta]\!] \quad \text{then}$$

$$i_M^{-1}([\![\Delta;;\vdash M]\!]\{\})[\![\Delta \vdash K : (x:\tau)^\top]\!]S = \top \quad \text{implies} \quad \Sigma, let\ x \Leftarrow M\ in\ K \downarrow$$

To prove adequacy, we will use a logical relation $\mathbb{R}$ which relates elements in the invariant domain $\mathbb{D}$ and typing judgements of values, continuations, computations and states. For the definition of the relation, we first define a relational structure R on FM-Cpo$_\perp^4$ and then lift our domain-constructing functor $F : (\text{FM-Cpo}_\perp^{op})^4 \times (\text{FM-Cpo}_\perp)^4 \to (\text{FM-Cpo}_\perp)^4$ to an admissible action on relations from R. We prove the existence of the relation in the line of A.Pitts[40] and M.Shinwell[52]. The relation $\mathbb{R} \in \mathsf{R}(\mathbb{D})$ will then be the invariant R-relation for F such that $i : F(\mathbb{R}, \mathbb{R}) \subset \mathbb{R}$ and $i^{-1} : \mathbb{R} \subset F(\mathbb{R}, \mathbb{R})$.

Define:

$tv = \{\Delta;;\vdash v : \tau$	$\mid \Delta;;\vdash v : \tau$	is a derivable typing judgement for a value term v }
$tk = \{\Delta\ \vdash k : (x:\tau)^\top$	$\mid \Delta\ \vdash k : (x:\tau)^\top$	is a derivable typing judgement for a continuation term k }
$tc = \{\Delta;;\vdash M : T\tau$	$\mid \Delta;;\vdash M : T\tau$	is a derivable typing judgement for a computation term M }
$ts = \{\Sigma : \Delta$	$\mid \Sigma : \Delta$	is a derivable typing judgement for a store Σ }

Definition 4. *The relational structure* R *on FM-Cpo*$_\perp^4$

A relational structure R *on FM-Cpo*$_\perp^4$*: For each domain* $D = (D_1, D_2, D_3, D_4) \in$ *FM-Cpo*$_\perp^4$ *let*
$\mathsf{R}(D_1) =$ *all subsets of* $D_1 \times tv$ *which contain* $\{\perp\} \times tv$
$\mathsf{R}(D_2) =$ *all subsets of* $D_2 \times tk$ *which contain* $\{\perp\} \times tk$
$\mathsf{R}(D_3) =$ *all subsets of* $D_3 \times tc$ *which contain* $\{\perp\} \times tc$
$\mathsf{R}(D_4) =$ *all subsets of* $D_4 \times ts$ *which contain* $\{\perp\} \times ts$
Define $\mathsf{R}(D)$ *to be* $\mathsf{R}(D_1) \times \mathsf{R}(D_2) \times \mathsf{R}(D_3) \times \mathsf{R}(D_4)$.

Definition 5.
For $D \in$ *FM-Cpo*$_\perp^4$, $\quad f = (f_1, f_2, f_3, f_4) : D \multimap D', \quad R \in \mathsf{R}(D)$,
$r = ((d_1, \Delta_1;;\vdash V : \tau_1), (d_2, \Delta_2 \vdash K : (x:\tau_2)^\top), (d_3, \Delta_3;;\vdash M : T\tau_3), (d_4, \Sigma : \Delta_4)) \in R \qquad$ *let*
$fr = ((f_1 d_1, \Delta_1;;\vdash V : \tau_1), (f_2 d_2, \Delta_2 \vdash K : (x:\tau_2)^\top), (f_3 d_3, \Delta_3;;\vdash M : T\tau_3), (f_4 d_4, \Sigma : \Delta_4))$

For $D, D' \in$ *FM-Cpo*$_\perp^4$, $\quad f : D \multimap D', \quad R \in \mathsf{R}(D), \quad R' \in \mathsf{R}(D')$
Define : $f : R \subset R'$ iff $\quad r \in R \Longrightarrow fr \in R'$

Definition 6. *A relation* $R \in \mathsf{R}(D)$ *is admissible iff for all* $D', S \in \mathsf{R}(D')$*the set* $\{\ f \mid f : S \subset R\ \}$ *contains* $\perp$ *and is closed under least upper bounds of countable FM-chains.*

Lemma 5. R *has inverse images.*
For all $f : D \multimap E$ *and* $S \in \mathsf{R}(E)$ *exists a relation* $f^*S \in \mathsf{R}(D)$ *such that*
$\forall g : D' \multimap D, R \in \mathsf{R}(D').\ g : R \subset f^*S \Leftrightarrow (f \circ g) : R \subset S$

For $f : D \multimap E,\ S \in \mathcal{R}(E)$ define
$f^*S = \{\ r = ((d_1, \Delta;;\vdash V : \tau), (d_2, \Delta \vdash K : (x:\tau)^\top), (d_3, \Delta;;\vdash M : T\tau), (d_4, \Sigma : \Delta))\ |$
$\quad (d_1, d_2, d_3, d_4) \in D \textit{ and } fr \in S\ \}$

$f^*S \in \mathsf{R}(D)$ and fulfills the inverse image requirement. Since $S \in \mathsf{R}$ it contains $(\bot \times tv) \times (\bot \times tk) \times (\bot \times tc) \times (\bot \times ts)$ and by strictness of f the same is true of f^*S.

Let $g : D' \multimap D,\ R \in \mathsf{R}(D')$. Assume that $g : R \subset f^*S$, then by definition $r \in R \Rightarrow gr \in f^*S$, and again by definition $gr \in f^*S \Rightarrow fgr \in S$, hence $r \in R \Rightarrow (f \circ g)r \in S$ that is $(f \circ g) : R \subset S$. The other direction, assume $(f \circ g) : R \subset S$, then by definition $r \in R \Rightarrow fgr \in S$. Since $g : D' \multimap D$ we know that the domain-element part of $gr \in D$, and then by definition $fgr \in S \Rightarrow gr \in f^*S$, hence $g : R \subset f^*S$.

Lemma 6. *R has intersections.*
Let $\mathcal{S} \subseteq \mathsf{R}(D)$, then exists a relation $\cap\mathcal{S} \in \mathsf{R}(D)$ satisfying $r \in \cap\mathcal{S} \Leftrightarrow \forall S \in \mathcal{S}. r \in S$.
If $\mathcal{S}$ is a set of admissible relations on D then $\cap\mathcal{S}$ is an admissible relation on D.

$\cap\mathcal{S}$ is the set-theoretical intersection.

Definition 7. *Relational lifting of F:*
Let $D^- = (D_1^-, D_2^-, D_3^-, D_4^-) \in FM\text{-}Cpo_\bot^4$ and let $R^- = (R_1^-, R_2^-, R_3^-, R_4^-) \in \mathsf{R}(D^-)$.
Let $D^+ = (D_1^+, D_2^+, D_3^+, D_4^+) \in FM\text{-}Cpo_\bot^4$ and let $R^+ = (R_1^+, R_2^+, R_3^+, R_4^+) \in \mathsf{R}(D^+)$.

Define $F(R^-, R^+) = F((R_1^-, R_2^-, R_3^-, R_4^-), (R_1^+, R_2^+, R_3^+, R_4^+)) = (R_V, R_K, R_M, R_S)$ where
$R_V = \{\ \ (\bot,\ \Delta;;\vdash V : \tau) \mid (\Delta;;\vdash V : \tau) \in tv\ \}$
$\quad \cup \{\ (in_{\mathbf{1}}*,\ \Delta;;\vdash () : unit)\ \}$
$\quad \cup \{\ (in_{\mathbb{Z}}n,\ \Delta;;\vdash \underline{n} : int)\ \}$
$\quad \cup \{\ (in_{\mathbb{L}}l,\ \Delta;;\vdash l : \Delta(l)\text{-}ref) \mid l \in dom(\Delta)\ \}$
$\quad \cup \{\ (in_{\oplus}d,\ \Delta;;\vdash V : \tau_1 + \tau_2) \mid d = in_i d' \wedge V = in_i V' \wedge \exists \Delta_0 \subseteq \Delta.(d', \Delta_0;;\vdash V' : \tau_i) \in R_1^+, i \in 1,2\ \}$
$\quad \cup \{\ (in_{\otimes}d,\ \Delta;;\vdash V : \tau_1 \times \tau_2) \mid d = (d_1, d_2) \wedge V = (V_1, V_2) \wedge$
$\qquad\qquad \exists \Delta_0 \subseteq \Delta.(d_1, \Delta_0;;\vdash V_1 : \tau_1) \in R_1^+ \wedge (d_2, \Delta_0;;\vdash V_2 : \tau_2) \in R_1^+\ \}$
$\quad \cup \{\ (in_{\multimap}d,\ \Delta;;\vdash V : \tau \to T\tau') \mid \forall \Delta' \supseteq \Delta.\forall(d',\ \Delta';;\vdash V' : \tau) \in R_1^-.(dd',\ \Delta';;\vdash VV' : T\tau') \in R_3^+\ \}$
$\quad \cup \{\ (in_{\mu}d,\ \Delta;;\vdash foldV : \mu\alpha.\tau) \mid \exists \Delta_0 \subseteq \Delta.(d, \Delta_0;;\vdash V : \tau[\mu\alpha.\tau/\alpha]) \in R_1^+\ \}$
$\quad \cup \{\ (in_{\forall}d,\ \Delta;;\vdash \Lambda\alpha.M : \forall\alpha.T\tau) \mid \exists \Delta_0 \subseteq \Delta.\forall\tau' \textit{ with}\_ \vdash \tau' : type.\ (d, \Delta_0;;\vdash M : T\tau[\tau'/\alpha]) \in R_3^+\ \}$

$R_K = \{\ (\bot,\ \Delta \vdash K : (x:\tau)^\top\ \}$
$\quad \cup \{\ (k,\ \Delta \vdash K : (x:\tau)^\top) \mid \forall \Delta' \supseteq \Delta.\forall(s, \Sigma : \Delta') \in R_4^-.\forall(v, \Delta';;\vdash V : \tau) \in R_1^-.$
$\qquad\qquad (ksv = \top) \Rightarrow (\Sigma, let\ x \Leftarrow val\ V\ in\ K\ \downarrow)\ \}$

$R_M = \{\ (\bot,\ \Delta;;\vdash M : T\tau)\ \}$
$\quad \cup \{\ (m,\ \Delta;;\vdash M : T\tau) \mid \forall \Delta' \supseteq \Delta.\forall(k, \Delta' \vdash K : (x:\tau)^\top \in R_2^-.\forall(s, \Sigma : \Delta') \in R_4^-.$
$\qquad\qquad (mks = \top) \Rightarrow (\Sigma, let\ x \Leftarrow M\ in\ K\ \downarrow)\ \}$

$R_S\ = \{\ (\bot,\ \Sigma : \Delta)\ \}$
$\quad \cup \{\ (S,\ \Sigma : \Delta) \mid \forall l \in dom(\Delta).(S(l)\ ,\ \Delta;;\vdash \Sigma(l) : \Delta(l)) \in R_1^+\ \}$

When $R^-, R^+ \in \mathsf{R}$ then $F(R^-, R^+) \in \mathsf{R}$.

It is immediate from the definition that each of the four projections of $F(R^-, R^+)$ include $\{(\bot, \textit{typing conclusion})\})$.

Lemma 7. *If R^+ is admissible, then $F(R^-, R^+)$ is admissible.*

Proof Assume R^+ is admissible. We want to show $F(R^-, R^+)$ is admissible for all $R^- \in \mathsf{R}(D^-)$. It suffices to show for each of the four projections, that it is closed under least upper bounds of finitely supported chains and contains $\{(\bot, typingjudgement)\}$. We have by definition $F(R^-,R^+)_1 \supseteq \{\ (\bot,\ \Delta;\vdash V:\tau)\ \}$, $F(R^-,R^+)_2 \supseteq \{\ (\bot,\ \Delta;\vdash K(x:\tau)^\top\ \}$, $F(R^-,R^+)_3 \supseteq \{\ (\bot,\ \Delta;\vdash M:T\tau)\ \}$ and $F(R^-,R^+)_4 \supseteq \{\ (\bot,\ \Sigma:\Delta)\ \}$.

$\mathcal{R}_S$: Assume a chain $(S^i,\ \Sigma:\Delta)_{i\in\omega}$ in $\mathcal{R}_S$.
If S^i is constantly $\bot$ we are done. Else it holds from some point onwards that $\forall l \in dom(\Delta).(S^i(l)\ ,\ \Delta;\vdash \Sigma(l):\Delta(l)) \in R_1^+$. $(S^i(l)\ ,\ \Delta;\vdash \Sigma(l):\Delta(l))_{i\in\omega}$ is a chain in R_1^+. Since R^+ is admissible it holds $\forall l \in dom(\Delta)$ that $(\bigsqcup_i(S^i(l))\ ,\ \Delta;\vdash \Sigma(l):\Delta(l)) =$ $((\bigsqcup_i S^i)(l)\ ,\ \Delta;\vdash \Sigma(l):\Delta(l)) \in R_1^+$. Hence $(\bigsqcup_i S^i,\ \Sigma:\Delta)_{i\in\omega}$ in $\mathcal{R}_S$.

$\mathcal{R}_M$: Assume a chain $(m^i,\ \Delta;;\vdash M:T\tau)_{i\in\omega}$ in $\mathcal{R}_M$.
If m^i is constantly $\bot$ we are done. Else it holds from some point onwards that $\forall\Delta'\supseteq\Delta$. $\forall(k,\Delta'\vdash K:(x:\tau)^\top \in R_2^-.\forall(s,\Sigma:\Delta')\in R_4^-.\ (m^iks=\top)\Rightarrow(\Sigma, let\ x\Leftarrow M\ in\ K\ \downarrow)$.
We want to show $((\bigsqcup_i m^i)ks=\top)\Rightarrow(\Sigma, let\ x\Leftarrow M\ in\ K\ \downarrow)$.
$(m^iks)_{i\in\omega}$ is a chain in $\mathbb{O}$ and $(\bigsqcup_i m^i)ks = \bigsqcup_i(m^iks)$. Assume $(\bigsqcup_i m^i)ks = \bigsqcup_i(m^iks) = \top$. This implies that there exist $j\in\omega$ such that $\forall i\geq j.m^iks=\top$. This again implies $(\Sigma, let\ x\Leftarrow M\ in\ K\ \downarrow)$.

$\mathcal{R}_K$: The proof is similar to the proof for $\mathcal{R}_M$ so we omit it.

$\mathcal{R}_V$: Assume a chain $(v^i,\ \Delta;;\vdash V:\tau)_{i\in\omega}$ in $\mathcal{R}_V$.
If v^i is constantly $\bot$ we are done. Else $\exists j\in\omega.\forall i\geq j.\ v^i\neq\bot$ and the proof is by *case analysis* over outermost type constructors.

(unit)(int)(loc): Chains of these types must be constant from some point onwards.

$(\forall)$: Assume a chain $(in_\forall d^i,\ \Delta;;\vdash \Lambda\alpha.M:\forall\alpha.T\tau)_{i\in\omega}\in R_V$. Then $\forall i.\exists\Delta_0\subseteq\Delta.\forall\tau':type.\ (d^i,\ \Delta_0;;\vdash M:T(\tau[\tau'/\tau]))\in R_3^+$. There are only finitely many $\Delta_0\subseteq\Delta$ and hence one such Δ_0' must occur infinitely often. This gives a subsequence $(in_\forall d^{ij},\ \Delta;;\vdash\Lambda\alpha.M:\forall\alpha.T\tau)_{ij}$ such that $\bigsqcup(in_\forall d^i)=\bigsqcup(in_\forall d^{ij})=in_\forall(\bigsqcup d^{ij})$. Now since R_3^+ is admissible and $\forall ij.\forall\tau':type.\ (d^{ij},\ \Delta_0';\vdash M:T(\tau[\tau'/\tau]))\in R_3^+$, then also $\forall\tau':type.\ (\bigsqcup d^{ij},\ \Delta_0';;\vdash M:T(\tau[\tau'/\tau]))\in R_3^+$. Then $(\bigsqcup(in_\forall d^i),\ \Delta;;\vdash\Lambda\alpha.M:\forall\alpha.T\tau)\in R_V$.

$(+)(\times)(\mu)$: That least upper bounds are included follows from the admissibility of R^+.

$(\to)$: Assume a chain $(in_{\multimap}d^i,\ \Delta;;\vdash V:\tau\to T\tau')_{i\in\omega}$. Let $\Delta'\supseteq\Delta$, $(d',\ \Delta';;\vdash V':\tau)\in R^-$. We want to show that $((\bigsqcup d^i)d',\ \Delta';\vdash VV':T\tau')\in R_3^+$. $(\bigsqcup d^i)d'=\bigsqcup(d^id')$. $(d^id',\ \Delta';;\vdash VV':T\tau')_{i\in\omega}$ is a chain in R_3^+, since R^+ is admissible its least upper bound $((\bigsqcup d^i)d',\ \Delta';;\vdash VV':T\tau')\in R_3^+$.

□

Let $\quad f^-:E^-\multimap D^-,\qquad f^+:D^+\multimap E^+$, then $\qquad F(f^-,f^+):F(D^-,D^+)\multimap F(E^-,E^+)$

Lemma 8. *Let* $\quad R^-\in\mathsf{R}(D^-),\qquad S^-\in\mathsf{R}(E^-),\qquad R^+\in\mathsf{R}(D^+),\qquad S^+\in\mathsf{R}(E^+)$
assume $\ f^-:S^-\subset R^-\ $ *and* $\ f^+:R^+\subset S^+$
then $\qquad F(f^-,f^+)=h=(h_1,h_2,h_3,h_4):F(R^-,R^+)\subset F(S^-,S^+)$

Proof

Assume $(S,\ \Sigma:\Delta)\in F(R^-,R^+)_4$. We want to show $(h_4S,\ \Sigma:\Delta)\in F(S^-,S^+)_4$.
If $S=\bot$ then by strictness $(h_4S,\ \Sigma:\Delta)=(\bot,\ \Sigma:\Delta)\in F(S^-,S^+)_4$. Else $(h_4S,\ \Sigma:\Delta)=(\lambda l.f_1^+(Sl),\ \Sigma:\Delta)$ and we need to show that $\forall l\in dom(\Delta).(f_1^+(Sl),\ \Delta;;\vdash\Sigma(l):\Delta(l))\in S^+$. This follows from the assumptions $f^+:R^+\subset S^+$ and $\forall\in dom(\Delta).Sl,\ \Delta;;\vdash\Sigma(l):\Delta(l))\in R^+$.

Assume $(m,\ \Delta;\vdash M : T\tau) \in F(R^-, R^+)_3$. We want to show $(h_3 m,\ \Delta;\vdash M : T\tau) \in F(S^-, S^+)_3$. If $m = \bot$ then by strictness $(h_3 m,\ \Delta;\vdash M : T\tau) = (\bot,\ \Delta;\vdash M : T\tau) \in F(S^-, S^+)_4$. Else $(h_3 m,\ \Delta;\vdash M : T\tau) = (\lambda k.\lambda s.m(f_2^- k)(f_4^- s),\ \Delta;\vdash M : T\tau)$ and we need to show that $\forall \Delta' \supseteq \Delta. \forall (k, \Delta' \vdash K : (x : \tau)^\top) \in S_2^-. \forall (s, \Sigma : \Delta') \in S_4^-. (m(f_2^- k)(f_4^- s) = \top) \Rightarrow (\Sigma,\ \mathit{let}\ x \Leftarrow M\ \mathit{in}\ K \downarrow)$. Since $f^- : S^- \subset R^-$ then $(f_2^- k,\ \Delta' \vdash K : (x : \tau)^\top) \in R_2^-$ and $(f_4^- s,\ \Sigma : \Delta') \in R_4^-$. Then since $(m,\ \Delta;\vdash M : T\tau) \in F(R^-, R^+)_3$ it follows that $(m(f_2^- k)(f_4^- s) = \top) \Longrightarrow (\Sigma,\ \mathit{let}\ x \Leftarrow M\ \mathit{in}\ K \downarrow)$.

By similar arguments $(k,\ \Delta \vdash K : (x : \tau)^\top) \in F(R^-, R^+)_2$ implies $(h_2 k,\ \Delta \vdash K : (x : \tau)^\top) \in F(S^-, S^+)_2$. We omit the proof.

Assume $(v,\ \Delta;;\vdash V : \tau) \in F(R^-, R^+)_1$. We want to show $(h_1 v,\ \Delta;;\vdash V : \tau) \in F(S^-, S^+)_1$. If $v = \bot$ then by strictness $(h_1 v,\ \Delta;;\vdash V : \tau) = (\bot,\ \Delta;;\vdash V : \tau) \in F(S^-, S^+)_1$.

Else $v \neq \bot$ and the proof is by cases of outermost type constructors.

- Assume $(v,\ \Delta;;\vdash () : \mathit{unit}) \in F(R^-, R^+)_1$. Then $v = in_{\mathbf{1}} * = h_1 v$ and $(in_{\mathbf{1}} *,\ \Delta;;\vdash () : \mathit{unit}) \in F(S^-, S^+)_1$.
- Assume $(v,\ \Delta;;\vdash \underline{n} : \mathit{int}) \in F(R^-, R^+)_1$. Then $v = in_{\mathbb{Z}} n = h_1 v$ and $(in_{\mathbb{Z}} n,\ \Delta;;\vdash \underline{n} : \mathit{int}) \in F(S^-, S^+)_1$.
- Assume $(v,\ \Delta;;\vdash l : \Delta(l)\ \mathit{ref}) \in F(R^-, R^+)_1$. Then $v = in_{\mathbb{L}} l = h_1 v$ and $(in_{\mathbb{L}} l,\ \Delta;;\vdash l : \Delta(l)\ \mathit{ref}) \in F(S^-, S^+)_1$.
- Assume $(v,\ \Delta;;\vdash V : \tau_1 + \tau_2) \in F(R^-, R^+)_1$. Then $v = in_{\oplus}(in_i d')$ and $V = in_i V'$ and $(d',\ \Delta;;\vdash V' : \tau_i) \in R_1^+$.
  $h_1 v = in_{\oplus}(in_i f_1^+ d')$ and by the assumption $f^+ : R^+ \subset S^+$ we have that $(f^+ d',\ \Delta;;\vdash V' : \tau_i) \in S_1^+$. So $(h_1 v,\ \Delta;;\vdash V : \tau_1 + \tau_2) \in F(S^-, S^+)_1$.
- Assume $(v,\ \Delta;;\vdash V : \tau_1 \times \tau_2) \in F(R^-, R^+)_1$. The proof that $(h_1 v,\ \Delta;;\vdash V : \tau_1 \times \tau_2) \in F(S^-, S^+)_1$is similar to the proof for sum type, we omit it.
- Assume $(v,\ \Delta;;\vdash V : \mu\alpha.\tau) \in F(R^-, R^+)_1$. The proof that $(h_1 v,\ \Delta;;\vdash V : \mu\alpha.\tau) \in F(S^-, S^+)_1$ is similar to the proof for sum type, we omit it.
- Assume $(v,\ \Delta;;\vdash V : \tau \to T\tau') \in F(R^-, R^+)_1$. It must be the case that $v = in_{\multimap}\lfloor d \rfloor$, and $h_1 v = in_{\multimap}\lfloor \lambda x. f_3^+(d(f_1^- x)) \rfloor$.
  Let $\Delta' \supseteq \Delta$, $(w,\ \Delta';;\vdash W : \tau) \in S_1^-$, we need to show $((\lambda x. f_3^+(d(f_1^- x)))w,\ \Delta';;\vdash VW) \in S_3^+$. Since $f^- : S^- \subset R^-$ then $(f_1^- w,\ \Delta';;\vdash W : \tau) \in R_1^-$. By the assumption $(in_{\multimap}\lfloor d \rfloor,\ \Delta;;\vdash V : \tau \to T\tau') \in F(R^-, R^+)_1$, then $(d(f_1^- w),\ \Delta';;\vdash VW) \in R_3^+$. Since $f^+ : R^+ \subset S^+$ then $((f_3^+(d(f_1^- w))),\ \Delta';;\vdash VW) = ((\lambda x. f_3^+(d(f_1^- x)))w,\ \Delta';\vdash VW) \in S_3^+$.
- Assume $(v,\ \Delta;;\vdash \Lambda\alpha.M : \forall\alpha.T\tau) \in F(R^-, R^+)_1$. It must be the case that $v = in_{\forall}\lfloor d \rfloor$, and $\exists \Delta_0 \subseteq \Delta. \forall \tau' : \mathit{type}.\ (d,\ \Delta_0;;\vdash\ M : T\tau[\tau'/\alpha]) \in R_3^+$.
  $h_1 v = in_{\forall}\lfloor f_3^+(d) \rfloor$. Since $f^+ : R^+ \subset S^+$ then $\exists \Delta_0 \subseteq \Delta. \forall \tau' : \mathit{type}.\ (f_3^+ d,\ \Delta_0;;\vdash\ M : T\tau[\tau'/\alpha]) \in S_3^+$. Hence $(h_1 v,\ \Delta;;\vdash \Lambda\alpha.M : \forall\alpha.T\tau) \in F(S^-, S^+)$.

□

We now conclude that the action of F is admissible. It follows (cf.[52]p.)[2] that:

Theorem 1. *There exists an invariant* R*-relation* $\mathbb{R} = (\mathbb{R}_V, \mathbb{R}_K, \mathbb{R}_M, \mathbb{R}_S) \in \mathsf{R}(\mathbb{D})$ *such that* $i : F(\mathbb{R}, \mathbb{R}) \subset \mathbb{R}$ *and* $i^{-1} : \mathbb{R} \subset F(\mathbb{R}, \mathbb{R})$.

[2] Later we show a full existence proof for the logical relation we define on top of the denotational semantics

By the definition of the action of F on $\mathcal{R}$-relations it holds that

$$\begin{aligned}
\mathbb{R}_V \cong\ & \{\ (\bot,\quad \Delta;;\vdash V:\tau)\ \}\\
& \cup \{\ (in_{\mathbf{1}}*,\ \Delta;;\vdash ():unit)\ \}\\
& \cup \{\ (in_{\mathbb{Z}}n,\ \Delta;;\vdash \underline{n}:int)\ \}\\
& \cup \{\ (in_{\mathbb{L}}l,\ \Delta;;\vdash l:\Delta(l)\text{-}ref)\ \}\\
& \cup \{\ (in_{+}d,\ \Delta;;\vdash V:\tau_1+\tau_2) \mid d=in_i d' \wedge V=in_i V' \wedge \exists\Delta_0\subseteq\Delta.\ (d',\ \Delta_0;;\vdash V':\tau_i)\in\mathbb{R}_V, i\in 1,2\ \}\\
& \cup \{\ (in_{\times}d,\ \Delta;;\vdash V:\tau_1\times\tau_2) \mid d=(d_1,d_2) \wedge V=(V_1,V_2) \wedge \exists\Delta_0\subseteq\Delta.\ (d_i,\ \Delta_0;;\vdash V_i:\tau_i)\in\mathbb{R}_V, i\in 1,2\ \}\\
& \cup \{\ (in_{\multimap}d,\ \Delta;;\vdash V:\tau\to T\tau') \mid \forall \text{ closed } \Delta'\supseteq\Delta.\forall(d',\ \Delta';;\vdash V':\tau)\in\mathbb{R}_V.(dd',\ \Delta';;\vdash VV':T\tau')\in\mathbb{R}_M\ \}\\
& \cup \{\ (in_{\mu}d,\ \Delta;;\vdash foldV:\mu\alpha.\tau) \mid \exists\Delta_0\subseteq\Delta.\ (d,\ \Delta_0;;\vdash V:\tau[\mu\alpha.\tau/\alpha])\in\mathbb{R}_V\ \}\\
& \cup \{\ (in_{\forall}d,\ \Delta;;\vdash \Lambda\alpha.M:\forall\alpha.T\tau) \mid \exists\Delta_0\subseteq\Delta.\forall\tau' \ with\ \_\vdash\tau':type.\ (d,\ \Delta_0;;\vdash M:T\tau[\tau'/\alpha])\in\mathbb{R}_M\ \}
\end{aligned}$$

$$\begin{aligned}
\mathbb{R}_K \cong\ & \{\ (\bot,\ \Delta\vdash K:(x:\tau)^{\top}\ \}\\
& \cup \{\ (k,\ \Delta\vdash K:(x:\tau)^{\top}) \mid \forall \text{ closed } \Delta'\supseteq\Delta.\forall(s,\Sigma:\Delta')\in\mathbb{R}_S.\forall(v,\ \Delta';;\vdash V:\tau)\in\mathbb{R}_V.\\
& \quad (ksv=\top)\Rightarrow(\Sigma, let\ x\Leftarrow val\ V\ in\ K\ \downarrow)\ \}
\end{aligned}$$

$$\begin{aligned}
\mathbb{R}_M \cong\ & \{\ (\bot,\ \Delta;;\vdash M:T\tau)\ \}\\
& \cup \{\ (m,\ \Delta;;\vdash M:T\tau) \mid \forall \text{ closed } \Delta'\supseteq\Delta.\forall(k,\ \Delta'\vdash K:(x:\tau)^{\top}\in\mathbb{R}_K.\forall(s,\ \Sigma:\Delta')\in\mathbb{R}_S.\\
& \quad (mks=\top)\Rightarrow(\Sigma, let\ x\Leftarrow M\ in\ K\ \downarrow)\ \}
\end{aligned}$$

$$\begin{aligned}
\mathbb{R}_S \cong\ & \{\ (\bot,\ \Sigma:\Delta)\ \}\\
& \cup \{\ (S,\ \Sigma:\Delta) \mid \forall l\in dom(\Delta).\ (S(l)\ ,\ \Delta;;\vdash\Sigma(l):\Delta(l))\in\mathbb{R}_V\ \}
\end{aligned}$$

Lemma 9. *Weakening*

1. *If* $(m,\ \Delta;;\vdash M:T\tau)\in\mathbb{R}_M$ *then* $\forall\ \Delta'\supseteq\Delta.\ (m,\ \Delta';;\vdash M:\tau)\in\mathbb{R}_M$.
2. *If* $(k,\ \Delta\vdash K:(x:\tau)^{\top})\in\mathbb{R}_K$ *then* $\forall\ \Delta'\supseteq\Delta.\ (k,\ \Delta'\vdash K:(x:\tau)^{\top})\in\mathbb{R}_K$.
3. *If* $(v,\ \Delta;;\vdash V:\tau)\in\mathbb{R}_V$ *then* $\forall\ \Delta'\supseteq\Delta.\ (v,\ \Delta';;\vdash V:\tau)\in\mathbb{R}_V$.

Proof First note that we are relating to derivable typing judgments when $\Delta'\supseteq\Delta$.

1. Assume $(m,\ \Delta;;\vdash M:T\tau)\in\mathbb{R}_3$ and let $\Delta'\supseteq\Delta$. If $m=\bot$ also $(m,\ \Delta';;\vdash M:T\tau)\in\mathbb{R}_3$. Else let $\Delta''\supseteq\Delta'$, $(k,\ \Delta''\vdash K:(x:\tau)^{\top}\in\mathbb{R}_K,\ (s,\Sigma:\Delta'')\in\mathbb{R}_S$. Assume $mks=\top$. Since $(m,\ \Delta;;\vdash M:T\tau)\in\mathbb{R}_M$ and $\Delta''\supseteq\Delta$ this implies $(\Sigma, let\ x\Leftarrow M\ in\ K\ \downarrow)$.
   We conclude $(m,\ \Delta';;\vdash M:T\tau)\in\mathbb{R}_M$
2. The proof is similar as for (1) so we omit it.
3. Assume $(v,\ \Delta;;\vdash V:\tau)\in\mathbb{R}_V$ and let $\Delta'\supseteq\Delta$. If $v=\bot$ also $(v,\ \Delta';;\vdash V:\tau)\in\mathbb{R}_V$. Else the proof is by cases of type constructors.

   (unit) and (int) are immediate from the definition.

   ($\sigma\ ref$): When $\Delta'\supseteq\Delta$ and $l:\sigma\in\Delta$ then $l:\sigma\in\Delta'$.Hence $(v,\ \Delta;;\vdash V:\sigma\ ref)\in\mathbb{R}_V$ implies $(v,\ \Delta';;\vdash V:\sigma\ ref)\in\mathbb{R}_V$.

   $(+)(\times)(\mu)(\forall)$: Follows by transitivity of $\subseteq$ for stores $(\Delta_0\subseteq\Delta\subseteq\Delta')$.

   $(\to)$: Follows by transitivity of $\subseteq$ for stores $(\Delta''\supseteq\Delta'\subseteq\Delta)$.

□

Lemma 10. $\forall\Delta.\forall-\vdash\tau.\ ((\lambda S.(\lambda d.\top)_{\bot})_{\bot},\ \Delta\vdash val\ x:(x:\tau))\in\mathbb{R}_K$.

Let $(s,\ \Sigma:\Delta)\in\mathbb{R}_S$ and $(v,\ \Delta;;\vdash V:\tau)\in\mathbb{R}_V$ and assume $(\lambda S.(\lambda d.\top)_{\bot})_{\bot}sv=\top$ then $s\neq\bot$ and $v\neq\bot$. $\Sigma, let\ x\Leftarrow val\ V\ in\ val\ x\downarrow$ is well-formed and hence terminates.

Definition 8. *Extension of relation to open terms, $\mathbb{R}_{\Xi\Gamma} = (\mathbb{R}_{\Xi\Gamma\vdash V}, \mathbb{R}_K, \mathbb{R}_{\Xi\Gamma\vdash M}, \mathbb{R}_S)$*
For $\Xi = \alpha_1, \ldots, \alpha_m$ and $\Gamma = x_1 : \tau_1, \ldots, x_n : \tau_n$ with Γ well typed by Ξ define:

$$\begin{aligned}
&\mathbb{R}_{\Xi\Gamma\vdash V} = \{\ (f,\ \Delta;\Xi;\Gamma \vdash V : \tau)\ |\\
&f \in (\mathbb{V}^n \multimap \mathbb{V})\ \wedge\ \Delta;\Xi;\Gamma \vdash V : \tau\ \wedge\\
&\quad \forall \textit{ closed value types } \sigma_1 \ldots \sigma_m.\ \forall \Delta' \supseteq \Delta.\\
&\quad \forall (d_1,\ \Delta';;\vdash V_1 : \tau_1[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_{\mathbb{V}} \ldots (d_n,\ \Delta';;\vdash V_n : \tau_n[\overline{\sigma_j/\alpha_j}]) \textit{ with } \rho = d_1 \otimes \ldots \otimes d_n\\
&\qquad (f(\rho),\ \Delta';;\vdash V[\overline{V_i/x_i}] : \tau[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_V\ \}
\end{aligned}$$

$$\begin{aligned}
&\mathbb{R}_{\Xi\Gamma\vdash M} = \{\ (f,\ \Delta;\Xi;\Gamma \vdash M : T\tau)\ |\\
&f \in (\mathbb{V}^n \multimap \mathbb{M})\ \wedge\ \Delta;\Xi;\Gamma \vdash M : T\tau\ \wedge\\
&\quad \forall \textit{ closed value types } \sigma_1 \ldots \sigma_m.\ \forall \supseteq \Delta.\\
&\quad \forall (d_1,\ \Delta';;\vdash V_1 : \tau_1[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_{\mathbb{V}} \ldots (d_n,\ \Delta';;\vdash V_n : \tau_n[\overline{\sigma_j/\alpha_j}]) \textit{ with } \rho = d_1 \otimes \ldots \otimes d_n\\
&\qquad (f(\rho),\ \Delta';;\vdash M[\overline{V_i/x_i}] : \tau[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_M\ \}
\end{aligned}$$

Lemma 11. *If $(m,\ \Delta;;x : \tau_1 \vdash M : T\tau_2) \in \mathbb{R}_{x:\tau_1\vdash M}$ and $(k,\ \Delta \vdash K : (y : \tau_2)^\top) \in \mathbb{R}_K$ then $(\lambda S'.\lambda d'.\ m(x \mapsto d')kS'),\ \Delta \vdash let\ y \Leftarrow M\ in\ K : (x : \tau_1)^\top) \in \mathbb{R}_K$.*

Proof Let $\Delta' \supseteq \Delta$, $(S,\ \Sigma : \Delta') \in \mathbb{R}_S$, $(w,\ \Delta';;\vdash W : \tau_1) \in \mathbb{R}_V$.
Assume $(\lambda S'.\lambda d'.\ m(x \mapsto d')kS')Sw = m(x \mapsto w)kS = \top$. We want to show that this implies $\Sigma,\ let\ x \Leftarrow val\ W\ in\ (let\ y \Leftarrow M\ in\ K) \downarrow$.

By assumption $(m(x \mapsto w),\ \Delta;;\vdash M[W/x] : T\tau_2) \in \mathbb{R}_M$. So it follows that $\Sigma,\ let\ y \Leftarrow M[W/x]\ in\ K \downarrow$. By judgments of termination then $\Sigma,\ let\ x \Leftarrow val\ W\ in\ (let\ y \Leftarrow M\ in\ K) \downarrow$ □

Lemma 12. *Typing rules preserve the $\mathbb{R}_\Gamma$ relation of a term and its denotation.*

Proof is in the appendix.

Theorem 2. *Fundamental theorem*

If	$\Delta;\Xi;\Gamma \vdash V : \tau$	*then*	$(\llbracket \Delta;\Xi;\Gamma \vdash V \rrbracket\ ,\ \Delta;\Xi;\Gamma \vdash V : \tau) \in \mathbb{R}_{\Xi\Gamma\vdash V}$
If	$\Delta \vdash K : (x : \tau)^\top$	*then*	$(\llbracket \Delta \vdash K : (x : \tau)^\top \rrbracket\ ,\ \Delta \vdash K : (x : \tau)^\top) \in \mathbb{R}_K$
If	$\Delta;\Xi;\Gamma \vdash M : T\tau$	*then*	$(\llbracket \Delta;\Xi;\Gamma \vdash M \rrbracket\ ,\ \Delta;\Xi;\Gamma \vdash M : T\tau) \in \mathbb{R}_{\Xi\Gamma\vdash M}$
If	$\Sigma : \Delta$	*then*	$(S\ ,\ \Sigma : \Delta) \in \mathbb{R}_S$ *whenever* $S \in \llbracket \Sigma : \Delta \rrbracket$

Proof

1. Proof of fundamental theorem for values and computations by induction on typing derivation using the previous lemma 12.
2. For continuations the proof is by induction over the construction of the continuation term. The base case is $\forall \Delta.\forall - \vdash \tau.\ (\llbracket \Delta \vdash val\ x : (x : \tau)^\top \rrbracket,\ \Delta \vdash val\ x : (x : \tau)^\top) \in \mathbb{R}_K$. Since $\llbracket \Delta \vdash val\ x : (x : \tau)^\top \rrbracket = \lambda S.\lambda d.\llbracket \Delta;;x : \tau \vdash val\ x : T\tau \rrbracket (x \mapsto d)((\lambda S^0.(\lambda d^0.\top)_\bot)_\bot)S = \lambda S.\lambda d.((\lambda S^0.(\lambda d^0.\top)_\bot)_\bot)Sd = ((\lambda S^0.(\lambda d^0.\top)_\bot)_\bot)$, then this follows from lemma 10.
For the inductive case assume $(\llbracket \Delta \vdash K : (y : \tau') \rrbracket,\ \Delta \vdash K : (y : \tau')) \in \mathbb{R}_K$. To show for $\Delta;;x : \tau \vdash M : T\tau'$ that $(\llbracket \Delta \vdash let\ y \Leftarrow M\ in\ K \rrbracket,\ \Delta \vdash let\ y \Leftarrow M\ in\ K : (x : \tau)^\top) \in \mathbb{R}_K$. Let $\Delta' \supseteq \Delta$ and $(s,\ \Sigma : \Delta') \in \mathbb{R}_S$ and $(v,\ \Delta';;\vdash V : \tau) \in \mathbb{R}_V$ and assume $\llbracket \Delta \vdash let\ y \Leftarrow M\ in\ K \rrbracket sv = \top$. We need to show that $\Sigma,\ let\ x \Leftarrow val\ V\ in\ (let\ y \Leftarrow M\ in\ K) \downarrow$. By judgement of termination it suffices to show $\Sigma,\ let\ y \Leftarrow M[V/x]\ in\ K \downarrow$. By 1) it holds that $(\llbracket \Delta';,x : \tau \vdash M : T\tau' \rrbracket (x \mapsto v),\ \Delta';;\vdash M[V/x] : T\tau') \in \mathbb{R}_M$. And then since $\llbracket \Delta \vdash let\ y \Leftarrow M\ in\ K \rrbracket sv = \llbracket \Delta;;x : \tau \vdash M : T\tau' \rrbracket (x \mapsto v)(\llbracket \Delta \vdash K : (y : \tau')^\top \rrbracket)s$ then $\Sigma,\ let\ y \Leftarrow M[V/x]\ in\ K \downarrow$ follows.
3. For states if follows from typing rules for states together with fundamental lemma for values.

□

Corollary 3. *Adequacy*

$$\text{If } \Delta;;\vdash M : T\tau, \qquad \Delta \vdash K : (x:\tau)^{\top}, \qquad \Sigma : \Delta \quad \text{and} \quad S \in [\![\Sigma : \Delta]\!] \quad \text{then}$$

$$[\![\Delta;;\vdash M : T\tau]\!]\{\}[\![\Delta \vdash K : (x:\tau)^{\top}]\!]S = \top \quad \text{implies} \quad \Sigma, let\ x \Leftarrow M\ in\ K \ \downarrow$$

Proof By fundamental theorem $([\![\Delta;;\vdash M : T\tau]\!],\ \Delta;;\vdash M : T\tau) \in \mathbb{R}_M$ and $([\![\Delta \vdash K : (x:\tau)^{\top}]\!],\ \Delta \vdash K : (x:\tau)^{\top}) \in \mathbb{R}_K$ and $(S, [\![\Sigma : \Delta]\!]) \in \mathbb{R}_S$, then by the definition of $\in \mathbb{R}_M$ adequacy follows.

□

By soundness and adequacy of the denotational semantics we derive:

Proposition 3. *Contextual equivalence.*
For all $\Delta;\Xi;\Gamma \vdash G_1 : \gamma$ and $\Delta;\Xi;\Gamma \vdash G_2 : \gamma$ it holds that

if $[\![\Delta;\Xi;\Gamma \vdash G_1 : \gamma]\!] = [\![\Delta;\Xi;\Gamma \vdash G_2 : \gamma]\!]$ then $G_1 =_{ctx} G_2$.

Proof Assume $\Delta;\Xi;\Gamma \vdash G_1 : \gamma$ and $\Delta;\Xi;\Gamma \vdash G_2 : \gamma$ and $\Sigma : \Delta$ and $[\![\Delta;\Xi;\Gamma \vdash G_1 : \gamma]\!] = [\![\Delta;\Xi;\Gamma \vdash G_2 : \gamma]\!]$. Let $s \in [\![\Sigma : \Delta]\!]$ and let $C[\,] : \Delta;\Xi;\Gamma \vdash \gamma \Rightarrow \Delta;;\vdash T\tau$ be a context.

Further assume $\Sigma,\ let\ x \Leftarrow C[G_1]\ in\ val\ x \downarrow$. By soundness this implies $[\![\Delta;;\vdash C[G_1] : T\tau]\!]\{\}[\![\Delta \vdash val\ x : (x:\tau)^{\top}]\!]S = \top$. By compositionallity of the denotational semantics $[\![\Delta;;\vdash C[G_1] : T\tau]\!]\{\} = [\![\Delta;;\vdash C[G_2] : T\tau]\!]\{\}$. So $[\![\Delta;;\vdash C[G_2] : T\tau]\!]\{\}[\![\Delta \vdash val\ x : (x:\tau)^{\top}]\!]S = \top$. By adequacy then $\Sigma,\ let\ x \Leftarrow C[G_2]\ in\ val\ x \downarrow$.

The other direction is similar. □

So simple equivalences can be proven only on the basis of the denotational semantics. We want however to be able to prove more programs contextual equivalent, that we can do in this way. In the remaining part of this report, we define relations between denotations, which makes it possible to prove more intricate equivalences. It is essential for the relations we define, that they are based on a denotational semantics which is sound and adequate.

4 Logical relations for contextual equivalence

We now aim at defining an invariant parameterized relation on $\mathbb{D}$ and $F(\mathbb{D}, \mathbb{D})$, which we can use to prove contextual equivalence for more programs. Since our denotations belong to a recursive domain we cannot prove the existence of our relation by induction, the existence of the relation requires a separate proof. In this presentation we have chosen to define two logical relations, where one extends the other. The last extension specifically handles explicit divergence, where two programs may diverge at different 'steps'(c.f. Introduction). The reason we have chosen to define two relations is that the definition of the full parameters may seem quite complicated, and by separating we hope to make the definitions easier to comprehend and use. Equivalence proof for programs which do not require the full definition of parameters are essentially the same in both relations. The full definition though sometimes requires considering extra cases, which will turn out to be all right from assumptions. So the equivalence proofs can be expressed simpler in the first relation for programs, which do not need the full definition.

For each relation we first define the parameters, which we index our relation by. Then in the following section we will define a parameterized logical relation, on which we will base our proof method for contextual equivalence. Then finally there will be sections with examples of equivalence proofs.

Informally explained the parameters give properties of two related states. Computations related under a parameter have corresponding termination behavior under the assumption that they are applied to states which hold the properties stated in the parameter and continuations which "behave well" for the parameter. The parameters express requirements on disjoint areas of store; there is a finite visible area and a finite number of hidden invariants. As explained in the introduction, we have tried to refine the definition of parameters from [13], so that it is possible via parameters, to express in more detail why we expect two programs to be equivalent. When we have defined such parameters that express our intuition and maybe also how they will later be restricted, then the certification of equivalence can be an almost automatic analysis. The equivalence proof may though seem rather lengthly as there are often many 'turns of the handle' unfolding definitions of denotations and of relatedness and other essentially straightforward steps. In the introduction we presented informally the use of parameters in some proofs of contextual equivalence. We will also define order relations on parameters. Informally a bigger parameter describes the states at a later time of computations when more visible locations have been allocated and/or more hidden invariants have been build up.

5 Parameters for a parameterized logical relation

Before we can define parameters we need some preliminary definitions. It is possible to define actions of permutations on all constructs in a straightforward way, so that relatedness is preserved under permutations. However, for proof of contextual equivalence, this is not necessary. FM-theory gives that termination properties are independent of permutations, when permutations are applied also to contexts and stores. So if we can prove that two programs have identical termination properties in all relevant contexts, then this will hold also under permutations.

First we define accessibility maps, almost as Benton and Leperchey [7] and the previous papers in this thesis. Accessibility maps ease statements and proofs of disjointness properties between different areas of states which are required to fulfill different invariants. Recall for the following definitions $F(\mathbb{D}, \mathbb{D})_S = (\mathbb{L}_\perp \multimap \mathbb{V})$.

Definition 9. *Accessibility map*
An accessibility map is a continuous function $A : F(\mathbb{D}, \mathbb{D})_S \to \mathcal{P}_{fin}(\mathbb{L})$ such that:

– $\forall S_1, S_2 \in F(\mathbb{D}, \mathbb{D})_S.\ (\forall l \in A(S_1).\ S_1 l = S_2 l) \Rightarrow A(S_1) = A(S_2)$

For any finite set $X \in \mathcal{P}(\mathbb{L})$ there is a constant accessibility map A_X, where it holds that $\forall S. A_X(S) = X$. A special accessibility map is $A_\emptyset$, with $\forall S.\ A_\emptyset(S) = \emptyset$.

When we want to express a local invariant required for a pair of states, and this invariant can be decided only on the basis of the areas of the states viewed by a pair of accessibility maps, then we use a finitary state relation. If we want to express a requirement only on one state, we can use a finitary state predicate.

Definition 10. *Finitary state relations.*

1. *The set of finitary state relations: Let A_1, A_2 be accessibility maps.*
   $P \subseteq F(\mathbb{D},\mathbb{D})_S{}^2$ *is an (A_1, A_2)-state-relation iff*
   $\quad\forall (S_1, S_2), (S_1', S_2') \in F(\mathbb{D},\mathbb{D})_S{}^2.$
   $\quad\quad\forall l_1 \in A_1(S_1).S_1 l_1 = S_1' l_1 \ \wedge\ \forall l_2 \in A_2(S_2).S_2 l_2 = S_2' l_2 \Rightarrow$
   $\quad\quad\quad(S_1, S_2) \in P \Leftrightarrow (S_1', S_2') \in P$
2. *The set of finitary state predicates: Let A be an accessibility map.*
   $P \subseteq F(\mathbb{D},\mathbb{D})_S$ *is an A-state-predicate iff*
   $\quad\forall S, S' \in F(\mathbb{D},\mathbb{D})_S.$
   $\quad\quad\forall l \in A(S).Sl = S'l \Rightarrow$
   $\quad\quad\quad S \in P \Leftrightarrow S' \in P$

A special finitary state relation is the $(A_\emptyset, A_\emptyset)$-state-relation $T = F(\mathbb{D},\mathbb{D})_S^2$. It holds for all (S_1, S_2) that (S_1, S_2) belong to T.

Whether a pair of states belong to a finitary state relation can be decided on the basis of only the parts of the states, which can be viewed by the (finite) accessibility maps. Given two states we can directly decide if they belong to the finitary state relation, it is independent of the "current step in the computations". In the same way it can be decided whether a state belongs to a finitary state predicate directly on the basis of the part of the state viewed by the accessibility map.

We will now define local parameters, which we will later use to express hidden invariants of two related states together with sets of visible locations in each side. The use will be defined precisely, when we define our parameterized relation. The intuition is that a local parameter has its own private area of the store in each side, together with finite sets of visible locations. The private areas are used for hidden invariants for testing conditions and for storing related values. An example:

A local parameter may require for a pair of states (S_1, S_2) that:

i) The location l_a is visible in left hand side, and the location l_b is visible in right hand side. And, at any time, for the local parameter to hold, it is required that the values stored in $S_1 l_a$ and $S_2 l_b$ are values of type τ related at the actual time (expressed by a parameter).
ii) The location l_x is local in left hand side, and the locations l_p, l_y, l_z are local in right hand side.
iii) The value stored in $S_1 l_p$ is an integer value.
iv) If $S_1 l_p = 0$ then the values stored in $(S_1 l_x, S_2 l_y)$ are values of type σ related at the actual time. If $S_1 l_p \neq 0$ then the values stored in $(S_1 l_x, S_2 l_z)$ are values of type σ related at the actual time.

When analyzing computations applied to continuations it is important to have a clear hold on the distribution of responsibilities of changes in hidden areas between the computation and the continuation. In simple situations properties are preserved by both the computations and the continuations separately. *Ordinary parameters* express such invariants. The example above and the parameters in [13] are ordinary parameters in this sense.

It is possible that computations make irreversible changes to states as we have seen in the 'Awkward example' in the introduction. To cover such situations we will define local extensions as part of a local parameter definition. We notice, that a requirement that relates fewer states does not in a simple way tell us how computation relations are. This is due to that related computations are expected to preserve invariants when they write but at the same time can rely on invariants when they read, and reading and writing drags in opposite directions. Consider an example, we have one location l and q_0 says that in the state l holds either 0 or 1, q_1 says that l must hold 1. Among computations which only read l, then computations which are requiring to read (0 or 1) will be satisfied with reading a 1, but computations which are requiring to read a 1 will not be satisfied with reading (0 or 1). Among computations which only writes l, then computations which write a 1 will also write (0 or 1), but computations which writes (0 or 1) will not preserve that l holds a 1. This is why we only require that an extension does not use more area of states and has the same or more visible locations. Additionally it is part of the definition of our relation, that relatedness requires that all future extensions will be preserved.

As explained in the introduction, we have here tried to refine the parameters, so that they can express more refined distribution of responsibilities between computations and continuations. Such refined parameters are called *specialized parameters*. These will be used in parts of proofs, where we have some knowledge of which continuation a computation will be applied to (for an example see section 7.2). We hope that we will also see the benefit of the specialized parameters, when we compare our proof method with proof methods based on bisimulations [28].

A specialized parameter may be thought of as a kind of conjunction of parameter-instantiations. We use the modified conjunction sign $\bar{\wedge}$ in the parameters; it is not a real logical conjunction, but meant to give intuitive meaning. Values and computations will be related under such a specialized $\bar{\wedge}$ parameter, if they preserve each parameter-instantiation separately. Continuations will be related under a $\bar{\wedge}$ parameter together with a choice of one conjunct-clause. Assuming that the continuations are applied to states which have the properties given by the chosen $\bar{\wedge}$ instantiation in hidden areas and where all stored values preserve each $\bar{\wedge}$ instantiation separately, then the related continuations will have similar termination properties. For instance we can use this, when we know that the initial part of the continuations modify the hidden areas in such a way that a specific $\bar{\wedge}$-clause choice is established, and this $\bar{\wedge}$-clause is necessary for relatedness of some later exported functions (c.f. example 7.2). So, the idea is that computations must preserve each $\bar{\wedge}$-clause of a specialized local parameter separately. Computations cannot change stores between $\bar{\wedge}$-clauses. Specialized local parameters for computations do not express, which of the $\bar{\wedge}$ requirements, they expect. For continuations it is different. A specialized local parameter for continuations expresses which $\bar{\wedge}$-clause the continuations expect the stores to fulfill. Continuations may then change stores such that another $\bar{\wedge}$-clause is fulfilled at the later step. Because the stored values are expected to preserve each $\bar{\wedge}$-clause, the continuations may in this way ensure that future computation will give equivalent termination.

We differentiate accordingly between vm-parameters for values and computations, k-parameters for continuations and s-parameters for states. In a sense then k-parameters are instantiations of vm-parameters and s-parameters are instantiations of k-parameters. And going in the opposite direction k-parameters are erasures of s-parameters, and vm-parameters are erasures of k-parameters. This will be explained in detail below. In the simple situations where each of computation and continuation preserves an invariant which ensures termination-approximation only one k-instantiation of a vm-parameter is possible. These are the ordinary parameters. For ordinary parameters a k-parameter can be identified with its erasure. We use specialized parameters in more complicated situations where computations must be applied to continuations, which so to say brings the combination back to a situation, where termination approximation is ensured. To cover these situations adds some extra complexity to the definition of local parameters.

In proofs of contextual equivalence, the starting point will be to show two computations or values related under a simple ordinary parameter that only expresses which visible locations are known, but does not have any hidden invariants. When the proof develops we will often want to show relatedness of subexpressions under more complicated parameters using knowledge about the initial steps of further execution. In the introduction we have shown some examples, where we will use specialized parameters in a proof of contextual equivalence.

We now first give some extra preliminary definitions. Notice how these definitions relate to the situations mentioned in the introduction and above.

Definition 11. *A match of finite store types Z.*

A match of finite store types is a finite set $Z = \{(l_{11}, l_{12}, \tau_1) \ldots (l_{n1}, l_{n2}, \tau_n)\}$ where each τ_i is closed and it holds that $\forall i \neq j \in 1 \ldots n.\ l_{i1} \neq l_{j1} \ \wedge \ l_{i2} \neq l_{j2}$. A match of finite store types Z defines two store types Z_1 and Z_2 given by first/second components of Z together with the type, and a bijection between their domains. It holds that $dom(Z_1)$ and $dom(Z_2)$ has the same finite cardinality.

A match of finite store types will be used to describe related visible areas of a pair of states. Because we use FM-domains it would be sufficient to use the same location names in both sides. We have chosen the above definition because it makes it easier to express proofs of contextual equivalence in situations where some initially hidden locations are later exported, and where these locations are not always the same but it depends on some other properties which locations will be exported.

In the definitions below we use 'disjunct signs' $\bar{\vee}$ and 'conjunct signs' $\bar{\wedge}$, these are not logical operators. The notation is meant to enhance intuition. Precise interpretation of the local parameters comes with the definition of the parameterized relation. Informally the idea is that functions and computations may change states from one $\bar{\vee}$-clause to another $\bar{\vee}$-clause within the same $\bar{\wedge}$-clause, but will preserve each $\bar{\wedge}$-clause separately. Continuations may change states between $\bar{\wedge}$-clauses as well as between $\bar{\vee}$-clauses. *Ordinary* parameters do not have any $\bar{\wedge}$'s and so continuations as well as computations can only change states between $\bar{\vee}$-clauses.

Here first is a brief description of parameters for values and computations:

A *parameter* consists of a number of local parameters, where relatedness of a pair of states will require that each local parameter holds its own area of store in each side.

A *local parameter* is set of local-parameter-components. Relatedness of a pair of computations will require that each component gives an invariant for states that is preserved if the computations are executed in states that hold the invariant. ($\bar{\wedge}$ is used between components).

A *local-parameter-component* is a tree of local-parameter-nodes, we can think of the root as the actual local-parameter-component, and the tree as defining possible irreversible changes of states (made by related computations) within the area owned by the local parameter.

A *local-parameter-node* is a set of local-parameter-parts, related computations may change states back and forth between the parts of a node. ($\bar{\vee}$ is used between parts).

A *local-parameter-part* defines some specific requirements for relatedness of a pair of states. The requirements can be decided on the basis of finite local areas of states, possibly together with knowing the current "time" of computation as expressed by a parameter.

We will also define two orderings $\triangleright$ and $\blacktriangleright$ between parameters. These orderings will be based on set inclusion (number of local parameters), local extensions $\succeq$ (corresponding to irreversible changes of states by computations) and forgetting of components.

5.1 Parameters, ordinary and specialized

We will now define general parameters which will be either ordinary or specialized. Ordinary parameters are used, when we relate computations where we don't know any specific properties of the continuations, they will be applied to. Specialized parameters are used, when we know the initial steps of the continuations, they are applied to. We use specialized parameters, if we know that states initially belong in one local-parameter-component and the initial steps of the continuations change states to another local-parameter-component.

Definition 12. *Local-parameter-part Q*
Given accessibility maps A_1 and A_2.
Then Q is an (A_1, A_2)-local-parameter-part with associated sets L_1^Q, L_2^Q if

> *$Q = (P, LL)$ and P is an (A_1, A_2)-state-relation and $LL =$ $\{ (l^{11}, l^{12}, \tau^1), \ldots, (l^{k1}, l^{k2}, \tau^k) \}$ is a finite* set *of location pairs and closed value types, $k \geq 0$.*
> *$L_1^Q = \pi_1(LL)$ and $L_2^Q = \pi_2(LL)$.*

The informal understanding of (P, LL) for a pair of states (S_1, S_2) is that $(S_1, S_2) \in P$ and LL hold related values.

Definition 13. *Local-parameter-node q*
Let I be a finite index set.
$q = (\{Q_i | i \in I\}, A_1^q, A_2^q, Z^q) = ([Q_1 \bar{\vee} \ldots \bar{\vee} Q_k], A_1^q, A_2^q, Z^q)$, is a local-parameter-node iff

- *A_1^q, A_2^q are accessibility maps.*
- *Each Q_i is an (A_1^q, A_2^q)-local-parameter-part*
- *Z^q is a match of finite store types.*
- $\bigcup_i L_1^{Q_i} \cap \pi_1(Z^q) = \emptyset \ \wedge \ \bigcup_i L_2^{Q_i} \cap \pi_2(Z^q) = \emptyset$

To the local-parameter-node $q = (\{Q_i | i \in I\}, A_1^q, A_2^q, Z^q)$ are associated the accessibility maps $\mathring{A}_1^q, \mathring{A}_2^q$ where
$\forall S.\ \mathring{A}_1^q(S) = A_1^{q_k}(S) \cup \pi_1 Z^q \cup \bigcup_{i \in I} L_1^{Q_i}$ *and*
$\mathring{A}_2^r(S) = A_2^{q_k}(S) \cup \pi_2 Z^q \cup \bigcup_{i \in I} L_2^{Q_i}$.
$\mathring{A}_1^r, \mathring{A}_2^r$ are the most inclusive accessibility maps associated with the local-parameter-node, encompass locations meant to be visible as well as hidden.

To the local-parameter-node q are associated fixed finite, sets of locations $L_1^q = \bigcup_i L_1^{Q_i}$ and $L_2^q = \bigcup_i L_2^{Q_i}$.

We order local-parameter-nodes such that a bigger parameter has the same or more visible locations and a bigger parameter always "owns" the same or smaller parts of the stores.

Definition 14. *$\geq$ order on local-parameter-nodes*
Let d, e be local-parameter-nodes, and $d = (\{Q_i^d | i \in I_d\}, A_1^d, A_2^d, Z^d)$, $e = (\{Q_i^e | i \in I_e\}, A_1^e, A_2^e, Z^e)$
$e \geq d$ iff

- *$Z^e \supseteq Z^d$ and*
- $\forall (S_1, S_2).\ \mathring{A}_1^d(S_1) \supseteq \mathring{A}_1^e(S_1) \wedge \mathring{A}_2^d(S_2) \supseteq \mathring{A}_2^e(S_2)$.

Definition 15. *Local-parameter-component $\hat{q}$*

A local-parameter-component $\hat{q}$ is a finite tree where each node q_i is a local-parameter-node and it holds that

- *$\forall q_1, q_2 \in q$. if q_1 is an ancestor of q_2 then $q_2 \geq q_1$*

To the local-parameter-component $\hat{q}$ with root-node q_0 is associated the match of finite store types $Z^q = Z^{q_0}$.

To the local-parameter-component $\hat{q}$ with root node q_0 are associated the accessibility maps $\mathring{A}_1^q = \mathring{A}_1^{q_0}$ and $\mathring{A}_2^q = \mathring{A}_2^{q_0}$.
$\mathring{A}_1^q, \mathring{A}_2^q$ are the most inclusive accessibility maps associated with the local-parameter-component (all locations possibly "owned" by subtrees are included c.f. order on tree-nodes).

To the local-parameter-component $\hat{q}$ are associated fixed finite, sets of locations W_1^q and W_2^q.
$W_1^q = \bigcup_{q_j \in \hat{q}} \left(\pi_1(Z^{q_j} \cup L_1^{q_j})\right)$ *and* $W_2^q = \bigcup_{q_j \in \hat{q}} \left(\pi_2(Z^{q_j} \cup L_2^{q_j})\right)$
where for $q_j = ((P_i, LL_i)|i \in I\}, A_1^{q_j}, A_2^{q_j}, Z^{q_j})$ *we let* $L_1^{q_j} = \cup_i L_1^{Q_i},\ L_2^{q_j} = \cup_i L_2^{Q_i}$.

For notational convenience when not otherwise indicated we let the root node of $\hat{q}$ be named q.

Intuitively the fixed sets W_1^q, W_2^q give locations (visible as well as hidden) that we know can be associated with q without knowing any specific state.

Intuitively we identify a local-parameter-component with its root. The rest of the tree is there to tell us how a local parameter may be locally extended.

Definition 16. *$\succeq$ relation on local-parameter-components.*
Let $\hat{q}'$ and $\hat{q}$ be local-parameter-components.

$\hat{q}' \succeq \hat{q}$ *iff* $\hat{q}' \in subtrees(\hat{q})$

We now define local parameters. A local parameter for values and computations is a finite set of local-parameter-components. A local parameter for continuations and a local parameter for states will "belong" to *one* of the components. The idea is that the values and computations preserve each component in states, but continuations may change states between components.

Definition 17. *Local parameters*

(vm) A local vm-parameter has the form $r = \{\hat{q}_1 \dots \hat{q}_n\} = (\hat{q}_1 \bar{\wedge} \dots \bar{\wedge} \hat{q}_n)$ *where*
- $\forall i \in 1 \dots n.\ \hat{q}_i$ *is a local parameter component*
- $n \geq 1$

(k) If $r = (\hat{q}_1 \bar{\wedge} \dots \bar{\wedge} \hat{q}_n)$ *is a local vm-parameter, then* $\forall j \in 1 \dots n.\ (r|\hat{q}_j)$ *is a local k-parameter.* $\underline{\hat{q}_j \textit{ is a choice of } \bar{\wedge}\textit{-clause}}$.

(s) If $(r|\hat{q})$ *is a local k-parameter, and* $root(\hat{q}) = (\{Q_i|i \in I\}, A_1^q, A_2^q, Z^q) = ([Q_1 \bar{\vee} \dots \bar{\vee} Q_k], A_1^q, A_2^q, Z^q)$, *then for each* $i \in I$ $(r|\hat{q}|Q_i)$ *is a local s-parameter.* $\underline{Q_i \textit{ is a choice of } \bar{\vee}\textit{-clause}}$.

A local parameter where $n = 1$ is ordinary, *so $(\hat{q})$ is an ordinary vm-parameter, $(\hat{q}|\hat{q})$ is an ordinary k-parameter and for all $i \in I$ $(\hat{q}|\hat{q}|Q_i)$ is an ordinary s-parameter.*

To the local parameter $r = \{\hat{q}_1 \dots \hat{q}_n\}$ is associated the match of finite store types

$Z^{\cap r} = \bigcap_{k \in 1..n} Z^{q_k}$.

$Z^{\cap r}$ gives the locations visible for every $\bar{\wedge}$-clause in the local parameter r

To the local parameter r are associated the fixed finite sets of locations $W_1^r = \bigcup_{i \in 1..n} W_1^{q_i}$ *and* $W_2^r = \bigcup_{i \in 1..n} W_2^{q_i}$.

To the local parameters $r = \{\hat{q}_1 \dots \hat{q}_n\}$ are associated the accessibility maps $\mathring{A}_1^r, \mathring{A}_2^r$ where

$\forall S.\ \mathring{A}_1^r(S) = \bigcup_{k \in 1..n} \mathring{A}_1^{q_k}(S)$ *and* $\mathring{A}_2^r(S) = \bigcup_{k \in 1..n} \mathring{A}_2^{q_k}(S)$

$\mathring{A}_1^r, \mathring{A}_2^r$ are the most inclusive accessibility maps associated with the local parameter, encompass all locations that may be associated with r, locations meant to be visible as well as hidden.

We sometimes use $\{(l_{11}, l_{12}, \tau_1) \dots (l_{n1}, l_{n2}, \tau_n)\}$ or $T_{\{(l_{11}, l_{12}, \tau_1) \dots (l_{n1}, l_{n2}, \tau_n)\}}$ as shorthand for the (ordinary) local parameter $\{((T, \emptyset_{LL}), A_\emptyset, A_\emptyset, \{(l_{11}, l_{12}, \tau_1) \dots (l_{n1}, l_{n2}, \tau_n)\})\}$. Such a parameter is used when we just add n visible locations in both sides. Visible locations will be expected to hold related values.

Definition 18. *$\succeq$ order on local-parameters*
Let $r' = \{\hat{q}'_1 \dots \hat{q}'_n\},\ r = \{\hat{q}_1 \dots \hat{q}_n\}$ be local vm-parameters (same n)

$r' \succeq r$ *iff* $\forall j \in 1 \dots n.\ \hat{q}'_j \succeq \hat{q}_j$. *($\hat{q}'_j$ is a subtree of $\hat{q}_j$)*

Let $rk' = (r'|\hat{q}'_i),\ rk = (r|\hat{q}_j)$ be local k-parameters with $r' = \{\hat{q}'_1 \dots \hat{q}'_n\},\ r = \{\hat{q}_1 \dots \hat{q}_n\}$

$rk' \succeq rk$ *iff* $r' \succeq r$ *and* $i = j$.

Definition 19. *Parameters*

(vm) *A vm-parameter p is a finite set $p = \{r_1 \dots r_n\}$ where*
- *Each $r_i \in p$ is a local vm-parameter*
- $\forall i \neq j \in 1 \dots n.\ W_1^{r_i} \cap W_1^{r_j} = \emptyset\ \wedge\ W_2^{r_i} \cap W_2^{r_j} = \emptyset$
- $\exists S_1, S_2 \neq \bot. \forall i, j \in 1 \dots n, i \neq j. \mathring{A}_1^{ri}(S_1) \cap \mathring{A}_1^{rj}(S_1) = \emptyset \wedge \mathring{A}_2^{ri}(S_2) \cap \mathring{A}_2^{rj}(S_2) = \emptyset$

(k) *If $p = \{r_1 \dots r_n\}$ is a vm-parameter and for each $j \in 1 \dots n$ $(r_j|\hat{q}_j)$ is a local k-parameter, then $(pk) = \{(r_1|\hat{q}_1) \dots (r_n|\hat{q}_n)\}$ is a k-parameter.*
A k-parameter is a vm-parameter together with choices of one $\bar{\wedge}$-clause for each local vm-parameter.

(s) *If $(pk) = \{(r_1|\hat{q}_1) \dots (r_n|\hat{q}_n)\}$ is a k-parameter and*
for each $j \in 1 \dots n$ $(r_j|\hat{q}_j|Q_j)$ is a local s-parameter
then $(pks) = \{(r_1|\hat{q}_1|Q_1) \dots (r_n|\hat{q}_n|Q_n)\}$ is an s-parameter.
An s-parameter is a k-parameter together with choices of one $\bar{\vee}$-clause for each local k-parameter.

(pk) is the k-erasure of (pks), and p is the vm-erasure of (pks) and of (pk).

(pks) is an s-instantiation of (pk) and of p. (pk) is a k-instantiation of p.

To the vm-parameter p are associated
- *Accessibility maps $\mathring{A}_1^p,\ \mathring{A}_2^p$ where $\forall S.\ \mathring{A}_1^p(S) = \bigcup \mathring{A}_1^{ri}(S)\ \wedge\ \mathring{A}_2^p(S) = \bigcup \mathring{A}_2^{ri}(S)$.*

To the k-parameter (pk) and to the s-parameter (pks) are associated
- *Accessibility maps $\mathcal{A}_1^{(pk)} = \mathcal{A}_1^{(pks)},\ \mathcal{A}_2^{(pk)} = \mathcal{A}_2^{(pks)}$, where*
  $\forall S.\ \mathcal{A}_1^{(pk)}(S) = \bigcup_j (A_1^{q_j}(S) \cup L_1^{q_j})\ \wedge\ \mathcal{A}_2^{(pk)}(S) = \bigcup_j (A_2^{q_j}(S) \cup L_2^{q_j}),\ j \in 1 \dots n.$

To the vm-parameter p is associated
- *a match of finite store types $Z^p = \biguplus_j Z^{\cap r_j}$, where $j \in 1 \dots n$.*

To the k-parameter (pk) and to the s-parameter (pks) is associated
- *A match of finite store types $Z^{(pk)} = Z^{(pks)} = \biguplus_j Z^{q_j}$, where $j \in 1 \dots n$.*

A parameter is ordinary iff all its local parameters are ordinary.
A parameter or a local parameter which is not ordinary is said to be specialized.
Any parameter and any local parameter is either ordinary or specialized.

All types that occur in parameters are closed.

$\mathfrak{p}^{vm}$ is the set of vm-parameters.	$\mathfrak{o}^{vm}$ is the set of ordinary vm-parameters.
$\mathfrak{p}^{k}$ is the set of k-parameters.	$\mathfrak{o}^{k}$ is the set of ordinary k-parameters.
$\mathfrak{p}^{s}$ is the set of s-parameters.	$\mathfrak{o}^{s}$ is the set of ordinary s-parameters.

$\mathfrak{s}^{vm}$ is the set of specialized vm-parameters.	$\mathfrak{p}^{vm} \setminus \mathfrak{o}^{vm} = \mathfrak{s}^{vm}$.
$\mathfrak{s}^{k}$ is the set of specialized k-parameters.	$\mathfrak{p}^{k} \setminus \mathfrak{o}^{k} = \mathfrak{s}^{k}$.
$\mathfrak{s}^{s}$ is the set of specialized s-parameters.	$\mathfrak{p}^{s} \setminus \mathfrak{o}^{s} = \mathfrak{s}^{s}$.

Definition 20. *Notation for* **sets** *of instantiations $p^{\mathbf{K}}$, $p^{\mathbf{S}}$*

– *If $p \in \mathfrak{p}^{vm}$ then $p^{\mathbf{K}}$ denotes the set of k-instantiations of p and $p^{\mathbf{S}}$ denotes the set of s-instantiations of p.*
– *If $p \in \mathfrak{p}^{k}$ then $p^{\mathbf{S}}$ denotes the set of s-instantiations of p.*

Definition 21. *Notation for erasures p^k, p^{vm}*

– *If $p \in \mathfrak{p}^{s}$ then p^k denotes the (unique) k-erasure of p and p^{vm} denotes the (unique) vm-erasure of p.*
– *If $p \in \mathfrak{p}^{k}$ then p^{vm} denotes the vm-erasure of p.*

Definition 22. *$\succeq$ relation on parameters.*
Let $p' = \{r'_1 \dots r'_n\}$ and $p = \{r_1 \dots r_n\}$ be vm-parameters.

$$p' \succeq p \textit{ iff } \forall i \in 1 \dots n.\ r'_i \succeq r_i$$

Let $pk' = \{(r'_1|\hat{q}'_1) \dots (r'_n|\hat{q}'_n)\}$ and $pk = \{(r_1|\hat{q}_1) \dots (r_n|\hat{q}_n)\}$ be k-parameters.

$$pk' \succeq pk \textit{ iff } \forall i \in 1 \dots n.\ (r'_i|\hat{q}'_i) \succeq (r_i|\hat{q}_i)$$

When we remove some but not all $\bar{\wedge}$-clauses from a local parameter then we get another local parameter. In this way we can derive parameters from parameters. As explained the idea is that related computations must preserve each $\bar{\wedge}$ instantiation. In our relation we will require that relatedness for values and computations is preserved when we go to a parameter derived by removal of $\bar{\wedge}$-clauses.

Definition 23. *Parameters derived from parameters*

– *Let $p \in \mathfrak{p}^{vm}$, $p = \{r_1, \dots, r_n\} = \{\{\hat{q}_{11} \dots \hat{q}_{1k_1}\}, \dots, \{\hat{q}_{n1} \dots \hat{q}_{nk_n}\}\}$.*
Let $\forall i \in 1 \dots n.\ r_i \supseteq r'_i \neq \emptyset$.
Then $p' = \{r'_1, \dots, r'_n\}$ is a parameter derived from p.
So p' is derived from p by removal of $\bar{\wedge}$-clauses.

$\mathfrak{sub}(p)$ *denotes the set of parameters derived from p.*

– *Let $p' = \{r'_1, \dots, r'_n\}$ be derived from $p = \{r_1, \dots, r_n\} \in \mathfrak{p}^{vm}$ such that $\forall i \in 1 \dots n$ either $\big(r'_i = r_i\big)$ or $\big(r'_i = \{\hat{q}\}$ where $\hat{q} \in r_i\big)$.*
Then p' is a parameter 1-derived from p.
So in p' some *local parameters in p are replaced by exactly one of their conjuncts.*

$\mathfrak{sub}^1(p)$ *denotes the set of parameters 1-derived from p.*

– *Let $p' = \{r'_1, \dots, r'_n\}$ be derived from $p = \{r_1, \dots, r_n\} \in \mathfrak{p}^{vm}$ such that $\forall i \in 1 \dots n.\ r'_i = \{\hat{q}\}$ where $\hat{q} \in r_i$.*
Then p' is an ordinary parameter, and p' is a parameter o-derived from p.
$\mathfrak{ord}(p)$ *denotes the set of ordinary parameters o-derived from p.*

A derived parameter is a parameter: Disjointness properties are preserved. It is possible that locations move from hidden to visible.

$$p \in \mathfrak{sub}(p),\ p \in \mathfrak{sub}^1(p),\ p \in \mathfrak{sub}^{o1}(p) \quad \text{and} \quad \mathfrak{ord}(p) \subseteq \mathfrak{sub}^{o1}(p) \subseteq \mathfrak{sub}^1(p) \subseteq \mathfrak{sub}(p)$$

We will see below, that if values are related under a parameter p, then they are also related under each parameter derived from p and hence also each parameter 1-derived, o1-derived or o-derived from p. In proofs of contextual equivalence this will sometimes be beneficial.

Definition 24. *Orders $\rhd$ and $\blacktriangleright$ on parameters*

$\rhd$: *The relation $\rhd$ on vm-parameters and k-parameters is defined as the reflexive transitive closure of the relations $\succeq$ and $\supseteq_{ord}$ (local extension and adding ordinary local parameters), where*

Let $p' \in \mathfrak{p}^{vm}$ and $p \in \mathfrak{p}^{vm}$ i.e. both are vm-parameters. Then
$p' \supseteq_{ord} p \overset{def}{\Longleftrightarrow} p = \{r_1, \ldots r_n\}$,$p' = \{r_1, \ldots, r_n, r_{n+1}, \ldots r_{n+m}\}$ *and* $\forall i \in n{+}1 \ldots n{+}m.\ r_i$ *is* ordinary. $(m \geq 0)$
Let pk' and pk both be k-parameters. Then
$pk' \supseteq_{ord} pk \overset{def}{\Longleftrightarrow} pk'^{vm} \supseteq_{ord} pk^{vm}$ *(their erasures are $\supseteq_{ord}$ ordered) and $pk' \supseteq pk$*

$\blacktriangleright$: *The relation $\blacktriangleright$ on vm-parameters is defined as the reflexive transitive closure of the relations $\succeq$, $\supseteq$ and $\geq_{\bar{\wedge}}$ (local extension, adding local parameters and removal of $\bar{\wedge}$-clauses), where*

Let $p' \in \mathfrak{p}^{vm}$ and $p \in \mathfrak{p}^{vm}$ i.e. both are vm-parameters. Then
$p' \geq_{\bar{\wedge}} p \overset{def}{\Longleftrightarrow} p' \in \mathfrak{sub}(p)$

Lemma 13. *$\rhd$ and $\blacktriangleright$ are order relations.*

Lemma 14.
Assume $r' \rhd r$, then either both r' and r are ordinary parameters or both r' and r are specialized parameters with the relation $\succeq$ on all the non-ordinary constituents.

Lemma 15.
For $p', p \in \mathfrak{p}^{vm}$ it holds that $p' \rhd p \Rightarrow p' \blacktriangleright p$ (but not the other direction).

The next lemma explains though, that it is often possible via a series of $\blacktriangleright$-extensions to come back to a $\rhd$ -extension. This can be useful in equivalence proofs.

Lemma 16.

– *If $p' \in \mathfrak{s}^{vm}$ and $p \in \mathfrak{o}^{vm}$ and $p' \blacktriangleright p$, then $\forall p'_o \in \mathfrak{ord}(p').\ p'_o \rhd p$.*
– *If $p' \blacktriangleright p \in \mathfrak{p}^{vm}$, $p' = \{r_1 \ldots r_n, r_{n+1} \ldots r_m\}$, $p = \{r_1 \ldots r_n\}$, and p'' is 1-derived from p' such that $\forall j \in (n{+}1) \ldots m.\ r_j$ is replaced by one of its conjuncts $\hat{q}_j$, then p'' is a parameter o-derived from p', and it holds that $p'' \rhd p$.*

Lemma 17. *If $p' \blacktriangleright p$ or $p' \rhd p$ then $Z^{p'} \supseteq Z^p$*

We are now ready to define our parameterized relation.

6 Parameterized relation

Recall the informal ideas of the parameters. The parameters express properties of related states.

Related values and computations will preserve each $\bar{\wedge}$-clause separately. This is expressed via reference to the set of parameters that $\blacktriangleright$-extend p. Related functions take values related in a $\blacktriangleright$-extended parameter to related computations. Related computations take continuations and states related under corresponding instantiations of a $\blacktriangleright$-extended parameter to termination approximation.

Continuations related under a specialized parameter 'knows' how to bring the situation back to normal for this specialization, it behaves well (give termination approximation) when applied to states and values related under an *only* $\triangleright$-extended parameter.

Since our denotations belong to a recursive domain, the existence of our parameterized logical relation again involves a separate proof. The proof requires that the relations are preserved under approximations. On the other hand we want the parameters to express invariants for hidden local areas of related states, and such properties of states may not be preserved under approximations. Therefore our relations are really given by 4-tuples, which we think of as two pairs: the 4-tuples have the form $(d_1', d_2' \mid\mid d_1, d_2)$, where $d_1' \sqsubseteq d_1$ and $d_2' \sqsubseteq d_2$ and d_1, d_2 are a kind of parameters. We can now let the approximation be carried out over the primed domain elements d_1', d_2', and preserve the invariant on the non-primed elements d_1, d_2. Correspondingly, relatedness of computations is stated as a two-sided termination approximation when applied to related continuations and states (probably under an extended parameter). Termination of application of an approximated computation m_1' to an approximated continuation k_1' and an approximated state S_1' implies termination in the other side of the non-approximated elements, $m_1'k_1'S_1' = \top \Longrightarrow m_2k_2S_2 = \top$ and also $m_2'k_2'S_2' = \top \Longrightarrow m_1k_1S_1 = \top$. Thus we combine in the definition of the relation domain-theoretical-approximation in one side and termination-approximation to the other side. With this separation of domain theoretical approximation from the local properties that the parameters express, we can prove that the relation exists. We can then extract a binary relation, defined via reference to the 4-ary relation, such that the binary relation implies contextual equivalence.

6.1 Relational Structure on $\mathbf{FMcpo}_\perp^4$

Definition 25. *Relational structure on FM-cpo$_\perp^4$*
Let $D = (D_V, D_K, D_M, D_S) \in$ FM-cpo$_\perp^4$.

The set of relations on D is defined as:

$\mathcal{R}(D) = \hat{R}_V \times \hat{R}_K \times \hat{R}_M \times \hat{R}_S$
where

$\hat{R}_V =$ *all subsets of* $D_V^2 \times D_V^2 \times \{\tau | \tau$ *closed value type*$\} \times \mathfrak{p}^{vm}$ *which include* $\{(\perp, \perp)\} \times D_V^2 \times \{\tau | \tau$ *closed value type*$\} \times \mathfrak{p}^{vm}$

$\hat{R}_K =$ *all subsets of* $D_K^2 \times D_K^2 \times \{(x:\tau)^\top | (x:\tau)^\top$ *continuation type*$\} \times \mathfrak{p}^k$ *which include* $\{(\perp, \perp)\} \times D_K^2 \times \{(x:\tau)^\top | (x:\tau)^\top$ *continuation type*$\} \times \mathfrak{p}^k$

$\hat{R}_M =$ *all subsets of* $D_M^2 \times D_M^2 \times \{T\tau | T\tau$ *closed computation type*$\} \times \mathfrak{p}^{vm}$ *which include* $\{(\perp, \perp)\} \times D_M^2 \times \{T\tau | T\tau$ *closed computation type*$\} \times \mathfrak{p}^{vm}$

$\hat{R}_S =$ *all subsets of* $D_S^2 \times D_S^2 \times \mathfrak{p}^s$ *which include* $\{(\perp, \perp)\} \times D_S^2 \times \mathfrak{p}^s$

For an element in one of the four projection relations we use the notation $(d'_1, d'_2 \parallel d_1, d_2, (type), p)$, *where we think of* $d_1, d_2, (type), p$ *as a kind of combined parameter.*

Definition 26. *Application of a pair of morphisms to a relation.*

For $D, E \in$ *FM-cpo*$^4_\perp$ *for* $f = (f_v, f_k, f_m, f_s) : D \multimap E$ *and* $g = (g_v, g_k, g_m, g_s) : D \multimap E$, *and relation* $R = (R_V, R_K, R_M, R_S) \in \mathcal{R}(D)$, *define*

$$\begin{array}{ll}(f_v, g_v)R_V & = \{(f_v v'_1, f_v v'_2 \parallel g_v v_1, g_v v_2, \tau_1, p_1) \mid (v'_1, v'_2 \parallel v_1, v_2, \tau_1, p_1) \in R_V\} \\ (f_k, g_k)R_K & = \{(f_k k'_1, f_k k'_2 \parallel g_k k_1, g_k k_2, (x:\tau_2)^\top, p_2) \mid (k'_1, k'_2 \parallel k_1, k_2, (x:\tau_2)^\top, p_2) \in R_K\} \\ (f_m, g_m)R_M & = \{(f_m m'_1, f_m m'_2 \parallel g_m m_1, g_m m_2, T\tau_3, p_3) \mid (m'_1, m'_2 \parallel m_1, m_2, T\tau_3, p_3) \in R_M\} \\ (f_S, g_S)R_S & = \{(f_s s'_1, f_s s'_2 \parallel g_s s_1, g_s s_2, p_4)) \mid (s'_1, s'_2 \parallel s_1, s_2, p_4) \in R_S\}\end{array}$$

and define $(f, g)R = ((f_v, g_v)R_V, (f_k, g_k)R_K, (f_m, g_m)R_M, (f_s, g_s)R_S)$

Definition 27. $(f, g) : R \subset S$
For $f = (f_v, f_k, f_m, f_s) : D \multimap E,\ g = (g_v, g_k, g_m, g_s) : D \cong E$ *with* $f \sqsubseteq g$
and relations $R \in \mathcal{R}(D),\ S \in \mathcal{R}(E)$ *define*

$$(f, g) : R \subset S \overset{def}{\iff} (f, g)R \subseteq S$$

where on the left hand side $\_ : \_ \subset \_$ *is being defined, and on the right hand side* $\subseteq$ *is set theoretical inclusion.*

Lemma 18.

- $\forall R \in \mathcal{R}(D).\ (id_D, id_D) : R \subset R$
- $\forall R \in \mathcal{R}(D).\forall S \in \mathcal{R}(D').\forall g : D \cong D'\ (\perp, g) : R \subset S$
- $(f, g) : R \subset S$ *and* $(f', g') : S \subset T \Rightarrow (f' \circ f, g' \circ g) : R \subset T$
- $(id_D, id_D) : R \subset R'$ *and* $(id_D, id_D) : R' \subset R \Rightarrow R = R'$.

Definition 28. *Downwards closed relation*
A relation $(R_V, R_K, R_M, R_S) \in \mathcal{R}(D)$ *is* downwards closed *if,*

for each $j \in \{V, K, M, S\}$.
$$d''_1 \sqsubseteq d'_1\ \wedge\ d''_2 \sqsubseteq d'_2\ \wedge\ (d'_1, d'_2 \parallel d_1, d_2, (type), p) \in R_j \Rightarrow (d''_1, d''_2 \parallel d_1, d_2, (type), p) \in R_j$$

Definition 29. *Admissible relation.*
A relation $R = (R_V, R_K, R_M, R_S) \in \mathcal{R}(D)$ *is admissible if*

$\forall S \in \mathcal{R}(D)$ *the set* $\{f \mid (f, id_D) : S \subset R\} \subseteq (E \multimap D)$ *contains* $\perp$ *and is closed under least upper bounds of countable finitely supported chains.* $\mathcal{R}_{adm}(D)$ *denotes the set of admissible relations on* D.

Definition 30. *Parameter-weakened relation (Weakening)*
A relation $(R_V, R_K, R_M, R_S) \in \mathcal{R}(D)$ *is* parameter-weakened *if,*
$\forall p_1, p_0 \in \mathfrak{p}^{vm},\ \forall (pk_1), (pk_0) \in \mathfrak{p}^k$ *it holds that*

$$\begin{array}{lll} - \quad p_1 \blacktriangleright p_0 & \wedge\ (v'_1, v'_2 \parallel v_1, v_2, \tau, p_0) \in R_V & \Rightarrow (v'_1, v'_2 \parallel v_1, v_2, \tau, p_1) \in R_V \\ - (pk_1) \rhd (pk_0) & \wedge\ (k'_1, k'_2 \parallel k_1, k_2, (x:\tau)^\top, (pk_0)) \in R_K & \Rightarrow (k'_1, k'_2 \parallel k_1, k_2, (x:\tau)^\top, (pk_1)) \in R_K \\ - \quad p_1 \blacktriangleright p_0 & \wedge\ (m'_1, m'_2 \parallel m_1, m_2, T\tau, p_0) \in R_M & \Rightarrow (m'_1, m'_2 \parallel m_1, m_2, T\tau, p_1) \in R_M \end{array}$$

These properties may co-exist.

Definition 31. *adm*$^+$*relation*
A relation $R \in \mathcal{R}(D)$ *is an* adm$^+$relation *if it is admissible, downwards closed and parameter-weakened.*
We let $\mathcal{R}_{adm^+}(D)$ *denote the set of adm*$^+$*relations over* D

6.2 Lifting F to an adm$^+$ action on relations

We aim to show that there exists a relational lifting of the functor F s.t.
$\forall R^- \in \mathcal{R}(\mathbb{D}), R^+ \in \mathcal{R}(\mathbb{D}). F(R^-, R^+) \in \mathcal{R}(F(\mathbb{D}, \mathbb{D}))$ and an *adm*$^+$ relation
$\nabla = (\nabla_V, \nabla_K, \nabla_M, \nabla_S) \in \mathcal{R}_{adm^+}(\mathbb{D})$ satisfying the equations in definition 32 and
$(i, i) : F(\nabla, \nabla) \subset \nabla \ \wedge\ (i^{-1}, i^{-1}) : \nabla \subset F(\nabla, \nabla)$.

Definition 32. *adm*$^+$*action of F on relations.*
Let $R^- \in \mathcal{R}(\mathbb{D}), R^+ \in \mathcal{R}(\mathbb{D})$

Define $F(R^-, R^+) \in \mathcal{R}(F(\mathbb{D}, \mathbb{D}))$,
$F(R^-, R^+) = (F(R^-, R^+)_V, F(R^-, R^+)_K, F(R^-, R^+)_M, F(R^-, R^+)_S)$ *where*

$$
\begin{aligned}
F(R^-, R^+)_V = &\{(\bot,\ \bot \parallel v_1,\ v_2,\ \tau,\ p) \mid p \in \mathfrak{p}^{vm} \textit{ the set of all vm-parameters } \} \cup \\
&\{(v_1',\ v_2' \parallel v_1,\ v_2,\ \tau,\ p) \mid p \in \mathfrak{p}^{vm} \wedge \\
&\quad v_1' \sqsubseteq v_1 \neq \bot \ \wedge\ v_2' \sqsubseteq v_2 \neq \bot \wedge \\
&\quad (v_1',\ v_2' \parallel v_1,\ v_2, \tau,\ p) \in \tilde{F}(R^-, R^+)_V \ \} \\
&\textit{where}
\end{aligned}
$$

$$
\begin{aligned}
\tilde{F}(R^-, R^+)_V = &\{(v_1',\ v_2' \parallel in_{\mathbf{1}}\lfloor * \rfloor,\ in_{\mathbf{1}}\lfloor * \rfloor,\ unit,\ p) \ \} \cup \\
&\{(v_1',\ v_2' \parallel in_{\mathbb{Z}}\lfloor n \rfloor,\ in_{\mathbb{Z}}\lfloor n \rfloor,\ int,\ p) \mid n \in \mathbb{Z} \ \} \cup \\
&\{(v_1',\ v_2' \parallel in_{\mathbb{L}}\lfloor l_1 \rfloor,\ in_{\mathbb{L}}\lfloor l_2 \rfloor,\ \tau\ ref,\ p) \mid (l_1, l_2, \tau) \in Z^p \ \ \} \cup \\
\\
&\{(v_1',\ v_2' \parallel in_{\oplus} in_i d_1,\ in_{\oplus} in_i d_2,\ \tau_1 + \tau_2,\ p) \mid d_1, d_2 \in \mathbb{V}_{\downarrow} \ \wedge\ \exists d_1', d_2' \in \mathbb{V}. \\
&\quad ((v_1' = \bot \wedge d_1' = \bot) \vee v_1' = in_{\oplus} in_i d_1' \neq \bot) \wedge ((v_2' = \bot \wedge d_2' = \bot) \vee v_2' = in_{\oplus} in_i d_2' \neq \bot) \wedge \\
&\quad (d_1',\ d_2' \parallel d_1,\ d_2,\ \tau_i,\ p) \in R_V^+,\ i \in \{1, 2\} \ \} \cup \\
\\
&\{(v_1',\ v_2' \parallel in_{\otimes}(d_{1a}, d_{1b}),\ in_{\otimes}(d_{2a}, d_{2b}),\ \tau_a \times \tau_b,\ p) \mid d_{1a}, d_{1b}, d_{2a}, d_{2b} \in \mathbb{V}_{\downarrow} \wedge \\
&\quad \exists d_{1a}', d_{1b}', d_{2a}', d_{2b}' \in \mathbb{V}. \\
&\quad (d_{1a}',\ d_{2a}' \parallel d_{1a},\ d_{2a},\ \tau_a,\ p) \in R_V^+ \ \wedge\ (d_{1b}',\ d_{2b}' \parallel d_{1b},\ d_{2b},\ \tau_b,\ p) \in R_V^+ \ \wedge \\
&\quad ((v_1' = \bot \wedge (d_{1a}' = \bot \vee d_{1b}' = \bot)) \ \vee\ (v_1' = in_{\otimes}(d_{1a}', d_{1b}') \neq \bot)) \ \wedge \\
&\quad ((v_2' = \bot \wedge (d_{2a}' = \bot \vee d_{2b}' = \bot)) \ \vee\ (v_2' = in_{\otimes}(d_{2a}', d_{2b}') \neq \bot))\} \cup \\
\\
&\{(v_1',\ v_2' \parallel in_{\mu} d_1,\ in_{\mu} d_2,\ \mu\alpha.\tau,\ p) \mid d_1, d_2 \in \mathbb{V}_{\downarrow} \ \wedge\ \exists d_1', d_2' \in \mathbb{V}. \\
&\quad ((v_1' = \bot \wedge d_1' = \bot) \vee v_1' = in_{\mu} d_1' \neq \bot) \wedge ((v_2' = \bot \wedge d_2' = \bot) \vee v_2' = in_{\mu} d_2' \neq \bot) \wedge \\
&\quad (d_1',\ d_2' \parallel d_1,\ d_2,\ \tau[\mu\alpha.\tau/\alpha],\ p) \in R_V^+ \ \} \cup \\
\\
&\{(v_1',\ v_2' \parallel in_{\forall}\lfloor d_1 \rfloor,\ in_{\forall}\lfloor d_2 \rfloor,\ \forall\alpha.T\tau,\ p) \mid d_1, d_2 \in \mathbb{M} \ \wedge\ \exists d_1', d_2' \in \mathbb{M}. \\
&\quad ((v_1' = \bot \wedge d_1' = \bot) \ \vee\ (v_1' = in_{\forall}\lfloor d_1' \rfloor \neq \bot \ \wedge\ d_1' \sqsupseteq \bot)) \wedge \\
&\quad ((v_2' = \bot \wedge d_2' = \bot) \ \vee\ (v_2' = in_{\forall}\lfloor d_2' \rfloor \neq \bot \ \wedge\ d_2' \sqsupseteq \bot)) \wedge \\
&\quad \forall \sigma \textit{ with } \_ \vdash \sigma : type.\ (d_1',\ d_2' \parallel d_1,\ d_2,\ T\tau[\sigma/\alpha],\ p) \in R_M^+ \ \} \cup \\
\\
&\{(v_1',\ v_2' \parallel in_{\multimap}\lfloor d_1 \rfloor,\ in_{\multimap}\lfloor d_2 \rfloor,\ \tau \to \mathrm{T}\tau',\ p) \mid d_1, d_2 \in (\mathbb{V} \multimap \mathbb{M}) \ \wedge\ \exists d_1', d_2' \in (\mathbb{V} \multimap \mathbb{M}). \\
&\quad ((v_1' = \bot \wedge d_1' = \bot) \ \vee\ (v_1' = in_{\multimap}\lfloor d_1' \rfloor \neq \bot \ \wedge\ d_1' \sqsupseteq \bot)) \wedge \\
&\quad ((v_2' = \bot \wedge d_2' = \bot) \ \vee\ (v_2' = in_{\multimap}\lfloor d_2' \rfloor \neq \bot \ \wedge\ d_2' \sqsupseteq \bot)) \wedge \\
&\quad \forall p' \blacktriangleright p, \forall (w_1',\ w_2' \parallel w_1,\ w_2, \tau,\ p') \in R_V^-.\ (\ d_1' w_1',\ d_2' w_2' \parallel d_1 w_1,\ d_1 w_2,\ T\tau',\ p') \in R_M^+ \ \}
\end{aligned}
$$

Recall $\blacktriangleright$ on vm-parameters is defined as the reflexive transitive closure of the relations $\succeq$, $\supseteq$ and $\geq_{\bar{\wedge}}$, (which are local extension, superset and removal of $\bar{\wedge}$-clauses).

$F(R^-, R^+)_K = \{(k'_1,\ k'_2 \parallel k_1,\ k_2,\ (x:\tau)^\top,\ (pk)) \mid (pk) \in \mathfrak{p}^k \wedge$
$k'_1 \sqsubseteq k_1\ \wedge\ k'_2 \sqsubseteq k_2\ \wedge\ \ \forall (pk') \rhd (pk).$
(that is (pk) extended only with ordinary *local k-parameters or local extensions)*

$\forall (pks') \in (pk')^{\mathbf{S}}$ *(the set of s-instantiations of p' i.e. choices of $\bar\vee$-clauses),*
$\forall (s'_1,\ s'_2 \parallel s_1,\ s_2,\ (pks')) \in R^-_S.\ \forall (v'_1,\ v'_2 \parallel v_1,\ v_2, \tau,\ (pk')^{vm}) \in R^-_V.$
$(k'_1 s'_1 v'_1 = \top \Rightarrow k_2 s_2 v_2 = \top) \wedge$
$(k'_2 s'_2 v'_2 = \top \Rightarrow k_1 s_1 v_1 = \top)\ \}$

$F(R^-, R^+)_M = \{(m'_1,\ m'_2 \parallel m_1,\ m_2,\ T\tau,\ p) \mid p \in \mathfrak{p}^{vm} \wedge$
$m'_1 \sqsubseteq m_1\ \wedge\ m'_2 \sqsubseteq m_2 \wedge$
$\forall p' \blacktriangleright p. \forall (pk') \in p'^{\mathbf{K}}$ *(the set of k-instantiations of p' i.e. choices of a $\bar\wedge$-clause in each local parameter).*

$\forall (pks') \in (pk')^{\mathbf{S}}$ *(the set of s-instantiations of (pk') i.e. choices of $\bar\vee$-clause from each chosen $\bar\wedge$-clause in (pk')),*
$\forall (k'_1,\ k'_2 \parallel k_1,\ k_2,\ (x:\tau)^\top,\ (pk')) \in R^-_K.\ \forall (S'_1,\ S'_2 \parallel S_1,\ S_2,\ (pks')) \in R^-_S\ .$
$(m'_1 k'_1 S'_1 = \top \Rightarrow m_2 k_2 S_2 = \top) \wedge$
$(m'_2 k'_2 S'_2 = \top \Rightarrow m_1 k_1 S_1 = \top)\ \}$

$F(R^-, R^+)_S = \{(\bot,\ \bot \parallel S_1,\ S_2,\ (pks)) \mid (pks) \in \mathfrak{p}^s$ *the set of all s-parameters* $\} \cup$

$\{(S'_1,\ S'_2 \parallel\ \ S_1,\ S_2,\ (pks) \mid (pks) = \{(r_1|\hat q_1|Q_1), \ldots, (r_n|\hat q_n|Q_n)\} \in \mathfrak{p}^s$
($\hat q_i$ is a choice of $\bar\wedge$-clause in r_i, and Q_i is a choice of $\bar\vee$-clause in q_i) $\wedge$

$S'_1 \sqsubseteq S_1 \neq \bot\ \ \wedge\ S'_2 \sqsubseteq S_2 \neq \bot \wedge$

$\mathcal{A}_1^{(pks)}(S_1) \cap \pi_1(Z^{(pks)}) = \emptyset\ \wedge\ \mathcal{A}_2^{(pks)}(S_2) \cap \pi_2(Z^{(pks)}) = \emptyset$
(in each side are the visible locations in $Z^{(pks)}$ disjoint from the known hidden locations) $\wedge$

$\forall i \neq j.\ \mathring{A}_1^{ri}(S_1) \cap \mathring{A}_1^{rj}(S_1) = \emptyset\ \wedge\ \mathring{A}_2^{ri}(S_2) \cap \mathring{A}_2^{rj}(S_2) = \emptyset$
($\mathring{A}_1^{ri}, \mathring{A}_2^{ri}$ accessibility maps are the most inclusive for the local parameter r_i.
In each side is every location, visible, hidden or reserved, belonging to a local parameter outside the areas owned by any different local parameter) $\wedge$

$\forall (l_1, l_2, \tau) \in Z^{(pks)}.(S'_1(l_1),\ S'_2(l_2) \parallel S_1(l_1),\ S_2(l_2),\ \tau,\ (pks)^{vm}) \in R^+_V$
(All visible locations hold related values) $\wedge$

if $Q_i = (P_i, LL_i)$ then $(S_1, S_2) \in P_i$ $\wedge$
$\forall (l_1, l_2, \tau) \in LL_i.\ (S'_1(l_1),\ S'_2(l_2) \parallel S_1(l_1),\ S_2(l_2),\ \tau,\ (pks)^{vm}) \in R^+_V$
(The states belong to all simple state relations in the chosen $\bar\vee$-clauses.
Corresponding LL location-sets hold related values) $\}$

It follows from the definition that

- For all $(v'_1,\ v'_2 \parallel v_1,\ v_2,\ \tau,\ p) \in F(R^-, R^+)_V$ it holds that $(v'_1 = v'_2 = \bot)$ or $(v_1 \neq \bot \wedge v_2 \neq \bot)$.
- For all $(s'_1,\ s'_2 \parallel s_1,\ s_2,\ p) \in F(R^-, R^+)_S$ it holds that $(s'_1 = s'_2 = \bot)$ or $(s_1 \neq \bot \wedge s_2 \neq \bot)$.
- $\forall R^-, R^+. \forall (x:\tau)^\top. \forall p \in \mathfrak{p}^k. \forall k_1, k_2 \in F(\mathbb{D}, \mathbb{D})_K.\ (\bot, \bot \parallel k_1, k_2, (x:\tau)^\top, p) \in F(R^-, R^+)_K$
- $\forall R^-, R^+. \forall T\tau. \forall p \in \mathfrak{p}^{vm}. \forall m_1, m_2 \in F(\mathbb{D}, \mathbb{D})_M.\ (\bot, \bot \parallel m_1, m_2, T\tau, p) \in F(R^-, R^+)_M$

First we will show that when F acts on adm$^+$relations, then we get an adm$^+$relation. So we need to show that the action of F preserves downwards closure, admissibility and parameter weakening.

Lemma 19. *The action of F preserves downwards closure.*

For all $R^+, R^- \in \mathcal{R}(\mathbb{D})$.
If R^+ is downwards closed, then $F(R^-, R^+)$ is downwards closed.

Lemma 20. *The action of F on adm$^+$relations preserves admissibility.*

For all $R^+, R^- \in \mathcal{R}(\mathbb{D})$. If R^+ is adm$^+$, then $F(R^-, R^+)$ is admissible.

Lemma 21. *The action of F preserves parameter weakening.*

For all $R^+, R^- \in \mathcal{R}(\mathbb{D})$.
If R^+ is parameter weakened, then $F(R^-, R^+)$ is parameter weakened.

Proofs of the above three lemmas are in the appendix.

As the action of F on relations on $\mathbb{D}$ preserves downwards closure, admissibility and parameter weakening it follows that

Corollary 4. *The action of F preserves adm$^+$.*

For all $R^+, R^- \in \mathcal{R}(\mathbb{D})$.
If $R^+ \in \mathcal{R}_{adm^+}(\mathbb{D})$, then $F(R^-, R^+) \in \mathcal{R}_{adm^+}(\mathbb{D})$.

Lemma 22. *The action of F on functions $\mathbb{D} \multimap \mathbb{D}$ preserves the relation $(\_\,, id) : \_ \subset \_$.*
$\forall R^+, S^+, R^-, S^- \in \mathcal{R}(\mathbb{D}).\ \forall f^+, f^- : \mathbb{D} \multimap \mathbb{D}.$

If $(f^-, id_{\mathbb{D}}) : S^- \subset R^-$ and $(f^+, id_{\mathbb{D}}) : R^+ \subset S^+$ then
$(F(f^-, f^+), F(id_{\mathbb{D}}, id_{\mathbb{D}})) = (F(f^-, f^+), id_{F(\mathbb{D},\mathbb{D})}) : F(R^-, R^+) \subset F(S^-, S^+)$.

Corollary 5. *Monotonicity.*
$\forall R^+, S^+, R^-, S^- \in \mathcal{R}(\mathbb{D})$. *If $S^- \subset R^-$ and $R^+ \subset S^+$ then* $F(R^-, R^+) \subset F(S^-, S^+)$.

The corollary follows from the lemma with $f^+ = f^- = id_{\mathbb{D}}$.

Proof of lemma 22
We show here the proof for $F(\mathbb{D}, \mathbb{D})_S$, where we see that we benefit from expressing our relations as 'four-tuples'. The rest of the proof is in the appendix.

Let $R^+, S^+, R^-, S^- \in \mathcal{R}(\mathbb{D})$, and
let $f^+, f^- : \mathbb{D} \multimap \mathbb{D}$. $f^- = (f^-_v, f^-_k, f^-_m, f^-_s)$, $f^+ = (f^+_v, f^+_k, f^+_m, f^+_s)$.

Assume $(f^-, id_{\mathbb{D}}) : S^- \subset R^-$ and $(f^+, id_{\mathbb{D}}) : R^+ \subset S^+$.
We aim to show $(F(f^-, f^+), F(id_{\mathbb{D}}, id_{\mathbb{D}})) : F(R^-, R^+) \subset F(S^-, S^+)$.

$(f^-, id_{\mathbb{D}}) : S^- \subset R^- \ \wedge\ (f^+, id_{\mathbb{D}}) : R^+ \subset S^+$ implies $f^- \sqsubseteq id_{\mathbb{D}} \ \wedge\ f^+ \sqsubseteq id_{\mathbb{D}}$.
By the functorial properties of F we then have $F(f^-, f^+) \sqsubseteq F(id_{\mathbb{D}}, id_{\mathbb{D}}) = id_{F(\mathbb{D},\mathbb{D})}$.

Let $F(f^-, f^+) = h = (h_v, h_k, h_m, h_s)$ and let for now $id_F = id_{F(\mathbb{D},\mathbb{D})}$.
We need to show that $s \in F(R^-, R^+) \Rightarrow ((h, id_F)s) \in F(S^-, S^+)$.

- $F(\mathbb{D},\mathbb{D})_S$
  Assume $(s_1', s_2' \parallel s_1, s_2, (pks)) \in F(R^-, R^+)_S$, $(pks) = \{(r_1|q_1|Q_1), \ldots, (r_n|q_n|Q_n)\} \in \mathfrak{p}^s$. We aim to show $(h_s s_1', h_s s_2' \parallel id_s s_1, id_s s_2, (pks)) \in F(S^-, S^+)$.

  If $s_1' = \bot = s_2'$ then since h_s is strict so $(h_s s_1', h_s s_2' \parallel id_s s_1, id_s s_2, p) = (\bot, \bot \parallel s_1, s_2, (pks)) \in F(S^-, S^+)_S$.
  Else $(h_s s_1', h_s s_2' \parallel id_s s_1, id_s s_2, (pks)) = (\lambda l. f_v^+(s_1' l), \lambda l. f_v^+(s_2' l) \parallel s_1, s_2, (pks))$ and
  $(s_1', s_2' \parallel s_1, s_2, p) \in F(R^-, R^+)_S$ so
  (a) $s_1' \sqsubseteq s_1 \neq \bot \ \wedge \ s_2' \sqsubseteq s_2 \neq \bot$
  (b) $\mathcal{A}_1^{(pks)}(s_1) \cap \pi_1(Z^{(pks)}) = \emptyset \ \wedge \ \mathcal{A}_2^{(pks)}(s_2) \cap \pi_2(Z^{(pks)}) = \emptyset \ \wedge$
  $\forall i \neq j.\ \mathring{A}_1^{ri}(s_1) \cap \mathring{A}_1^{rj}(s_1) = \emptyset \ \wedge \ \mathring{A}_2^{ri}(s_2) \cap \mathring{A}_2^{rj}(s_2) = \emptyset$
  (c) $\forall (l_1, l_2, \tau) \in Z^{(pks)}.(s_1'(l_1),\ s_2'(l_2) \parallel s_1(l_1),\ s_2(l_2),\ \tau,\ (pks)^{vm}) \in R_V^+$
  (d) $\forall i \in 1..n.$ if $Q_i = (P_i, LL_i)$ then $(s_1, s_2) \in P_i$
  (e) if $Q_i = (P_i, LL_i)$ then $\forall (l_1, l_2, \tau) \in LL_i.\ (s_1'(l_1),\ s_2'(l_2) \parallel s_1(l_1),\ s_2(l_2),\ \tau,\ (pks)^{vm}) \in R_V^+$

  and we need to show that if $h_s(s_1') \neq \bot \ \vee \ h_s(s_2') \neq \bot$ then
  1. $h_s(s_1') \sqsubseteq s_1 \neq \bot \ \wedge \ h_s(s_2') \sqsubseteq s_2 \neq \bot$
  2. $\mathcal{A}_1^{(pks)}(s_1) \cap \pi_1(Z^{(pks)}) = \emptyset \ \wedge \ \mathcal{A}_2^{(pks)}(s_2) \cap \pi_2(Z^{(pks)}) = \emptyset \ \wedge$
  $\forall i \neq j.\ \mathring{A}_1^{ri}(s_1) \cap \mathring{A}_1^{rj}(s_1) = \emptyset \ \wedge \ \mathring{A}_2^{ri}(s_2) \cap \mathring{A}_2^{rj}(s_2) = \emptyset$
  3. $\forall (l_1, l_2, \tau) \in Z^{(pks)}.((h_s s_1')(l_1),\ (h_s s_2')(l_2) \parallel s_1(l_1),\ s_2(l_2),\ \tau,\ (pks)^{vm}) =$
  $(f_v^+(s_1' l_1), f_v^+(s_2' l_2) \parallel s_1(l_1), s_2(l_2),\ \tau, (pks)^{vm}) \in S_V^+$
  4. $\forall i \in 1..n.$ if $Q_i = (P_i, LL_i)$ then $(s_1, s_2) \in P_i$
  5. if $Q_i = (P_i, LL_i)$ then $\forall (l_1, l_2, \tau) \in LL_i.\ ((h_s s_1')(l_1),\ (h_s s_2')(l_2) \parallel s_1(l_1),\ s_2(l_2),\ \tau,\ (pks)^{vm}) =$
  $(f_v^+(s_1' l_1), f_v^+(s_2' l_2) \parallel s_1(l_1), s_2(l_2),\ \tau, (pks)^{vm}) \in S_V^+$
  1. Follows from (a) together with $h \sqsubseteq id$.
  2. Follows from (b) directly.
  3. Follows from (c) together with the assumption $(f^+, id_{\mathbb{D}}) : R^+ \subset S^+$.
  4. Follows from (d) directly.
  5. Follows from (e) together with $(f^+, id_{\mathbb{D}}) : R^+ \subset S^+$.

  We conclude that $(h_s s_1', h_s s_2' \parallel id_s s_1, id_s s_2, (pks)) \in F(S^-, S^+)_S$.

□

Theorem 3. *There exists an* adm$^+$ *relation* $\nabla = (\nabla_V, \nabla_K, \nabla_M, \nabla_S) \in \mathcal{R}_{adm^+}(\mathbb{D})$ *satisfying the equations in definition 32 and*
$(i, i) : F(\nabla, \nabla) \subset \nabla \ \wedge \ (i^{-1}, i^{-1}) : \nabla \subset F(\nabla, \nabla)$.

We now aim to prove theorem 3.
We have a partial order on the set of relations on a domain $D \in \{\mathbb{D}, F(\mathbb{D}, \mathbb{D})\}$ by

$$R \subset R' \overset{def}{\iff} (id_D, id_D) : R \subset R'.$$

This inherits as a partial order on the set $\mathcal{R}_{adm^+}(D)$.

Lemma 23. *1. If $\mathcal{S}$ is a set of relations on $D \in \{\mathbb{D}, F(\mathbb{D}, \mathbb{D})\}$ ($\mathcal{S} \subseteq \mathcal{R}(D)$), then there exist a relation $\bigcap \mathcal{S} \in \mathcal{R}(D)$ given by the set theoretical intersection, such that*
$\forall (g, j). \forall R.\ ((g, j) : R \subset \bigcap \mathcal{S}) \Leftrightarrow (\forall S \in \mathcal{S}.\ (g, j) : R \subset S)$.

2. This inherits to adm$^+$relations: If $\mathcal{S}$ is a set of adm$^+$relations on D ($\mathcal{S} \subseteq \mathcal{R}_{adm^+}(D)$), then there exist an adm$^+$relation $\bigcap \mathcal{S} \in \mathcal{R}_{adm^+}(D)$ such that
$\forall (g, j). \forall R.\ ((g, j) : R \subset \bigcap \mathcal{S}) \Leftrightarrow (\forall S \in \mathcal{S}.\ (g, j) : R \subset S)$.

Proof of lemma 23. Let $\mathcal{S}$ be a set of adm$^+$relations on $D \in$ FM-cpo$^4_\bot$ and let $\cap \mathcal{S}$ be the set theoretical intersection. We need to prove

– $\cap\mathcal{S} \in \mathcal{R}_{adm^+}(\mathbb{D})$:
$\cap\mathcal{S} \in \mathcal{R}(\mathbb{D})$ follows from the definition of the relational structure. Admissibility of $\cap\mathcal{S}$ follows from admissibility of each $S \in \mathcal{S}$. Downwards closure of $\cap\mathcal{S}$ follows from downwards closure of each $S \in \mathcal{S}$. Parameter weakening of $\cap\mathcal{S}$ follows from parameter weakening of each $S \in \mathcal{S}$.

– $\forall(g,j).\forall R.\ ((g,j) : R \subset \bigcap\mathcal{S}) \Leftrightarrow (\forall S \in \mathcal{S}.\ (g,j) : R \subset S)$:
$\Rightarrow$: Assume $(g,j) : R \subset \bigcap\mathcal{S}$. Then $\forall r \in R.(g,j)r \in \bigcap\mathcal{S})$. So $\forall S \in \mathcal{S}, \forall r \in R.(g,j)r \in S)$. Hence $\forall S \in \mathcal{S}.\ (g,j) : R \subset S)$.
$\Leftarrow$: Assume $\forall S \in \mathcal{S}.\ (g,j) : R \subset S$. Then $\forall S \in \mathcal{S}.\forall r \in R.\ (g,j)r \in S$. So $\forall r \in R.\ (g,j)r \in \bigcap\mathcal{S}$. Hence $(g,j) : R \subset \bigcap\mathcal{S}$.

□

Corollary 6.
For $D \in \{\mathbb{D}, F(\mathbb{D},\mathbb{D})\}$ it holds that the partially ordered set $(\mathcal{R}_{adm^+}(D)\ ,\ \subset)$ is a complete lattice.

Definition 33. *The relation $((f,j)^*S)$*
For $D, E \in$ FM-cpo$^4_\perp$, $D = (D_1, D_2, D_3, D_4)$, $j : D \cong E$,
$f : D \multimap E$, $f \sqsubseteq j$, $S \in \mathcal{R}(E)$

define $(f,j)^*S =$
$\{s = ((v'_1, v'_2 \parallel v_1, v_2, \tau_1, p_1), (k'_1, k'_2 \parallel k_1, k_2, (x:\tau_2)^\top, p_2), (m'_1, m'_2 \parallel m_1, m_2, T\tau_3, p_3), (s'_1, s'_2 \parallel s_1, s_2, p_4)) \mid$
$v'_1, v'_2, v_1, v_2 \in D_1 \ \wedge\ k'_1, k'_2, k_1, k_2 \in D_2 \ \wedge\ m'_1, m'_2, m_1, m_2 \in D_3 \ \wedge\ s'_1, s'_2, s_1, s_2 \in D_4 \ \wedge$
$(f,j)s \in S\ \}$

Lemma 24. *When $(f,j)^*S$ is defined as above, then*

1. *$(f,j)^*S \in \mathcal{R}(D)$.*
2. *If S is adm$^+$, then $(f,j)^*S$ is adm$^+$.*
3. *$\forall D', j', f', R$ with $j' : D' \cong D$, $f' : D' \multimap D$, $f' \sqsubseteq j', R \in \mathcal{R}(D')$.*
   $(f',j') : R \subset ((f,j)^*S) \Leftrightarrow (f \circ f', j \circ j') : R \subset S$

Proof

1. The domain parts are in the required domains, and for each of the four D_i, for any $(\perp, \perp \parallel d_1, d_2, (type), p)$ it holds that $(f_i, j_i)(\perp, \perp \parallel d_1, d_2, (type), p) = (\perp, \perp \parallel j_i d_1, j_i d_2, (type), p)$ $\in S$.
2. When S is adm$^+$ it is by definition admissible , downwards closed and parameter-weakened. And we need to show that then also $(f,j)^*S$ is admissible , downwards closed and parameter-weakened.

   Assume S is admissible. Assume a chain r^i in $(f,j)^*S \in \mathcal{R}(D)$. Then the least upper bound of the domain parts in r^i belong to D. Also $(f,j)(r^i)$ is a chain in S, and since S is admissible its least upper bound $\bigsqcup(f,j)(r^i)$ is in S. Then $\bigsqcup r^i \in (f,j)^*S$ by continuity of f, j.

   Assume S is parameter weakened. Assume an element $s = ((v'_1, v'_2 \parallel v_1, v_2, \tau_1, p_1), (k'_1, k'_2 \parallel k_1, k_2, (x:\tau_2)^\top, p_2), (m'_1, m'_2 \parallel m_1, m_2, T\tau_3, p_3), (s'_1, s'_2 \parallel s_1, s_2, p_4)) \in (f,j)^*S$, and assume $p'_1 \blacktriangleright p_1$, $p'_2 \triangleright p_2$, $p'_3 \blacktriangleright p_3$. We then have $(f,j)s = (f_v v'_1, f_v v'_2 \parallel j_v v_1, j_v v_2, \tau_1, p_1), (f_k k'_1, f_k k'_2 \parallel j_k k_1, j_k k_2, (x:\tau_2)^\top, p_2), (f_m m'_1, f_m m'_2 \parallel j_m m_1, j_m m_2, T\tau_3, p_3), (f_s s'_1, f_s s'_2 \parallel j_s s_1, j_s s_2, p_4)) \in S$. Since S is parameter weakened then $(f_v v'_1, f_v v'_2 \parallel j_v v_1, j_v v_2, \tau_1, p'_1), (f_k k'_1, f_k k'_2 \parallel j_k k_1, j_k k_2, (x:\tau_2)^\top, p'_2), (f_m m'_1, f_m m'_2 \parallel j_m m_1, j_m m_2, T\tau_3, p'_3), (f_s s'_1, f_s s'_2 \parallel j_s s_1, j_s s_2, p_4)) \in S$. And then $((v'_1, v'_2 \parallel v_1, v_2, \tau_1, p'_1), (k'_1, k'_2 \parallel k_1, k_2, (x:\tau_2)^\top, p'_2), (m'_1, m'_2 \parallel m_1, m_2, T\tau_3, p'_3), (s'_1, s'_2 \parallel s_1, s_2, p_4))$ $\in (f,j)^*S$. So $(f,j)^*S$ is parameter weakened.

Assume S is downwards closed. Assume an element $s = ((v'_1, v'_2 \parallel v_1, v_2, \tau_1, p_1), (k'_1, k'_2 \parallel k_1, k_2, (x : \tau_2)^\top, p_2), (m'_1, m'_2 \parallel m_1, m_2, T\tau_3, p_3), (s'_1, s'_2 \parallel s_1, s_2, p_4)) \in (f, j)^*S$, and assume $v''_1 \sqsubseteq v'_1, v''_2 \sqsubseteq v'_2, k''_1 \sqsubseteq k'_1, k''_2 \sqsubseteq k'_2, m''_1 \sqsubseteq m'_1, m''_2 \sqsubseteq m'_2, s''_1 \sqsubseteq s'_1, s''_2 \sqsubseteq s'_2$. We then have $(f, j)s = (f_v v'_1, f_v v'_2 \parallel j_v v_1, j_v v_2, \tau_1, p_1), (f_k k'_1, f_k k'_2 \parallel j_k k_1, j_k k_2,$
$(x : \tau_2)^\top, p_2), (f_m m'_1, f_m m'_2 \parallel j_m m_1, j_m m_2, T\tau_3, p_3), (f_s s'_1, f_s s'_2 \parallel j_s s_1, j_s s_2, p_4)) \in S$. Since S is downwards closed and f is monotone then $(f_v v''_1, f_v v''_2 \parallel j_v v_1, j_v v_2, \tau_1, p'_1), (f_k k''_1, f_k k''_2 \parallel j_k k_1, j_k k_2,$
$(x : \tau_2)^\top, p'_2), (f_m m''_1, f_m m''_2 \parallel j_m m_1, j_m m_2, T\tau_3, p'_3), (f_s s''_1, f_s s''_2 \parallel j_s s_1, j_s s_2, p_4)) \in S$. And then
$((v''_1, v''_2 \parallel v_1, v_2, \tau_1, p'_1), (k''_1, k''_2 \parallel k_1, k_2, (x : \tau_2)^\top, p'_2), (m''_1, m''_2 \parallel m_1, m_2, T\tau_3, p'_3), (s''_1, s''_2 \parallel s_1, s_2, p_4))$
$\in (f, j)^*S$. So $(f, j)^*S$ is downwards closed.

3. a) If j, j' are isomorphisms and compose then $(j \circ j')$ is an isomorphism. Since $f \sqsubseteq j,\ f' \sqsubseteq j'$, then also $f \circ f \sqsubseteq j \circ j'$.
b) $\Rightarrow$: Assume $(f', j') : R \subset ((f, j)^*S)$. This implies $s \in R \Rightarrow (f', j')(s) \in ((f, j)^*S)$. By the definition of $((f, j)^*S)$ then $(f, j)((f', j')(s)) = (f \circ f', j \circ j')(s) \in S$. Together with a) then $(f \circ f', j \circ j') : R \subset S$.
$\Leftarrow$: Assume $(f \circ f', j \circ j') : R \subset S$. This implies $s \in R \Rightarrow (f \circ f', j \circ j')(s) = (f, j)((f', j')(s)) \in S$. Since $f' : D' \multimap D$ and $j' : D' \cong D$ we have that the domain parts of $(f', j')(s)$ are in the required domains for $(f, j)^*S$ so $(f', j')(s) \in ((f, j)^*S)$ by the difinition of $((f, j)^*S)$. Hence $(f', j') : R \subseteq ((f, j)^*S)$.

$\square$

Lemma 25. *$(i, i) : F(R, R) \subset R$ and $(i^{-1}, i^{-1}) : R \subset F(R, R) \iff R = (i^{-1}, i^{-1})^*F(R, R)$*

Proof
$\Leftarrow$: Assume $R = (i^{-1}, i^{-1})^*F(R, R)$
i) $t \in R = (i^{-1}, i^{-1})^*F(R, R) \Rightarrow (i^{-1}, i^{-1})t \in F(R, R)$ by the definition of $(i^{-1}, i^{-1})^*F(R, R)$. But this implies $(i^{-1}, i^{-1}) : R \subset F(R, R)$ by the definition of $(\_, \_) : \_ \subset \_$.
ii) $t \in F(R, R) \Rightarrow (i^{-1}, i^{-1})((i, i)(t)) \in F(R, R)$ but this implies $(i, i)(t) \in (i^{-1}, i^{-1})^*F(R, R)$ since we know that the domain-element parts of $(i, i)(t)$ belong to the required domains. Then using the assumption we have $(i, i)t \in R$. We conclude $(i, i) : F(R, R) \subset R$.
$\Rightarrow$: Assume $(i, i) : F(R, R) \subset R$ and $(i^{-1}, i^{-1}) : R \subset F(R, R)$
i) $t_0 \in R \Rightarrow (i^{-1}, i^{-1})t_0 \in F(R, R)$, so $t_0 \in (i^{-1}, i^{-1})^*F(R, R)$
ii) $t_1 \in (i^{-1}, i^{-1})^*F(R, R) \Rightarrow (i^{-1}, i^{-1})(t_1) \in F(R, R) \Rightarrow (i, i)((i^{-1}, i^{-1})(t_1)) \in R$, so $t_1 \in R$
We conclude $R = (i^{-1}, i^{-1})^*F(R, R)$.
$\square$

We want to prove the existence of a relation $R \in \mathcal{R}_{adm^+}(\mathbb{D})$ such that

$$(i, i) : F(R, R) \subset R \wedge (i^{-1}, i^{-1}) : R \subset F(R, R),$$

by the lemma 25 this is the same as looking for a relation

$$R = (i^{-1}, i^{-1})^*F(R, R).$$

We notice that $R \in \mathcal{R}_{adm^+}(\mathbb{D}),\ R = (i^{-1}, i^{-1})^*F(R, R)$ *iff*
R is a fixed point of $\Phi : \mathcal{R}_{adm^+}(\mathbb{D}) \to \mathcal{R}_{adm^+}(\mathbb{D})$ where $\Phi(R) = (i^{-1}, i^{-1})^*F(R, R)$.
Therefore we want to show that Φ has as a fixed point.

$(i^{-1}, i^{-1})^*$ is monotone: $\mathcal{R}_{adm^+}(F(\mathbb{D}, \mathbb{D})) \to \mathcal{R}_{adm^+}(\mathbb{D})$ with the $\subset$ ordering on relations.
$S_0 \subset S_1 \Rightarrow (i^{-1}, i^{-1})^*S_0 \subset (i^{-1}, i^{-1})^*S_1$. Apparent from the definition of $(i^{-1}, i^{-1})^*S$.
But $F : \mathcal{R}_{adm^+}(\mathbb{D}) \to \mathcal{R}_{adm^+}(F(\mathbb{D}, \mathbb{D}))$ is not monotone, hence Φ is not monotone. We separate negative and positive occurrences, define

$$\Psi(R^-, R^+) \overset{def}{=} (i^{-1}, i^{-1})^*F(R^-, R^+).$$

Ψ is monotone: $\mathcal{R}^{op}_{adm^+}(\mathbb{D}) \times \mathcal{R}_{adm^+}(\mathbb{D}) \to \mathcal{R}_{adm^+}(\mathbb{D})$ with the $\subset$ ordering on relations.
If $S^- \subset R^- \wedge R^+ \subset S^+$ then $F(R^-, R^+) \subset F(S^-, S^+)$. Then also $\Psi(R^-, R^+) = (i^{-1}, i^{-1})^* F(R^-, R^+) \subset (i^{-1}, i^{-1})^* F(S^-, S^+) = \Psi(S^-, S^+)$.

We now look for a fixed point of Ψ. Define

$$\Psi^{\S}(R^-, R^+) = (\Psi(R^+, R^-), \Psi(R^-, R^+)).$$

Then $\Psi^{\S}$ is a monotone operator on the complete lattice $\mathcal{R}_{adm^+}(\mathbb{D})^{op} \times \mathcal{R}_{adm^+}(\mathbb{D})$. By the Knaster-Tarski fixed point theorem the least fixed point (∇^-, ∇^+) of $\Psi^{\S}$ exists, it is also the least prefixed point, and it holds that $\Psi^{\S}(\nabla^-, \nabla^+) = (\nabla^-, \nabla^+)$, i.e. (by the definition of $\Psi^{\S}$) $\Psi(\nabla^+, \nabla^-) = \nabla^-$ and $\Psi(\nabla^-, \nabla^+) = \nabla^+$.

We want to show that $\nabla^- = \nabla^+$ or equivalently $\nabla^- \subset \nabla^+$ and $\nabla^+ \subset \nabla^-$.

We have $\Psi^{\S}(\nabla^+, \nabla^-) = (\Psi(\nabla^-, \nabla^+), \Psi(\nabla^+, \nabla^-)) = (\nabla^+, \nabla^-)$. Since (∇^-, ∇^+) is the least prefixed point it then follows that $\nabla^+ \subset \nabla^-$. It remains to show $\nabla^- \subset \nabla^+$. We have $\nabla^- \subset \nabla^+ \Leftrightarrow (id_{\mathbb{D}}, id_{\mathbb{D}}) : \nabla^- \subset \nabla^+$. We know that ∇^+ is admissible so $\{e \mid (e, id_{\mathbb{D}}) : \nabla^- \subset \nabla^+\}$ contains $\perp$ and is closed under least upper bounds of chains. By the minimal invariant property of $\mathbb{D}$ we know that $id_{\mathbb{D}}$ is the least fixed point of $\delta : (\mathbb{D} \multimap \mathbb{D}) \to (\mathbb{D} \multimap \mathbb{D})$ defined by $\delta(e) = i \circ F(e, e) \circ i^{-1}$, so it suffices to show that $\{ e \mid (e, id_{\mathbb{D}}) : \nabla^- \subset \nabla^+ \}$ is closed under δ.

Assume $(e, id_{\mathbb{D}}) : \nabla^- \subset \nabla^+$ then by lemma 22 it holds that $(F(e, e), F(id_{\mathbb{D}}, id_{\mathbb{D}})) = (F(e, e), id_{F(\mathbb{D},\mathbb{D})}) : F(\nabla^+, \nabla^-) \subset F(\nabla^-, \nabla^+)$. We also have (by definition of Ψ) that
$\nabla^- = \Psi(\nabla^+, \nabla^-) = (i^{-1}, i^{-1})^* F(\nabla^+, \nabla^-)$ and $\nabla^+ = \Psi(\nabla^-, \nabla^+) = (i^{-1}, i^{-1})^* F(\nabla^-, \nabla^+)$.

$\nabla^- = (i^{-1}, i^{-1})^* F(\nabla^+, \nabla^-)$ implies $(i^{-1}, i^{-1}) : \nabla^- \subset F(\nabla^+, \nabla^-)$.
$\nabla^+ = (i^{-1}, i^{-1})^* F(\nabla^-, \nabla^+)$ implies $(i^{-1}, i^{-1}) : \nabla^+ \subset F(\nabla^-, \nabla^+)$.
It also holds that when $s \in F(\nabla^-, \nabla^+)$ so $(i^{-1}, i^{-1})((i, i)(s)) \in F(\nabla^-, \nabla^+)$, and since domain parts of $(i, i)(s)$ belong to the required domains then this implies $(i, i)(s) \in (i^{-1}, i^{-1})^* F(\nabla^-, \nabla^+) = \nabla^+$.
So it holds that

$$\begin{array}{rrcl} & (i^{-1}, i^{-1}) : \quad \nabla^- & \subset & F(\nabla^+, \nabla^-) \\ (e, id_{\mathbb{D}}) : \nabla^- \subset \nabla^+ \Rightarrow & (F(e, e), id_{F(\mathbb{D},\mathbb{D})}) : F(\nabla^+, \nabla^-) & \subset & F(\nabla^-, \nabla^+) \\ & (i, i) : F(\nabla^-, \nabla^+) & \subset & \nabla^+ \end{array}$$

By compositionallity of $(\_, \_) : \_ \subset \_$ it holds that
if $(e, id_{\mathbb{D}}) : \nabla^- \subset \nabla^+$ then $(\delta(e), id_{\mathbb{D}}) = (i \circ F(e, e) \circ i^{-1}, id_{\mathbb{D}}) : \nabla^- \subset \nabla^+$.
$\{e \mid (e, id_{\mathbb{D}}) : \nabla^- \subset \nabla^+ \}$ contains $\perp$ and is closed under least upper bounds of chains. As $(\perp, id_{\mathbb{D}}) : \nabla^- \subset \nabla^+$ then $(\delta^n(\perp))$ is a chain in $\{e \mid (e, id_{\mathbb{D}}) : \nabla^- \subset \nabla^+ \}$. So $(\bigsqcup \delta^n(\perp), id_{\mathbb{D}}) = (id_{\mathbb{D}}, id_{\mathbb{D}}) : \nabla^- \subset \nabla^+$ hence $\nabla^- \subset \nabla^+$.

Since $\nabla^- \subset \nabla^+$ and $\nabla^+ \subset \nabla^-$ we can conclude $\nabla^- = \nabla^+$ call this relation ∇.

Then $(i, i) : F(\nabla, \nabla) \subseteq \nabla$ and $(i^{-1}, i^{-1}) : \nabla \subset F(\nabla, \nabla)$.
$\nabla \in \mathcal{R}_{adm^+}(\mathbb{D})$ is the **invariant relation**.

Here we prove the properties "parameter weakening" and "downwards closure" for the relation ∇ along the lines of Pitt's proof of admissibility in [40]. Admissibility is necessary for the proof of existence of an invariant relation to come through. Parameter weakening and downwards closure are just properties we want. In the paper [13] we added existence quantors (over finite sets) to the definition of the action of our domain constructing functor on relations. We could then prove parameter weakening for the invariant relation. Here we get the property without the existence quantors in the definition of the relation. For parameter weakening to be provable here, it must be the case that it is preserved under intersections so that we have a complete lattice $(\mathcal{R}_{adm^+}(D)^{op} \times$

$\mathcal{R}_{adm^+}(D)$, $\subset$) and can use Knaster-Tarski's fixed point theorem. Second, we must be able to show that the property is preserved by the action of F. To ensure this we require the property explicitly (or let it be easily provable from the definition) on base-value-related, function-type-related, $\forall$-type-related, computations and continuations. Generally we hope that the four-tuple construction will make more properties expressible and then provable in this way.

6.3 Properties of the invariant relation $\boldsymbol{\nabla = (\nabla_V, \nabla_K, \nabla_M, \nabla_S) \in \mathcal{R}_{adm^+}(\mathbb{D})}$

∇ is admissible , downwards closed and parameter-weakened, and we have
$(i, i) : F(\nabla, \nabla) \subset \nabla \ \wedge \ (i^{-1}, i^{-1}) : \nabla \subset F(\nabla, \nabla)$.

Let $F(\nabla,\nabla) = (F(\nabla,\nabla)_V, F(\nabla,\nabla)_K, F(\nabla,\nabla)_M, F(\nabla,\nabla)_S)$, then further by definition of the action of F on relations it holds that:

$$
\begin{aligned}
F(\nabla,\nabla)_V = \{&(\bot,\ \bot \parallel v_1,\ v_2,\ \tau,\ p) \mid p \in \mathfrak{p}^{vm} \text{ the set of all vm-parameters, } v_1, v_2 \in F(\mathbb{D},\mathbb{D})_V\} \cup \\
&\{(v_1',\ v_2' \parallel v_1,\ v_2,\ \tau,\ p) \mid p \in \mathfrak{p}^{vm} \ \wedge \ v_1', v_2', v_1, v_2 \in F(\mathbb{D},\mathbb{D})_V \ \wedge \\
&\quad v_1' \sqsubseteq v_1 \neq \bot \ \wedge \ v_2' \sqsubseteq v_2 \neq \bot \ \wedge \\
&\quad (v_1',\ v_2' \parallel v_1,\ v_2, \tau,\ p) \in \diamond \ \}
\end{aligned}
$$

where

$$
\begin{aligned}
\diamond \quad = \ &\{(v_1',\ v_2' \parallel in_{\mathbf{1}}\lfloor * \rfloor,\ in_{\mathbf{1}}\lfloor * \rfloor,\ unit,\ p)\ \} \cup \\
&\{(v_1',\ v_2' \parallel in_{\mathbb{Z}}\lfloor n \rfloor,\ in_{\mathbb{Z}}\lfloor n \rfloor,\ int,\ p) \mid n \in \mathbb{Z}\ \} \cup \\
&\{(v_1',\ v_2' \parallel in_{\mathbb{L}}\lfloor l_1 \rfloor,\ in_{\mathbb{L}}\lfloor l_2 \rfloor,\ \tau\ ref,\ p) \mid (l_1, l_2, \tau) \in Z_p\ \ \} \cup \\
&\{(v_1',\ v_2' \parallel in_{\oplus} in_i d_1,\ in_{\oplus} in_i d_2,\ \tau_1 + \tau_2,\ p) \mid d_1, d_2 \in (\mathbb{D}_V)_{\downarrow} \ \wedge \ \exists d_1', d_2' \in \mathbb{D}_V. \\
&\quad ((v_1' = \bot \wedge d_1' = \bot) \vee v_1' = in_{\oplus} in_i d_1' \neq \bot) \wedge ((v_2' = \bot \wedge d_2' = \bot) \vee v_2' = in_{\oplus} in_i d_2' \neq \bot) \wedge \\
&\quad (d_1',\ d_2' \parallel d_1,\ d_2,\ \tau_i,\ p) \in \nabla_V,\ i \in \{1,2\}\ \} \cup \\
&\{(v_1',\ v_2' \parallel in_{\otimes}(d_{1a}, d_{1b}),\ in_{\otimes}(d_{2a}, d_{2b}),\ \tau_a \times \tau_b,\ p) \mid d_{1a}, d_{1b}, d_{2a}, d_{2b} \in (\mathbb{D}_V)_{\downarrow} \ \wedge \\
&\quad \exists d_{1a}', d_{1b}', d_{2a}', d_{2b}' \in \mathbb{D}_V. \\
&\quad (d_{1a}',\ d_{2a}' \parallel d_{1a},\ d_{2a},\ \tau_a,\ p) \in \nabla_V \ \wedge \ (d_{1b}',\ d_{2b}' \parallel d_{1b},\ d_{2b},\ \tau_b,\ p) \in \nabla_V \ \wedge \\
&\quad ((v_1' = \bot \wedge (d_{1a}' = \bot \vee d_{1b}' = \bot)) \ \vee \ (v_1' = in_{\otimes}(d_{1a}', d_{1b}') \neq \bot)) \ \wedge \\
&\quad ((v_2' = \bot \wedge (d_{2a}' = \bot \vee d_{2b}' = \bot)) \ \vee \ (v_2' = in_{\otimes}(d_{2a}', d_{2b}') \neq \bot))\} \cup \\
&\{(v_1',\ v_2' \parallel in_{\mu} d_1,\ in_{\mu} d_2,\ \mu\alpha.\tau,\ p) \mid d_1, d_2 \in (\mathbb{D}_V)_{\downarrow} \ \wedge \ \exists d_1', d_2' \in \mathbb{D}_V. \\
&\quad ((v_1' = \bot \wedge d_1' = \bot) \vee v_1' = in_{\mu} d_1' \neq \bot) \wedge ((v_2' = \bot \wedge d_2' = \bot) \vee v_2' = in_{\mu} d_2' \neq \bot) \wedge \\
&\quad (d_1',\ d_2' \parallel d_1,\ d_2,\ \tau[\mu\alpha.\tau/\alpha],\ p) \in \nabla_V\ \} \cup \\
&\{(v_1',\ v_2' \parallel in_{\forall}\lfloor d_1 \rfloor,\ in_{\forall}\lfloor d_2 \rfloor,\ \forall\alpha.T\tau,\ p) \mid d_1, d_2 \in \mathbb{D}_M \ \wedge \ \exists d_1', d_2' \in \mathbb{D}_M. \\
&\quad ((v_1' = \bot \wedge d_1' = \bot) \ \vee \ (v_1' = in_{\forall}\lfloor d_1' \rfloor \neq \bot \ \wedge \ d_1' \sqsupseteq \bot)) \ \wedge \\
&\quad ((v_2' = \bot \wedge d_2' = \bot) \ \vee \ (v_2' = in_{\forall}\lfloor d_2' \rfloor \neq \bot \ \wedge \ d_2' \sqsupseteq \bot)) \ \wedge \\
&\quad \forall\sigma \ with \ \_ \vdash \sigma : type.\ (d_1',\ d_2' \parallel d_1,\ d_2,\ T\tau[\sigma/\alpha],\ p) \in \nabla_M\ \} \cup \\
&\{(v_1',\ v_2' \parallel in_{\multimap}\lfloor d_1 \rfloor,\ in_{\multimap}\lfloor d_2 \rfloor,\ \tau \to \mathrm{T}\tau',\ p) \mid d_1, d_2 \in (\mathbb{D}_V \multimap \mathbb{D}_M) \ \wedge \ \exists d_1', d_2' \in (\mathbb{D}_V \multimap \mathbb{D}_M). \\
&\quad ((v_1' = \bot \wedge d_1' = \bot) \ \vee \ (v_1' = in_{\multimap}\lfloor d_1' \rfloor \neq \bot \ \wedge \ d_1' \sqsupseteq \bot)) \ \wedge \\
&\quad ((v_2' = \bot \wedge d_2' = \bot) \ \vee \ (v_2' = in_{\multimap}\lfloor d_2' \rfloor \neq \bot \ \wedge \ d_2' \sqsupseteq \bot)) \ \wedge \\
&\quad \forall p' \blacktriangleright p, \forall (w_1',\ w_2' \parallel w_1,\ w_2, \tau,\ p') \in \nabla_V.\ (\ d_1' w_1',\ d_2' w_2' \parallel d_1 w_1,\ d_1 w_2,\ T\tau',\ p') \in \nabla_M\ \}
\end{aligned}
$$

Recall $\blacktriangleright$ on vm-parameters is defined as the reflexive transitive closure of the relations $\succeq$, $\supseteq$ and $\geq_{\bar{\wedge}}$, (which are local extension, superset and removal of $\bar{\wedge}$-clauses).

$F(\nabla,\nabla)_K = \{(k_1',\ k_2' \parallel k_1,\ k_2,\ (x:\tau)^\top,\ (pk)) \mid (pk) \in \mathfrak{p}^k\ \wedge$
$k_1' \sqsubseteq k_1\ \wedge\ k_2' \sqsubseteq k_2\ \wedge\ \ \forall (pk') \rhd (pk).$
(that is (pk) extended only with *ordinary* local k-parameters or local extensions)

$\forall (pks') \in (pk')^{\mathbf{S}}$ (the set of s-instantiations of p' i.e. choices of $\bar{\vee}$-clauses),
$\forall (s_1',\ s_2' \parallel s_1,\ s_2,\ (pks')) \in \nabla_S.\ \forall (v_1',\ v_2' \parallel v_1,\ v_2, \tau,\ (pk')^{vm}) \in \nabla_V.$
$(k_1's_1'v_1' = \top \Rightarrow k_2s_2v_2 = \top)\ \wedge$
$(k_2's_2'v_2' = \top \Rightarrow k_1s_1v_1 = \top)\ \}$

$F(\nabla,\nabla)_M = \{(m_1',\ m_2' \parallel m_1,\ m_2,\ T\tau,\ p) \mid p \in \mathfrak{p}^{vm}\ \wedge$
$m_1' \sqsubseteq m_1\ \wedge\ m_2' \sqsubseteq m_2\ \wedge$
$\forall p' \blacktriangleright p. \forall (pk') \in p'^{\mathbf{K}}$ (the set of k-instantiations of p' i.e.
choices of a $\bar{\wedge}$-clause in each local parameter).

$\forall (pks') \in (pk')^{\mathbf{S}}$ (the set of s-instantiations of (pk') i.e.
choices of $\bar{\vee}$-clause from each chosen $\bar{\wedge}$-clause in (pk')),
$\forall (k_1',\ k_2' \parallel k_1,\ k_2,\ (x:\tau)^\top,\ (pk')) \in \nabla_K.\ \forall (S_1',\ S_2' \parallel S_1,\ S_2,\ (pks')) \in \nabla_S$.
$(m_1'k_1'S_1' = \top \Rightarrow m_2k_2S_2 = \top)\ \wedge$
$(m_2'k_2'S_2' = \top \Rightarrow m_1k_1S_1 = \top)\ \}$

$F(\nabla,\nabla)_S = \{(\bot,\ \bot \parallel S_1,\ S_2,\ (pks)) \mid (pks) \in \mathfrak{p}^s$ the set of all s-parameters $\}\ \cup$

$\{(S_1',\ S_2' \parallel\ S_1,\ S_2,\ (pks) \mid (pks) = \{(r_1|q_1|Q_1), \ldots, (r_n|q_n|Q_n)\} \in \mathfrak{p}^s$
(q_i is a choice of $\bar{\wedge}$-clause in r_i, and Q_i is a choice of $\bar{\vee}$-clause in q_i) $\wedge$

$S_1' \sqsubseteq S_1 \neq \bot\ \ \wedge\ S_2' \sqsubseteq S_2 \neq \bot\ \wedge$

$\mathcal{A}_1^{(pks)}(S_1) \cap \pi_1(Z^{(pks)}) = \emptyset\ \wedge\ \mathcal{A}_2^{(pks)}(S_2) \cap \pi_2(Z^{(pks)}) = \emptyset$
(in each side are the visible locations in $Z^{(pks)}$ disjoint from the known hidden locations) $\wedge$

$\forall i \neq j.\ \mathring{A}_1^{ri}(S_1) \cap \mathring{A}_1^{rj}(S_1) = \emptyset\ \wedge\ \mathring{A}_2^{ri}(S_2) \cap \mathring{A}_2^{rj}(S_2) = \emptyset$
($\mathring{A}_1^{ri}, \mathring{A}_2^{ri}$ accessibility maps are the most inclusive for the local parameter r_i.
In each side is every location, visible, hidden or reserved, belonging to a local parameter
outside the areas owned by any different local parameter) $\wedge$

$\forall (l_1, l_2, \tau) \in Z^{(pks)}.(S_1'(l_1),\ S_2'(l_2) \parallel S_1(l_1),\ S_2(l_2),\ \tau,\ (pks)^{vm}) \in \nabla_V$
(All visible locations hold related values) $\wedge$

$\forall i \in 1..n.$ if $Q_i = (P_i, LL_i)$ then $(S_1, S_2) \in P_i\ \wedge$
$\forall (l_1, l_2, \tau) \in LL_i.\ (S_1'(l_1),\ S_2'(l_2) \parallel S_1(l_1),\ S_2(l_2),\ \tau,\ (pks)^{vm}) \in \nabla_V$
(The states belong to all simple state relations in the chosen $\bar{\vee}$-clauses.
Corresponding LL location-sets hold related values) $\}$

– For all $(v_1',\ v_2' \parallel v_1,\ v_2,\ \tau,\ p) \in F(\nabla,\nabla)_V$ it holds that $(v_1' = v_2' = \bot)$ or $(v_1 \neq \bot \wedge v_2 \neq \bot)$.
– For all $(s_1',\ s_2' \parallel s_1,\ s_2,\ p) \in F(\nabla,\nabla)_S$ it holds that $(s_1' = s_2' = \bot)$ or $(s_1 \neq \bot \wedge s_2 \neq \bot)$.
– $\forall (x:\tau)^\top. \forall p \in \mathfrak{p}^k. \forall k_1, k_2 \in F(\mathbb{D},\mathbb{D})_K.\ (\bot, \bot \parallel k_1, k_2, (x:\tau)^\top, p) \in F(\nabla,\nabla)_K$
– $\forall T\tau. \forall p \in \mathfrak{p}^{vm}. \forall m_1, m_2 \in F(\mathbb{D},\mathbb{D})_M.\ (\bot, \bot \parallel m_1, m_2, T\tau, p) \in F(\nabla,\nabla)_M$

There are certain properties of values, computations and states which have to do with parameters derived from other parameters by removal of $\bar{\wedge}$-clauses. Such parameters are $\blacktriangleright$ related, and the parameter weakening property gives that relatedness of values and computations will be preserved under $\blacktriangleright$ extensions. We have no general parameter weakening property for states, but a special limited weakening concerned with removal of con-clauses. The next lemma says that if states s_1', s_2', s_1, s_2 are related under an s-parameter pks and pks' is an s-parameter derived from pks by removal of some $\bar{\wedge}$-clauses but such that those instantiated in pks are not touched, then s_1', s_2', s_1, s_2 will also be related under pks'.

Lemma 26. *Let $p = \{r_i\}$ be a vm-parameter, and let $pks = \{(r_i|q_i|Q_i)\} \in p^{\mathbf{S}}$.*
Assume $(s_1', s_2' \parallel s_1, s_2, pks) \in \nabla_S$.
If $\forall i. r_i \supseteq r_i' \neq \emptyset$ and $pks' = \{(r_i'|q_i|Q_i)\}$,
then $p' = (pks')^{vm} = \{r_i'\}$ and $(s_1', s_2' \parallel s_1, s_2, pks') \in \nabla_S$.

Proof By definition of $\blacktriangleright$ it holds that $p' = (pks')^{vm} = \{r_i'\} \blacktriangleright p$.
The approximation properties $s_1' \sqsubseteq s_1$ and $s_2' \sqsubseteq s_2$ are not changed. By assumption

(a) $\mathcal{A}_1^{(pks)}(s_1) \cap \pi_1(Z^{(pks)}) = \emptyset \ \wedge \ \mathcal{A}_2^{(pks)}(s_2) \cap \pi_2(Z^{(pks)}) = \emptyset$ and
(b) $\forall i \neq j.\ \mathring{A}_1^{ri}(s_1) \cap \mathring{A}_1^{rj}(s_1) = \emptyset \ \wedge \ \mathring{A}_2^{ri}(s_2) \cap \mathring{A}_2^{rj}(s_2) = \emptyset$ and
(c) $\forall i \in 1..n.$ if $Q_i = (P_i, LL_i)$ then $(s_1, s_2) \in P_i \ \wedge$
$\forall (l_1, l_2, \tau) \in LL_i.\ (s_1'(l_1),\ s_2'(l_2) \parallel s_1(l_1),\ Ss_2(l_2),\ \tau,\ (pks)^{vm}) \in \nabla_V$ and
(d) $\forall (l_1, l_2, \tau) \in Z^{(pks)}.(s_1'(l_1),\ s_2'(l_2) \parallel s_1(l_1),\ s_2(l_2),\ \tau,\ (pks)^{vm}) \in \nabla_V$

We need to prove

1. $\mathcal{A}_1^{(pks')}(s_1) \cap \pi_1(Z^{(pks')}) = \emptyset \ \wedge \ \mathcal{A}_2^{(pks')}(s_2) \cap \pi_2(Z^{(pks')}) = \emptyset$ and
2. $\forall i \neq j.\ \mathring{A}_1^{ri'}(s_1) \cap \mathring{A}_1^{rj'}(s_1) = \emptyset \ \wedge \ \mathring{A}_2^{ri'}(s_2) \cap \mathring{A}_2^{rj'}(s_2) = \emptyset$ and
3. $\forall i \in 1..n.$ if $Q_i = (P_i, LL_i)$ then $(s_1, s_2) \in P_i \ \wedge$
$\forall (l_1, l_2, \tau) \in LL_i.\ (s_1'(l_1),\ s_2'(l_2) \parallel s_1(l_1),\ Ss_2(l_2),\ \tau,\ (pks')^{vm}) \in \nabla_V$ and
4. $\forall (l_1, l_2, \tau) \in Z^{(pks')}.(s_1'(l_1),\ s_2'(l_2) \parallel s_1(l_1),\ s_2(l_2),\ \tau,\ (pks')^{vm}) \in \nabla_V$

1. follows from (a). $Z^{(pks')} = \biguplus Z^{q_i}$ is defined on the basis of the instantiated q_i's in (pks'). These are the same as in (pks), so $Z^{(pks')} = Z^{(pks)}$. Also $\mathcal{A}_1^{(pks')}, \mathcal{A}_2^{(pks')}, \mathcal{A}_1^{(pks)}, \mathcal{A}_2^{(pks)}$ are defined on the basis of the instantiated q_i's.
2. follows from (b) together with $\forall i.\ r_i' \subseteq r_i \Rightarrow (\mathring{A}_1^{ri'}(s_1) \subseteq \mathring{A}_1^{ri}(s_1) \wedge \mathring{A}_2^{ri'}(s_2) \subseteq \mathring{A}_2^{ri}(s_2))$.
3. There are no changes to finitary state relations, so also the LL's will be the same. As $(pks')^{vm} \blacktriangleright (pks)^{vm}$ then by weakening for values and (c) all values stored in LL's are still related.
4. follows from (d) and $Z^{(pks')} = Z^{(pks)}$ and $(pks')^{vm} \blacktriangleright (pks)^{vm}$ together with parameter weakening for stored values.

□

6.4 Relating denotations of open terms

We will now define a parameterized binary relation $\nabla^{\Xi\Gamma}$ between denotations of open expressions. The relation is defined by reference to the relation ∇. The binary relation will be shown to imply contextual equivalence at (ordinary) parameters, which only give a set of visible locations and no hidden invariants.

Definition 34. *The relation $\nabla^{\Xi\Gamma}$*

$\nabla_V^{\Xi\Gamma}$ *For $\Xi = \alpha_1 \ldots \alpha_m,\ \Gamma = x_1 : \tau_1, \ldots, x_n : \tau_n$ and $\Delta; \Xi; \Gamma \vdash V_1 : \tau$ and $\Delta; \Xi; \Gamma \vdash V_2 : \tau$*
Let p be an ordinary vm-parameter with associated Z^p such that $Z_1^p = Z_2^p = \Delta$,
let $v_1 = [\![\Delta; \Xi; \Gamma \vdash V_1]\!]$ and $v_2 = [\![\Delta; \Xi; \Gamma \vdash V_2]\!]$, define

$(v_1, v_2, \tau, p) \in \nabla_V^{\Xi\Gamma} \overset{def}{\iff}$

$\forall \overline{\sigma_j}$ *with* $\_ \vdash \sigma_j : type,\ j = 1,..,m.\ \forall\ p' \blacktriangleright p.$
$\forall (v'_{11}, v'_{21} \parallel v_{11}, v_{21}, \tau_1[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V, \ldots, (v'_{1n}, v'_{2n} \parallel v_{1n}, v_{2n}, \tau_n[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$, *with*
$\rho'_1 = v'_{11} \otimes \ldots \otimes v'_{1n} \in [\![\Gamma]\!],\ \rho_1 = v_{11} \otimes \ldots \otimes v_{1n} \in [\![\Gamma]\!],$
$\rho'_2 = v'_{21} \otimes \ldots \otimes v'_{2n} \in [\![\Gamma]\!],\ \rho_2 = v_{21} \otimes \ldots \otimes v_{2n} \in [\![\Gamma]\!].$

it holds that $(v_1(\rho'_1), v_2(\rho'_2) \parallel v_1(\rho_1), v_2(\rho_2), \tau[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V.$

$\nabla_M^{\Xi\Gamma}$ *For* $\Xi = \alpha_1 \ldots \alpha_m,\ \Gamma = x_1 : \tau_1, \ldots, x_n : \tau_n$ *and* $\Delta; \Xi; \Gamma \vdash M_1 : T\tau$ *and* $\Delta; \Xi; \Gamma \vdash M_2 : T\tau$
Let p *be an ordinary vm-parameter with associated* Z^p *such that* $Z_1^p = Z_2^p = \Delta$,
let $m_1 = [\![\Delta; \Xi; \Gamma \vdash M_1]\!]$ *and* $m_2 = [\![\Delta; \Xi; \Gamma \vdash M_2]\!]$, *define*
$(m_1, m_2, T\tau, p) \in \nabla_M^{\Xi\Gamma} \overset{def}{\iff}$

$\forall \overline{\sigma_j}$ *with* $\_ \vdash \sigma_j : type,\ j = 1,..,m.\ \forall p' \blacktriangleright p.$
$\forall (v'_{11}, v'_{21} \parallel v_{11}, v_{21}, \tau_1[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V, \ldots, (v'_{1n}, v'_{2n} \parallel v_{1n}, v_{2n}, \tau_n[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$, *with*
$\rho'_1 = v'_{11} \otimes \ldots \otimes v'_{1n} \in [\![\Gamma]\!],\ \rho_1 = v_{11} \otimes \ldots \otimes v_{1n} \in [\![\Gamma]\!],$
$\rho'_2 = v'_{21} \otimes \ldots \otimes v'_{2n} \in [\![\Gamma]\!],\ \rho_2 = v_{21} \otimes \ldots \otimes v_{2n} \in [\![\Gamma]\!].$

it holds that $(m_1(\rho'_{1i}), m_2(\rho'_{2i}) \parallel m_1(\rho_{1i}), m_2(\rho_{2i}), T\tau[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M.$

Proposition 4. *Typing rules preserve* $\nabla^{\Xi\Gamma}$ *relation of denotations.*
For typing rules with no premisses it holds that for all $p \in \mathfrak{o}^{vm}$ *with* $Z_1^p = Z_2^p = \Delta$

if $\dfrac{}{\Delta; \Xi; \Gamma \vdash V : \tau}$ *and* $v = [\![\Delta; \Xi; \Gamma \vdash V : \tau]\!]$ *then* $(v, v, \tau, p) \in \nabla_V^{\Xi\Gamma}$

For typing rules with j *premisses it holds that for all* $p \in \mathfrak{o}^{vm}$ *with* $Z_1^p = Z_2^p = \Delta$

if $\dfrac{\Delta; \Xi_1; \Gamma_1 \vdash G_{11} : \gamma_1 \ \ldots\ldots\ \Delta; \Xi_j; \Gamma_j \vdash G_{j1} : \gamma_j}{\Delta; \Xi; \Gamma \vdash G'_1 : \gamma'}$ *and by the same typing rule*

$\dfrac{\Delta; \Xi_1; \Gamma_1 \vdash G_{12} : \gamma_1 \ \ldots\ldots\ \Delta; \Xi_j; \Gamma_j \vdash G_{j2} : \gamma_j}{\Delta; \Xi; \Gamma \vdash G'_2 : \gamma'}$ *and for the well typed terms*

$d_{11} = [\![\Delta; \Xi_1; \Gamma_1 \vdash G_{11} : \gamma_1]\!],\ d_{12} = [\![\Delta; \Xi_1; \Gamma_1 \vdash G_{12} : \gamma_1]\!], \ldots,$
$d_{j1} = [\![\Delta; \Xi_j; \Gamma_j \vdash G_{j1} : \gamma_j]\!],\ d_{j2} = [\![\Delta; \Xi_j; \Gamma_j \vdash G_{j2} : \gamma_j]\!]$, *and*
$\forall i \in 1 \ldots j,\ (d_{i1}, d_{i2}, \gamma_i, p) \in \nabla_X^{\Xi_i\Gamma_i}$ *(where in each case* X *is the relevant* $X \in V, M$*).*
And $d'_1 = [\![\Delta; \Xi; \Gamma' \vdash G'_1 : \gamma']\!],\ d'_2 = [\![\Delta; \Xi; \Gamma' \vdash G'_2 : \gamma']\!].$

then it holds that $(d'_1, d'_2, \gamma', p) \in \nabla_X^{\Xi\Gamma}$

Proof
To shorten formulations in the proof we let here *arbitrarily* $p \in \mathfrak{o}^{vm}$ be a parameter such that $Z^p = \{(l_1, l_1, \hat{\sigma}_1) \ldots (l_n, l_n, \hat{\sigma}_n)\}$ where $\Delta = Z_1^p = Z_2^p$.

For $\Xi = \alpha_1 \ldots \alpha_m$, $\Gamma = x_1 : \tau_1, \ldots, x_n : \tau_n$ with $\Xi \vdash \Gamma$ and
arbitrary $\sigma_1 \ldots \sigma_m$ with $\_ \vdash \sigma_k : type$,

arbitrary $p' \blacktriangleright p$, *arbitrary* $(v'_{1i}, v'_{2i} \parallel v_{1i}, v_{2i}, \tau_i[\overline{\sigma_k/\alpha_k}], p') \in \nabla_V,\ i = 1,..,n.$

Let $\rho'_1 = v'_{11} \otimes \ldots \otimes v'_{1n} \in [\![\Gamma]\!]$. Let $\rho_1 = v_{11} \otimes \ldots \otimes v_{1n} \in [\![\Gamma]\!]$.
Let $\rho'_2 = v'_{21} \otimes \ldots \otimes v'_{2n} \in [\![\Gamma]\!]$. Let $\rho_2 = v_{21} \otimes \ldots \otimes v'_{2n} \in [\![\Gamma]\!]$.

As $\forall i.\ v'_{1i} \sqsubseteq v_{1i} \wedge v'_{2i} \sqsubseteq v_{2i}$ and $\big((v'_{1i} = v'_{2i} = \bot) \vee (v_{1i} \neq \bot \wedge v_{2i} \neq \bot)\big)$, then it holds that
$\rho'_1 \sqsubseteq \rho_1 \wedge \rho'_2 \sqsubseteq \rho_2$
$(\rho'_1 = \rho'_2 = \bot)\ \vee (\rho_1 \neq \bot \wedge \rho_2 \neq \bot)$
$(\rho_1 = \bot \vee \rho_2 = \bot) \Rightarrow (\rho'_1 = \bot \wedge \rho'_2 = \bot)$
$(\rho'_1 \neq \bot \vee \rho'_2 \neq \bot) \Rightarrow (\rho_1 \neq \bot \wedge \rho_2 \neq \bot)$.

We now show for each typing rule that relatedness of denotations is preserved. As the σ types, the parameter extension p' and the ρ environments are arbitrary, this covers all cases

id: $$\frac{}{\Delta; \Xi; x_1 : \tau_1 \ldots x_j : \tau_j \ldots x_n : \tau_n \vdash x_j : \tau_j}$$

Let $\Gamma = x_1 : \tau_1 \ldots x_j : \tau_j \ldots x_n : \tau_n$.
There are no assumptions, so we need to prove that it holds
$([\![\Delta; \Xi; \Gamma \vdash x_j : \tau_j]\!], [\![\Delta; \Xi; \Gamma \vdash x_j : \tau_j]\!], \tau, p) \in \nabla_V^{\Xi\Gamma}$ that is
$([\![\Delta; \Xi; \Gamma \vdash x_j : \tau_j]\!](\rho'_{1i}), [\![\Delta; \Xi; \Gamma \vdash x_j : \tau_j]\!](\rho'_{2i})) \parallel$
$[\![\Delta; \Xi; \Gamma \vdash x_j : \tau_j]\!](\rho_{1i}), [\![\Delta; \Xi; \Gamma \vdash x_j : \tau_j]\!](\rho_{2i})), \tau[\overline{\sigma_k/\alpha_k}], p') \in \nabla_V$.

If $\rho'_1 = \bot \wedge \rho'_2 = \bot$ then for any $j \in \{1, .., n\}$,
$([\![\Delta; \Xi; \Gamma \vdash x_j : \tau_j]\!](\rho'_1), [\![\Delta; \Xi; \Gamma \vdash x_j : \tau_j]\!](\rho'_2)),$
$[\![\Delta; \Xi; \Gamma \vdash x_j : \tau_j]\!](\rho_1), [\![\Delta; \Xi; \Gamma \vdash x_j : \tau_j]\!](\rho_2)) =$
$(\bot, \bot, [\![\Delta; \Xi; \Gamma \vdash x_j : \tau_j]\!](\rho_1), [\![\Delta; \Xi; \Gamma \vdash x_j : \tau_j]\!](\rho_2))$ and
$(\bot, \bot \parallel [\![\Delta; \Xi; \Gamma \vdash x_j : \tau_j]\!](\rho_1), [\![\Delta; \Xi; \Gamma \vdash x_j : \tau_j]\!](\rho_2), \tau_j[\overline{\sigma_k/\alpha_k}], p') \in \nabla_V,$

If $\rho'_1 \neq \bot \vee \rho'_2 \neq \bot$ and $\rho_1 \neq \bot \wedge \rho_2 \neq \bot$ then for $j \in \{1, .., n\}$,
$([\![\Delta; \Xi; \Gamma \vdash x_j : \tau_j]\!](\rho'_1), [\![\Delta; \Xi; \Gamma \vdash x_j : \tau_j]\!](\rho'_2),$
$[\![\Delta; \Xi; \Gamma \vdash x_j : \tau_j]\!](\rho_1), [\![\Delta; \Xi; \Gamma \vdash x_j : \tau_j]\!](\rho_2)) =$
$(d'_{1j}, d'_{2j}, v_{1j}, v_{2j})$, where $d'_{1j} \in \{\bot, v'_{1j}\} \wedge d'_{2j} \in \{\bot, v'_{2j}\}$ and as
$(v'_{1j}, v'_{2j} \parallel v_{1j}, v_{2j}, \tau_j[\overline{\sigma_k/\alpha_k}], p') \in \nabla_V$
so by downwards closure $(d'_{1j}, d'_{2j} \parallel v_{1j}, v_{2j}, \tau_j[\overline{\sigma_k/\alpha_k}], p') \in \nabla_V$.

Hence $([\![\Delta; \Gamma \vdash x_j : \tau_j]\!], [\![\Delta; \Gamma \vdash x_j : \tau_j]\!], \tau_j, p) \in \nabla_V^{\Xi\Gamma}$.

loc: $$\frac{}{\Delta; \Xi; \Gamma \vdash l : \sigma\ ref}\ (l : \sigma \in \Delta)$$

$\Delta; \Xi; \Gamma \vdash l : \sigma\ ref$ requires $l : \sigma \in \Delta$.

We need to prove that it holds $([\![\Delta; \Xi; \Gamma \vdash l : \sigma ref]\!], [\![\Delta; \Xi; \Gamma \vdash l : \sigma ref]\!], \sigma ref, p) \in \nabla_V^{\Xi\Gamma}$. The requirement $(l : \sigma \in \Delta)$ means that σ is closed and so $\sigma[\overline{\sigma_j/\alpha_j}] = \sigma$. So we want to prove that
$([\![\Delta; \Xi; \Gamma \vdash l : \sigma ref]\!](\rho'_1), [\![\Delta; \Xi; \Gamma \vdash l : \sigma ref]\!](\rho'_2) \parallel$
$[\![\Delta; \Xi; \Gamma \vdash l : \sigma ref]\!](\rho_1), [\![\Delta; \Xi; \Gamma \vdash l : \sigma ref]\!](\rho_2)), \sigma ref, p') \in \nabla_V$.

If $\rho'_1 = \bot \wedge \rho'_2 = \bot$ then
$([\![\Delta; \Xi; \Gamma \vdash l : \sigma ref]\!](\rho'_1), [\![\Delta; \Xi; \Gamma \vdash l : \sigma ref]\!](\rho'_2),$
$[\![\Delta; \Xi; \Gamma \vdash l : \sigma ref]\!](\rho_1), [\![\Delta; \Xi; \Gamma \vdash l : \sigma ref]\!](\rho_2)) =$
$(\bot, \bot, [\![\Delta; \Xi; \Gamma \vdash l : \sigma ref]\!](\rho_1), [\![\Delta; \Xi; \Gamma \vdash l : \sigma ref]\!](\rho_2))$ and
$(\bot, \bot \parallel [\![\Delta; \Xi; \Gamma \vdash l : \sigma ref]\!](\rho_1), [\![\Delta; \Xi; \Gamma \vdash l : \sigma ref]\!](\rho_2), \sigma[\overline{\sigma_j/\alpha_j}]ref, p') \in \nabla_V,$

If $\rho'_1 \neq \bot \vee \rho'_2 \neq \bot$ and $\rho_1 \neq \bot \wedge \rho_2 \neq \bot$ then
$([\![\Delta; \Xi; \Gamma \vdash l : \sigma ref]\!](\rho'_1), [\![\Delta; \Xi; \Gamma \vdash l : \sigma ref]\!](\rho'_2),$
$[\![\Delta; \Xi; \Gamma \vdash l : \sigma ref]\!](\rho_1), [\![\Delta; \Xi; \Gamma \vdash l : \sigma ref]\!](\rho_2)) =$

$(d'_1, d'_2, i(in_{\mathbb{L}} l), i(in_{\mathbb{L}} l))$ where $d'_1, d'_2 \in \{\bot, i(in_{\mathbb{L}} l)\}$
Since $p' \blacktriangleright p$ then $Z^{p'} \supseteq Z^p$ so $(l, l, \sigma) \in Z^p \Rightarrow (l, l, \sigma) \in Z^{p'}$ so
$(d'_1, d'_2 \parallel i(in_{\mathbb{L}} l), i(in_{\mathbb{L}} l), \sigma[\overline{\sigma_j/\alpha_j}]\ ref, p') \in \nabla_V$

Hence $([\![\Delta; \Gamma \vdash l : \sigma ref]\!], [\![\Delta; \Gamma \vdash l : \sigma ref]\!], \sigma ref, p) \in \nabla_V^{\Xi\Gamma}$.

val: $$\frac{\Delta; \Xi; \Gamma \vdash V : \tau}{\Delta; \Xi; \Gamma \vdash val\ V : T\tau}$$

Let $v_1 = [\![\Delta; \Xi; \Gamma \vdash V_1 : \tau]\!]$ and $v_2 = [\![\Delta; \Xi; \Gamma \vdash V_2 : \tau]\!]$. Assume $(v_1, v_2, \tau, p) \in \nabla_V^{\Xi\Gamma}$.
This assumption implies $(v_1(\rho'_1), v_2(\rho'_2) \parallel v_1(\rho_1), v_2(\rho_2), \tau[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$

Let $m_1 = [\![\Delta; \Xi; \Gamma \vdash val\ V_1 : T\tau]\!]$ and $m_2 = [\![\Delta; \Xi; \Gamma \vdash val\ V_2 : T\tau]\!]$.
We need to show $(m_1, m_2, T\tau, p) \in \nabla_M^{\Xi\Gamma}$, that is we want to show
$(m_1(\rho'_1), m_2(\rho'_2) \parallel m_1(\rho_1), m_2(\rho_2), T\tau[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$, or
$(i^{-1}(m_1(\rho'_1)), i^{-1}(m_2(\rho'_2)) \parallel i^{-1}(m_1(\rho_1)), i^{-1}(m_2(\rho_2)), T\tau[\overline{\sigma_j/\alpha_j}], p') \in F(\nabla, \nabla)_M$.

$i^{-1}(m_1(\rho'_1)) = \lambda k.\lambda S.i^{-1}(k)S(v_1(\rho'_1))$, $i^{-1}(m_2(\rho'_2)) = \lambda k.\lambda S.i^{-1}(k)S(v_2(\rho'_2))$,
$i^{-1}(m_1(\rho_1)) = \lambda k.\lambda S.i^{-1}(k)S(v_1(\rho_1))$, $i^{-1}(m_2(\rho_2)) = \lambda k.\lambda S.i^{-1}(k)S(v_2(\rho_2))$.

If $v_1(\rho'_1) = v_2(\rho'_2) = \bot$ then $m_1(\rho'_1)$ and $m_2(\rho'_2)$ will both be the constant $\bot$ function in $\mathbb{M}$, and hence $(m_1(\rho'_1), m_2(\rho'_2) \parallel m_1(\rho_1), m_2(\rho_2), T\tau[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$.

Else, let $p'' \blacktriangleright p', (pk'') \in p''^{\mathbf{K}}, (pks'') \in pk''^{\mathbf{S}}$ and let $(k'_1, k'_2 \parallel k_1, k_2, (x : \tau[\overline{\sigma_j/\alpha_j}])^\top, (pk'')) \in \nabla_K, (S'_1, S'_2 \parallel S_1, S_2, (pks'')) \in \nabla_S$. We need to show
$m_1(\rho'_1))k'_1 S'_1 = \top \Rightarrow (m_2(\rho_2))k_2 S_2 = \top$ and $m_2(\rho'_2))k'_2 S'_2 = \top \Rightarrow (m_1(\rho_1))k_1 S_1 = \top$

By assumption and weakening $(v_1(\rho'_1), v_2(\rho'_2) \parallel v_1(\rho_1), v_2(\rho_2), \tau[\overline{\sigma_j/\alpha_j}], p'') \in \nabla_V$.
$i^{-1}(m_1(\rho'_1))k'_1 S'_1 = i^{-1}(k'_1) S'_1 v_1(\rho'_1)$, $i^{-1}(m_2(\rho'_2))k'_2 S'_2 = i^{-1}(k'_2) S'_2 v_2(\rho'_2)$,
$i^{-1}(m_1(\rho_1))k_1 S_1 = i^{-1}(k_1) S_1 v_1(\rho_1)$, $i^{-1}(m_2(\rho_2))k_2 S_2 = i^{-1}(k_2) S_2 v_2(\rho_2)$.

So it follows that since k's, S's and v's are correspondingly related so
$(m_1(\rho'_1))k'_1 S'_1 = k'_1 S'_1(v_1(\rho'_1)) = \top \Rightarrow k_2 S_2(v_2(\rho_2)) = (m_2(\rho_2))k_2 S_2 = \top\ \wedge$
$(m_2(\rho'_2))k'_2 S'_2 = k'_2 S'_2(v_2(\rho'_2)) = \top \Rightarrow k_1 S_1(v_1(\rho_1)) = (m_1(\rho_1))k_1 S_1 = \top$
Hence $(m_1(\rho'_1), m_2(\rho'_2) \parallel m_1(\rho_1), m_2(\rho_2), T\tau[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$.
We conclude $(m_1, m_2, T\tau[\overline{\sigma_j/\alpha_j}], p) \in \nabla_M^\Gamma$.

deref: $$\frac{\Delta; \Xi; \Gamma \vdash V : \tau\ ref}{\Delta; \Xi; \Gamma \vdash !V : T\tau}$$

Let $v_1 = [\![\Delta; \Xi; \Gamma \vdash V_1 : \tau\ ref]\!]$ and $v_2 = [\![\Delta; \Xi; \Gamma \vdash V_2 : \tau\ ref]\!]$.
Assume $(v_1, v_2, \tau\ ref, p) \in \nabla_V^{\Xi\Gamma}$, so $(v_1(\rho'_1), v_2(\rho'_2) \parallel v_1(\rho_1), v_2(\rho_2), \tau[\overline{\sigma_j/\alpha_j}]\ ref, p') \in \nabla_V$.
This implies that either $v_1(\rho'_1) = v_2(\rho'_2) = \bot$ or $\exists (l_1, l_2, \tau[\overline{\sigma_j/\alpha_j}]) \in Z^{p'}$.
$v_1(\rho'_1) \in \{\bot, i(in_{\mathbb{L}} l_1)\}$, $v_1(\rho_1) = i(in_{\mathbb{L}} l_1)$, $v_2(\rho'_2) \in \{\bot, i(in_{\mathbb{L}} l_2)\}$, $v_2(\rho_2) = i(in_{\mathbb{L}} l_2)$.

Let $m_1 = [\![\Delta; \Xi; \Gamma \vdash !V_1]\!]$ and $m_2 = [\![\Delta; \Xi; \Gamma \vdash !V_2]\!]$.
We need to show $(m_1(\rho'_1), m_2(\rho'_2) \parallel m_1(\rho_1), m_2(\rho_2), T\tau[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$.
If $v_1(\rho'_1) = v_2(\rho'_2) = \bot$ then $m_1(\rho'_1) = m_2(\rho'_2) = \bot$ and
$(m_1(\rho'_1), m_2(\rho'_2) \parallel m_1(\rho_1), m_2(\rho'_2), v_2(\rho_2), T\tau[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$. Else,
$i^{-1}(m_1(\rho'_1)) \in \{\bot, \lambda k.\lambda S.(i^{-1}k)S(Sl_1)\}$, $i^{-1}(m_1(\rho_1)) = \lambda k.\lambda S.(i^{-1}k)S(Sl_1)$,
$i^{-1}(m_2(\rho'_2)) \in \{\bot, \lambda k.\lambda S.(i^{-1}k)S(Sl_2)\}$, $i^{-1}(m_2(\rho_2)) = \lambda k.\lambda S.(i^{-1}k)S(Sl_2)$.

Let $p'' \blacktriangleright p', (pk'') \in p''^{\mathbf{K}}, (pks'') \in (pk'')^{\mathbf{S}}, (k'_1, k'_2 \parallel k_1, k_2, (x : \tau[\overline{\sigma_j/\alpha_j}])^\top, (pk'')) \in \nabla_K, (S'_1, S'_2 \parallel S_1, S_2, (pks'')) \in \nabla_S$.

Since $(l_1, l_2, \tau[\overline{\sigma_j/\alpha_j}]) \in Z^{p'}$ also $(l_1, l_2, \tau[\overline{\sigma_j/\alpha_j}]) \in Z^{p''}$, then $(S_1', S_2' \parallel S_1, S_2, (pks'')) \in \nabla_S \Rightarrow$ $(S_1' = \bot \wedge S_2' = \bot) \vee (S_1' l_1, S_2' l_2 \parallel S_1 l_1, S_2 l_2, \tau[\overline{\sigma_j/\alpha_j}], p'') \in \nabla_V$.
If $(S_1' = \bot \wedge S_2' = \bot)$ then $(m_1(\rho_1'))k_1' S_1' = (m_2(\rho_2'))k_2' S_2' = \bot$.
Else it follows that since continuations states and values are correspondingly related
$(m_1(\rho_1'))k_1' S_1' \sqsubseteq k_1' S_1'(S_1' l_1) = \top \Rightarrow k_2 S_2(S_2 l_2) = (m_2(\rho_2))k_2 S_2 = \top \wedge$
$(m_2(\rho_2'))k_2' S_2' \sqsubseteq k_2' S_2'(S_2' l_2) = \top \Rightarrow k_1 S_1(\underline{S_1 l_1}) = (m_1(\rho_1))k_1 S_1 = \top$
Hence $(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T\tau[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$.
We conclude $(m_1, m_2, T\tau, p) \in \nabla_M^{\Xi\Gamma}$.

alloc: $$\frac{\Delta; \Xi; \Gamma \vdash V : \tau}{\Delta; \Xi; \Gamma \vdash ref\ V : T(\tau\ ref)}$$

Let $v_1 = [\![\Delta; \Xi; \Gamma \vdash V_1 : \tau]\!]$ and $v_2 = [\![\Delta; \Xi; \Gamma \vdash V_2 : \tau]\!]$. Assume $(v_1, v_2, \tau, p) \in \nabla_V^{\Xi\Gamma}$.
Let $m_1 = [\![\Delta; \Xi; \Gamma \vdash ref\ V_1 : T(\tau\ ref)]\!]$ and $m_2 = [\![\Delta; \Xi; \Gamma \vdash ref\ V_2 : T(\tau\ ref)]\!]$.
We need to show $(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T(\tau[\overline{\sigma_j/\alpha_j}]\ ref), p') \in \nabla_M$.

If $(m_1(\rho_1')) = \bot \wedge (m_2(\rho_2')) = \bot$ then $(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T(\tau[\overline{\sigma_j/\alpha_j}]\ ref), p') \in \nabla_M$. Else $\rho_1' = \bot \Rightarrow (m_1(\rho_1')) = \bot$, $\rho_1 = \bot \Rightarrow (m_1(\rho_1)) = \bot$, $\rho_2' = \bot \Rightarrow (m_1(\rho_2')) = \bot$, $\rho_2 = \bot \Rightarrow (m_1(\rho_2)) = \bot$.

$\rho_1' \neq \bot \Rightarrow i^{-1}(m_1(\rho_1')) = \lambda k.\lambda S.(i^{-1}k)(S[l_1' \mapsto (v_1(\rho_1'))])(i \circ in_{\mathbb{L}} l_1')$
for some/any $l_1' \notin supp(\lambda l'.(i^{-1}k)(S[l' \mapsto (v_1(\rho_1'))](i \circ in_{\mathbb{L}} l'))$.
$\rho_1 \neq \bot \Rightarrow i^{-1}(m_1(\rho_1)) = i^{-1}(m_1(\rho_1)) = \lambda k.\lambda S.(i^{-1}k)(S[l_1 \mapsto (v_1(\rho_1))])(i \circ in_{\mathbb{L}} l_1)$
for some/any $l_1 \notin supp(\lambda l'.(i^{-1}k)(S[l' \mapsto (v_1(\rho_1))](i \circ in_{\mathbb{L}} l'))$.
$\rho_2' \neq \bot \Rightarrow i^{-1}(m_2(\rho_2')) = i^{-1}(m_2(\rho_2')) = \lambda k.\lambda S.(i^{-1}k)(S[l_2' \mapsto (v_2(\rho_2'))])(i \circ in_{\mathbb{L}} l_2')$
for some/any $l_2' \notin supp(\lambda l'.(i^{-1}k)(S[l' \mapsto (v_2(\rho_2'))](i \circ in_{\mathbb{L}} l'))$.
$\rho_2 \neq \bot \Rightarrow i^{-1}(m_2(\rho_2)) = i^{-1}(m_2(\rho_2)) = \lambda k.\lambda S.(i^{-1}k)(S[l_2 \mapsto (v_2(\rho_2))])(i \circ in_{\mathbb{L}} l_2)$
for some/any $l_2 \notin supp(\lambda l'.(i^{-1}k)(S[l' \mapsto (v_2(\rho_2))](i \circ in_{\mathbb{L}} l'))$.

Let arbitrarily $p'' \blacktriangleright p'$, $(pk'') \in p''^{\mathbf{K}}$, $(pks'') \in pk''^{\mathbf{S}}$, $(k_1', k_2' \parallel k_1, k_2, (x : \tau[\overline{\sigma_j/\alpha_j}]\ ref)^\top, (pk'')) \in \nabla_K$, $(S_1', S_2' \parallel S_1, S_2, (pks'')) \in \nabla_S$.

If $\rho_1' = \rho_2' = \bot$ or $S_1' = S_2' = \bot$ then $(m_1(\rho_1'))k_1' S_1' = (m_1(\rho_2'))k_2' S_2' = \bot$. Else
Let $l_0 \notin (\mathring{A}_{p''2}(S_1) \cup \mathring{A}_{p''2}(S_2) \cup$
$supp(\lambda l'.(i^{-1}k)(S[l' \mapsto (v_1(\rho_1'))(i \circ in_{\mathbb{L}} l'))\ \cup\ supp(\lambda l'.(i^{-1}k)(S[l' \mapsto (v_1(\rho_1))(i \circ in_{\mathbb{L}} l')) \cup$
$supp(\lambda l'.(i^{-1}k)(S[l' \mapsto (v_2(\rho_2'))(i \circ in_{\mathbb{L}} l')))\ \cup\ supp(\lambda l'.(i^{-1}k)(S[l' \mapsto (v_2(\rho_2))(i \circ in_{\mathbb{L}} l')))$.
Then
$i^{-1}(m_1(\rho_1'))k_1' S_1' \sqsubseteq (i^{-1}k_1')(S_1'[l_0 \mapsto (v_1(\rho_1'))])(i \circ in_{\mathbb{L}} l_0)$,
$i^{-1}(m_1(\rho_1))k_1 S_1 = (i^{-1}k_1)(S_1[l_0 \mapsto (v_1(\rho_1))])(i \circ in_{\mathbb{L}} l_0)$,
$i^{-1}(m_2(\rho_2'))k_2' S_2' \sqsubseteq (i^{-1}k_2')(S_2'[l_0 \mapsto (v_2(\rho_2'))])(i \circ in_{\mathbb{L}} l_0)$,
$i^{-1}(m_2(\rho_2))k_2 S_2 = (i^{-1}k_2)(S_2[l_0 \mapsto (v_2(\rho_2))])(i \circ in_{\mathbb{L}} l_0)$.

We define a parameter (pks^3) that extends (pks'') by adding $(l_0, l_0, \tau[\overline{\sigma_j/\alpha_j}])$ to the visible locations. Let $r = ((T, \emptyset_{LL}), A_\emptyset, A_\emptyset, \{(l_0, l_0, \tau[\overline{\sigma_j/\alpha_j}])\})$.
Let $(pks^3) = (pks'') \uplus \{(r|r|(T, \emptyset_{LL}))\}$ so $(pk^3) = (pks^3)^k = (pk)'' \uplus \{(r|r)\}$, $p^3 = (pks^3)^{vm} = p'' \uplus \{r\}$ and $p^3 \rhd p''$, $(pk^3) \rhd (pk'')$.
Then $Z^{pks^3} = Z^{pk^3} = Z^{pks''} \uplus \{(l_0, l_0, \tau[\overline{\sigma_j/\alpha_j}])\}$, $Z^{p^3} = Z^{p''} \uplus \{(l_0, l_0, \tau[\overline{\sigma_j/\alpha_j}])\}$

It holds that $(i \circ in_{\mathbb{L}} l_0, i \circ in_{\mathbb{L}} l_0, i \circ in_{\mathbb{L}} l_0, i \circ in_{\mathbb{L}} l_0, \tau[\overline{\sigma_j/\alpha_j}]\ ref, p^3) \in \nabla_V$. To prove the required termination approximation we want to show
$(S_1'[l_0 \mapsto v_1(\rho_1')], S_2'[l_0 \mapsto v_2(\rho_2')] \parallel S_1[l_0 \mapsto v_1(\rho_1)], S_2[l_0 \mapsto v_2(\rho_2)], p^3) \in \nabla_S$.

By assumption $(S_1', S_2' \parallel S_1, S_2, (pks'')) \in \nabla_S$. The states have only been changed outside the areas for the most consuming accessibility maps $\mathring{A}_{p''1}, \mathring{A}_{p''2}$ involved in p''. Hence the updated states $(S_1[l_0 \mapsto (v_1(\rho_1))],\ S_2[l_0 \mapsto (v_2(\rho_2))])$ still belong to the same simple state predicates and relations involved in p''. The associated location lists hold values related in p'' by weakening these are also related in $p^3 \rhd p''$. $Z^{pks^3} = Z^{pks''} \uplus \{(l_0, l_0, \tau)\}$. Since the original states were related and by weakening the locations in $Z^{pks''}$ hold values related in p^3. By assumption and weakening also $(v_1(\rho_1'), v_2(\rho_2') \parallel v_1(\rho_1), v_2(\rho_2), \tau[\overline{\sigma_j/\alpha_j}], p^3) \in \nabla_V$.

In more detail we have that
$(pks^3) = \{(r_1|\hat{q}_1|Q_1), \ldots, (r_n|\hat{q}_n|Q_n), (r_{n+1}|\hat{q}_{n+1}|Q_{n+1})\}$ where $r_{n+1} = r$ with trivial instantiations, and so relatedness of the updated states in pks^3 requires

1. $S_1'[l_0 \mapsto v_1(\rho_1')] \sqsubseteq S_1[l_0 \mapsto v_1(\rho_1)] \neq \bot \ \wedge\ S_2'[l_0 \mapsto v_2(\rho_2')] \sqsubseteq S_2[l_0 \mapsto v_2(\rho_2)] \neq \bot$
2. $\mathcal{A}_1^{pks^3}(S_1[l_0 \mapsto v_1(\rho_1)]) \cap \pi_1(Z^{pks^3}) = \emptyset \ \wedge\ \mathcal{A}_2^{pks^3}(S_2[l_0 \mapsto v_2(\rho_2)]) \cap \pi_2(Z^{pks^3}) = \emptyset$
3. $\forall i \neq j \in 1\ldots$ n+1. $\mathring{A}_1^{ri}(S_1[l_0 \mapsto v_1(\rho_1)]) \cap \mathring{A}_1^{rj}(S_1[l_0 \mapsto v_1(\rho_1)]) = \emptyset \ \wedge\ \mathring{A}_2^{ri}(S_2[l_0 \mapsto v_2(\rho_2)]) \cap \mathring{A}_2^{rj}(S_2[l_0 \mapsto v_2(\rho_2)]) = \emptyset$
4. $\forall (l_1, l_2, \sigma) \in Z^{pks^3}.(S_1'[l_0 \mapsto v_1(\rho_1')](l_1),\ S_2'[l_0 \mapsto v_2(\rho_2')](l_2) \parallel S_1[l_0 \mapsto v_1(\rho_1)](l_1),\ S_2[l_0 \mapsto v_2(\rho_2)](l_2),\ \sigma,\ p^3) \in \nabla_V$
5. $\forall i \in 1..$n+1. if $Q_{ij_i} = (P_i, LL_i)$ then $(S_1[l_0 \mapsto v_1(\rho_1)], S_2[l_0 \mapsto v_2(\rho_2)]) \in P_i \ \wedge$
   $\forall (l_1, l_2, \sigma) \in LL_i.\ (S_1'[l_0 \mapsto v_1(\rho_1')](l_1),\ S_2'[l_0 \mapsto v_2(\rho_2')](l_2) \parallel S_1[l_0 \mapsto v_1(\rho_1)](l_1),\ S_2[l_0 \mapsto v_2(\rho_2)](l_2),\ \sigma,\ p^3) \in \nabla_V$

1. Follows from: assumptions $S_1' \sqsubseteq S_1 \neq \bot \ \wedge\ S_2' \sqsubseteq S_2 \neq \bot$ and $\rho_1' \sqsubseteq \rho_1 \ \wedge\ \rho_2' \sqsubseteq \rho_2$
2. Follows from: $r = ((T, \emptyset_{LL}), A_\emptyset, A_\emptyset, \{(l_0, l_0, \tau)\})$, so
   $\pi_1(Z^{pks^3}) = \pi_1(Z^{pks''}) \uplus \{l_0\}$, $\mathcal{A}_1^{pks^3}(S_1[l_0 \mapsto v_1(\rho_1)]) = \mathcal{A}_1^{pks''}(S_1),\ l_0 \notin \mathcal{A}_1^{pks''}(S_1)$ and
   $\pi_2(Z^{pks^3}) = \pi_2(Z^{pks''}) \uplus \{l_0\}$, $\mathcal{A}_2^{pks^3}(S_2[l_0 \mapsto v_2(\rho_2)]) = \mathcal{A}_2^{pks''}(S_2),\ l_0 \notin \mathcal{A}_2^{pks''}(S_2)$
3. Follows from: $\mathring{A}_1^{r_{n+1}}(S_1[l_0 \mapsto v_1(\rho_1)]) = \mathring{A}_2^{r_{n+1}}(S_2[l_0 \mapsto v_2(\rho_2)]) = \{l_0\}$ and
   $\forall i \neq j \in 1\ldots n.\ \mathring{A}_1^{ri}(S_1) \cap \mathring{A}_1^{rj}(S_1) = \emptyset \ \wedge\ \mathring{A}_2^{ri}(S_2) \cap \mathring{A}_2^{rj}(S_2) = \emptyset$ and
   $\forall i \in 1\ldots n.\ \mathring{A}_1^{ri}(S_1[l_0 \mapsto v_1(\rho_1)]) = \mathring{A}_1^{ri}(S_1), l_0 \notin \mathring{A}_1^{ri}(S_1) \ \wedge\ \mathring{A}_2^{ri}(S_2[l_0 \mapsto v_2(\rho_2)]) = \mathring{A}_2^{ri}(S_2), l_0 \notin \mathring{A}_2^{ri}(S_2)$
4. Follows from: $Z^{pks^3} = Z^{pks''} \uplus \{(l_0, l_0, \tau[\overline{\sigma_j/\alpha_j}])\}, \forall (l_1, l_2, \sigma) \in Z^{pks''}.(S_1'(l_1),\ S_2'(l_2) \parallel S_1(l_1),\ S_2(l_2),\ \sigma,\ p'') \in \nabla_V$, weakening and the assumption $(v_1, v_2, \tau, p) \in \nabla_V^{\Xi\Gamma}$.
5. Follows from: Simple state relations and predicates only depend on the areas given by the accessibility maps and $Q_{n+1} = (T, \emptyset_{LL})$ and $\forall i \in 1..n.\ l_0 \notin A_1^{q_i}(S_1), l_0 \notin A_2^{q_i}(S_2)$ and if $Q_i = (P_i, LL_i)$ then $(S_1, S_2) \in P_i \wedge \forall (l_1, l_2, \sigma) \in LL_i.\ (S_1'(l_1), S_2'(l_2) \parallel S_1(l_1), S_2(l_2), \sigma, p^{vm}) \in \nabla_V$.

We have
$(S_1'[l_0 \mapsto (v_1(\rho_1'))]), S_1[l_0 \mapsto (v_1(\rho_1))]), S_2'[l_0 \mapsto (v_2(\rho_2'))]), S_2[l_0 \mapsto (v_2(\rho_2))]), pks^3) \in \nabla_S$.
It then follows that $(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T(\tau[\overline{\sigma_j/\alpha_j}]\ ref), p') \in \nabla_M$ and so $(m_1, m_2, T(\tau\ ref), p) \in \nabla_M^{\Xi\Gamma}$.

assign: $$\frac{\Delta; \Xi; \Gamma \vdash V_a : \tau\ ref \qquad \Delta; \Xi; \Gamma \vdash V_b : \tau}{\Delta; \Xi; \Gamma \vdash V_a := V_b : Tunit}$$

Let $v_{1a} = [\![\Delta; \Xi; \Gamma \vdash V_{1a} : \tau\ ref]\!]$, $v_{2a} = [\![\Delta; \Xi; \Gamma \vdash V_{2a} : \tau\ ref]\!]$,
$v_{1b} = [\![\Delta; \Xi; \Gamma \vdash V_{1b} : \tau]\!]$, $v_{2b} = [\![\Delta; \Xi; \Gamma \vdash V_{2b} : \tau]\!]$.

Assume $(v_{1a}, v_{2a}, \tau\ ref, p) \in \nabla_V^{\Xi\Gamma}$ and $(v_{1b}, v_{2b}, \tau, p) \in \nabla_V^{\Xi\Gamma}$. So
$(v_{1a}(\rho_1'), v_{2a}(\rho_2') \parallel v_{1a}(\rho_1), v_{2a}(\rho_2), \tau[\overline{\sigma_j/\alpha_j}]\ ref, p') \in \nabla_V$ and
$(v_{1b}(\rho_1'), v_{2b}(\rho_2') \parallel v_{1b}(\rho_1), v_{2b}(\rho_2), \tau[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$.

The assumption implies that either $v_{1a}(\rho_1') = v_{2a}(\rho_2') = \bot$ or $\exists(l_1, l_2, \tau[\overline{\sigma_j/\alpha_j}]) \in Z^{p'}$.
$i^{-1}(v_{1a}(\rho_1')) \sqsubseteq i^{-1}(v_{1a}(\rho_1)) = in_{\mathbb{L}} l_1,\ i^{-1}(v_{2a}(\rho_2')) \sqsubseteq i^{-1}(v_{2a}(\rho_2)) = in_{\mathbb{L}} l_2$.
Let $m_1 = [\![\Delta; \Xi; \Gamma \vdash V_{1a} := V_{1b} : Tunit]\!]$ and $m_2 = [\![\Delta; \Xi; \Gamma \vdash V_{2a} := V_{2b} : Tunit]\!]$.
If $v_{1a}(\rho_1') = v_{2a}(\rho_2') = \bot$ then $m_1(\rho_1') = m_2(\rho_2') = \bot$ and we have
$(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), Tunit, p') \in \nabla_M$.
Else
$i^{-1}(m_1(\rho_1')) \sqsubseteq \lambda k.\lambda S.(i^{-1}k)(S[l_1 \mapsto (v_{1b}(\rho_1'))])i(in_{\mathbf{1}}*)$,
$i^{-1}(m_1(\rho_1)) = \lambda k.\lambda S.(i^{-1}k)(S[l_1 \mapsto (v_{1b}(\rho_1))])i(in_{\mathbf{1}}*)$,
$i^{-1}(m_2(\rho_2')) \sqsubseteq \lambda k.\lambda S.(i^{-1}k)(S[l_2 \mapsto (v_{2b}(\rho_2'))])i(in_{\mathbf{1}}*)$,
$i^{-1}(m_2(\rho_2)) = \lambda k.\lambda S.(i^{-1}k)(S[l_2 \mapsto (v_{2b}(\rho_2))])i(in_{\mathbf{1}}*)$.

We want to show $(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), Tunit, p') \in \nabla_M$.
Let $p'' \blacktriangleright p', (pk'') \in p''^{\mathbf{K}}, (pks'') \in (pk'')^{\mathbf{S}}$.
$(k_1', k_2' \parallel k_1, k_2, (x : unit), (pk'')) \in \nabla_K,\ (S_1', S_2' \parallel S_1, S_2, (pks'')) \in \nabla_S$. We will show

$$\begin{aligned}
&(m_1(\rho_1'))k_1'S_1' \sqsubseteq (k_1')(S_1'[l_1 \mapsto (v_{1b}(\rho_1'))])i(in_{\mathbf{1}}*) = \top \Rightarrow\\
&\qquad (m_2(\rho_2))k_2S_2 = (k_2)(S_2[l_2 \mapsto (v_{2b}(\rho_2))])i(in_{\mathbf{1}}*) = \top \wedge\\
&(m_2(\rho_2'))k_2'S_2' \sqsubseteq (k_2')(S_2'[l_2 \mapsto (v_{2b}(\rho_2'))])i(in_{\mathbf{1}}*) = \top \Rightarrow\\
&\qquad (m_1(\rho_1))k_1S_1 = (k_1)(S_1[l_1 \mapsto (v_{1b}(\rho_1))])i(in_{\mathbf{1}}*) = \top
\end{aligned}$$

By assumption $(v_{1b}(\rho_1'), v_{2b}(\rho_2') \parallel v_{1b}(\rho_1), v_{2b}(\rho_2), \tau[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$ then by parameter weakening also $(v_{1b}(\rho_1'), v_{2b}(\rho_2') \parallel v_{1b}(\rho_1), v_{2b}(\rho_2), \tau[\overline{\sigma_j/\alpha_j}], p'') \in \nabla_V$.
Since $p'' \blacktriangleright p'$ then $Z^{p''} \supseteq Z^{p'}$. So $(l_1, l_2, \tau[\overline{\sigma_j/\alpha_j}]) \in Z^{p'}$ implies $(l_1, l_2, \tau[\overline{\sigma_j/\alpha_j}]) \in Z^{p''}$.
And since $(S_1', S_2' \parallel S_1, S_2, (pks'')) \in \nabla_S$ then $\forall(l_a, l_b, \sigma) \in Z^{(pks'')}$. $(S_1'[l_1 \mapsto (v_{1b}(\rho_1'))]l_a$, $S_2'[l_1 \mapsto (v_{2b}(\rho_2'))]l_b \parallel S_1[l_1 \mapsto (v_{1b}(\rho_1))]l_a, S_2[l_1 \mapsto (v_{2b}(\rho_2))]l_b, \sigma, p'') \in \nabla_V$.
The states S_1, S_2 have only been changed outside the areas for the accessibility maps involved in the hidden areas of local parameters in p''. Hence the updated states $S_1[l_1 \mapsto (v_{1b}(\rho_1))],\ S_2[l_2 \mapsto (v_{2b}(\rho_2))]$ still belong to the same simple state relations involved there. The associated location lists hold values related in p''.
$A_1^{p''}(S_1) \cap Z_1^{p''} = \emptyset \wedge l_1 \in \pi_1(Z^{p''}) \Rightarrow A_1^{p''}(S_1) = A_1^{p''}(S_1[l_1 \mapsto (v_{1b}(\rho_1))])$, and
$A_2^{p''}(S_2) \cap \pi_2(Z^{p''}) = \emptyset \wedge l_2 \in \pi_2(Z^{p''}) \Rightarrow A_2^{p''}(S_2) = A_2^{p''}(S_2[l_2 \mapsto (v_{2b}(\rho_2))])$. Also $\forall l \in A_1^{p''}(S_1)$. $(S_1)l = (S_1[l_1 \mapsto (v_{1b}(\rho_1))])l$ and $\forall l \in A_2^{p''}(S_2)$. $(S_2)l = (S_2[l_2 \mapsto (v_{2b}(\rho_2))])l$.
$(l_1, l_2, \tau[\overline{\sigma_j/\alpha_j}]) \in Z_{p'} \subseteq Z_{p''}$.

So $(S_1'[l_1 \mapsto (v_{1b}(\rho_1'))], S_2'[l_2 \mapsto (v_{2b}(\rho_2'))] \parallel S_1[l_1 \mapsto (v_{1b}(\rho_1))], S_2[l_2 \mapsto (v_{2b}(\rho_2))], p'') \in \nabla_S$.
Also $(i(in_{\mathbf{1}}*), i(in_{\mathbf{1}}*), i(in_{\mathbf{1}}*), i(in_{\mathbf{1}}*), unit, p'') \in \nabla_V$.
Now continuations, updated states and $(in_{\mathbf{1}}*)$-values are correspondingly related, so we get the required termination properties. Then $(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), Tunit, p') \in \nabla_M$.
We conclude $(m_1, m_2, Tunit, p) \in \nabla_M^{\Xi\Gamma}$.

rec: $$\frac{\Delta; \Xi; \Gamma, f : \tau \to T\tau', x : \tau \vdash M : T\tau'}{\Delta; \Xi; \Gamma \vdash rec\ f(x : \tau) = M : \tau \to T\tau'}$$

Let $m_1 = [\![\Delta; \Xi; \Gamma, f : \tau \to T\tau', x : \tau \vdash M_1 : T\tau']\!]$, and
let $m_2 = [\![\Delta; \Xi; \Gamma, f : \tau \to T\tau', x : \tau \vdash M_2 : T\tau']\!]$,
Assume $(m_1, m_2, T\tau', p) \in \nabla_M^{\Xi\Gamma_0}$, where $\Gamma_0 = \Gamma \cup \{\ f : \tau \to T\tau', x : \tau\ \}$.

Let $v_1 = [\![\Delta; \Xi; \Gamma \vdash rec\ f(x : \tau) = M_1 : \tau \to T\tau']\!]$ and

let $v_2 = \llbracket \Delta; \Xi; \Gamma \vdash rec\ f(x:\tau) = M_2 : \tau \to T\tau' \rrbracket$.
We aim to prove $(v_1, v_2, \tau \to T\tau', p) \in \nabla_V^{\Xi\Gamma}$, that is
$(v_1(\rho_1'), v_2(\rho_2') \parallel v_1(\rho_1), v_2(\rho_2), (\tau \to T\tau')[\overline{\sigma_j/\alpha_j}], p') =$
$(v_1(\rho_1'), v_2(\rho_2') \parallel v_1(\rho_1), v_2(\rho_2), (\tau[\overline{\sigma_j/\alpha_j}]) \to (T\tau'[\overline{\sigma_j/\alpha_j}]), p') \in \nabla_V$.

If $\rho_1' = \rho_2' = \bot$ then $v_1(\rho_1') = v_2(\rho_2') = \bot$.
When $\rho \neq \bot$ we have $\llbracket \Delta; \Xi; \Gamma \vdash rec\ f(x:\tau) = \ M : \tau \to T\tau' \ \rrbracket\rho =$
$i \circ in_{\multimap} \lfloor fix(\lambda f' \in (\mathbb{V} \multimap \mathbb{M}).(\lambda x' \in \mathbb{V}.\llbracket \Delta; \Xi; \Gamma, f : \tau \to T\tau', x : \tau \vdash M : T\tau' \rrbracket(\rho \otimes f \mapsto i \circ in_{\multimap} \lfloor f' \rfloor \otimes x \mapsto x')))\rfloor = i \circ in_{\multimap} \lfloor \bigsqcup_{n\in\omega} g_n \rfloor$
where
$g_n \in (\mathbb{V} \multimap \mathbb{M})$, $g_0 = \bot_{\mathbb{V} \multimap \mathbb{M}}$ and
$g_{n+1} = \ \lambda x_0 \in \mathbb{V}.\llbracket \Delta; \Xi; \Gamma, f : \tau \to T\tau', x : \tau \vdash M : T\tau' \rrbracket(\rho \otimes f \mapsto i \circ in_{\multimap} \lfloor g_n \rfloor \otimes x \mapsto x_0)$.
so we have that when $\rho_1' \neq \bot \vee \rho_2' \neq \bot$ then
$v_1(\rho_1) = i \circ in_{\multimap} \lfloor \bigsqcup_{n\in\omega} g_n^1 \rfloor$, $v_2(\rho_2) = i \circ in_{\multimap} \lfloor \bigsqcup_{n\in\omega} g_n^2 \rfloor$.
$v_1(\rho_1') \sqsubseteq i \circ in_{\multimap} \lfloor \bigsqcup_{n\in\omega} g_n^{1'} \rfloor$, $v_2(\rho_2') \sqsubseteq i \circ in_{\multimap} \lfloor \bigsqcup_{n\in\omega} g_n^{2'} \rfloor$
where
$g_0^1 = g_0^2 = g_0^{1'} = g_0^{2'} = \bot_{\mathbb{V} \multimap \mathbb{M}}$ and
$g_{n+1}^1 = \lambda x_0.m_1(\rho_1 \otimes i \circ in_{\multimap} \lfloor g_n^1 \rfloor \otimes x_0)$,
$g_{n+1}^2 = \lambda x_0.m_1(\rho_2 \otimes i \circ in_{\multimap} \lfloor g_n^2 \rfloor \otimes x_0)$,
$g_{n+1}^{1'} = \lambda x_0.m_1(\rho_1' \otimes i \circ in_{\multimap} \lfloor g_n^{1'} \rfloor \otimes x_0)$,
$g_{n+1}^{2'} = \lambda x_0.m_1(\rho_2' \otimes i \circ in_{\multimap} \lfloor g_n^{2'} \rfloor \otimes x_0)$
It holds that
$\bigsqcup_{n\in\omega} g_n^1 = (\lambda x' \in \mathbb{V}.m_1(\rho_1 \otimes i \circ in_{\multimap} \lfloor (\bigsqcup_{n\in\omega} g_n^1) \rfloor \otimes x'))$ and
$\bigsqcup_{n\in\omega} g_n^2 = (\lambda x' \in \mathbb{V}.m_2(\rho_2 \otimes i \circ in_{\multimap} \lfloor (\bigsqcup_{n\in\omega} g_n^2) \rfloor \otimes x'))$

We have $g_0^{1'} \sqsubseteq g_0^1 \sqsubseteq \bigsqcup g_n^1 \wedge g_0^{2'} \sqsubseteq g_0^2 \sqsubseteq \bigsqcup g_n^2$ and from the definition and the assumptions $\rho_1' \sqsubseteq \rho_1 \wedge \rho_2' \sqsubseteq \rho_2$ then by induction $\forall n. g_n^{1'} \sqsubseteq g_n^1 \sqsubseteq \bigsqcup g_n^1 \wedge g_n^{2'} \sqsubseteq g_n^2 \sqsubseteq \bigsqcup g_n^2$.

Also $\forall d_1, d_2.\ \bot_{\mathbb{V} \multimap \mathbb{M}}(d_1) = \bot_{\mathbb{V} \multimap \mathbb{M}}(d_2) = \bot_{\mathbb{M}}$.
Since $\forall q \in \mathfrak{p}^{vm}.\forall d_1, d_2 \in \mathbb{V}.\ (\bot, \bot \parallel \bigsqcup g_n^1(d_1), \bigsqcup g_n^1(d_2), T\tau'[\overline{\sigma_j/\alpha_j}], q) \in F(\nabla, \nabla)_M$
so $(i \circ in_{\multimap} \lfloor g_0^1 \rfloor, i \circ in_{\multimap} \lfloor g_0^2 \rfloor \parallel v_1(\rho_1), v_2(\rho_2), (\tau \to T\tau')[\overline{\sigma_j/\alpha_j}], p) \in F(\nabla, \nabla)_V$, and also for p'

We will show by induction on n that
$\forall n \in \omega.\ (i \circ in_{\multimap} \lfloor g_n^{1'} \rfloor, i \circ in_{\multimap} \lfloor g_n^{2'} \rfloor \parallel v_1(\rho_1), v_2(\rho_2), (\tau \to T\tau')[\overline{\sigma_j/\alpha_j}], p') \in F(\nabla, \nabla)_V$.

Assume $(i \circ in_{\multimap} \lfloor g_n^{1'} \rfloor, i \circ in_{\multimap} \lfloor g_n^{2'} \rfloor \parallel v_1(\rho_1), v_2(\rho_2), (\tau \to T\tau')[\overline{\sigma_j/\alpha_j}], p') \in F(\nabla, \nabla)_V$.
Recall $(i \circ in_{\multimap} \lfloor g_n^{1'} \rfloor, i \circ in_{\multimap} \lfloor g_n^{2'} \rfloor \parallel v_1(\rho_1), v_2(\rho_2), (\tau \to T\tau')[\overline{\sigma_j/\alpha_j}], p') =$
$(i \circ in_{\multimap} \lfloor g_n^{1'} \rfloor, i \circ in_{\multimap} \lfloor g_n^{2'} \rfloor \parallel i \circ in_{\multimap} \lfloor \bigsqcup_{n\in\omega} g_n^1 \rfloor, i \circ in_{\multimap} \lfloor \bigsqcup_{n\in\omega} g_n^2 \rfloor, (\tau \to T\tau')[\overline{\sigma_j/\alpha_j}], p')$.

To show that $(i \circ in_{\multimap} \lfloor g_{n+1}^{1'} \rfloor, i \circ in_{\multimap} \lfloor g_{n+1}^{2'} \rfloor \parallel v_1(\rho_1), v_2(\rho_2), (\tau \to T\tau')[\overline{\sigma_j/\alpha_j}], p') \in F(\nabla, \nabla)_V$
we must have $\forall p'' \blacktriangleright p'.\forall (d_1', d_2' \parallel d_1, d_2, \tau[\overline{\sigma_j/\alpha_j}], p'') \in \nabla_V$.

$(\lambda x_0.m_1(\rho_1' \otimes i \circ in_{\multimap} \lfloor g_n^{1'} \rfloor \otimes x_0)d_1',\ \lambda x_0.m_2(\rho_{2'} \otimes i \circ in_{\multimap} \lfloor g_n^{2'} \rfloor \otimes x_0)d_2' \parallel \bigsqcup g_n^1(d_1),\ \bigsqcup g_n^2(d_2),\ T\tau'[\overline{\sigma_j/\alpha_j}],\ p'') =$
$(m_1(\rho_1' \otimes i \circ in_{\multimap} \lfloor g_n^{1'} \rfloor \otimes d_1'),\ m_2(\rho_2' \otimes i \circ in_{\multimap} \lfloor g_n^{2'} \rfloor \otimes d_2') \parallel m_1(\rho_1 \otimes i \circ in_{\multimap} \lfloor (\bigsqcup_{n\in\omega} g_n^1) \rfloor \otimes d_1, m_2(\rho_2 \otimes$
$i \circ in_{\multimap} \lfloor (\bigsqcup_{n\in\omega} g_n^2) \rfloor \otimes d_2, T\tau'[\overline{\sigma_j/\alpha_j}],\ p'') \in \nabla_M$

Since by assumption $(m_1, m_2, T\tau'[\overline{\sigma_j/\alpha_j}], p) \in \nabla_M^{\Xi\Gamma_0}$ and we have
$(i \circ in_{\multimap} \lfloor g_n^{1'} \rfloor, i \circ in_{\multimap} \lfloor g_n^{2'} \rfloor \parallel i \circ in_{\multimap} \lfloor \bigsqcup_{n\in\omega} g_n^1 \rfloor, i \circ in_{\multimap} \lfloor \bigsqcup_{n\in\omega} g_n^2 \rfloor, (\tau \to T\tau')[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$
and $(d_1', d_2' \parallel d_1, d_2, \tau[\overline{\sigma_j/\alpha_j}], p'') \in \nabla_V$ it holds that
$(m_1(\rho_1' \otimes i \circ in_{\multimap} \lfloor g_n^{1'} \rfloor \otimes d_1'),\ m_2(\rho_2' \otimes i \circ in_{\multimap} \lfloor g_n^{2'} \rfloor \otimes d_2') \parallel m_1(\rho_1 \otimes i \circ in_{\multimap} \lfloor \bigsqcup_{n\in\omega} g_n^1 \rfloor \otimes d_1),\ m_2(\rho_2 \otimes$
$i \circ in_{\multimap} \lfloor \bigsqcup_{n\in\omega} g_n^2 \rfloor \otimes d_2),\ T\tau'[\overline{\sigma_j/\alpha_j}],\ p'') \in \nabla_M$. So

$(i \circ in_{\multimap}\lfloor g^{1'}_{n+1}\rfloor, i \circ in_{\multimap}\lfloor g^{2'}_{n+1}\rfloor \parallel i \circ in_{\multimap}\lfloor \bigsqcup_{n\in\omega} g^1_n\rfloor, i \circ in_{\multimap}\lfloor \bigsqcup_{n\in\omega} g^2_n\rfloor, (\tau \to T\tau')[\overline{\sigma_j/\alpha_j}], p') =$
$(i \circ in_{\multimap}\lfloor g^{1'}_{n+1}\rfloor, i \circ in_{\multimap}\lfloor g^{2'}_{n+1}\rfloor \parallel v_1(\rho_1), v_2(\rho_2), (\tau \to T\tau')[\overline{\sigma_j/\alpha_j}], p') \in F(\nabla, \nabla)_V$.

We have shown that $\forall n \in \omega.\ (i \circ in_{\multimap}\lfloor g^{1'}_n\rfloor, i \circ in_{\multimap}\lfloor g^{2'}_n\rfloor \parallel v_1(\rho_1), v_2(\rho_2), (\tau \to T\tau')[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$. Then since ∇_V is admissible also $(\bigsqcup i \circ in_{\multimap}\lfloor g^{1'}_n\rfloor, \bigsqcup i \circ in_{\multimap}\lfloor g^{2'}_n\rfloor \parallel v_1(\rho_1), v_2(\rho_2), (\tau \to T\tau')[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$. So $(v_1(\rho'_1), v_2(\rho'_2) \parallel v_1(\rho_1), v_2(\rho_2), (\tau \to T\tau')[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$, hence $(v_1, v_2, \tau \to T\tau', p) \in \nabla^{\Xi\Gamma}_V$.

app: $$\frac{\Delta;\Xi;\Gamma \vdash V_a : \tau \to T\tau' \qquad \Delta;\Xi;\Gamma \vdash V_b : \tau}{\Delta;\Xi;\Gamma \vdash V_a V_b : T\tau'}$$

Let $v_{1a} = [\![\Delta;\Xi;\Gamma \vdash V_{1a} : \tau \to T\tau']\!]$, $v_{2a} = [\![\Delta;\Xi;\Gamma \vdash V_{2a} : \tau \to T\tau']\!]$, $v_{1b} = [\![\Delta;\Xi;\Gamma \vdash V_{1b} : \tau]\!]$, $v_{2b} = [\![\Delta;\Xi;\Gamma \vdash V_{2b} : \tau]\!]$.
Assume $(v_{1a}, v_{2a}, \tau \multimap T\tau', p) \in \nabla^{\Xi\Gamma}_V$ and $(v_{1b}, v_{2b}, \tau, p) \in \nabla^{\Xi\Gamma}_V$.
Let $m_1 = [\![\Delta;\Gamma \vdash V_{1a}V_{1b}]\!]$ and $m_2 = [\![\Delta;\Gamma \vdash V_{2a}V_{2b}]\!]$. We aim to show $(m_1, m_2, T\tau', p) \in \nabla^{\Xi\Gamma}_M$, that is $(m_1(\rho'_1), m_2(\rho'_2) \parallel m_1(\rho_1), m_2(\rho_2), T\tau'[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$.

The assumption $(v_{1a}, v_{2a}, \tau \multimap T\tau', p) \in \nabla^{\Xi\Gamma}_V$ implies either $v_{1a}(\rho'_1) = v_{2a}(\rho'_2) = \bot$ or $\exists f'_1, f_1, f'_2, f_2$.
$i^{-1}(v_{1a}(\rho_1)) = in_{\multimap}\lfloor f_1\rfloor\ \wedge (i^{-1}(v_{1a}(\rho'_1)) = \bot \wedge f'_1 = \bot) \vee i^{-1}(v_{1a}(\rho'_1)) = in_{\multimap}\lfloor f'_1\rfloor\ \wedge$
$i^{-1}(v_{2a}(\rho_2)) = in_{\multimap}\lfloor f_2\rfloor\ \wedge (i^{-1}(v_{2a}(\rho'_2)) = \bot \wedge f'_2 = \bot) \vee i^{-1}(v_{2a}(\rho'_2)) = in_{\multimap}\lfloor f'_2\rfloor\ \wedge$
$\forall p'' \blacktriangleright p'. \forall (d'_1, d'_2 \parallel d_1, d_2, \tau[\overline{\sigma_j/\alpha_j}], p'') \in \nabla_V. (f'_1 d'_1, f'_2 d'_2 \parallel f_1 d_1, f_2 d_2, T\tau'[\overline{\sigma_j/\alpha_j}], p'') \in \nabla_M$.
In the first case $m_1(\rho'_1) = m_2(\rho'_2) = \bot$, and $(m_1(\rho'_1), m_2(\rho'_2) \parallel m_1(\rho_1), m_2(\rho_2), T\tau'[\overline{\sigma_j/\alpha_j}], p) \in \nabla_M$. In the second case we get that since $(v_{1b}(\rho'_1), v_{2b}(\rho'_2) \parallel v_{1b}(\rho_1), v_{2b}(\rho_2), \tau[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$ then $(f'_1(v_{1b}(\rho'_1)), f'_2(v_{2b}(\rho'_2)) \parallel f_1(v_{1b}(\rho_1)), f_2(v_{2b}(\rho_2)), T\tau'[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$. And we have $m_1(\rho'_1) = f'_1(v_{1b}(\rho'_1))$, $m_2(\rho'_2) = f'_2(v_{2b}(\rho'_2))$, $m_1(\rho_1) = f_1(v_{1b}(\rho_1))$, $m_2(\rho_2) = f_2(v_{0b}(\rho_2))$ so $(m_1(\rho'_1), m_2(\rho'_2) \parallel m_1(\rho_1), m_2(\rho_2), T\tau'[\overline{\sigma_j/\alpha_j}], p) \in \nabla_M$.
We conclude $(m_1, m_2, T\tau', p) \in \nabla^{\Xi\Gamma}_M$

let: $$\frac{\Delta;\Xi;\Gamma \vdash M_a : T\tau_a \qquad \Delta;\Xi;\Gamma, x : \tau_a \vdash M_b : T\tau_b}{\Delta;\Xi;\Gamma \vdash let\ x \Leftarrow M_a\ in\ M_b : T\tau_b}$$

Let $m_{1a} = [\![\Delta;\Xi;\Gamma \vdash M_{1a} : T\tau_a]\!]$, $m_{2a} = [\![\Delta;\Xi;\Gamma \vdash M_{2a} : T\tau_a]\!]$, $m_{1b} = [\![\Delta;\Xi;\Gamma, x : \tau_a \vdash M_{1b} : T\tau_b]\!]$, $m_{2b} = [\![\Delta;\Xi;\Gamma, x : \tau_a \vdash M_{2b} : T\tau_b]\!]$.
Assume $(m_{1a}, m_{2a}, T\tau_a, p) \in \nabla^{\Xi\Gamma}_M$, and assume $(m_{1b}, m_{2b}, T\tau_b, p) \in \nabla^{\Xi(\Gamma, x:\tau_a)}_M$,
then $(m_{1a}(\rho'_1), m_{2a}(\rho'_2) \parallel m_{1a}(\rho_1), m_{2a}(\rho_2), T\tau_a[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$, and
for any $(v'_{1x}, v'_{2x} \parallel v_{1x}, v_{2x}, \tau_a[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$ it holds that
$(m_{1b}(\rho'_1 \otimes v'_{1x}), m_{2b}(\rho'_2 \otimes v'_{2x}) \parallel m_{1b}(\rho_1 \otimes v_{1x}), m_{2b}(\rho_2 \otimes v_{2x}), T\tau_b[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$.

Let $m_1 = [\![\Delta;\Xi;\Gamma \vdash let\ x \Leftarrow M_{1a}\ in\ M_{1b} : T\tau_b]\!]$ and
let $m_2 = [\![\Delta;\Xi;\Gamma \vdash let\ x \Leftarrow M_{2a}\ in\ M_{2b} : T\tau_b]\!]$.
We need to show $(m_1, m_2, T\tau_b, p) \in \nabla^{\Xi\Gamma}_M$ that is (in all cases)
$(m_1(\rho'_1), m_2(\rho'_2) \parallel m_1(\rho_1), m_2(\rho_2), T\tau_b[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$.
If $\rho'_1 = \rho'_2 = \bot$ then $m_1(\rho'_1) = m_2(\rho'_2) = \bot$ and we are done. Else we have
$i^{-1}(m_1(\rho'_1)) = \lambda k.\lambda S.\ (i^{-1}(m_{1a}(\rho'_1)))(\lambda S_0.\lambda d_x.(\ i^{-1}(m_{1b}(\rho'_1 \otimes d_x))kS_0)S$,
$i^{-1}(m_1(\rho_1)) = \lambda k.\lambda S.\ (i^{-1}(m_{1a}(\rho_1)))(\lambda S_0.\lambda d_x.(\ i^{-1}(m_{1b}(\rho_1 \otimes d_x))kS_0)S$,
$i^{-1}(m_2(\rho'_2)) = \lambda k.\lambda S.\ (i^{-1}(m_{2a}(\rho'_2)))(\lambda S_0.\lambda d_x.(\ i^{-1}(m_{2b}(\rho'_2 \otimes d_x))kS_0)S$,
$i^{-1}(m_2(\rho_2)) = \lambda k.\lambda S.\ (i^{-1}(m_{2a}(\rho_2)))(\lambda S_0.\lambda d_x.(\ i^{-1}(m_{2b}(\rho_2 \otimes d_x))kS_0)S$.

Let $p'' \blacktriangleright p'$, $(pk'') \in p''^{\mathbf{K}}$, $(pks'') \in pk''^{\mathbf{S}}$, $(k'_1, k'_2 \parallel k_1, k_2, (x : \tau_b[\overline{\sigma_j/\alpha_j}])^{\top}, (pk'')) \in \nabla_K$, $(S'_1, S'_2 \parallel S_1, S_2, (pks'')) \in \nabla_S$.

$(i^{-1}(m_1(\rho_1')))k_1'S_1' = (i^{-1}(m_{1a}(\rho_1')))(\lambda S_0.\lambda d_x.(\ i^{-1}(m_{1b}(\rho_1' \otimes d_x))k_1'S_0)S_1'$,
$(i^{-1}(m_1(\rho_1)))k_1S_1 = (i^{-1}(m_{1a}(\rho_1)))(\lambda S_0.\lambda d_x.(\ i^{-1}(m_{1b}(\rho_1 \otimes d_x))k_1S_0)S_1$,
$(i^{-1}(m_2(\rho_2')))k_2'S_2' = (i^{-1}(m_{2a}(\rho_2')))(\lambda S_0.\lambda d_x.(\ i^{-1}(m_{2b}(\rho_2' \otimes d_x))k_2'S_0)S_2'$,
$(i^{-1}(m_2(\rho_2)))k_2S_2 = (i^{-1}(m_{2a}(\rho_2)))(\lambda S_0.\lambda d_x.(\ i^{-1}(m_{2b}(\rho_2 \otimes d_x))k_2S_0)S_2$.

By assupmtion $(m_{1a}(\rho_1'), m_{2a}(\rho_2') \parallel m_{1a}(\rho_1), m_{2a}(\rho_2), T\tau_a[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$ and
$(S_1', S_2' \parallel S_1, S_2, (pks'')) \in \nabla_S$. If $(m_{1a}(\rho_1') = m_{2a}(\rho_2') = \bot$ or $S_1' = S_2' = \bot$ we get the required termination approximation. Else we want to show that
$((\lambda S_0.\lambda d_x.\ (i^{-1}(m_{1b}(\rho_1' \otimes d_x))k_1'S_0), (\lambda S_0.\lambda d_x.\ (i^{-1}(m_{2b}(\rho_2' \otimes d_x))k_2'S_0) \parallel$
$(\lambda S_0.\lambda d_x.\ (i^{-1}(m_{1b}(\rho_1 \otimes d_x))k_1S_0), (\lambda S_0.\lambda d_x.\ (i^{-1}(m_{2b}(\rho_2 \otimes d_x))k_2S_0), (x : \tau_a[\overline{\sigma_j/\alpha_j}])^\top, (pk'')) \in$
$F(\nabla, \nabla)_K$.

Let $(pk^3) \rhd (pk''), (pks^3) \in (pk^3)^{\mathbf{S}}, p^3 = (pk^3)^{vm}$ so $p^3 \rhd p''$.
Let $(S_{10}', S_{20}' \parallel S_{10}, S_{20}, (pks^3)) \in \nabla_S,\ (d_1', d_2' \parallel d_1, d_2, \tau_a[\overline{\sigma_j/\alpha_j}], p^3) \in \nabla_V$.
$(\lambda S_0.\lambda d_x.(i^{-1}(m_{1b}(\rho_1' \otimes d_x))k_1'S_0))S_{10}'d_1' = (i^{-1}(m_{1b}(\rho_1' \otimes d_1')))k_1'S_{10}'$,
$(\lambda S_0.\lambda d_x.\ (i^{-1}(m_{1b}(\rho_1 \otimes d_x))k_1S_0)S_{10}d_1 = (i^{-1}(m_{1b}(\rho_1 \otimes d_1)))k_1S_{10}$,
$(\lambda S_0.\lambda d_x.\ (i^{-1}(m_{2b}(\rho_2' \otimes d_x))k_2'S_0)S_{20}'d_2' = (i^{-1}(m_{2b}(\rho_2' \otimes d_2')))k_2'S_{20}'$,
$(\lambda S_0.\lambda d_0.\ (i^{-1}(m_{2b}(\rho_2 \otimes d_x))k_2S_0)S_{20}d_2 = (i^{-1}(m_{2b}(\rho_2 \otimes d_2)))k_2S_{20}$.

We have $p^3 \rhd p'' \blacktriangleright p' \rhd p$. So $p^3 \blacktriangleright p$. $(d_1', d_2' \parallel d_1, d_2, \tau_a[\overline{\sigma_j/\alpha_j}], p^3) \in \nabla_V$,
$\forall i.(v_{1i}', v_{2i}' \parallel v_{1i}, v_{2i}, \tau_i[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$, it follows by weakening that $\forall i.(v_{1i}', v_{2i}' \parallel v_{1i}, v_{2i}, \tau_i[\overline{\sigma_j/\alpha_j}], p^3) \in$
∇_V. Also $(m_{1b}, m_{2b}, T\tau_b[\overline{\sigma_j/\alpha_j}], p) \in \nabla_M^{\Xi(\Gamma, x:\tau_a)}$.

Then $((m_{1b}(\rho_1' \otimes d_1')), (m_{2b}(\rho_2' \otimes d_2')) \parallel (m_{1b}(\rho_1 \otimes d_1)), (m_{2b}(\rho_2 \otimes d_2)), T\tau_b[\overline{\sigma_j/\alpha_j}], p^3) \in \nabla_M$.
By assumption $(k_1', k_2' \parallel k_1, k_2, (x : \tau_b[\overline{\sigma_j/\alpha_j}])^\top, (pk'')) \in \nabla_K$. Since $(pk^3) \rhd (pk'')$ then by parameter weakening $(k_1', k_2' \parallel k_1, k_2, (x : \tau_b[\overline{\sigma_j/\alpha_j}])^\top, (pk^3)) \in \nabla_K$.
Since also $(S_{10}', S_{20}' \parallel S_{10}, S_{20}, (pks^3)) \in \nabla_S$ we can conclude that
$(m_1, m_2, T\tau_b, p) \in \nabla_M^{\Xi\Gamma}$.

fold : $\dfrac{\Delta; \Xi; \Gamma \vdash V : \tau[\mu\alpha.\tau(\alpha)/\alpha]}{\Delta; \Xi; \Gamma \vdash fold\ V : \mu\alpha.\tau}$

Let $v_{10} = [\![\Delta; \Gamma \vdash V_1]\!]$ and $v_{20} = [\![\Delta; \Gamma \vdash V_2]\!]$.
Assume $(v_{10}, v_{20}, \tau[\mu\alpha.\tau/\alpha], p) \in \nabla_V^{\Xi\Gamma}$.

The assumption implies $(v_{10}(\rho_1'), v_{20}(\rho_2') \parallel v_{10}(\rho_1), v_{20}(\rho_2), \tau[\mu\alpha.\tau/\alpha][\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$.
α is not free in $\tau[\mu\alpha.\tau/\alpha]$. If $\alpha \notin \Xi$ then let $i \in \{1 \ldots k\}$, if $\alpha = \alpha_m \in \Xi$ let $i \in \{1 \ldots k\} \setminus \{m\}$.
So $\tau[\mu\alpha.\tau/\alpha][\overline{\sigma_j/\alpha_j}] = (\tau[\overline{\sigma_i/\alpha_i}])[\mu\alpha.(\tau[\overline{\sigma_i/\alpha_i}])/\alpha]$ and $\mu\alpha.(\tau[\overline{\sigma_i/\alpha_i}]) = (\mu\alpha.\tau)[\overline{\sigma_j/\alpha_j}]$, and
$(v_{10}(\rho_1'), v_{20}(\rho_2') \parallel v_{10}(\rho_1), v_{20}(\rho_2), (\tau[\overline{\sigma_i/\alpha_i}])[\mu\alpha.(\tau[\overline{\sigma_i/\alpha_i}])/\alpha], p') \in \nabla_V$.
Then either $v_{10}(\rho_1') = v_{20}(\rho_2') = \bot$ or $v_{10}(\rho_1') \sqsubseteq v_{10}(\rho_1) \in (\mathbb{V})_\downarrow \wedge v_{20}(\rho_2') \sqsubseteq v_{20}(\rho_2) \in (\mathbb{V})_\downarrow$.

Let $v_1 = [\![\Delta; \Xi; \Gamma \vdash fold\ V_1 : \mu\alpha.\tau]\!]$ and $v_2 = [\![\Delta; \Xi; \Gamma \vdash fold\ V_2 : \mu\alpha.\tau]\!]$.
If $v_{10}(\rho_1') = v_{20}(\rho_2') = \bot$ then $v_1(\rho_1') = v_2(\rho_2') = \bot$ and so
$(v_1(\rho_1'), v_2(\rho_2') \parallel v_1(\rho_1), v_2(\rho_2), (\mu\alpha.\tau)[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$. Else $i^{-1}(v_1(\rho_1')) = in_\mu(v_{10}(\rho_1'))$, $i^{-1}(v_2(\rho_2')) = in_\mu(v_{20}(\rho_2'))$, $i^{-1}(v_1(\rho_1)) = in_\mu(v_{10}(\rho_1))$, $i^{-1}(v_2(\rho_2)) = in_\mu(v_{20}(\rho_2))$. By assumption $(v_{10}(\rho_1'), v_{20}(\rho_2') \parallel v_{10}(\rho_1), v_{20}(\rho_2), (\tau[\overline{\sigma_i/\alpha_i}])[\mu\alpha.(\tau[\overline{\sigma_i/\alpha_i}])/\alpha], p') \in \nabla_V$ then
$(v_1(\rho_1'), v_2(\rho_2') \parallel v_1(\rho_1), v_2(\rho_2), \mu\alpha.\tau[\overline{\sigma_i/\alpha_i}], p') = (v_1(\rho_1'), v_2(\rho_2') \parallel v_1(\rho_1), v_2(\rho_2), (\mu\alpha.\tau)[\overline{\sigma_j/\alpha_j}], p') \in$
∇_V. We see $(v_1, v_2, \mu\alpha.\tau, p) \in \nabla_V^{\Xi\Gamma}$.

unfold: $$\frac{\Delta;\Xi;\Gamma \vdash V : \mu\alpha.\tau}{\Delta;\Xi;\Gamma \vdash unfold\ V : T(\tau[\mu\alpha.\tau/\alpha])}$$

Let $v_1 = [\![\Delta;\Gamma \vdash V_1 : \mu\alpha.\tau]\!]$ and $v_2 = [\![\Delta;\Gamma \vdash V_2 : \mu\alpha.\tau]\!]$. Assume $(v_1, v_2, \mu\alpha.\tau, p) \in \nabla_V^{\Xi\Gamma}$.
Let $m_1 = [\![\Delta;\Gamma \vdash unfold\ V_1 : T(\tau[\mu\alpha.\tau/\alpha])]\!]$ and $m_2 = [\![\Delta;\Gamma \vdash unfold\ V_2 : T(\tau[\mu\alpha.\tau/\alpha])]\!]$.
We need to show $(m_1, m_2, T(\tau[\mu\alpha.\tau/\alpha]), p) \in \nabla_M^{\Xi\Gamma}$
The assumption $(v_1(\rho_1'), v_2(\rho_2') \parallel v_1(\rho_1), v_2(\rho_2), (\mu\alpha.\tau)[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$ implies
either $v_1(\rho_1') = v_2(\rho_2') = \bot$ or $\exists d_1', d_2', d_1, d_2$.
$(i^{-1}(v_1(\rho_1)) = in_\mu d_1 \neq \bot)\ \wedge\ ((i^{-1}(v_1(\rho_1')) = d_1' = \bot \vee (i^{-1}(v_1(\rho_1')) = in_\mu d_1' \neq \bot) \wedge$
$(i^{-1}(v_2(\rho_2)) = in_\mu d_2 \neq \bot) \wedge ((i^{-1}(v_2(\rho_2')) = d_2' = \bot \vee (i^{-1}(v_2(\rho_2')) = in_\mu d_2' \neq \bot) \wedge$
$(d_1', d_2' \parallel d_1, d_2, \tau[\mu\alpha.\tau/\alpha][\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$. By weakening also $\forall p'' \blacktriangleright p'.(d_1', d_2' \parallel d_1, d_2, \tau[\mu\alpha.\tau/\alpha][\overline{\sigma_j/\alpha_j}], p'') \in \nabla_V$. If $v_1(\rho_1') = v_2(\rho_2') = \bot$ then $m_1(\rho_1') = m_2(\rho_2') = \bot$ and then also $(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T\tau[\mu\alpha.\tau/\alpha][\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$. Else
$i^{-1}(m_1(\rho_1')) = \lambda k.\lambda S.i^{-1}(k)S(d_1'),\ i^{-1}(m_2(\rho_2')) = \lambda k.\lambda S.i^{-1}(k)S(d_2'),$
$i^{-1}(m_1(\rho_1)) = \lambda k.\lambda S.i^{-1}(k)S(d_1),\ i^{-1}(m_2(\rho_2)) = \lambda k.\lambda S.i^{-1}(k)S(d_2).$
Applied to continuations and states related under a parameter $\blacktriangleright$-extending p' this gives application of related continuations to states and values related under the same parameter as the continuations. Then also $(m_1(\rho_1'), m_2(\rho_2'), \parallel m_1(\rho_1), m_2(\rho_2), T(\tau[\mu\alpha.\tau/\alpha][\overline{\sigma_j/\alpha_j}]), p') \in \nabla_M$.
We conclude $(m_1, m_2, T(\tau[\mu\alpha.\tau/\alpha]), p) \in \nabla_M^{\Xi\Gamma}$.

Λ: $$\frac{\Delta;\Xi,\alpha;\Gamma \vdash M : T\tau \quad \Xi \vdash \Gamma}{\Delta;\Xi;\Gamma \vdash \Lambda\alpha.M : \forall\alpha.T\tau}$$

Assume α is not a free type variable in Γ.
Let $m_1 = [\![\Delta;\Xi,\alpha;\Gamma \vdash M_1 : T\tau]\!]$ and $m_2 = [\![\Delta;\Xi,\alpha;\Gamma \vdash M_2 : T\tau]\!]$.
Assume $(m_1, m_2, T\tau, p) \in \nabla_M^{\Xi\alpha\Gamma}$, then $\forall\sigma$ with $\_ \vdash \sigma : type$ it holds that
$(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_1), T\tau[\overline{\sigma_j/\alpha_j}, \sigma/\alpha], p') \in \nabla_M$.

Let $v_1 = [\![\Delta;\Xi;\Gamma \vdash \Lambda\alpha.M_1 : \forall\alpha.T\tau]\!]$ and $v_2 = [\![\Delta;\Xi;\Gamma \vdash \Lambda\alpha.M_2 : \forall\alpha.T\tau]\!]$.
We want to show $(v_1, v_2, \forall\alpha.T\tau, p) \in \nabla_V^{\Xi\Gamma}$ or (in all cases)
$(v_1(\rho_1'), v_2(\rho_2') \parallel v_1(\rho_1), v_2(\rho_2), \forall\alpha.T\tau[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$.

We have $\rho_1' = \rho_2' = \bot \Rightarrow v_1(\rho_1') = v_2(\rho_2') = \bot$. Else $v_1(\rho_1') \sqsubseteq in_\forall\lfloor m_1(\rho_1')\rfloor,\ v_2(\rho_2') \sqsubseteq in_\forall\lfloor m_2(\rho_2')\rfloor,\ v_1(\rho_1) = in_\forall\lfloor m_1(\rho_1)\rfloor,\ v_2(\rho_2) = in_\forall\lfloor m_1(\rho_2)\rfloor$.
By the definition of ∇ we want to show $\forall\sigma$ with $\vdash \sigma : type$ it holds that
$(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_1), T\tau[\overline{\sigma_j/\alpha_j}, \sigma/\alpha], p') \in \nabla_M$.
This follows from the assumptions.

Λapp: $$\frac{\Delta;\Xi;\Gamma \vdash V : \forall\alpha.T\tau \qquad \Xi \vdash \tau' : type}{\Delta;\Xi;\Gamma \vdash V\tau' : T(\tau[\tau'/\alpha])}$$

Assume $\Xi \vdash \tau' : type$ and let $v_1 = [\![\Delta;\Xi;\Gamma \vdash V_1 : \forall\alpha.T\tau]\!]$ and $v_2 = [\![\Delta;\Xi;\Gamma \vdash V_2 : \forall\alpha.T\tau]\!]$.
Assume $(v_1, v_2, \forall\alpha.T\tau, p) \in \nabla_V^{\Xi\Gamma}$ so $(v_1(\rho_1'), v_2(\rho_2') \parallel v_1(\rho_1), v_2(\rho_2), (\forall\alpha.T\tau)[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$.
Now if α does not occur among $\overline{\alpha_j}$ then let $\overline{\alpha_i}$ be the same as $\overline{\alpha_j}$ and $\overline{\sigma_i}$ be the same as $\overline{\sigma_j}$, if $\alpha_k = \alpha$ then let $\overline{\alpha_i}$ stand for $\overline{\alpha_j}\backslash\alpha_k$ and let similarly $\overline{\sigma_i}$ stand for $\overline{\sigma_j}\backslash\sigma_k$ then $(\forall\alpha.T\tau)[\overline{\sigma_j/\alpha_j}] = \forall\alpha.T\tau[\overline{\sigma_i/\alpha_i}]$. By the definition of ∇_V either $v_1(\rho_1') = v_2(\rho_2') = \bot$, or $\exists d_1', d_2', d_1, d_2$.
$(v_1(\rho_1) = in_\forall\lfloor d_1\rfloor) \wedge ((v_1(\rho_1') = \bot \wedge d_1' = \bot) \vee ((v_1(\rho_1') = in_\forall\lfloor d_1'\rfloor)) \wedge$
$(v_2(\rho_2) = in_\forall\lfloor d_2\rfloor) \wedge ((v_2(\rho_2') = \bot \wedge d_2' = \bot) \vee ((v_2(\rho_2') = in_\forall\lfloor d_2'\rfloor))$ and
$\forall\sigma$ with $\_ \vdash \sigma : type$ it holds that
$(d_1', d_2' \parallel d_1, d_2, (T\tau[\overline{\sigma_i/\alpha_i}])[\sigma/\alpha], p') = (d_1', d_2' \parallel d_1, d_2, (T\tau[\overline{\sigma_i/\alpha_i}, \sigma/\alpha], p') \in \nabla_M$, as $\tau'[\overline{\sigma_j/\alpha_j}]$ is closed so $(d_1', d_2' \parallel d_1, d_2, (T\tau[\overline{\sigma_i/\alpha_i}, (\tau'[\overline{\sigma_j/\alpha_j}])/\alpha], p') \in \nabla_M$.

Let $m_1 = [\![\Delta;\Xi;\Gamma \vdash V_1\tau' : T(\tau[\tau'/\alpha])]\!]$ and $m_2 = [\![\Delta;\Xi;\Gamma \vdash V_2\tau' : T(\tau[\tau'/\alpha])]\!]$.

We want to show $(m_1, m_2, T\tau[\tau'/\alpha], p) \in \nabla_M^{\Xi\Gamma}$ or
$(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T(\tau[\tau'/\alpha])\overline{[\sigma_j/\alpha_j]}, p') =$
$(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T\tau\overline{[\sigma_i/\alpha_i}, (\tau'\overline{[\sigma_j/\alpha_j]})/\alpha], p') \in \nabla_M$.

If $v_1(\rho_1') = v_2(\rho_2') = \bot$ then $m_1(\rho_1') = m_2(\rho_2') = \bot$, and
$(\bot, \bot \parallel m_1(\rho_1), m_2(\rho_2), T(\tau[\tau'/\alpha])\overline{[\sigma_j/\alpha_j]}, p') \in \nabla_M$. Else it follows from the assumptions.
We conclude $(m_1, m_2, T\tau[\tau'/\alpha], p) \in \nabla_M^{\Xi\Gamma}$.

□

For store type Δ let id_Δ be the match of finite store types $\{(l, l, \tau) | l : \tau \in \Delta\}$.

Theorem 4. *Fundamental Theorem*
For all ordinary parameters p with $Z^p = id_\Delta$.

$$\begin{array}{llll} \textit{If} & \Delta; \Xi; \Gamma \vdash V : \tau & \textit{then} & (\llbracket \Delta; \Xi; \Gamma \vdash V : \tau \rrbracket, \llbracket \Delta; \Xi; \Gamma \vdash V : \tau \rrbracket, \tau, p) \in \nabla_V^{\Xi\Gamma} \\ \textit{If} & \Delta; \Xi; \Gamma \vdash M : T\tau & \textit{then} & (\llbracket \Delta; \Xi; \Gamma \vdash M : T\tau \rrbracket, \llbracket \Delta; \Xi; \Gamma \vdash M : T\tau \rrbracket, T\tau, p) \in \nabla_M^{\Xi\Gamma} \end{array}$$

Proof by induction over typing derivations using proposition 4.
Base cases, by proposition 4 it holds that

$$\begin{array}{ll} \text{When } \Delta; \Xi; \Gamma \vdash x_j : \tau_j & \text{then } (\llbracket \Delta; \Xi; \Gamma \vdash x_j : \tau_j \rrbracket, \llbracket \Delta; \Xi; \Gamma \vdash x_j : \tau_j \rrbracket, p) \in \nabla_V^{\Xi\Gamma} \\ \text{When } \Delta; \Xi; \Gamma \vdash () : unit & \text{then } (\llbracket \Delta; \Xi; \Gamma \vdash () : unit \rrbracket, \llbracket \Delta; \Xi; \Gamma \vdash () : unit \rrbracket), unit, p) \in \nabla_V^{\Xi\Gamma} \\ \text{When } \Delta; \Xi; \Gamma \vdash n : int & \text{then } (\llbracket \Delta; \Xi; \Gamma \vdash n : int \rrbracket, \llbracket \Delta; \Xi; \Gamma \vdash n : int \rrbracket, int, p) \in \nabla_V^{\Xi\Gamma} \\ \text{When } \Delta; \Xi; \Gamma \vdash l : \tau\ ref & \text{then } (\llbracket \Delta; \Xi; \Gamma \vdash l : \tau\ ref \rrbracket, \llbracket \Delta; \Xi; \Gamma \vdash l : \tau\ ref \rrbracket, \tau\ ref, p) \in \nabla_V^{\Xi\Gamma} \end{array}$$

The theorem holds for all base cases, then again using proposition 4 it holds for all typing judgments of value terms and computation terms.

□

The next lemma has to do with four-tuples which are really two pairs.

Lemma 27.
Let $p \in \mathfrak{o}^{vm}$

$(m_1, m_2, T\tau, p) \in \nabla_M^{\emptyset\emptyset} \Rightarrow$

$\forall p' \rhd p. \forall (pk') \in p'^{\mathbf{K}}. \forall (pks') \in (pk')^{\mathbf{S}}.$
$\forall (k_1, k_2 \parallel k_1, k_2, (x : \tau)^\top, (pk')) \in \nabla_K. \forall (S_1, S_2 \parallel S_1, S_2, (pks')) \in \nabla_S.$
$(i^{-1}m_1)k_1 S_1 = \top \iff (i^{-1}m_2)k_2 S_2 = \top.$

Proof

$(m_1, m_2, T\tau, p) \in \nabla_M^{\emptyset\emptyset} \Rightarrow$
$(m_1, m_2 \parallel m_1, m_2, T\tau, p) \in \nabla_M \Rightarrow$
$(i^{-1}(m_1), i^{-1}(m_2) \parallel i^{-1}(m_1), i^{-1}(m_2), T\tau, p) \in F(\nabla, \nabla)_M$.

It follows from the definition of $F(\nabla, \nabla)_M$ that $\forall p' \rhd p. \forall (pk') \in p'^{\mathbf{K}}. \forall (pks') \in (pk')^{\mathbf{S}}.$
$\forall (k_1, k_2 \parallel k_1, k_2, (x : \tau)^\top, (pk')) \in \nabla_K. \forall (S_1, S_2 \parallel S_1, S_2, (pks')) \in \nabla_S.$
$(i^{-1}(m_1))k_1 S_1 = \top \Rightarrow (i^{-1}(m_2))k_2 S_2 = \top$ and
$(i^{-1}(m_1))k_1 S_1 = \top \Leftarrow (i^{-1}(m_2))k_2 S_2 = \top$

□

Let T_Δ be the vm-parameter $\{(T, A_\emptyset, A_\emptyset, id_\Delta)\}$.
Let T_Δ^K be the parameter $\{((T, A_\emptyset, A_\emptyset, id_\Delta) | (T, A_\emptyset, A_\emptyset, id_\Delta))\}$.
Let T_Δ^S be the parameter $\{((T, A_\emptyset, A_\emptyset, id_\Delta) | (T, A_\emptyset, A_\emptyset, id_\Delta) | T)\}$.
Since there is only one possible instantiation $T_\Delta \in \mathfrak{o}^{vm}$, $T_\Delta^K \in \mathfrak{o}^k$ and $T_\Delta^S \in \mathfrak{o}^s$.

Lemma 28.

1. $\forall \_ \vdash \tau : type.\ \forall (pk) \in \mathfrak{o}^k$ *with* $Z^{pk} = id_\Delta$. $([\![\Delta \vdash val\ x : (x:\tau)^\top]\!], [\![\Delta \vdash val\ x : (x:\tau)^\top]\!] \parallel [\![\Delta \vdash val\ x : (x:\tau)^\top]\!], [\![\Delta \vdash val\ x : (x:\tau)^\top]\!], (x:\tau)^\top, (pk)) \in \nabla_K$

2. $\forall \Sigma : \Delta.\ \forall S \in [\![\Sigma : \Delta]\!].\ (S, S \parallel S, S, T^S_\Delta) \in \nabla_S$

Proof (We have omitted the isomorphisms i, i^{-1}).

1. Assume $\_ \vdash \tau : type$.
$[\![\Delta \vdash val\ x : (x:\tau)^\top]\!] =$
$\lambda S.\lambda d.[\![\Delta;; x:\tau \vdash val\ x : T\tau]\!][x \mapsto d]((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S =$
$\lambda S.\lambda d.(\lambda k_0.\lambda S_0.k_0 S_0 [\![\Delta;; x:\tau \vdash x:\tau]\!][x \mapsto d])((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S =$
$\lambda S.\lambda d.(\lambda k_0.\lambda S_0.k_0 S_0 d)((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S =$
$\lambda S.\lambda d.((\lambda S'.(\lambda d'.\top)_\bot)_\bot)Sd.$

   Let $(pk') \rhd (pk), (pks') \in (pk')^{\mathbf{S}}, (S'_1, S'_2 \parallel S_1, S_2, (pks')) \in \nabla_S, (v'_1, v'_2 \parallel v_1, v_2, \tau, (pk')^{vm}) \in \nabla_V$.
By definition of ∇ it holds that either $(v'_1 = v'_2 = \bot)$ or $(v_1 \neq \bot \wedge v_2 \neq \bot)$ and either $(S'_1 = S'_2 = \bot)$ or $(S_1 \neq \bot \wedge S_2 \neq \bot)$
$[\![\Delta \vdash val\ x : (x:\tau)^\top]\!]S'_1 v'_1 = ((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S'_1 v'_1$
$[\![\Delta \vdash val\ x : (x:\tau)^\top]\!]S_1 v_1 = ((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S_1 v_1$
$[\![\Delta \vdash val\ x : (x:\tau)^\top]\!]S'_2 v'_2 = ((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S'_2 v'_2$
$[\![\Delta \vdash val\ x : (x:\tau)^\top]\!]S_2 v_2 = ((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S_2 v_2$
We need to show
$((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S'_1 v'_1 = \top \ \Rightarrow\ ((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S_2 v_2 = \top$ and
$((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S'_2 v'_2 = \top \ \Rightarrow\ ((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S_1 v_1 = \top$.
Assume $((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S'_1 v'_1 = \top$. Then $v'_1 \neq \bot$ and $S'_1 \neq \bot$. $v'_1 \neq \bot \Rightarrow (v_1 \neq \bot \wedge v_2 \neq \bot)$, and $S'_1 \neq \bot \Rightarrow (S_1 \neq \bot \ \wedge\ S_2 \neq \bot)$. This implies that $((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S_2 v_2 = \top$.
The other direction is proved similarly.

2. We show $S \in [\![\Sigma : \Delta]\!]$ implies $(S, S \parallel S, S, T^S_\Delta) \in \nabla_S$.
It holds that $S \sqsubseteq S \neq \bot$ and $\forall l \in dom(\Delta).\ Sl = [\![\Delta;;\vdash \Sigma(l) : \Delta(l)]\!]$.
Then, by the previous theorem 4, $\forall l \in dom(\Delta).(Sl, Sl, Sl, Sl, \Delta l, T_\Delta) \in \nabla_V$.
It holds that $dom(\Delta) \cap A_\emptyset(S) = \emptyset$. Also $(S, S) \in T$.

□

Theorem 5. *Contextual equivalence*
For all $\Delta, \Xi, \gamma, \tau$, *for all value- or computation terms* G_1, G_2,
for all contexts $C[\_] : (\Delta; \Xi; \Gamma \vdash \gamma) \Rightarrow (\Delta;;\vdash T\tau)$ *(let* $j \in V, M$*)*

If $\Delta; \Xi; \Gamma \vdash G_1 : \gamma$ *and* $\Delta; \Xi; \Gamma \vdash G_2 : \gamma$ *and*
$([\![\Delta; \Xi; \Gamma \vdash G_1 : \gamma]\!], [\![\Delta; \Xi; \Gamma \vdash G_2 : \gamma]\!], \gamma, T_\Delta) \in \nabla^{\Xi\Gamma}_j$ *then*

$\forall \Sigma : \Delta.\ \ (\Sigma, let\ x \Leftarrow C[G_1]\ in\ val\ x \downarrow \Longleftrightarrow \Sigma, let\ x \Leftarrow C[G_2]\ in\ val\ x \downarrow)$

Proof

By induction over the structure of $C[\_]$ and using that typing rules preserve relatedness in $\nabla^{\Xi\Gamma}$and fundamental theorem 4 it holds that
$([\![\Delta; \Xi; \Gamma \vdash G_1 : \gamma]\!], [\![\Delta; \Xi; \Gamma \vdash G_2 : \gamma]\!], \gamma, T_\Delta) \in \nabla^{\Xi\Gamma}_j \Longrightarrow$
$([\![\Delta;;\vdash C[G_1] : T\tau]\!], [\![\Delta;;\vdash C[G_2] : T\tau]\!], T\tau, T_\Delta) \in \nabla^{\emptyset\emptyset}_M$.

By the previous lemma 28 when τ is closed

$([\![\Delta \vdash val\ x : (x:\tau)^\top]\!], [\![\Delta \vdash val\ x : (x:\tau)^\top]\!] \parallel [\![\Delta \vdash val\ x : (x:\tau)^\top]\!], [\![\Delta \vdash val\ x : (x:\tau)^\top]\!],$
$(x:\tau)^\top, T_\Delta) \in \nabla_K$
and
$\forall S \in [\![\Sigma : \Delta]\!].\ (S, S \parallel S, S, T_\Delta) \in \nabla_S.$

By lemma 27 it then holds that
$\forall S \in [\![\Sigma : \Delta]\!]$
$(i^{-1}[\![\Delta; \vdash C[M_1]]\!]\{\})[\![\Delta; \vdash val\ x : (x)^\top]\!]S = \top \iff$
$(i^{-1}[\![\Delta; \vdash C[M_2]]\!]\{\})[\![\Delta; \vdash val\ x : (x)^\top]\!]S = \top.$

By soundness and adequacy of the denotational semantics this implies
$\Sigma, let\ x \Leftarrow C[M_1]\ in\ val\ x\ \downarrow \iff \Sigma, let\ x \Leftarrow C[M_2]\ in\ val\ x\ \downarrow.$

□

7 Examples

In this section we give examples of proofs of program equivalences. Some of the examples were discussed informally in the introduction. Here we often omit the isomorphisms i, i^{-1} and injection functions. We abbreviate some let constructions by ; and sometimes we also simplify notation in other ways. When we have shown that some proof cases go through almost trivially, then we sometimes later omit similar cases.

7.1 Example. The Awkward Example

This example is taken from Pitts and Stark [39], it was discussed informally in the introduction. The two programs M and N are closed computations.

M: $let\ a \Leftarrow ref\ 0\ in$
$\quad val\ \big(rec\ f_M(g : unit \to T\tau) : Tint =$
$\quad\quad let\ x \Leftarrow a := 1\ in\ let\ y \Leftarrow g()\ in\ !a)\big)$

N: $val\ (rec\ f_N(g : unit \to T\tau) : Tint = let\ z \Leftarrow g()\ in\ val\ 1)$

We want to show that

$([\![; \vdash M : T\sigma]\!], [\![; \vdash N : T\sigma]\!], T((unit \to T\tau) \to Tint)), T_\emptyset) \in \nabla_M^{\emptyset\emptyset}$, that is
$([\![M]\!], [\![N]\!] \parallel [\![M]\!], [\![N]\!], T\sigma, T_\emptyset) \in \nabla_M.$

Let $p^1 \blacktriangleright T_\emptyset, (pk^1) \in p^{1\mathbf{K}}, (pks^1) \in pk^{1\mathbf{S}}$. Assume $(k_1', k_2' \parallel k_1, k_2, (x:\sigma)^\top, (pk^1)) \in \nabla_K$, $(s_1', s_2' \parallel s_1, s_2, (pks^1)) \in \nabla_S$. The case $s_1' = s_2' = \bot$ is trivial, else

$[\![; \vdash M]\!]\ k_1'\ s_1' = k_1'\ (s_1'[l_a \mapsto 0])\ ([\![; a \vdash recf_M]\!]\{a \mapsto l_a\})$ where l_a is a fresh location.
$[\![; \vdash M]\!]\ k_1\ s_1 = k_1\ (s_1[l_a \mapsto 0])\ ([\![; a \vdash recf_M]\!]\{a \mapsto l_a\})$ where l_a is a fresh location.
$[\![; \vdash N]\!]\ k_2'\ s_2' = k_2'\ s_2'\ ([\![; \vdash recf_N]\!]).$
$[\![; \vdash N]\!]\ k_2\ s_2 = k_2\ s_2\ ([\![; \vdash recf_N]\!]).$

By assumption $(k_1', k_2' \parallel k_1, k_2, (x:\sigma)^\top, (pk^1)) \in \nabla_K$, so we now want to show that there exists $(pk)^2 \rhd (pk^1)$ and $(pks^2) \in (pk^2)^{\mathbf{S}}$ such that

$(\llbracket ;a\vdash recf_M\rrbracket\{a\mapsto l_a\}, \llbracket ;\vdash recf_N\rrbracket \parallel \llbracket ;a\vdash recf_M\rrbracket\{a\mapsto l_a\}, \llbracket ;\vdash recf_N\rrbracket, \sigma, (pk^2)^{vm}) \in \nabla_V$ and $(s'_1[l_a\mapsto 0], s'_2 \parallel s_1[l_a\mapsto 0], s_2, (pks^2)) \in \nabla_S$.

Define local parameter $r = \langle q_a \prec \langle q_b\rangle\rangle$ where
$q_a = (\{(S_1,S_2)|S_1 l_a = 0\}, A_{l_a}, A_\emptyset, \emptyset_Z\ )$ and $q_b = (\{(S_1,S_2)|S_1 l_a = 1\ \}, A_{l_a}, A_\emptyset, \emptyset_Z)$.

Let $p^2 = p^1 \cup \{r\}$ then $p^2 \rhd p^1$. Let $(pk^2) = (pk^1)\cup\{(r|r)\}$ then $(pk^2)\in p^{2\mathbf{K}}$ and $(pk^2)\rhd(pk^1)$, and let $(pks^2) = (pks^1)\cup\{(r|r|\{(S_1,S_2)|S_1 l_a = 0\})\}$ then $(pks^2)\in pk^{2\mathbf{S}}$ and $(pks^2)\rhd(pks^1)$.

By assumption $(s'_1, s'_2 \parallel s_1, s_2, (pks^1)) \in \nabla_S$. Since l_a is fresh and by parameter weakening it follows that $(s'_1[l_a\mapsto 0], s'_2 \parallel s_1[l_a\mapsto 0], s_2, (pks^2)) \in \nabla_S$.

We need to show
$(\llbracket ;a\vdash recf_M\rrbracket\{a\mapsto l_a\}, \llbracket ;\vdash recf_N\rrbracket \parallel \llbracket ;a\vdash recf_M\rrbracket\{a\mapsto l_a\}, \llbracket ;\vdash recf_N\rrbracket, \sigma, p^2) \in \nabla_V$. For this let $p^4 \blacktriangleright p^3 \blacktriangleright p^2$, $(pk^4)\in p^{4\mathbf{K}}$, $(pks^4)\in pk^{4\mathbf{S}}$ and assume $(g'_1, g'_2 \parallel g_1, g_2, unit\to T\tau, p^3)\in\nabla_V$, $(K'_1, K'_2 \parallel K_1, K_2, (x:\tau)^\top, (pk^4))\in\nabla_K$ and $(S'_1, S'_2 \parallel S_1, S_2, (pks^4))\in\nabla_S$.
As before, the cases $g'_1 = g'_2 = \bot$, $K'_1 = K'_2 = \bot$ or $S'_1 = S'_2 = \bot$ are trivial, we will not always mentions such cases. Also we will sometimes omit writing "either $\bot$ or ..." or $\sqsubseteq$ on the primed places. This will be in cases were we want to prove termination approximation, and when such is proved for some cases then it is always also present for lower results.

$\llbracket ;a\vdash recf_M\rrbracket\{a\mapsto l_a\} g'_1 K'_1 S'_1 \sqsubseteq (g'_1(in_{\mathbf{1}}*))(\lambda S'.\lambda d'.\llbracket ;a\vdash !a\rrbracket\{a\mapsto l_a\}K'_1 S')(S'_1[l_a\mapsto in_{\mathbb{Z}}1])$,
$\llbracket ;a\vdash recf_M\rrbracket\{a\mapsto l_a\} g_1 K_1 S_1 = (g_1(in_{\mathbf{1}}*))(\lambda S'.\lambda d'.\llbracket ;a\vdash !a\rrbracket\{a\mapsto l_a\}K_1 S')(S_1[l_a\mapsto in_{\mathbb{Z}}1])$,
$\llbracket ;\vdash recf_N\rrbracket\{\} g'_2 K'_2 S'_2 \sqsubseteq (g'_2(in_{\mathbf{1}}*))(\lambda S'.\lambda d'.\llbracket ;\vdash val\ 1\rrbracket\{\}K'_2 S')S'_2$,
$\llbracket ;\vdash recf_N\rrbracket\{\} g_2 K_2 S_2 = (g_2(in_{\mathbf{1}}*))(\lambda S'.\lambda d'.\llbracket ;\vdash val\ 1\rrbracket\{\}K_2 S')S_2$.

Since $(g'_1, g'_2 \parallel g_1, g_2, unit\to T\tau, p^3)\in\nabla_V$ by assumption, then
$(g'_1(in_{\mathbf{1}}*)), g'_2(in_{\mathbf{1}}*)) \parallel g_1(in_{\mathbf{1}}*)), g_2(in_{\mathbf{1}}*)), T\tau, p^3)\in\nabla_M$.

So we want to show that these computations are applied to continuations and states related in a parameter $\blacktriangleright$-extended form p^3.

Because $p^4\blacktriangleright p^3$ it holds that either $r\in p^4$ or the local extension $\langle q_b\rangle\in p^4$.
If $\langle q_b\rangle\in p^4$ then let $q^5 = q^4$, $(pk^5) = (pk^4)$ and $(pks^5) = (pks^4)$.
If $r\in p^4$ then let $p^5 = p^4\setminus\{r\}\cup\{\langle q_b\rangle\}$, $(pk^5) = (pk^4)\setminus\{(r|r)\}\cup\{(\langle q_b\rangle|\langle q_b\rangle)\}$ and $(pks^5) = (pks^4)\setminus\{(r|r|\{(S_1,S_2)|S_2 l_a = 0\})\}\cup\{(\langle q_b\rangle|\langle q_b\rangle|\{(S_1,S_2)|S_2 l_a = 1\})\}$.
In both cases $p^5\rhd p^4$, and it holds that $\langle q_b\rangle\in p^5$.

By assumption $(S'_1, S'_2 \parallel S_1, S_2, (pks^4))\in\nabla_S$. If $\langle q_b\rangle\in p^4$ then $(pks^5) = (pks^4)$, $S_1 l_a = 1$ and $S_1[l_a\mapsto in_{\mathbb{Z}}1] = S_1$ so $(S'_1[l_a\mapsto in_{\mathbb{Z}}1], S'_2 \parallel S_1[l_a\mapsto in_{\mathbb{Z}}1], S_2, (pks^5))\in\nabla_S$. If $r\in p^4$ then $(pks^5) = (pks^4)\setminus\{(r|r|\{(S_1,S_2)|S_2 l_a = 0\})\}\cup\{(\langle q_b\rangle|\langle q_b\rangle|\{(S_1,S_2)|S_2 l_a = 1\})\}$ and so by parameter weakening for the stored values $(S'_1[l_a\mapsto in_{\mathbb{Z}}1], S'_2 \parallel S_1[l_a\mapsto in_{\mathbb{Z}}1], S_2, (pks^5))\in\nabla_S$.

We need to show that the continuations are related under the parameter (pk^5).

Let $(pk^6)\rhd(pk^5)$, $(pks^6)\in(pk^5)^{\mathbf{S}}$, $p^6 = (pk^6)^{vm}$.
Assume $(\bar S'_1, \bar S'_2 \parallel \bar S_1, \bar S_2, (pks^6))\in\nabla_S$ and $(d'_1, d_1, d'_2, d_2, \tau, p^6)\in\nabla_V$.

$(\lambda S'.\lambda d'.\llbracket\emptyset;a\vdash !a\rrbracket\{a\mapsto l_a\}K_1 S')\bar S_1 d_1 =$
  if $(d_1 = \bot \vee \bar S_1 = \bot)$ then $\bot$ else $K_1\bar S_1(\bar S_1 l_a) =$
  if $(d_1 = \bot \vee \bar S_1 = \bot)$ then $\bot$ else $K_1\bar S_1(in_{\mathbb{Z}}1)$.

$(\lambda S'.\lambda d'.\llbracket\emptyset;a\vdash !a\rrbracket\{a\mapsto l_a\}K'_1 S')\bar S'_1 d'_1 =$

if $(d_1' = \bot \vee \bar{S}_1' = \bot)$ then $\bot$ else $K_1' \bar{S}_1' (\bar{S}_1' l_a) =$
if $(d_1' = \bot \vee \bar{S}_1' = \bot)$ then $\bot$ else $K_1' \bar{S}_1' (v_1')$ where $v_1' \sqsubseteq in_{\mathbb{Z}} 1$.

$(\lambda S'.\lambda d'.[\![\emptyset; \vdash val\ 1]\!]\{\} K_2 S') \bar{S}_2 d_2 =$
if $(d_2 = \bot \vee \bar{S}_2 = \bot)$ then $\bot$ else $K_2 \bar{S}_2 (in_{\mathbb{Z}} 1)$.
$(\lambda S'.\lambda d'.[\![\emptyset; \vdash val\ 1]\!]\{\} K_2' S') \bar{S}_2' d_2' =$
if $(d_2 = \bot \vee \bar{S}_2 = \bot)$ then $\bot$ else $K_2' \bar{S}_2' (in_{\mathbb{Z}} 1)$.

By assumption $(K_1', K_2' \parallel K_1, K_2, (x : \tau), (pk^4)) \in \nabla_K$ and $(\bar{S}_1', \bar{S}_2' \parallel \bar{S}_1, \bar{S}_2, (pks^6)) \in \nabla_S$. Also $(v_1', in_{\mathbb{Z}} 1 \parallel in_{\mathbb{Z}} 1, in_{\mathbb{Z}} 1, int, p^6) \in \nabla_V$. By assumption and definition $p^6 \rhd p^5 \rhd p^4$. So we get the desired termination properties to conclude, that
$((\lambda S'.\lambda d'.[\![\emptyset; a \vdash !a]\!]\{a \mapsto l_a\} K_1' S'), (\lambda S'.\lambda d'.[\![\emptyset; \vdash val\ 1]\!]\{\} K_2' S') \parallel$
$(\lambda S'.\lambda d'.[\![\emptyset; a \vdash !a]\!]\{a \mapsto l_a\} K_1 S'), (\lambda S'.\lambda d'.[\![\emptyset; \vdash val\ ]\!]\{\} K_2 S'), (x : \tau)^\top, (pk^5)) \in \nabla_K$.
Then we can futher conclude that
$([\![; a \vdash recf_M]\!]\{a \mapsto l_a\}, [\![; \vdash recf_N]\!] \parallel [\![; a \vdash recf_M]\!]\{a \mapsto l_a\}, [\![; \vdash recf_N]\!], \sigma, p^2) \in \nabla_V$.
And then finally we have $([\![; \vdash M : T\sigma]\!], [\![; \vdash N : T\sigma]\!], T((unit \to T\tau) \to Tint)), T_\emptyset) \in \nabla_M^{\emptyset\emptyset}$.

7.2 Example. Knowing the initial steps of continuations

The programs M and N below are open computations with one free variable g of function type $\sigma = (unit \to Tunit) \to T\tau$. The programs were presented in the Introduction.

```
M:   let w ⇐ ref 0 in
       g(rec f_M(u : unit) = (w := 1));
       val ((rec get_M(u : unit) = (!w)) , (rec set_M(n : int) = (w := n)))

N:   let x ⇐ ref 0 in    //flag: inside program N
     let y ⇐ ref 0 in    //flag: argument function f_N has not been applied inside program N
     let v ⇐ ref 0 in
       g(rec f_N(u : unit) = (if (!x = 0) then (y := 1) else (v := 1));
       if (!y ≠ 0) then (v := 1);
       x := 1;
       val ((rec get_N(u : unit) = (!v)) , (rec set_N(n : int) = (v := n)))
```

We want to show that the denotations of M and N are related in the vm-parameter
$p = T_\emptyset \in \mathfrak{o}^{vm}$, that is

$$([\![; g \vdash M]\!], [\![; g \vdash N]\!], T\tau, p) \in \nabla_M^{\emptyset\{g:\sigma\}}$$

Let $p^1 \blacktriangleright T_\emptyset$, $(g_1', g_2' \parallel g_1, g_2, ((unit \to Tunit) \to T\tau), p^1) \in \nabla_V$, so we want to prove that
$([\![; ; g \vdash M]\!](g \mapsto g_1'), [\![; ; g \vdash N]\!](g \mapsto g_2') \parallel [\![; ; g \vdash M]\!](g \mapsto g_1), [\![; ; g \vdash N]\!](g \mapsto g_2), T\tau, p^1) \in \nabla_M$.
If $g_1' = g_2' = \bot$ we are done. Else let $p^2 \blacktriangleright p^1$, $(pk^2) \in p^{2\mathbf{K}}$, $(pks^2) \in (pk^2)^{\mathbf{S}}$,
$(K_1', K_2' \parallel K_1, K_2, (z : \tau)^\top, (pk^2)) \in \nabla_K$, $(S_1', S_2' \parallel S_1, S_2, (pks^2)) \in \nabla_S$.
Again if $K_1' = K_2' = \bot$ or $S_1' = S_2' = \bot$ we are done. Else

$[\![M]\!] g_1 K_1 S_1 =$
$(g_1 [\![w \vdash rec\ f_M(u : unit) = (w := 1))]\!](w \mapsto l_w))$
$(\lambda S^0.\lambda d^0.[\![w \vdash val\ (rec\ get_M, rec\ set_M)]\!](w \mapsto l_w) K_1 S^0)$
$S_1[l_w \mapsto 0]$,
where l_w is fresh.

$[\![M]\!] g_1' K_1' S_1'$ similar, we use the same location name l_w.

$[\![N]\!]g_2K_2S_2 =$
$(g_2[\![x, y, v \vdash rec\ f_N(u : unit) = \text{if } (!x = 0) \text{ then } (y := 1) \text{ else } (v := 1)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v))$
$(\lambda S^0 \lambda d^0.[\![x, y, v \vdash \text{if } (!y \neq 0) \text{ then } (v := 1);$
$(x := 1);\ val\ (rec\ get_N, rec\ set_N)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v)K_2S^0)$
$S_2[l_x \mapsto 0, l_y \mapsto 0, l_v \mapsto 0],$
where l_x, l_y, l_v are fresh.

$[\![N]\!]g_2'K_2'S_2'$ similar; we use the same location names.

Define local parameter with empty match of store type $\emptyset_Z$ and constant accessibility maps

$$r_3 = \big(\big((S_2l_x \neq 0 \wedge \exists n \in \mathbb{Z}.\ S_1l_w = S_2l_v = n, \emptyset_{LL}), \emptyset_Z, A_{l_w}, A_{l_v,l_y}\big)\bar{\wedge} \\ \big((S_2l_x = 0 \wedge S_2l_v = 0 \wedge S_1l_w = S_2l_y \in \{0,1\}, \emptyset_{LL}), \emptyset_Z, A_{l_w}, A_{l_v,l_y}\big)$$

By freshness of l_w and l_x, l_y, l_v we have a vm-parameter $p^3 = \{r_3\} \cup p^2$, and $p^3 \blacktriangleright p^2 \blacktriangleright p^1$.

First we want to show that we apply the related g-functions to related arguments. For this show
$([\![w \vdash rec\ f_M(u : unit) = (w := 1))]\!](w \mapsto l_w),$
$[\![x, y, v \vdash rec\ f_N(u : unit).\ \text{if } (!x = 0) \text{ then } (y := 1) \text{ else } (v := 1)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v) \parallel$
$[\![w \vdash recf(u : unit) = (w := 1))]\!](w \mapsto l_w),$
$[\![x, y, v \vdash rec\ f_N(u : unit).\ \text{if } (!x = 0) \text{ then } (v := 0; y := 1) \text{ else } (v := 1)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v),$
$unit \to Tunit,\ p^3) \in \nabla_V.$

Let $p^4_{ab} \blacktriangleright p^3$. We will see that the proof comes through by assumptions about relatedness of continuations. Therefore we only need to differentiate two cases.
Let $(pk^4_a) \in p^{4\mathbf{K}}$ either with r_3 $\bar{\wedge}$-instantiation $(S_2l_x \neq 0 \wedge S_1l_w = S_1l_v)$ or with r_3 extended by removal of $\bar{\wedge}$ clauses such that only $(S_2l_x \neq 0 \wedge S_1l_w = S_1l_v)$ is present in p^4_{ab}. And let $(pks^4_a) \in pk^{4\mathbf{S}}_a$.
Let $(pk^4_b) \in p^{4\mathbf{K}}$ either with $r_3\bar{\wedge}$-inst $(S_2l_x = 0 \wedge S_2l_v = 0 \wedge S_1l_w = S_2l_y \in \{0,1\})$ or with r_3 extended by removal of $\bar{\wedge}$ clauses such that only $(S_2l_x = 0 \wedge S_2l_v = 0 \wedge S_1l_w = S_2l_y \in \{0,1\})$ is present in p^4_{ab}. And let $(pks^4_b) \in pk^{4\mathbf{S}}_b$.

Assume
$(k'_{a1}, k'_{a2} \parallel k_{a1}, k_{a2}, (x : unit)^\top, (pk^4_a)) \in \nabla_K,\ (s'_{a1}, s'_{a2} \parallel s_{a1}, s_{a2}, (pks^4_a)) \in \nabla_S,$
$(k'_{b1}, k'_{b2} \parallel k_{b1}, k_{b2}, (x : unit)^\top, (pk^4_b)) \in \nabla_K,\ (s'_{b1}, s'_{b2} \parallel s_{b1}, s_{b2}, (pks^4_b)) \in \nabla_S.$

Again if $k'_{a1} = k'_{a2} = \perp$ or $s'_{a1} = s'_{a2} = \perp$ we are done in the first case, if if $k'_{b1} = k'_{b2} = \perp$ or $s'_{b1} = s'_{b2} = \perp$ we are done in the second case. Else

$[\![w \vdash rec\ f_M(u : unit) = (w := 1))]\!](w \mapsto l_w) * k_{a1}s_{a1} = k_{a1}(s_{a1}[l_w \mapsto 1])*$

$[\![x, y, v \vdash rec\ f_N(u : unit)\ \text{if } (!x = 0) \text{ then } (y := 1) \text{ else } (v := 1)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v) * k_{a2}s_{a2} = k_{a2}(s_{a2}[l_v \mapsto 1])*$

In each side the primed case is similar (or $\perp$).

By assumption, in this case, the continuations and the original states are related with requirement $(S_2l_x \neq 0 \wedge S_1l_w = S_1l_v)$. The states have only been changed in l_w, l_v. In the updated states l_w, l_v hold the same integer value, so the updated states are still related. Also the $*$ values are related. So we apply related continuations to values and states related under instantiations of the same parameter. Hence we get the required termination behaviours.

$[\![w \vdash rec\ f_M(u : unit) = (w := 1))]\!](w \mapsto l_w) * k_{b1}s_{b1} = k_{b1}(s_{b1}[l_w \mapsto 1])*$

$[\![x, y, v \vdash rec\ f_N(u : unit)$ if $(!x = 0)$ then $(y := 1)$ else $(v := 1)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v) * k_{b2}s_{b2} =$ $k_{b2}(s_{b2}[l_y \mapsto 1])*$

By assumption, in this case, the continuations and the original states are related with requirement $(S_2l_x = 0 \wedge S_2l_v = 0 \wedge S_1l_w = S_2l_y \in \{0, 1\})$. The states have only been changed in l_w, l_y. In the updated states l_w, l_y hold the same value 1, so the updated states are still related. Also the $*$ values are related. So we apply related continuations to states and values related under instantiations of the same parameter. Hence we get the required termination behaviours.

We conclude that the functions are related
$([\![w \vdash rec\ f_M(u : unit) = (w := 1))]\!](w \mapsto l_w),$
$[\![x, y, v \vdash rec\ f_N(u : unit).$ if $(!x = 0)$ then $(y := 1)$ else $(v := 1)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v) \parallel$
$[\![w \vdash recf(u : unit) = (w := 1))]\!](w \mapsto l_w),$
$[\![x, y, v \vdash rec\ f_N(u : unit).$ if $(!x = 0)$ then $(v := 0; y := 1)$ else $(v := 1)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v),$
$unit \to Tunit,\ p^3) \in \nabla_V$.

Since $p^3 \blacktriangleright p^1$ and by assumption $(g'_1, g'_2 \parallel g_1, g_2, \sigma, p^1) \in \nabla_V$ then
$g'_1([\![w \vdash rec\ f_M(u : unit) = (w := 1))]\!](w \mapsto l_w)),$
$g'_2([\![x, y, v \vdash rec\ f_N(u : unit).$ if $(!x = 0)$ then $(y := 1)$ else $(v := 1)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v)) \parallel$
$g_1([\![w \vdash rec\ f_M(u : unit) = (w := 1))]\!](w \mapsto l_w)),$
$g_2([\![x, y, v \vdash rec\ f_N(u : unit).$ if $(!x = 0)$ then $(y := 1)$ else $(v := 1)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v)),$
$T\tau,\ p^3) \in \nabla_M$

So we need to prove that these related computations are applied to continuations and states related in instantiations of a parameter $\blacktriangleright$-extending p^3 in:

$[\![M]\!]g_1K_1S_1 =$
  $(g_1[\![w \vdash rec\ f_M(u : unit) = (w := 1))]\!](w \mapsto l_w))$
    $(\lambda S^0.\lambda d^0.[\![w \vdash val\ (rec\ get_M, rec\ set_M)]\!](w \mapsto l_w)K_1S^0)$
      $S_1[l_w \mapsto 0],$
$[\![M]\!]g'_1K'_1S'_1$ similar.

$[\![N]\!]g_2K_2S_2 =$
  $(g_2[\![x, y, v \vdash rec\ f_N(u : unit) =$ if $(!x = 0)$ then $(y := 1)$ else $(v := 1)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v))$
    $(\lambda S^0\lambda d^0.[\![x, y, v \vdash$ if $(!y \neq 0)$ then $(v := 1);$
        $(x := 1);\ val\ (rec\ get_N, rec\ set_N)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v)K_2S^0)$
      $S_2[l_x \mapsto 0, l_y \mapsto 0, l_v \mapsto 0],$
$[\![N]\!]g'_2K'_2S'_2$ similar.

Let $(pk^3) \in p^{3\mathbf{K}},\ (pk^3) \blacktriangleright (pk^2)$ with r_3 $\bar{\wedge}$-instantiation
$(S_2l_x = 0 \wedge S_2l_v = 0 \wedge S_1l_w = S_2l_y \in \{0, 1\})$ and let $(pks^3) \in pk^{3\mathbf{S}}$.

By assumption $(S'_1, S'_2 \parallel S_1, S_2, (pks^2)) \in \nabla_S$, then by weakening for the stored values, it holds that $(S'_1[l_w \mapsto 0], S'_2[l_x \mapsto 0, l_y \mapsto 0, l_v \mapsto 0] \parallel S_1[l_w \mapsto 0], S_2[l_x \mapsto 0, l_y \mapsto 0, l_v \mapsto 0], (pks^3)) \in \nabla_S$.

It remains to prove that the continuations are apropriately related
$(\lambda S^0.\lambda d^0.[\![w \vdash val\ (rec\ get_M, rec\ set_M)]\!](w \mapsto l_w)K'_1S^0)$
$(\lambda S^0\lambda d^0.[\![x, y, v \vdash$ if $(!y \neq 0)$ then $(v := 1);$
    $(x := 1); val\ (rec\ get_N, rec\ set_N)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v)K'_2S^0) \parallel$

$(\lambda S^0.\lambda d^0.[\![w \vdash val\ (rec\ get_M, rec\ set_M)]\!](w \mapsto l_w)K_1 S^0)$,
$(\lambda S^0 \lambda d^0.[\![x, y, v \vdash \text{if } (!y \neq 0) \text{ then } (v := 1);$
$\qquad (x := 1); val\ (rec\ get_N, rec\ set_N)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v)K_2 S^0), (x : \tau), (pk^3)) \in \nabla_K$.

Let
$(pk^4) \rhd (pk^3)$, $(pks^4) \in (pk^4)^{\mathbf{S}}$, $(S_1^{4\prime}, S_2^{4\prime} \parallel S_1^4, S_2^4, (pks^4)) \in \nabla_S$, $(v_1^{4\prime}, v_2^{4\prime} \parallel v_1^4, v_2^4, \tau, (pk^4)^{vm}) \in \nabla_V$.

Again the $\perp$ cases trivially give termination approximation. Else

$(\lambda S^0.\lambda d^0.[\![w \vdash val\ (rec\ get_M, rec\ set_M)]\!](w \mapsto l_w)K_1 S^0)(S_1^4)(v_1^4) =$
$\quad K_1(S_1^4)([\![w \vdash rec\ (get_M, rec\ set_M)]\!](w \mapsto l_w))$

$(\lambda S^0.\lambda d^0.[\![w \vdash val\ (rec\ get_M, rec\ set_M)]\!](w \mapsto l_w)K_1 S^0)(S_1^{4\prime})(v_1^{4\prime}) =$
$\quad K_1(S_1^{4\prime})([\![w \vdash rec\ (get_M, rec\ set_M)]\!](w \mapsto l_w))$

$(\lambda S^0 \lambda d^0.[\![x, y, v \vdash \text{if } (!y \neq 0) \text{ then } (v := 1); (x := 1);$
$\qquad val\ (rec\ get_N, rec\ set_N)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v)K_2 S^0)(S_2^4)(v_2^4) =$

- $K_2 S_2^4[l_x \mapsto 1]([\![v \vdash rec\ (get_N, rec\ set_N)]\!](v \mapsto l_v))$ if $S_2^4 l_y = 0$
- $K_2 S_2^4[l_v \mapsto 1, l_x \mapsto 1]([\![v \vdash rec\ (get_N, rec\ set_N)]\!](v \mapsto l_v))$ if $S_2^4 l_y = n \neq 0$
- else $\perp$

$(\lambda S^0 \lambda d^0.[\![x, y, v \vdash \text{if } (!y \neq 0) \text{ then } (v := 1); (x := 1);$
$\qquad val\ (rec\ get_N, rec\ set_N)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v)K_2 S^0)(S_2^{4\prime})(v_2^{4\prime}) =$

- $K_2 S_2^{4\prime}[l_x \mapsto 1]([\![v \vdash rec\ (get_N, rec\ set_N)]\!](v \mapsto l_v))$ if $S_2^{4\prime} l_y = 0$
- $K_2 S_2^{4\prime}[l_v \mapsto 1, l_x \mapsto 1]([\![v \vdash rec\ (get_N, rec\ set_N)]\!](v \mapsto l_v))$ if $S_2^{4\prime} l_y = n \neq 0$
- else $\perp$

By approximation properties for ∇ it holds that $S_2^{4\prime} l_y \sqsubseteq S_2^4 l_y$, this may give some extra $\perp$'s in the last case.

If we can prove that we get related continuations applied to states and values related in a $\rhd$ extended parameter, then we can conclude that we have the required termination approximation. It is enough to show that we at the primed places get something below in the $\sqsubseteq$ order. So it is enough to consider the cases where $S_2^{4\prime} l_y = S_2^4 l_y$.

By assumption $(K_1', K_2' \parallel K_1, K_2, (z : \tau)^\top, (pk^2)) \in \nabla_K$.

Define $\hat{r}_3$ by removal of one of the $\bar{\wedge}$ clauses in r_3,
$\hat{r}_3 = \big(\big((S_2 l_x \neq 0 \wedge S_1 l_w = S_2 l_v, \emptyset_{LL}), \emptyset_Z, A_{l_w}, A_{l_v, l_y}\big)$. Then $\hat{r}_3$ has only one trivial instantiation for continuations as well as for states, and $\hat{r}_3$ is an ordinary local parameter. $\hat{r}_3$ use the same locations as r_3.
Define $\hat{p}^3 = \{\hat{r}_3\} \cup p^2$, then $\hat{p}^3 \rhd p^2$. Also $\hat{p}_3 \in \mathfrak{sub}(p_3)$ and so $\hat{p}_3 \blacktriangleright p_3$.
Define $(\hat{p}k^3)$ as (pk^2) with only the trivial instantiation of $\hat{r}_3$ added, then $(\hat{p}k^3) \rhd (pk^2)$ and $(\hat{p}k^3) \in \hat{p}_3^{\mathbf{K}}$. Define $(\hat{p}ks^3)$ as (pks^2) with only the trivial instantiation of $\hat{r}_3$ added, then $(\hat{p}ks^3) \in (\hat{p}k^3)^{\mathbf{S}}$ and $(\hat{p}ks^3) \in (\hat{p}^3)^{\mathbf{S}}$.

By assumption $(S_1^{4\prime}, S_2^{4\prime} \parallel S_1^4, S_2^4, (pks^4)) \in \nabla_S$, where$(pk^4) \rhd (pk^3)$ and (pk_3) has the r_3 $\bar{\wedge}$-instantiation $(S_2 l_x = 0 \wedge S_2 l_v = 0 \wedge S_1 l_w = S_2 l_y \in \{0, 1\})$. Then $S_2^4 l_y = 0$ implies that $S_1^4 l_w = 0$, and in this case then $S_2^4[l_x \mapsto 1](l_v) = 0 = S_1^4 l_w$. Also $S_2^4 l_y \neq 0$ implies that $S_1^4 l_w = 1 \wedge S_2^4 l_v = 0$, in this case then $S_2^4[l_v \mapsto 1, l_x \mapsto 1](l_v) = 1 = S_1^4 l_w$.

Let $\hat{p}^4$, $(\hat{p}k^4)$ and $(\hat{p}ks^4)$ be defined as p^4, (pk^4) and (pks^4) with the only change that the instantiations of r_3 is replaced by $\hat{r}_3$ with trivial instantiations. Then $\hat{p}^4 \rhd \hat{p}^3$ and since $\hat{p}^3 \rhd p^2$ so $\hat{p}^4 \rhd p^2$. Also $\hat{p}^4 \in \mathfrak{sub}(p^4)$ and so $\hat{p}^4 \blacktriangleright p^4$. The only change in the updated states above in both cases are within the area owned by r_3 and the updates fulfill the new instantiation. Also disjointness properties required for state-relatedness are preserved. Since $\hat{p}^4 \blacktriangleright p^4$ it holds that the stored related values are still related. So the updated states from above are related in ∇_S under the parameter $(\hat{p}ks^4)$.

It remains to show that the pairs of setters and getters exported by M and N are related under the parameter $\hat{p}^4$. This requires that each of setter-values and getter-values are related. Recall that for values related at int type we require that $\exists n \in \mathbb{Z}. v_1' \in \{\bot, n\} \wedge v_1 = n \wedge v_2' \in \{\bot, n\} \wedge$ $v_2 = n$.

Let $\hat{p}^5 \blacktriangleright \hat{p}^4$, $(\hat{p}k^5) \in \hat{p^5}^{\mathbf{K}}$, $(\hat{p}ks^5) \in \hat{pk^5}^{\mathbf{S}}$. Assume $(\hat{k}_1', \hat{k}_2' \parallel \hat{k}_1, \hat{k}_2, (int)^\top, \hat{p}k^5)$, $(\hat{s}_1', \hat{s}_2' \parallel \hat{s}_1, \hat{s}_2, \hat{p}ks^5)$ and $(v_1', v_2' \parallel v_1, v_2, int, \hat{p}^5)$
Getters:
$[\![rec\ get_M(u : unit) = !w]\!](w \mapsto l_w) * \hat{k}_1\hat{s}_1 = \hat{k}_1(\hat{s}_1)(\hat{s}_1 l_w)$.
$[\![rec\ get_M(u : unit) = !w]\!](w \mapsto l_w) * \hat{k}_1'\hat{s}_1' \sqsubseteq \hat{k}_1'(\hat{s}_1')(\hat{s}_1' l_w)$.
$[\![rec\ get_N(u : unit) = !v]\!](v \mapsto l_v) * \hat{k}_2\hat{s}_2 = \hat{k}_2(\hat{s}_2)(\hat{s}_2 l_v)$.
$[\![rec\ get_N(u : unit) = !v]\!](v \mapsto l_v) * \hat{k}_2'\hat{s}_2' \sqsubseteq \hat{k}_2'(\hat{s}_2')(\hat{s}_2' l_v)$.
Since states are assumed related by the requirements of $\hat{r}_3$ it holds that $\exists n \in \mathbb{Z}.\ \hat{s}_1 l_w = \hat{s}_2 l_v = n$. By apporixmation properties for states $\hat{s}_1' l_w \in \{\bot, n\} \wedge \hat{s}_2' l_v \in \{\bot, n\}$. So we have continuations applied to states and values correspondingly related. We conclude that the getters are related.
Setters:
The assumption $(v_1', v_2' \parallel v_1, v_2, int, \hat{p}^5)$ implies that either $v_1' = v_2' = \bot$ or $\exists n \in \mathbb{Z}.\ v_1' \sqsubseteq v_1 = n \wedge v_2' \sqsubseteq v_2 = n$. The $\bot$ case is immediate. Else
$[\![rec\ set_M(n : int) = w := n]\!](w \mapsto l_w)(v_1)\hat{k}_1\hat{s}_1 = \hat{k}_1(\hat{s}_1[l_w \mapsto v_1])*$.
$[\![rec\ set_M(n : int) = w := n]\!](w \mapsto l_w)(v_1')\hat{k}_1'\hat{s}_1' \sqsubseteq \hat{k}_1'(\hat{s}_1'[l_w \mapsto v_1'])*$.
$[\![rec\ get_N(n : int) = v := n]\!](v \mapsto l_v)(v_2)\hat{k}_2\hat{s}_2 = \hat{k}_2(\hat{s}_2[l_v \mapsto v_2])*$.
$[\![rec\ get_N(n : int) = v := n]\!](v \mapsto l_v)(v_2')\hat{k}_2'\hat{s}_2' \sqsubseteq \hat{k}_2'(\hat{s}_2'[l_v \mapsto v_2'])*$.
The states have only been updated within the area owned by $\hat{r}_3$ and the invariant is preserved because the integer values v_1, v_2 must be identical. Also approximation requirement for states are preserved. Values *'s are related in any parameter. So we have continuations applied to states and values correspondingly related. We conclude that the setters are related.

We conclude that
$(\lambda S^0.\lambda d^0.[\![w \vdash val\ (rec\ get_M, rec\ set_M)]\!](w \mapsto l_w)K_1'S^0)$
$(\lambda S^0\lambda d^0.[\![x, y, v \vdash \text{if } (!y \neq 0) \text{ then } (v := 1); (x := 1);$
$\qquad val\ (rec\ get_N, rec\ set_N)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v)K_2'S^0) \parallel$
$(\lambda S^0.\lambda d^0.[\![w \vdash val\ (rec\ get_M, rec\ set_M)]\!](w \mapsto l_w)K_1S^0)$,
$(\lambda S^0\lambda d^0.[\![x, y, v \vdash \text{if } (!y \neq 0) \text{ then } (v := 1); (x := 1);$
$\qquad val\ (rec\ get_N, rec\ set_N)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v)K_2S^0), (x : \tau), (pk^3)) \in \nabla_K$.

And then we can conclude $([\![M]\!], [\![N]\!], T(int\ ref), T) \in \nabla_V^{\{g:\sigma\}}$.

7.3 Example. Export of locations from local area to visible.

The programs M and N below are open computations with one free variable g of function type. The programs were presented in the Introduction.

```
M:
let w ⇐ ref 0 in
  let z ⇐ g(rec f (a:unit)= (w:= 1)) in
    val w
```

```
N:
let x ⇐ ref 0 in
let y ⇐ ref 0 in
let v ⇐ ref 0 in
  let $z_0$ ⇐ g(rec f (a:unit)= (if (!x:= 0) then (v:=0; y:= 1) else (v:=1)) in
    let $z_1 \Leftarrow$ if $(!y \neq 0)$ then $(v := 1)$ in
    let $z_2 \Leftarrow (x := 1)$ in
      val v
```

We want to show the denotations of the two open computations are related in the ordinary parameter $T_\emptyset$, i.e. that $([\![M]\!], [\![N]\!], T(int\ ref), T_\emptyset) \in \nabla_M^{\emptyset, \{g:\sigma\}}$, where $\sigma = (unit \to Tunit) \to T\tau$

Let $p^1 \blacktriangleright T_\emptyset,\ (g_1', g_2' \parallel g_1, g_2, \sigma, p^1) \in \nabla_V$.
Let $p^2 \blacktriangleright p^1,\ (pk^2) \in p^{2\mathbf{K}},\ (pks^2) \in (pk^2)^{\mathbf{S}}$,
$(K_1', K_2' \parallel K_1, K_2, (z:\tau)^\top, (pk^2)) \in \nabla_K,\ (S_1', S_2' \parallel S_1, S_2, (pks^2)) \in \nabla_S$.

$[\![M]\!] g_1 K_1 S_1 =$
$\quad (g_1 [\![w \vdash rec\ f(a : unit) = (w := 1))]\!](w \mapsto l_w))$
$\qquad (\lambda S^0.\lambda d^0.[\![w \vdash val\ w]\!](w \mapsto l_w) K_1 S^0)$
$\qquad\quad S_1[l_w \mapsto 0]$,
where l_w is fresh.

$[\![N]\!] g_2 K_2 S_2 =$
$\quad (g_2 [\![x, y, v \vdash rec\ f(a : unit) = \text{if } (!x = 0) \text{ then } (y := 1) \text{ else } (v := 1)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v))$
$\qquad (\lambda S^0 \lambda d^0.[\![x, y, v \vdash \text{let } z_1 \Leftarrow \text{if } (!y \neq 0) \text{ then } (v := 1) \text{ in}$
$\qquad\qquad\qquad \text{let } z_2 \Leftarrow (x := 1) \text{ in } val\ v]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v) K_2 S^0)$
$\qquad\quad S_2[l_x \mapsto 0, l_y \mapsto 0, l_v \mapsto 0]$,
where l_x, l_y, l_v are fresh.

Define local vm-parameter $r_3 = \big([S_2 l_x \neq 0, \emptyset_{LL}], A_\emptyset, A_{\{l_x, l_y\}}, \{(l_w, l_v, int)\}\big) \bar{\wedge} \big([S_2 l_x = 0 \wedge S_2 l_v = 0 \wedge S_1 lw = S_2 l_y \in \{0, 1\}], A_{\{l_w\}}, A_{\{l_x, l_y, l_v\}}, \emptyset_Z\big)$.

By freshness of l_w and l_x, l_y, l_v we have a vm-parameter $p^3 = \{r_3\} \cup p^2$, and $p^3 \blacktriangleright p^2 \blacktriangleright p^1$

First we want to show that
$([\![w \vdash rec\ f(a : unit) = (w := 1))]\!](w \mapsto l_w)$,
$[\![x, y, v \vdash rec\ f(a : unit) = \text{if } (!x = 0) \text{ then } (v := 0; y := 1) \text{ else } (v := 1)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v) \parallel$
$[\![w \vdash rec\ f(a : unit) = (w := 1))]\!](w \mapsto l_w)$,
$[\![x, y, v \vdash rec\ f(a : unit) = \text{if } (!x = 0) \text{ then } (v := 0; y := 1) \text{ else } (v := 1)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v)$,
$unit \to Tunit,\ p^3) \in \nabla_V$.

Let $p^4 \blacktriangleright p^3$

Let $(pk^4_a) \in p^{4^{\mathbf{K}}}$ either s.t. $r_3 \in p^4$ with $\bar{\wedge}$-instantiation
$([S_2 l_x \neq 0, \emptyset_{LL}], \{(l_w, l_v, int)\}, A_\emptyset, A_{\{l_x,l_y\}})$ or s.t. p^4 extends p^3 by removal of $\bar{\wedge}$-clause and only $([S_2 l_x \neq 0, \emptyset_{LL}], \{(l_w, l_v, int)\}, A_\emptyset, A_{\{l_x,l_y\}}) \in P^4$ and so also in the k-instantiation.
Let $(pks^4_a) \in pk^{4^{\mathbf{S}}}_a$.

Let $(pk^4_b) \in p^{4^{\mathbf{K}}}$ either s.t. $r_3 \in p^4$ with $\bar{\wedge}$-instantiation
$([S_2 l_x = 0 \wedge S_2 l_v = 0 \wedge S_1 l_w = S_2 l_y \in \{0,1\}], \emptyset_Z, A_{\{l_w\}}, A_{\{l_x,l_y,l_v\}})$ or s.t. p^4 extends p^3 by removal of $\bar{\wedge}$-clause and only $([S_2 l_x = 0 \wedge S_2 l_v = 0 \wedge S_1 l_w = S_2 l_y \in \{0,1\}], \emptyset_Z, A_{\{l_w\}}, A_{\{l_x,l_y,l_v\}}) \in P^4$ and so also in the k-instantiation.
Let $(pks^4_b) \in pk^{4^{\mathbf{S}}}_b$.
Assume
$(k'_{a1}, k'_{a2} \parallel k_{a1}, k_{a2}, (x : unit)^\top, (pk^4_a)) \in \nabla_K$, $(s'_{a1}, s'_{a2} \parallel s_{a1}, s_{a2}, (pks^4_a)) \in \nabla_S$,
$(k'_{b1}, k'_{b2} \parallel k_{b1}, k_{b2}, (x : unit)^\top, (pk^4_b)) \in \nabla_K$, $(s'_{b1}, s'_{b2} \parallel s_{b1}, s_{b2}, (pks^4_b)) \in \nabla_S$.

$[\![w \vdash rec\ f(a : unit) = (w := 1))]\!](w \mapsto l_w) * k_{a1} s_{a1} = k_{a1}(s_{a1}[l_w \mapsto 1])*$

$[\![x, y, v \vdash rec\ f(a : unit)$ if $(!x = 0)$ then $(v := 0; y := 1)$ else
$(v := 1)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v) * k_{a2} s_{a2} =$
$k_{a2}(s_{a2}[l_v \mapsto 1])*$

By assumption the continuations and the original states are related with ordinary instantiation $([S_2 l_x \neq 0, \emptyset_{LL}], \{(l_w, l_v, int)\}, A_\emptyset, A_{\{l_x,l_y\}})$ where (l_w, l_v, int) belong to the visible location areas. The states have only been changed in l_w, l_v. In the updated states l_w, l_v hold related integer values, so the updated states are still related. Also the $*$ values are related. Hence we get the required termination behaviours.

$[\![w \vdash rec\ f(a : unit) = (w := 1))]\!](w \mapsto l_w) * k_{b1} s_{b1} = k_{b1}(s_{b1}[l_w \mapsto 1])*$

$[\![x, y, v \vdash rec\ f(a : unit) =$ if $(!x = 0)$ then $(v := 0; y := 1)$ else
$(v := 1)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v) * k_{b2} s_{b2} =$
$k_{b2}(s_{b2}[l_v \mapsto 0, l_y \mapsto 1])*$

By assumption the continuations are related with instantiation
$([S_2 l_x = 0 \wedge S_2 l_v = 0 \wedge S_1 l_w = S_2 l_y \in \{0,1\}], \emptyset_Z, A_{\{l_w\}}, A_{\{l_x,l_y,l_v\}})$. The original states are related, the states have only been changed in l_w, l_y, l_v. The updated states are still related with instantiation $(S_2 l_x = 0 \wedge S_2 l_v = 0 \wedge S_1 l_w = S_2 l_y \in \{0,1\})$. Also the $*$ values are related. Hence we get the required termination behaviours.

We conclude that $([\![w \vdash recf(a : unit) = (w := 1))]\!](w \mapsto l_w)$,
$[\![x, y, v \vdash \lambda a.$ if $(!x = 0)$ then $(v := 0; y := 1)$ else $(v := 1)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v) \parallel$
$[\![w \vdash recf(a : unit) = (w := 1))]\!](w \mapsto l_w)$,
$[\![x, y, v \vdash \lambda a.$ if $(!x = 0)$ then $(v := 0; y := 1)$ else $(v := 1)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v)$,
$unit \to Tunit,\ p^3) \in \nabla_V$

Since $p^3 \blacktriangleright p^1$ and by assumption $(g'_1, g'_2 \parallel g_1, g_2, \sigma, p^1) \in \nabla_V$ then
$g'_1([\![w \vdash recf(a : unit) = (w := 1))]\!](w \mapsto l_w)$,
$g'_2[\![x, y, v \vdash recf(a : unit) =$ if $(!x = 0)$ then $(v := 0; y := 1)$ else
$(v := 1)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v) \parallel$
$g_1[\![w \vdash recf(a : unit) = (w := 1))]\!](w \mapsto l_w)$,
$g_2[\![x, y, v \vdash recf(a : unit) =$ if $(!x = 0)$ then $(v := 0; y := 1)$ else $(v := 1)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v)$,
$T\tau,\ p^3) \in \nabla_M$

So we need to prove that these related computations are applied to continuations and states related in instantiations of a parameter ►-extending p^3 in:

$[\![M]\!]g_1K_1S_1 =$
$(g_1[\![w \vdash rec\ f(a : unit) = (w := 1))]\!](w \mapsto l_w))$
$(\lambda S^0.\lambda d^0.[\![w \vdash val\ w]\!](w \mapsto l_w)K_1S)\ S_1[l_w \mapsto 0]$

$[\![N]\!]g_2K_2S_2 =$
$(g_2[\![x, y, v \vdash rec\ f(a : unit) =$ if $(!x = 0)$ then $(v := 0; y := 1)$ else
$(v := 1)]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v))\ (\lambda S^0\lambda d^0.[\![x, y, v \vdash$ let $z_1 \Leftarrow$ if $(!y \neq 0)$ then $(v := 1)$ in
  let $z_2 \Leftarrow (x := 1)$ in $val\ v]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v)K_2S^0)$
$S_2[l_x \mapsto 0, l_y \mapsto 0, l_v \mapsto 0]$

Let $(pk^3) \in p^{3\mathbf{K}}$, $(pk^3) \supseteq (pk^2)$ with r_3 instantiation $(S_2l_x = 0 \wedge S_2l_v = 0 \wedge S_1l_w = S_2l_y \in \{0, 1\})$.
Let $(pks^3) \in pk^{3\mathbf{S}}$, $(pks^3) \supseteq (pks^2)$ (then (pks^3) has the r_3 instantiation
$(S_2l_x = 0 \wedge S_2l_v = 0 \wedge S_1l_w = S_2l_y \in \{0, 1\})$).

By assumption $(S_1', S_2' \parallel S_1, S_2, (pks^2)) \in \nabla_S$, then by weakening for the stored values, it holds that $(S_1'[l_w \mapsto 0], S_1[l_w \mapsto 0], S_2'[l_x \mapsto 0, l_y \mapsto 0, l_v \mapsto 0], S_2[l_x \mapsto 0, l_y \mapsto 0, l_v \mapsto 0], (pks^3)) \in \nabla_S$

It remains to prove that
$(\lambda S^0.\lambda d^0.[\![w \vdash val\ w]\!](w \mapsto l_w)K_1'S^0)$
$(\lambda S^0\lambda d^0.[\![x, y, v \vdash$ let $z_1 \Leftarrow$ if $(!y \neq 0)$ then $(v := 1)$ in
  let $z_2 \Leftarrow (x := 1)$ in $val\ v]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v)K_2'S^0) \parallel$
$(\lambda S^0.\lambda d^0.[\![w \vdash val\ w]\!](w \mapsto l_w)K_1S^0)$,
$(\lambda S^0\lambda d^0.[\![x, y, v \vdash$ let $z_1 \Leftarrow$ if $(!y \neq 0)$ then $(v := 1)$ in
  let $z_2 \Leftarrow (x := 1)$ in $val\ v]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v)K_2S^0), (x : \tau), (pk^3)) \in \nabla_K$.

Let
$(pk^4) \triangleright (pk^3)$, $(pks^4) \in (pk^4)^{\mathbf{S}}$, $(S_1^{4\prime}, S_2^{4\prime} \parallel S_1^4, S_2^4, (pks^4)) \in \nabla_S$, $(v_1^{4\prime}, v_2^{4\prime} \parallel v_1^4, v_2^4, \tau, (pk^4)^{vm}) \in \nabla_V$.

$(\lambda S^0.\lambda d^0.[\![w \vdash val\ w]\!](w \mapsto l_w)K_1S^0)S_1^4v_1^4 = [\![w \vdash val\ w]\!](w \mapsto l_w)K_1S_1^4 = K_1S_1^4l_w$

$(\lambda S^0\lambda d^0.[\![x, y, v \vdash$ let $z_1 \Leftarrow$ if $(!y \neq 0)$ then $(v := 1)$ in
  let $z_2 \Leftarrow (x := 1)$ in $val\ v]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v)K_2S^0)S_2^4v_2^4 =$
$[\![x, y, v \vdash$ let $z_1 \Leftarrow$ if $(!y \neq 0)$ then $(v := 1)$ in let $z_2 \Leftarrow (x := 1)$ in $val\ v]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v)$
$K_2S_2^4 =$

- $K_2S_2^4[l_x \mapsto 1]l_v$ if $S_2^4l_y = 0$
- $K_2S_2^4[l_v \mapsto 1, l_x \mapsto 1]l_v$ if $S_2^4l_y = n \neq 0$
-     else $\perp$

Since the original states were assumed related it holds that $S_2^4l_y \in \{0, 1\}$.
Let $p_1^4 \in \mathfrak{sub}((pk^4)^{vm})$ be $(pk^4)^{vm}$ where r_3 is replaced by the ordinary local parameter
$r_{31} = ([S_2l_x \neq 0], A_\emptyset, A_{\{l_x,l_y\}}\{(lw, lv, int)\})$, then $p_1^4 \triangleright p^2$. Let $(pks_1^4) \in p_1^{4\mathbf{S}}$ be (pks^4) with the r_3 instantiations replaced by $(r_{31}|r_{31}|(S_2l_x \neq 0))$. Then $(pks_1^4)^{vm} = p_1^4$.

Recall $(K_1', K_2' \parallel K_1, K_2, (x : \tau), p^2) \in \nabla_K$. We will prove that $(l_w, l_v \parallel l_w, l_v, int\ ref, p_1^4) \in \nabla_V$, and that in both cases the updated states are related under (pks_1^4). The required termination properties then follow by assumption and from $p_1^4 \triangleright p^2$.

By assumption $(S_1^{4\prime}, S_2^{4\prime} \parallel S_1^4, S_2^4, (pks^4)) \in \nabla_S$. By definition of the parameters $Z^{(pks_1^4)} = Z^{(pks^4)} \cup \{(l_w, l_v, int)\}$. If $S_2^4 l_y = 0$ then also $S_1^4 l_w = 0,\ S_2^4 l_v = 0$ and $S_2^4[l_x \mapsto 1]l_v = 0$. If $S_2^4 l_y \neq 0$ then also $S_1^4 l_w = 1$ and $S_2^4[l_v \mapsto 1, l_x \mapsto 1]l_v = 1$. So in both cases the updated states hold related integer values at locations l_w, l_v.
By weakening the stored values related in $(pks^4)^{vm}$ are related in any parameter $\blacktriangleright$-extending $((pks^4)^{vm})$ and hence in (pks_1^4). So we have that visible as well as hidden values required to be related are still related. Disjointness properties are also fulfilled. So the updated states are related under (pks_1^4).

We conclude that
$(\lambda S^0.\lambda d^0.[\![w \vdash val\ w]\!](w \mapsto l_w)K_1' S^0)$,
$(\lambda S^0 \lambda d^0.[\![x, y, v \vdash \text{let } z_1 \Leftarrow \text{if } (!y \neq 0) \text{ then } (v := 1) \text{ in}$
$\quad \text{let } z_2 \Leftarrow (x := 1) \text{ in } val\ v]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v)K_2' S^0) \parallel$
$(\lambda S^0.\lambda d^0.[\![w \vdash val\ w]\!](w \mapsto l_w)K_1 S^0)$,
$(\lambda S^0 \lambda d^0.[\![x, y, v \vdash \text{let } z_1 \Leftarrow \text{if } (!y \neq 0) \text{ then } (v := 1) \text{ in}$
$\quad \text{let } z_2 \Leftarrow (x := 1) \text{ in } val\ v]\!](x \mapsto l_x, y \mapsto l_y, v \mapsto l_v)K_2 S^0), (x : \tau)^\top, (pk^3)) \in \nabla_K$

And then we can conclude $([\![M]\!], [\![N]\!], T(int\ ref), T) \in \nabla_V^{\{g:\sigma\}}$.

7.4 Example. Allocation with reservation.

M:
$val\ (recf_M(n : int) = ref\ n)$

N:
let flag $\Leftarrow$ ref 0 in
let cell $\Leftarrow$ ref 0 in
$val\ recf_N(n : int)$=(if $(!flag = 0)$ then $(flag := 1; cell := n; val\ cell)$ else $(ref\ n)$)

M,N are closed computations. We want to show $([\![M]\!], [\![N]\!], T(int \to T(int\ ref)), T_\emptyset) \in \nabla_M^{\emptyset\emptyset}$.
Let $p \blacktriangleright T_\emptyset,\ (pk) \in p^{\mathbf{K}},\ (pks) \in (pk)^{\mathbf{S}}$. Assume
$(k_1', k_2' \parallel k_1, k_2, (x : int\ ref), (pk)) \in \nabla_K,\ (s_1', s_2' \parallel s_1, s_2, (pks)) \in \nabla_S$.

$[\![M]\!]k_1 s_1 = k_1 s_1 [\![rec\ f_M]\!]$.

$[\![N]\!]k_2 s_2 = k_2(s_2[l_f \mapsto 0, l_c \mapsto 0])([\![rec\ f_N]\!](flag \mapsto l_f, cell \mapsto l_c))$
where l_f, l_c are fresh.

So we want to find a parameter $\triangleright$-extending (pk) such that updated states and denotations of functions are related therein.

Define local parameter $r = \langle q_0 \prec \langle q_1 \rangle \rangle$
$q_0 = ((S_2 l_f = 0), A_\emptyset, A_{\{l_f, l_c\}}, \emptyset_Z)$ and $q_1 = ((S_2 l_f \neq 0), A_\emptyset, A_{\{l_f\}}, \emptyset_Z)$

Let $p^1 = p \cup \{r\}$, then $p^1 \triangleright p$.

We need to show that the functions are related in p^1.

Case a):
Let $p_a^2 \blacktriangleright p^1$ such that $\{r\}$ has been locally extended from q_0 to q_1. This means $q_1 \in p_a^2,\ r \notin p_a^2$.
$(pk_a^2) \in p_a^{2\mathbf{K}},\ (pks_a^2) \in (pk_a^2)^{\mathbf{S}}$. Assume
$(K_1', K_2' \parallel K_1, K_2, (x : int\ ref), (pk_a^2)) \in \nabla_K,\ (S_1', S_2' \parallel S_1, S_2, (pks_a^2)) \in \nabla_S$.

$([\![recf_M]\!]n)K_1S_1 = K_1S_1[l_1 \mapsto n]l_1$ where l_1 is fresh for $K_1,\ S_1$ and any location viewed by the parameter.

$([\![recf_N]\!]n)K_2S_2 = K_2S_2[l_2 \mapsto n]l_2$ where l_2 is fresh for $K_2,\ S_2$ and any location viewed by the parameter.

So now we want to show that the updated states are related in a $\rhd$-extension of (pk_a^2) and that the location values are related in the erasure of the same parameter.

Let $p_a^3 = p_a^2 \cup T_{(l_1,l_2,int)}$, then $p_a^3 \rhd p_a^2$. Let $(pk_a^3) \in (p_a^3)^{\mathbf{K}},\ (pk_a^3) \rhd (pk_a^2)$. By assumption the continuations are related in (pk_a^2). The updated states are related
$(S_1'[l_1 \mapsto n], S_2'[l_2 \mapsto n] \parallel S_1[l_1 \mapsto n], S_2[l_2 \mapsto n], pks_a^3)$ and the location values
$(l_1, l_2 \parallel l_1, l_2, int\ ref, p_a^3)$. So we get the required termination approximation.

Case b):
Let $p_b^2 \blacktriangleright p^1$ such that $\{r\}$ has not been locally extended, so $r \in p_b^2$. Let
$(pk_b^2) \in p_b^{2\mathbf{K}},\ (pks_b^2) \in (pk_b^2)^{\mathbf{S}}$. Assume
$(K_1', K_2' \parallel k_1, K_2, (x : int\ ref), (pk_b^2)) \in \nabla_K,\ (S_1', S_2' \parallel S_1, S_2, (pks_b^2)) \in \nabla_S$.

$([\![recf_M]\!]n)K_1S_1 = K_1S_1[l_1 \mapsto n]l_1$ where l_1 is fresh for $K_1,\ S_1$ and any location viewed by the parameter.

$([\![recf_N]\!]n)K_2S_2 = K_2S_2[l_f \mapsto 1, l_c \mapsto n]l_c$

As the continuations are assumed related, then we now want to show that the updated states are related in a parameter $\rhd$-extending pk_b^2 and that the location values $l_1,\ l_c$ are related in the erasure of the same parameter.

Let $p_b^3 = ((p_b^2 \setminus r) \cup q_1) \cup T_{(l_1,l_c,int)}$, then $p_b^3 \rhd p_b^2$. This follows from l_c cannot appear in $Z^{p_b^2}$. Also it is a requirement that all stored related values are related also in $(p_b^2 \setminus r) \cup q_1$ and her again l_c cannot appear in $Z^{(p_b^2 \setminus r) \cup q_1}$.

Let $(pk_b^3) \in (p_b^3)^{\mathbf{K}},\ (pk_b^3) \rhd (pk_b^2)$ and let $(pks_b^3) \in (pk_b^3)^{\mathbf{S}},\ (pks_b^3) \rhd (pks_b^2)$. By assumption the continuations are related in (pk_b^2). The updated states are related
$(S_1'[l_1 \mapsto n], S_2'[l_f \mapsto 1, l_c \mapsto n] \parallel S_1[l_1 \mapsto n], S_2[l_f \mapsto 1, l_c \mapsto n], pks_b^3)$ and the location values
$(l_1, l_c \parallel l_1, l_c, int\ ref, p_b^3)$. So we get the required termination approximation.

7.5 Example. Setter getter generator with ∀ type.

```
M = $\Lambda\alpha$. val (rec $f$ ($g$: $\alpha$): $T((\alpha \to T\text{unit}) \times (\text{unit} \to T\alpha))$ =
                    let $y \Leftarrow$ ref $g$ in
                    let set $\Leftarrow$ val (rec $f_{1M}(g_1 : \alpha) : T\text{unit} = y := g_1$) in
                    let get $\Leftarrow$ val (rec $f_{2M}(x : \text{unit}) : T\alpha = !y$) in
                        (set,get))
```

```
N = Λα. val (rec f (g: α): T((α → Tunit) × (unit → Tα)) =
                  let y_0 ⇐ ref g in
                  let y_1 ⇐ ref g in
                  let p ⇐ ref 0 in
                  let set ⇐ val (rec f_1N(g_1 : α) : Tunit =
                                    if iszero(!p) then
                                      (p := 1; y_1 := g_1)
                                    else
                                      (p := 0; y_0 := g_1)) in
                  let get ⇐ val (rec f_2N(x : unit) : Tα =
                                      if iszero(!p) then !y_0 else !y_1) in
                     (set,get) )
```

M and N are closed values of polymorphic type M,N:$\forall\alpha.T(\alpha \to ((\alpha \to Tunit),(unit \to T\alpha)))$. To shorten notation we let $\forall\alpha.T\tau = \forall\alpha.T(\alpha \to ((\alpha \to Tunit),(unit \to T\alpha)))$.

We want to show $([\![;;\vdash M]\!]\{\}, [\![;;\vdash N]\!]\{\}, \forall\alpha.T\tau, T_\emptyset) \in \nabla_V^{\emptyset,\emptyset}$ that is $([\![M]\!],\ [\![N]\!] \parallel [\![M]\!],\ [\![N]\!], \forall\alpha.T\tau, T_\emptyset) \in \nabla_V$.

$[\![;;\vdash M]\!]\{\} = in_\forall\lfloor[\![;\alpha;\vdash val\ rec\ f_M]\!]\{\}\rfloor$ and $[\![;;\vdash N]\!]\{\} = in_\forall\lfloor[\![;\alpha;\vdash val\ rec\ f_N]\!]\{\}\rfloor$ so we need to show, $\forall - \vdash \sigma : type.$ $([\![;\alpha;\vdash val\ rec\ f_M]\!],\ [\![;\alpha;\vdash val\ rec\ f_N]\!] \parallel [\![;\alpha;\vdash val\ rec\ f_M]\!],\ [\![;\alpha;\vdash val\ rec\ f_N]\!], \sigma \to T(\sigma \to Tunit, unit \to T\sigma), T_\emptyset) \in \nabla_M$.

Let $- \vdash \sigma$ be any closed type, and shorten $\tau' = \sigma \to T(\sigma \to Tunit, unit \to T\sigma)$. Let $p \blacktriangleright T_\emptyset,\ (pk) \in p^{\mathbf{K}},\ (pks) \in (pk)^{\mathbf{S}}$ and assume $(k_1', k_2' \parallel k_1, k_2, (x : \tau')^\top, (pk)) \in \nabla_K$ and $(s_1', s_2' \parallel s_1, s_2, (pks)) \in \nabla_S$.

$[\![;\alpha;\vdash val\ rec\ f_M]\!]\{\}k_1s_1 = k_1s_1[\![;\alpha;\vdash\ rec\ f_M]\!]\{\}$

$[\![;\alpha;\vdash val\ rec\ f_N]\!]\{\}k_2s_2 = k_2s_2[\![;\alpha;\vdash\ rec\ f_N]\!]\{\}$

Now we want to prove $([\![;\alpha;\vdash\ rec\ f_M]\!]\{\}, [\![;\alpha;\vdash\ rec\ f_N]\!]\{\} \parallel [\![;\alpha;\vdash\ rec\ f_M]\!]\{\}, [\![;\alpha;\vdash rec\ f_N]\!]\{\}, \sigma \to T(\sigma \to Tunit, unit \to T\sigma), p) \in \nabla_V$.
The denotations have the right format so we can let $[\![;\alpha;\vdash\ rec\ f_M]\!]\{\} = in_\multimap\lfloor d_M\rfloor$ and let $[\![;\alpha;\vdash\ rec\ f_N]\!]\{\} = in_\multimap\lfloor d_N\rfloor$. Now let $p^1 \blacktriangleright p$ and assume $(v_1', v_2' \parallel v_1, v_2, \sigma, p^1) \in \nabla$. Then next we aim to prove $(d_m v_1',\ d_N v_2' \parallel d_M v_1,\ d_N v_2, T(\sigma \to Tunit, unit \to T\sigma), p^1) \in \nabla_M$.
Let $p^2 \blacktriangleright p^1,\ (pk^2) \in (p^2)^{\mathbf{K}},\ (pks^2) \in (pk^2)^{\mathbf{S}}$ and assume $(K_1', K_2' \parallel K_1, K_2, (x : (\sigma \to Tunit, unit \to T\sigma))^\top, (pk^2)) \in \nabla_K$ and $(S_1', S_2' \parallel S_1, S_2, (pks^2)) \in \nabla_S$.

$(d_M v_1)K_1S_1 = K_1(S_1[l_y \mapsto v_1])(v_{setM}, v_{getM})$
$(d_N v_2)K_2S_2 = K_2(S_2[l_{y0} \mapsto v_1, l_{y1} \mapsto v_1, l_p \mapsto 0])(v_{setN}, v_{getN})$
where l_y, l_{y0}, l_{y1}, l_p are fresh.
Now we want to find a local parameter we can add to $\rhd$-extend, such that updated states and function values are related therein.
Let $r = ((S_2l_p = 0, (l_y, l_{y0}, \sigma))\triangledown(S_2l_p \neq 0, (l_y, l_{y0}, \sigma)), A_{\{l_y\}}, A_{\{l_{y0}, l_{y1}, l_p\}}, \emptyset_Z$ and let $p^3 = p^2 \cup \{r\}$.
Let (pks^3) be $(pks^2) \cup \{(r|r|(S_2l_p \neq 0, (l_y, l_{y0}, \sigma)))\}$. Since the locations are fresh and by parameter weakening for the stored values it holds that $(S_1'[l_y \mapsto v_1'], S_2'[l_{y0} \mapsto v_1', l_{y1} \mapsto v_1', l_p \mapsto 0] \parallel S_1[l_y \mapsto v_1], S_2[l_{y0} \mapsto v_1, l_{y1} \mapsto v_1, l_p \mapsto 0], (pks^3)) \in \nabla_S$.
It remains to show that setters and getters are related in p^3.

Setters: Let $p^5 \blacktriangleright p^4 \blacktriangleright p^3$, $(pk^5) \in (p^5)^{\mathbf{K}}$, $(pks^5) \in (pk^5)^{\mathbf{S}}$ and assume
$(w_1', w_2' \parallel w_1, w_2, \sigma, p^4) \in \nabla_V$, $(\kappa_1', \kappa_2' \parallel \kappa_1, \kappa_2, (x : unit)^\top, (pk^5)) \in \nabla_K$ and
$(\varsigma_1', \varsigma_2' \parallel \varsigma_1, \varsigma_2, (pks^5)) \in \nabla_S$.
$d_{setM} w_1 \kappa_1 \varsigma_1 = \kappa_1(\varsigma_1[l_y \mapsto w_1]*)$
$d_{setN} w_2 \kappa_2 \varsigma_2 = \kappa_2(\varsigma_2[l_p \mapsto 1, l_{y1} \mapsto w_2]*)$ if $\varsigma_2 l_p = 0$.
$d_{setN} w_2 \kappa_2 \varsigma_2 = \kappa_2(\varsigma_2[l_p \mapsto 0, l_{y0} \mapsto w_2]*)$ if $\varsigma_2 l_p \neq 0$.
If the original states were related in an instantiation of r to $(S_2 l_p = 0, (l_y, l_{y0}, \sigma)$ then the updated states are related in an instantiation of r to $(S_2 l_p \neq 0, (l_y, l_{y1}, \sigma)$ and vice versa. Also $*$ values are related. Since the continuations were related by assumption we get the desired termination behaviour. So we conlude that $(d_{setM}, d_{setN} \parallel d_{setM}, d_{setN}, \sigma \to Tunit, p^3) \in \nabla_V$.

Getters: Let $p^5 \blacktriangleright p^4 \blacktriangleright p^3$, $(pk^5) \in (p^5)^{\mathbf{K}}$, $(pks^5) \in (pk^5)^{\mathbf{S}}$ and assume
$(w_1', w_2' \parallel w_1, w_2, unit, p^4) \in \nabla_V$, $(\kappa_1', \kappa_2' \parallel \kappa_1, \kappa_2, (x : \sigma)^\top, (pk^5)) \in \nabla_K$ and
$(\varsigma_1', \varsigma_2' \parallel \varsigma_1, \varsigma_2, (pks^5)) \in \nabla_S$.
$d_{setM} w_1 \kappa_1 \varsigma_1 = \kappa_1 \varsigma_1(\varsigma_1 l_y)$
$d_{setN} w_2 \kappa_2 \varsigma_2 = \kappa_2 \varsigma_2(\varsigma_2 l_{y0})$ if $\varsigma_2 l_p = 0$.
$d_{setN} w_2 \kappa_2 \varsigma_2 = \kappa_2 \varsigma_2(\varsigma_2 l_{y1})$ if $\varsigma_2 l_p \neq 0$.
If the states were related in an instantiation of r to $(S_2 l_p = 0, (l_y, l_{y0}, \sigma)$ then the retrived values are related in p^5, and the same holds if the states were related in an instantiation of r to $(S_2 l_p \neq 0, (l_y, l_{y1}, \sigma)$. Since the continuations and the states were related by assumption we get the desired termination behaviour. So we conlude that
$(d_{getM}, d_{getN} \parallel d_{getM}, d_{getN}, unit \to T\sigma, p^3) \in \nabla_V$.
Then we can further conclude $(d_M v_1', d_N v_2' \parallel d_M v_1, d_N v_2, T(\sigma \to Tunit, unit \to T\sigma), p^1) \in \nabla_M$ and then also $(\llbracket ; \alpha ; \vdash\ rec\ f_M \rrbracket\{\}, \llbracket ; \alpha ; \vdash\ rec\ f_N \rrbracket\{\} \parallel \llbracket ; \alpha ; \vdash\ rec\ f_M \rrbracket\{\}, \llbracket ; \alpha ; \vdash\ rec\ f_N \rrbracket\{\}, \sigma \to T(\sigma \to Tunit, unit \to T\sigma), p) \in \nabla_V$. As σ was any arbitrary closed type we can now finally conclude $(\llbracket ; ; \vdash M \rrbracket\{\}, \llbracket ; ; \vdash N \rrbracket\{\}, \forall\alpha.T\tau, T_\emptyset) \in \nabla_V^{\emptyset;\emptyset}$ that is
$(\llbracket M \rrbracket,\ \llbracket N \rrbracket \parallel \llbracket M \rrbracket,\ \llbracket N \rrbracket, \forall\alpha.T\tau, T_\emptyset) \in \nabla_V$.

7.6 Non-example. Snapback is not in the relation at most types

We define
$snapback = in_{\multimap}\lfloor \lambda f \in \mathbb{V}.(\lambda K \in \mathbb{K}.\lambda S \in \mathbb{S}.if\ \big(exists\ \hat{f}.f = in_{\multimap}\lfloor \hat{f} \rfloor\big)\ then\big((\hat{f}*)(\lambda\hat{s}.\lambda d.KSd)S\big)\ else\ \bot)\rfloor$.

snapback is not the denotation of any term, but it is an element in $\mathbb{V}$.

At most types of the form $(unit \to T\tau) \to T\tau$ snapback is not related to itself at the parameters T_Δ. We will show a counter example, and then look at the general principle behind. So we give examples of related functions, related continuations and related states such that when *snapback* is applied to the them termination properties differ.

We now aim to show
$(snapback, snapback, (unit \to T(unit \to Tunit)) \to T(unit \to Tunit), T_\Delta) \notin \nabla_V$.
Let $g = in_{\multimap}\lfloor \lambda v \in \mathbb{V}.\lambda k \in \mathbb{K}.\lambda s \in \mathbb{S}.ks(in_{\multimap}\lfloor \lambda v' \lambda k'.\lambda s'.k's'in_{\mathbf{1}}\lfloor * \rfloor \rfloor)\rfloor$.

For all $n \in \mathbb{Z}$ let $g_n = in_{\multimap}\lfloor \lambda v \in \mathbb{V}.(\lambda k \in \mathbb{K}.\lambda s \in \mathbb{S}.k(s[l \mapsto n])(in_{\multimap}\lfloor \lambda v'.\lambda k'.\lambda s'.\ if\ s'l = n\ then\ (k's'in_{\mathbf{1}}\lfloor * \rfloor\rfloor)\ else\ \bot))\rfloor$, where $l \notin dom(\Delta) \cup supp(k) \cup supp(s)$.
It holds that
$(g, g_n \parallel g, g_n, unit \to T(unit \to Tunit), T_\Delta) \in \nabla_V$.

To see this, let $q_2 \blacktriangleright q_1 \blacktriangleright T_\Delta$, $(qk_2) \in q_2^{\mathbf{K}}$, $(qks_2) \in (qk_2)^{\mathbf{S}}$ and assume
$(V_1', V_2' \parallel V_1, V_2, unit, q_1) \in \nabla_V$, $(K_1', K_2' \parallel K_1, K_2, (x : unit \to Tunit), (qk_2)) \in \nabla_K$ and
$(S_1', S_2' \parallel S_1, S_2, (qks_2)) \in \nabla_S$. The $\bot$ cases at primed elements are trivial.

$gV_1K_1S_1 = K_1S_1(in_{\multimap}\lfloor \lambda v'\lambda k'.\lambda s'.k's'in_{\mathbf{1}}\lfloor * \rfloor\rfloor)$
$g_nV_2K_2S_2 = K_2(S_2[l \mapsto n])(in_{\multimap}\lfloor \lambda v'.\lambda k'.\lambda s'.\ if\ s'l = n\ then\ (k's'in_{\mathbf{1}}\lfloor * \rfloor\rfloor)\ else\ \bot)$, where
$l \notin dom(\mathit{\Delta}) \cup supp(K_2) \cup supp(S_2)$.

Define local parameter $r_n = (\{(s_1, s_2)|s_2l = n\}, \emptyset_{LL}), A_\emptyset, A_{\{l\}}, \emptyset_Z$. Let $q_{n3} = q_2 \cup \{r_n\}$, $(qks_{n3}) = (qks_2) \cup \{(r_n|r_n|(\{(s_1, s_2)|s_2l = n\})\}$ then $q_{n3} \in \mathfrak{p}^{vm}$, $q_{n3} \rhd q_2$, $(qks_{n3}) \rhd (qks_2)$. Since the original states were related in (qks_2) and l is fresh, then the updated states are related in (qks_{n3}). So it remains to show that the function-values are related in q_{n3}.

Let $q_5 \blacktriangleright q_4 \blacktriangleright q_{n3}$, $(qk_5) \in q_5^{\mathbf{K}}$, $(qks_5) \in (qk_5)^{\mathbf{S}}$ and assume $(V_1', V_2' \parallel V_1, V_2, unit, q_4) \in \nabla_V$, $(K_1', K_2' \parallel K_1, K_2, (x : unit \to Tunit), (qk_5)) \in \nabla_K$ and $(S_1', S_2' \parallel S_1, S_2, (qks_5)) \in \nabla_S$. The $\bot$ cases at primed elements are trivial. Else, since the states are related in a parameter extending (qks_{n3}) it must be the case that $S_2l = n$ so we get related continuations applied to related states and related values:
$(in_{\multimap}\lfloor \lambda v'\lambda k'.\lambda s'.k's'in_{\mathbf{1}}\lfloor * \rfloor\rfloor)V_1K_1S_1 = K_1S_1in_{\mathbf{1}}\lfloor * \rfloor$
$(in_{\multimap}\lfloor \lambda v'.\lambda k'.\lambda s'.\ if\ s'l = 1\ then\ (k's'in_{\mathbf{1}}\lfloor * \rfloor\rfloor)\ else\ \bot)V_2K_2S_2 = K_2S_2in_{\mathbf{1}}\lfloor * \rfloor$
so we conclude that the function-values are related under q_{n3} and then we can conclude that $(g, g_n \parallel g, g_n, unit \to T(unit \to Tunit), T_{\mathit{\Delta}}) \in \nabla_V$ for each n.

Let $(s_1', s_2' \parallel s_1, s_2, T_{\mathit{\Delta}}) \in \nabla_S$ where $s_1' \neq \bot$, and let
$k = \lambda s.\lambda d.(if\ d = in_{\multimap}\lfloor d' \rfloor\ then\ \big(d'(in_1\lfloor * \rfloor)(\lambda s'.(\lambda d'.(\top)_\bot)_\bot)s\big)\ else\ \bot)$. Then $(k, k \parallel k, k, (x : unit \to Tunit)^\top, T_{\mathit{\Delta}}) \in \nabla_K$. To see this let $(pk) \rhd T_{\mathit{\Delta}}$, $(pks) \in (pk)^{\mathbf{S}}, p = (pk)^{vm}$ and assume $(S_1', S_2' \parallel S_1, S_2, (pks)) \in \nabla_S$ and $(h_1', h_2' \parallel h_1, h_2, unit \to Tunit, p) \in \nabla_V$. The $\bot$ cases are triviel. Else this requires there are d_1', d_2', d_1, d_2 such that $h_1' \sqsubseteq h_1 = in_{\multimap}\lfloor d_1 \rfloor$ and $h_2' \sqsubseteq h_2 = in_{\multimap}\lfloor d_2 \rfloor$ where $(h_1' = \bot \wedge d_1' = \bot)$ or $h_1' = in_{\multimap}\lfloor d_1' \rfloor$ and $(h_2' = \bot \wedge d_2' = \bot)$ or $h_1' = in_{\multimap}\lfloor d_2' \rfloor$. Then $kS_1'V_1' = d_1'(in_1\lfloor * \rfloor)(\lambda s'.(\lambda d'.(\top)_\bot)_\bot)S_1'$,
$kS_1V_1 = d_1(in_1\lfloor * \rfloor)(\lambda s'.(\lambda d'.(\top)_\bot)_\bot)S_1$ and $kS_2'V_2' = d_2'(in_1\lfloor * \rfloor)(\lambda s'.(\lambda d'.(\top)_\bot)_\bot)S_2'$
$kS_2V_2 = d_2(in_1\lfloor * \rfloor)(\lambda s'.(\lambda d'.(\top)_\bot)_\bot)S_2$. Here
$(d_1'(in_1\lfloor * \rfloor), d_2'(in_1\lfloor * \rfloor) \parallel d_1(in_1\lfloor * \rfloor), d_2(in_1\lfloor * \rfloor), Tunit, p) \in \nabla_M$ by assumption about related $h's$. We have seen the $(\lambda s'.(\lambda d'.(\top)_\bot)_\bot)$ are related in all ordinary parameters. The states were related by assumption. So we conclude $(k, k \parallel k, k, (x : unit \to Tunit)^\top, T_{\mathit{\Delta}}) \in \nabla_K$.

Let $l \notin dom(\mathit{\Delta}) \cup supp(k) \cup s_2$. Chose n such that $in_{\mathbb{Z}}\lfloor n \rfloor$ is different from s_2l. We now apply *snapback* to related values, related continuations and related states, and we see that this does not give the required termination approximation.

$(snapback(g))ks_1' = (g*)(\lambda\hat{s}.\lambda d.ks_1'd)s_1' = (\lambda\hat{s}.\lambda d.ks_1'd)(s_1')(in_{\multimap}\lfloor \lambda v'\lambda k'.\lambda s'.k's'in_{\mathbf{1}}\lfloor * \rfloor\rfloor) =$
$ks_1'(in_{\multimap}\lfloor \lambda v'\lambda k'.\lambda s'.k's'in_{\mathbf{1}}\lfloor * \rfloor\rfloor) = (\lambda v'\lambda k'.\lambda s'.k's'in_{\mathbf{1}}\lfloor * \rfloor)(in_1\lfloor * \rfloor)(\lambda s'.(\lambda d'.(\top)_\bot)_\bot)s_1' = \top.$

$(snapback(g_n))ks_2 = (g_n*)(\lambda\hat{s}.\lambda d.ks_2d)s_2 =$
$(\lambda\hat{s}.\lambda d.ks_2d)(s_2[l \mapsto n])(in_{\multimap}\lfloor \lambda v'.\lambda k'.\lambda s'.\ if\ s'l = n\ then\ (k's'in_{\mathbf{1}}\lfloor * \rfloor\rfloor)\ else\ \bot) =$
$ks_2(in_{\multimap}\lfloor \lambda v'.\lambda k'.\lambda s'.\ if\ s'l = n\ then\ (k's'in_{\mathbf{1}}\lfloor * \rfloor)\ else\ \bot\rfloor) =$
$(\lambda v'.\lambda k'.\lambda s'.\ if\ s'l = n\ then\ (k's'in_{\mathbf{1}}\lfloor * \rfloor)\ else\ \bot)(in_1\lfloor * \rfloor)(\lambda s'.(\lambda d'.(\top)_\bot)_\bot)s_2 = \bot$
where we have used that $s_2l \neq n$.

We conclude $(snapback, snapback, (unit \to T(unit \to Tunit)) \to T(unit \to Tunit), T_{\mathit{\Delta}}) \notin \nabla_V$

The types *unit* and *int* and types build up only from those have the special property that relatedness in ∇ does not depend on the vm-parameters. The counterexamples use that new locations can be allocated and stored values expected to have certain properties, but *snapback* will erase this.

8 Parameterized logical relation with further extended parameters

We will now define an extended version of the relation we have used to prove equivalences in the previous section. The extension is designed to (additionally) handle explicit divergence. We do much of the same development again. First we define the new parameters, which are extended versions of the ones we have met so far. Then we define a new admissible action of the domain construction functor F on relations, and prove that the new relation exists. As before we extract a binary relation on which we base proofs of contextual equivalences. Finally we analyse the example from the introduction 1.4. We have chosen to use the same name ∇ again for the new relation, and also for sets of parameters we reuse the names. That should not give any problems, since the following development stand alone, so there is no doubt about what names refer to. In the following sections many definitions and proofs from before are repeated in versions with only differences at certain places following the new parameter definition.

8.1 Parameters further extended

Ordinary parameters in the new set up are exactly as ordinary parameters before. Ordinary parameters are almost as the parameters from [13] with local extensions added. As before ordinary parameter are preserved by each of related computations and related continuations separately. To the parameters from the previous section we now add a way to express, that one side has already diverged. This is essentially the only difference, but we need to define how it is handled. Especially we give a new version of the parameterized relation. We also again define order relations on parameters.

8.2 Parameters and specialized parameters

As said ordinary parameters are exactly as before. We will now define general parameters which will be either ordinary or specialized. Ordinary parameters are used, when we relate computations where we don't know any specific properties of the continuations, they will be applied to. Specialized parameters are used, when we know the initial steps of the continuations, they are applied to. We use specialized parameters, if we know that states initially belong in one local-parameter-component and the initial steps of the continuations change states to another local-parameter-component. We also use specialized parameters in situations, where both sides eventually diverge, but not at the same step of execution. First we define a new version of local-parameter-parts.

Definition 35. *Local-parameter-part Q*
Given accessibility maps A_1 and A_2.
Then Q is an (A_1, A_2)-local-parameter-part with associated sets L_1^Q, L_2^Q if one of the following holds:

1. *$Q = (P, LL)$ and P is an (A_1, A_2)-state-relation and $LL =$ $\{ (l^{11}, l^{12}, \tau^1), \ldots, (l^{k1}, l^{k2}, \tau^k) \}$ is a finite* set *of location pairs and closed value types, $k \geq 0$. $L_1^Q = \pi_1(LL)$ and $L_2^Q = \pi_2(LL)$.*
2. *$Q = (P, \perp)$ and P is an A_1-state-predicate. $L_1^Q = L_2^Q = \emptyset$.*
3. *$Q = (\perp, P)$ and P is an A_2-state-predicate. $L_1^Q = L_2^Q = \emptyset$.*

Recall that in ordinary parameters all local-parameter-parts must have the form (P, LL).

The informal understanding of (P, LL) for a pair of states (S_1, S_2) is that $(S_1, S_2) \in P$ and LL hold related values. The informal understanding of $(P, \perp)$ and $(\perp, P)$ for a state S is that $S \in P$ and the other side of what we aim to prove equivalent has already diverged.

Definition 36. *Local-parameter-node q*
Let I be a finite index set.
$q = (\{Q_i | i \in I\}, A_1^q, A_2^q, Z^q) = ([Q_1 \bar{\triangledown} \dots \bar{\triangledown} Q_k], A_1^q, A_2^q, Z^q)$, *is a local-parameter-node iff*

- A_1^q, A_2^q *are accessibility maps.*
- *Each Q_i is an (A_1^q, A_2^q)-local-parameter-part*
- *Either all Q_i have the form (P, LL) or all Q_i have the form $(P, \perp)$ or all Q_i have the form $(\perp, P)$.*
- Z^q *is a match of finite store types.*
- $\bigcup_i L_1^{Q_i} \cap \pi_1(Z^q) = \emptyset \ \wedge \ \bigcup_i L_2^{Q_i} \cap \pi_2(Z^q) = \emptyset$

To the local-parameter-node $q = (\{Q_i | i \in I\}, A_1^q, A_2^q, Z^q)$ are associated the accessibility maps $\mathring{A}_1^q, \mathring{A}_2^q$ where
$\forall S.\ \mathring{A}_1^q(S) = A_1^{q_k}(S) \cup \pi_1 Z^q \cup \bigcup_{i \in I} L_1^{Q_i}$ *and*
$\mathring{A}_2^r(S) = A_2^{q_k}(S) \cup \pi_2 Z^q \cup \bigcup_{i \in I} L_2^{Q_i}$.
$\mathring{A}_1^r, \mathring{A}_2^r$ *are the most inclusive accessibility maps associated with the local-parameter-node, encompass locations meant to be visible as well as hidden.*

To the local-parameter-node q are associated fixed finite sets of locations $L_1^q = \bigcup_i L_1^{Q_i}$ and $L_2^q = \bigcup_i L_2^{Q_i}$.

Definition 37. *$\geq$ order on local-parameter-nodes*
Let d, e be local-parameter-nodes, and $d = (\{Q_i^d | i \in I_d\}, A_1^d, A_2^d, Z^d)$, $e = (\{Q_i^e | i \in I_e\}, A_1^e, A_2^e, Z^e)$
$e \geq d$ iff

- $Z^e \supseteq Z^d$ *and*
- *If the Q^d's have the form $(\perp, P)$ then the Q^e's have the form $(\perp, P)$, and*
- *if the Q^d's have the form $(P, \perp)$ then the Q^e's have the form $(P, \perp)$ and*
- $\forall (S_1, S_2).\ \mathring{A}_1^d(S_1) \supseteq \mathring{A}_1^e(S_1) \wedge \mathring{A}_2^d(S_2) \supseteq \mathring{A}_2^e(S_2)$.

Definition 38. *Local-parameter-component $\hat{q}$*

A local-parameter-component $\hat{q}$ is a finite tree, where each node q_i is a local-parameter-node and it holds that

- $\forall q_1, q_2 \in q.$ *if q_1 is an ancestor of q_2 then $q_2 \geq q_1$*

To the local-parameter-component $\hat{q}$ with root-node q_0 is associated the match of finite store types $Z^q = Z^{q_0}$.

To the local-parameter-component $\hat{q}$ with root node q_0 are associated the accessibility maps $\mathring{A}_1^q = \mathring{A}_1^{q_0}$ and $\mathring{A}_2^q = \mathring{A}_2^{q_0}$.
$\mathring{A}_1^q, \mathring{A}_2^q$ *are the most inclusive accessibility maps associated with the local-parameter-coponent (all locations possibly "owned" by subtrees are included c.f. order on tree-nodes).*

To the local-parameter-component $\hat{q}$ are associated fixed finite sets of locations W_1^q and W_2^q.
$W_1^q = \bigcup_{q_j \in \hat{q}} \big(\pi_1(Z^{q_j} \cup L_1^{q_j})\big)$ *and* $W_2^q = \bigcup_{q_j \in \hat{q}} \big(\pi_2(Z^{q_j} \cup L_2^{q_j})\big)$
where for $q_j = ((P_i, LL_i) | i \in I\}, A_1^{q_j}, A_2^{q_j}, Z^{q_j})$ we let $L_1^{q_j} = \cup_i L_1^{Q_i}$, $L_2^{q_j} = \cup_i L_2^{Q_i}$.

For notational convenience when not otherwise indicated we let the root node of $\hat{q}$ be named q.

Intuitively the fixed sets W_1^q, W_2^q give locations (visible as well as hidden) that we know can be associated with q without knowing any state.

Intuitively we identify a local-parameter-component with its root. The rest of the tree is there to tell us how a local parameter may be locally extended.

Definition 39. *$\succeq$ relation on local-parameter-components.*
Let $\hat{q}'$ and $\hat{q}$ be local-parameter-components.

$\hat{q}' \succeq \hat{q}$ *iff* $\hat{q}' \in subtrees(\hat{q})$

We now define local parameters. As before a local parameter for values and computations is a finite set of local-parameter-components. A local parameter for continuations and a local parameter for states will "belong" to *one* of the components. The idea is that the values and computations preserve each component in states, but continuations may change states between components.

Definition 40. *Local parameters*

(vm) A local vm-parameter has the form $r = \{\hat{q}_1 \ldots \hat{q}_n\} = (\hat{q}_1 \bar{\wedge} \ldots \bar{\wedge} \hat{q}_n)$ where
- *$\forall i \in 1 \ldots n.\ \hat{q}_i$ is a local parameter component*
- *$n \geq 1$*

(k) If $r = (\hat{q}_1 \bar{\wedge} \ldots \bar{\wedge} \hat{q}_n)$ is a local vm-parameter, then $\forall j \in 1 \ldots n.\ (r|\hat{q}_j)$ is a local k-parameter. $\underline{\hat{q}_j \text{ is a choiche of } \bar{\wedge}\text{-clause}}$.

(s) If $(r|\hat{q})$ is a local k-parameter, and $root(\hat{q}) = (\{Q_i | i \in I\}, A_1^q, A_2^q, Z^q) = ([Q_1 \bar{\vee} \ldots \bar{\vee} Q_j \bar{\vee} \ldots], A_1^q, A_2^q, Z^q)$, then for each $i \in I$ $(r|\hat{q}|Q_i)$ is a local s-parameter. $\underline{Q_i \text{ is a choice of } \bar{\vee}\text{-clause}}$.

A local parameter where $n = 1$ and all local parameter parts are ordinary is an ordinary *local parameter. So if all parts in $\hat{q}$ have the form (P, LL) then $(\hat{q})$ is an ordinary vm-parameter, $(\hat{q}|\hat{q})$ is an ordinary k-parameter and for all $i \in I$ $(\hat{q}|\hat{q}|Q_i)$ is an ordinary s-parameter.*

To the local parameter $r = \{\hat{q}_1 \ldots \hat{q}_n\}$ is associated the match of finite store types

$Z^{\cap r} = \bigcap_{k \in 1..n} Z^{q_k}$.

$Z^{\cap r}$ gives the locations visible for every $\bar{\wedge}$-clause in the local parameter r

To the local parameter r are associated the fixed finite sets of locations $W_1^r = \bigcup_{i \in 1..n} W_1^{q_i}$ and $W_2^r = \bigcup_{i \in 1..n} W_2^{q_i}$.

To the local parameters $r = \{\hat{q}_1 \ldots \hat{q}_n\}$ are associated the accessibility maps $\mathring{A}_1^r, \mathring{A}_2^r$ where

$\forall S.\ \mathring{A}_1^r(S) = \bigcup_{k \in 1..n} \mathring{A}_1^{q_k}(S)$ *and* $\mathring{A}_2^r(S) = \bigcup_{k \in 1..n} \mathring{A}_2^{q_k}(S)$

$\mathring{A}_1^r, \mathring{A}_2^r$ are the most inclusive accessibility maps associated with the local parameter, encompass all locations that may be associated with r, locations meant to be visible as well as hidden.

Recall, an ordinary local vm-parameter must have exactly one conjunct $\{q\}$ so $n = 1$. Further it must hold that all local-parameter-parts have the form (P, LL).

We sometimes use $\{(l_{11}, l_{12}, \tau_1) \ldots (l_{n1}, l_{n2}, \tau_n)\}$ or $T_{\{(l_{11}, l_{12}, \tau_1) \ldots (l_{n1}, l_{n2}, \tau_n)\}}$ as shorthand for the (ordinary) local parameter $\{((T, \emptyset_{LL}), A_\emptyset, A_\emptyset, \{(l_{11}, l_{12}, \tau_1) \ldots (l_{n1}, l_{n2}, \tau_n)\})\}$. Such a parameter is used when we just add n visible locations in both sides. Visible locations will be expected to hold related values.

Definition 41. *$\succeq$ order on local-parameters*
Let $r' = \{\hat{q}_1' \ldots \hat{q}_n'\},\ r = \{\hat{q}_1 \ldots \hat{q}_n\}$ be local vm-parameters

$r' \succeq r$ iff $\forall j \in 1 \ldots n.\ \hat{q}_j' \succeq \hat{q}_j$. ($\hat{q}_j'$ is a subtree of $\hat{q}_j$)

Let $rk' = (r'|\hat{q}_i'),\ rk = (r|\hat{q}_j)$ be local k-parameters with $r' = \{\hat{q}_1' \ldots \hat{q}_n'\},\ r = \{\hat{q}_1 \ldots \hat{q}_n\}$

$rk' \succeq rk$ iff $r' \succeq r$ and $i = j$.

Definition 42. *Parameters*

(vm) A vm-parameter p is a finite set $p = \{r_1 \dots r_n\}$ where
- *Each $r_i \in p$ is a local vm-parameter*
- *$\forall i \neq j \in 1 \dots n.\ W_1^{r_i} \cap W_1^{r_j} = \emptyset \ \wedge\ W_2^{r_i} \cap W_2^{r_j} = \emptyset$*
- *$\exists S_1, S_2 \neq \bot. \forall i \neq j \in 1 \dots n. \mathring{A}_1^{ri}(S_1) \cap \mathring{A}_1^{rj}(S_1) = \emptyset \wedge \mathring{A}_2^{ri}(S_2) \cap \mathring{A}_2^{rj}(S_2) = \emptyset$*

(k) If $p = \{r_1 \dots r_n\}$ is a vm-parameter and for each $j \in 1 \dots n$ $(r_j|\hat{q}_j)$ is a local k-parameter, then $(pk) = \{(r_1|\hat{q}_1) \dots (r_n|\hat{q}_n)\}$ is a k-parameter.
A k-parameter is a vm-parameter together with choices of one $\bar{\wedge}$-clause for each local vm-parameter.

(s) If $(pk) = \{(r_1|\hat{q}_1) \dots (r_n|\hat{q}_n)\}$ is a k-parameter and
for each $j \in 1 \dots n$ $(r_j|\hat{q}_j|Q_j)$ is a local s-parameter and
($\exists k.$ Q_k has the form $(P, \bot) \Longrightarrow$ no Q_i has the form $(\bot, P)$) and
($\exists k.$ Q_k has the form $(\bot, P) \Longrightarrow$ no Q_i has the form $(P, \bot)$)
then $(pks) = \{(r_1|\hat{q}_1|Q_1) \dots (r_n|\hat{q}_n|Q_n)\}$ is a s-parameter.
An s-parameter is a k-parameter together with choices of one $\bar{\vee}$-clause for each local k-parameter.

(pk) is the k-erasure of (pks), and p is the vm-erasure of (pks) and of (pk).

(pks) is an s-instantiation of (pk) and of p. (pk) is a k-instantiation of p.

To the vm-parameter p are associated

– *Accessibility maps $\mathring{A}_1^p,\ \mathring{A}_2^p$ where $\forall S.\ \mathring{A}_1^p(S) = \bigcup \mathring{A}_1^{ri}(S) \ \wedge\ \mathring{A}_2^p(S) = \bigcup \mathring{A}_2^{ri}(S)$.*

To the k-parameter (pk) and to the s-parameter (pks) are associated

– *Accessibility maps $\mathcal{A}_1^{(pk)} = \mathcal{A}_1^{(pks)},\ \mathcal{A}_2^{(pk)} = \mathcal{A}_2^{(pks)}$, where*
$\forall S.\ \mathcal{A}_1^{(pk)}(S) = \bigcup_j (A_1^{q_j}(S) \cup L_1^{q_j}) \ \wedge\ \mathcal{A}_2^{(pk)}(S) = \bigcup_j (A_2^{q_j}(S) \cup L_2^{q_j})$, $j \in 1 \dots n$.

To the vm-parameter p is associated

– *a match of finite store types $Z^p = \biguplus_j Z^{\cap r_j}$, where $j \in 1 \dots n$.*

To the k-parameter (pk) and to the s-parameter (pks) is associated

– *A match of finite store types $Z^{(pk)} = Z^{(pks)} = \biguplus_j Z^{q_j}$, where $j \in 1 \dots n$.*

A parameter is ordinary iff all its local parameters are ordinary.
A parameter or a local parameter which is not ordinary is said to be specialized.
Any parameter and any local parameter is either ordinary or specialized.

$\mathfrak{p}^{vm}$ is the set of vm-parameters.	$\mathfrak{o}^{vm}$ is the set of ordinary vm-parameters.
$\mathfrak{p}^{k}$ is the set of k-parameters.	$\mathfrak{o}^{k}$ is the set of ordinary k-parameters.
$\mathfrak{p}^{s}$ is the set of s-parameters.	$\mathfrak{o}^{s}$ is the set of ordinary s-parameters.

$\mathfrak{s}^{vm}$ is the set of specialized vm-parameters.	$\mathfrak{p}^{vm} \setminus \mathfrak{o}^{vm} = \mathfrak{s}^{vm}$.
$\mathfrak{s}^{k}$ is the set of specialized k-parameters.	$\mathfrak{p}^{k} \setminus \mathfrak{o}^{k} = \mathfrak{s}^{k}$.
$\mathfrak{s}^{s}$ is the set of specialized s-parameters.	$\mathfrak{p}^{s} \setminus \mathfrak{o}^{s} = \mathfrak{s}^{s}$.

$\mathfrak{s}^{sP\bot}$ denotes the set of s-parameters which has a $(P, \bot)$ instantiation.
$\mathfrak{s}^{s\bot P}$ denotes the set of s-parameters which has a $(\bot, P)$ instantiation.

Definition 43. *Notation for* **sets** *of instantiations $p^{\mathbf{K}}$, $p^{\mathbf{S}}$*

– *If $p \in \mathfrak{p}^{vm}$ then $p^{\mathbf{K}}$ denotes the set of k-instantiations of p and $p^{\mathbf{S}}$ denotes the set of s-instantiations of p.*
– *If $p \in \mathfrak{p}^{k}$ then $p^{\mathbf{S}}$ denotes the set of s-instantiations of p.*

Definition 44. *Notation for erasures p^k, p^{vm}*

– *If $p \in \mathfrak{p}^{s}$ then p^k denotes the k-erasure of p and p^{vm} denotes the vm-erasure of p.*
– *If $p \in \mathfrak{p}^{k}$ then p^{vm} denotes the vm-erasure of p.*

Definition 45. *$\succeq$ relation on parameters.*
Let $p' = \{r'_1 \dots r'_n\}$ and $p = \{r_1 \dots r_n\}$ be vm-parameters.

$$p' \succeq p \text{ iff } \forall i \in 1 \dots n.\ r'_i \succeq r_i$$

Let $pk' = \{(r'_1|\hat{q}'_1) \dots (r'_n|\hat{q}'_n)\}$ and $pk = \{(r_1|\hat{q}_1) \dots (r_n|\hat{q}_n)\}$ be k-parameters.

$$pk' \succeq pk \text{ iff } \forall i \in 1 \dots n.\ (r'_i|\hat{q}'_i) \succeq (r_i|\hat{q}_i)$$

When we remove some but not all $\bar{\wedge}$-clauses from a local parameter then we get another local parameter. Again in the new relation we will require that relatedness for values and computations is preserved when we go to a parameter derived by removal of $\bar{\wedge}$-clauses.

Definition 46. *Parameters derived from parameters*

– *Let $p \in \mathfrak{p}^{vm}$, $p = \{r_1, \dots, r_n\} = \{\{q_{11} \dots q_{1k_1}\}, \dots, \{q_{n1} \dots q_{nk_n}\}\}$.*
Let $\forall i \in 1 \dots n.\ r_i \supseteq r'_i \neq \emptyset$.
Then $p' = \{r'_1, \dots, r'_n\}$ is a parameter derived from p.
So p' is derived from p by removal of $\bar{\wedge}$-clauses.

$\mathfrak{sub}(p)$ denotes the set of parameters derived from p.

– *Let $p' = \{r'_1, \dots, r'_n\}$ be derived from $p = \{r_1, \dots, r_n\} \in \mathfrak{p}^{vm}$ such that $\forall i \in 1 \dots n.\ \big(r'_i = \{\hat{q}\}$ where $\hat{q} \in r_i$ and q is an ordinary local parameter $\big)$.*
Then p' is an ordinary parameter, and p' is a parameter o-derived from p.
$\mathfrak{ord}(p)$ denotes the set of ordinary parameters derived from p.

Lemma 29. *A derived parameter is a parameter: the disjointness properties are preserved.*
It is possible that locations move from hidden to visible.

Order on parameters are defined as before:

Definition 47. *Orders $\rhd$ and $\blacktriangleright$ on parameters*

$\rhd$: *The relation $\rhd$ on vm-parameters and k-parameters is defined as the reflexive transitive closure of the relations $\succeq$ and $\supseteq_{ord}$ (local extension and adding ordinary local parameters).*

$\blacktriangleright$: *The relation $\blacktriangleright$ on vm-parameters is defined as the reflexive transitive closure of the relations $\succeq$, $\supseteq$ and $\geq_{\bar{\wedge}}$ (local extension, adding local parameters and removal of $\bar{\wedge}$-clauses).*

We still have the following lemmas

Lemma 30.
Assume $r' \rhd r$, then either both r' and r are ordinary parameters or both r' and r are specialized parameters with the relation $\succeq$ on all the non-ordinary constituents.

Lemma 31.
For $p', p \in \mathfrak{p}^{vm}$ it holds that $p' \rhd p \Rightarrow p' \blacktriangleright p$ (but not the other direction).

Lemma 32.

– *If $p' \in \mathfrak{s}^{vm}$ and $p \in \mathfrak{o}^{vm}$ and $p' \blacktriangleright p$, then $\forall p'_o \in \mathfrak{ord}(p')$. $p'_o \rhd p$.*

– *If $p' \blacktriangleright p \in \mathfrak{p}^{vm}$, $p' = \{r_1 \dots r_n, r_{n+1} \dots r_m\}$, $p = \{r_1 \dots r_n\}$, and p'' is derived from p' such that $\forall j \in (n{+}1) \dots m$. r_j is replaced by one of its conjuncts q_j where q_j is an ordinary local parameter, then $p'' \rhd p$.*

Lemma 33. *If $p' \blacktriangleright p$ or $p' \rhd p$ then $Z^{p'} \supseteq Z^p$*

Now we will define a relation parameterized by the new parameters.

9 Parameterized relation with new extended parameters

The relation is parameterized with the new parameters. In the definition of the relation the definition for values has the same formulations as before (but the parameter definitions are new). For states we now add, that for local instantiations to $(P, \bot)$ the left hand side non primed $s_1 \in P$ and similar for $(\bot, P)$. The primary difference is in the definition of relatedness for computations and continuations. Here we now differentiate three cases of the states they are applied to. If none of the instantiations in the state parameter has a $(P, \bot)$ or a $(\bot, P)$ then as before we require two sided termination approximation. If an instantiation in the state parameter is a $(P, \bot)$ then we instead require that the primed application in left hand side will always give $\bot$, and similarly for $(\bot, P)$. The existence of the new relation again requires a separate proof.

9.1 Relational Structure on FMcpo$^4_\bot$

The relational structure is formulated as before, but now with the new definition of parameters.

Definition 48. *Relational structure on FM-cpo$^4_\bot$*
Let $D = (D_V, D_K, D_M, D_S) \in$ FM-cpo$^4_\bot$.

The set of relations on D is defined as:

$$\begin{aligned}
\mathcal{R}(D) &= \hat{R}_V \times \hat{R}_K \times \hat{R}_M \times \hat{R}_S \\
&\quad \textit{where} \\
\hat{R}_V &= \textit{all subsets of } D_V^2 \times D_V^2 \times \{\tau | \tau \textit{ closed value type}\} \times \mathfrak{p}^{vm} \textit{ which} \\
&\quad \textit{include } \{(\bot,\bot)\} \times D_V^2 \times \{\tau | \tau \textit{ closed value type}\} \times \mathfrak{p}^{vm} \\
\hat{R}_K &= \textit{all subsets of } D_K^2 \times D_K^2 \times \{(x:\tau)^\top | (x:\tau)^\top \textit{ continuation type}\} \times \mathfrak{p}^k \textit{ which} \\
&\quad \textit{include } \{(\bot,\bot)\} \times D_K^2 \times \{(x:\tau)^\top | (x:\tau)^\top \textit{ continuation type}\} \times \mathfrak{p}^k \\
\hat{R}_M &= \textit{all subsets of } D_M^2 \times D_M^2 \times \{T\tau | T\tau \textit{ closed computation type}\} \times \mathfrak{p}^{vm} \textit{ which} \\
&\quad \textit{include } \{(\bot,\bot)\} \times D_M^2 \times \{T\tau | T\tau \textit{ closed computation type}\} \times \mathfrak{p}^{vm} \\
\hat{R}_S &= \textit{all subsets of } D_S^2 \times D_S^2 \times \mathfrak{p}^s \textit{ which} \\
&\quad \textit{include } \{(\bot,\bot)\} \times D_S^2 \times \mathfrak{p}^s
\end{aligned}$$

Application of a pair of morphisms to a relation is defined as before. Recall also the definition of $(f,g) : R \subset S$. For $f = (f_v, f_k, f_m, f_s) : D \multimap E$, $g = (g_v, g_k, g_m, g_s) : D \cong E$ with $f \sqsubseteq g$ and relations $R \in \mathcal{R}(D)$, $S \in \mathcal{R}(E)$ we have $(f,g) : R \subset S \overset{def}{\Longleftrightarrow} (f,g)R \subseteq S$. As before

– $\forall R \in \mathcal{R}(D).\ (id_D, id_D) : R \subset R$
– $\forall R \in \mathcal{R}(D). \forall S \in \mathcal{R}(D'). \forall g : D \cong D'\ (\bot, g) : R \subset S$

– $(f, g) : R \subset S$ and $(f', g') : S \subset T \Rightarrow (f' \circ f, g' \circ g) : R \subset T$
– $(id_D, id_D) : R \subset R'$ *and* $(id_D, id_D) : R' \subset R \Rightarrow R = R'$.

Admissiblity and downwards closure for relations is defined as before. Also parameter weakening is defined as before, but now with the new parameters. We present the definition again:

A relation $(R_V, R_K, R_M, R_S) \in \mathcal{R}(D)$ is *parameter-weakened* if,
$\forall p_1, p_0 \in \mathfrak{p}^{vm},\ \forall (pk_1), (pk_0) \in \mathfrak{p}^k$ it holds that

$$\begin{array}{lll}
- \quad p_1 \blacktriangleright p_0 & \wedge\ (v_1', v_2' \parallel v_1, v_2, \tau, p_0) \in R_V & \Rightarrow (v_1', v_2' \parallel v_1, v_2, \tau, p_1) \in R_V \\
- \ (pk_1) \rhd (pk_0) & \wedge\ (k_1', k_2' \parallel k_1, k_2, (x : \tau)^\top, (pk_0)) \in R_K & \Rightarrow (k_1', k_2' \parallel k_1, k_2, (x : \tau)^\top, (pk_1)) \in R_K \\
- \quad p_1 \blacktriangleright p_0 & \wedge\ (m_1', m_2' \parallel m_1, m_2, T\tau, p_0) \in R_M & \Rightarrow (m_1', m_2' \parallel m_1, m_2, T\tau, p_1) \in R_M
\end{array}$$

Definition 49. *adm$^+$relation*
A relation $R \in \mathcal{R}(D)$ is an adm$^+$relation *if it is*
admissible, downwards closed and parameter-weakened.
We let $\mathcal{R}_{adm^+}(D)$ denote the set of adm$^+$relations over D

We aim to show that there exists a new relational lifting of the functor F s.t. $\forall R^- \in \mathcal{R}(\mathbb{D}), R^+ \in \mathcal{R}(\mathbb{D}).F(R^-, R^+) \in \mathcal{R}(F(\mathbb{D}, \mathbb{D}))$ and a new *adm$^+$* relation $\nabla = (\nabla_V, \nabla_K, \nabla_M, \nabla_S) \in \mathcal{R}_{adm^+}(\mathbb{D})$ satisfying the equations in definition 50 and $(i, i) : F(\nabla, \nabla) \subset \nabla\ \wedge\ (i^{-1}, i^{-1}) : \nabla \subset F(\nabla, \nabla)$.

We define a new action of F on relations with the new parameters. We have added new requirements in the definition of relatedness for continuations and computations, where $(P, \bot)$ and $(\bot, P)$ instantiations in the states they are applied to is handled separately. For states and such state parameter instantiations we naturally require that the state in the respective side belong to P. Aside for these changes, the definitions are formulated as in the earlier sections.

Definition 50. *adm$^+$action of F on relations.*
Let $R^- \in \mathcal{R}(\mathbb{D}), R^+ \in \mathcal{R}(\mathbb{D})$

Define $F(R^-, R^+) \in \mathcal{R}(F(\mathbb{D}, \mathbb{D}))$,
$F(R^-, R^+) = (F(R^-, R^+)_V, F(R^-, R^+)_K, F(R^-, R^+)_M, F(R^-, R^+)_S)$ *where*

$$\begin{array}{rl}
F(R^-, R^+)_V = & \{(\bot,\ \bot \parallel v_1,\ v_2,\ \tau,\ p) \mid p \in \mathfrak{p}^{vm} \textit{ the set of all vm-parameters } \} \cup \\
& \\
& \{(v_1',\ v_2' \parallel v_1,\ v_2,\ \tau,\ p) \mid p \in \mathfrak{p}^{vm}\ \wedge \\
& \quad v_1' \sqsubseteq v_1 \neq \bot\ \wedge\ v_2' \sqsubseteq v_2 \neq \bot\ \wedge \\
& \quad (v_1',\ v_2' \parallel v_1,\ v_2, \tau,\ p) \in \tilde{F}(R^-, R^+)_V\ \} \\
& \textit{where} \\
\tilde{F}(R^-, R^+)_V = & \{(v_1',\ v_2' \parallel in_{\mathbf{1}}\lfloor * \rfloor,\ in_{\mathbf{1}}\lfloor * \rfloor,\ unit,\ p)\ \} \cup \\
& \{(v_1',\ v_2' \parallel in_{\mathbb{Z}}\lfloor n \rfloor,\ in_{\mathbb{Z}}\lfloor n \rfloor,\ int,\ p) \mid n \in \mathbb{Z}\ \} \cup \\
& \{(v_1',\ v_2' \parallel in_{\mathbb{L}}\lfloor l_1 \rfloor,\ in_{\mathbb{L}}\lfloor l_2 \rfloor,\ \tau\ ref,\ p) \mid (l_1, l_2, \tau) \in Z^p\ \ \} \cup \\
& \{(v_1',\ v_2' \parallel in_{\oplus} in_i d_1,\ in_{\oplus} in_i d_2,\ \tau_1 + \tau_2,\ p) \mid d_1, d_2 \in \mathbb{V}_\downarrow\ \wedge\ \exists d_1', d_2' \in \mathbb{V}. \\
& \quad ((v_1' = \bot \wedge d_1' = \bot) \vee v_1' = in_{\oplus} in_i d_1' \neq \bot) \wedge ((v_2' = \bot \wedge d_2' = \bot) \vee v_2' = in_{\oplus} in_i d_2' \neq \bot) \wedge \\
& \quad (d_1',\ d_2' \parallel d_1,\ d_2,\ \tau_i,\ p) \in R_V^+,\ i \in \{1, 2\}\ \} \cup \\
& \{(v_1',\ v_2' \parallel in_{\otimes}(d_{1a}, d_{1b}),\ in_{\otimes}(d_{2a}, d_{2b}),\ \tau_a \times \tau_b,\ p) \mid d_{1a}, d_{1b}, d_{2a}, d_{2b} \in \mathbb{V}_\downarrow\ \wedge \\
& \quad \exists d_{1a}', d_{1b}', d_{2a}', d_{2b}' \in \mathbb{V}. \\
& \quad (d_{1a}',\ d_{2a}' \parallel d_{1a},\ d_{2a},\ \tau_a,\ p) \in R_V^+\ \wedge\ (d_{1b}',\ d_{2b}' \parallel d_{1b},\ d_{2b},\ \tau_b,\ p) \in R_V^+\ \wedge \\
& \quad ((v_1' = \bot \wedge (d_{1a}' = \bot \vee d_{1b}' = \bot))\ \vee\ (v_1' = in_{\otimes}(d_{1a}', d_{1b}') \neq \bot))\ \wedge \\
& \quad ((v_2' = \bot \wedge (d_{2a}' = \bot \vee d_{2b}' = \bot))\ \vee\ (v_2' = in_{\otimes}(d_{2a}', d_{2b}') \neq \bot))\} \cup \\
& \{(v_1',\ v_2' \parallel in_{\mu} d_1,\ in_{\mu} d_2,\ \mu\alpha.\tau,\ p) \mid d_1, d_2 \in \mathbb{V}_\downarrow\ \wedge\ \exists d_1', d_2' \in \mathbb{V}. \\
& \quad ((v_1' = \bot \wedge d_1' = \bot) \vee v_1' = in_{\mu} d_1' \neq \bot) \wedge ((v_2' = \bot \wedge d_2' = \bot) \vee v_2' = in_{\mu} d_2' \neq \bot) \wedge \\
& \quad (d_1',\ d_2' \parallel d_1,\ d_2,\ \tau[\mu\alpha.\tau/\alpha],\ p) \in R_V^+\ \} \cup
\end{array}$$

$\{(v'_1,\ v'_2 \parallel in_\forall\lfloor d_1\rfloor,\ in_\forall\lfloor d_2\rfloor,\ \forall\alpha.T\tau,\ p) \mid d_1, d_2 \in \mathbb{M}\ \wedge\ \exists d'_1, d'_2 \in \mathbb{M}.$

$\quad((v'_1 = \bot \wedge d'_1 = \bot)\ \vee\ (v'_1 = in_\forall\lfloor d'_1\rfloor \neq \bot\ \wedge\ d'_1 \sqsupseteq \bot))\ \wedge$

$\quad((v'_2 = \bot \wedge d'_2 = \bot)\ \vee\ (v'_2 = in_\forall\lfloor d'_2\rfloor \neq \bot\ \wedge\ d'_2 \sqsupseteq \bot))\ \wedge$

$\quad\forall\sigma$ *with* $\_ \vdash \sigma : type.\ (d'_1,\ d'_2 \parallel d_1,\ d_2,\ T\tau[\sigma/\alpha],\ p) \in R^+_M\ \} \cup$

$\{(v'_1,\ v'_2 \parallel in_{\multimap}\lfloor d_1\rfloor,\ in_{\multimap}\lfloor d_2\rfloor,\ \tau \to \mathrm{T}\tau',\ p) \mid d_1, d_2 \in (\mathbb{V} \multimap \mathbb{M})\ \wedge\ \exists d'_1, d'_2 \in (\mathbb{V} \multimap \mathbb{M}).$

$\quad((v'_1 = \bot \wedge d'_1 = \bot)\ \vee\ (v'_1 = in_{\multimap}\lfloor d'_1\rfloor \neq \bot\ \wedge\ d'_1 \sqsupseteq \bot))\ \wedge$

$\quad((v'_2 = \bot \wedge d'_2 = \bot)\ \vee\ (v'_2 = in_{\multimap}\lfloor d'_2\rfloor \neq \bot\ \wedge\ d'_2 \sqsupseteq \bot))\ \wedge$

$\quad\forall p' \blacktriangleright p, \forall(w'_1,\ w'_2 \parallel w_1,\ w_2, \tau,\ p') \in R^-_V.\ (\ d'_1 w'_1,\ d'_2 w'_2 \parallel d_1 w_1,\ d_1 w_2,\ T\tau',\ p') \in R^+_M\ \}$

Recall $\blacktriangleright$ *on vm-parameters is defined as the reflexive transitive closure of the relations* $\succeq$, $\supseteq$ *and* $\geq_{\bar{\wedge}}$*, (which are local extension, supset and removal of* $\bar{\wedge}$*-clauses).*

$F(R^-, R^+)_K = \{(k'_1,\ k'_2 \parallel k_1,\ k_2,\ (x:\tau)^\top,\ (pk)) \mid (pk) \in \mathfrak{p}^k\ \wedge$

$\qquad k'_1 \sqsubseteq k_1\ \wedge\ k'_2 \sqsubseteq k_2\ \wedge\ \ \forall(pk') \rhd (pk).$

$\qquad\quad$ *(that is* (pk) *extended only with* ordinary *local k-parameters or local extensions)*

$\qquad\forall(pks') \in (pk')^{\mathbf{S}}$ *(the set of s-instantiations of* p' *i.e. choices of* $\bar{\vee}$*-clauses),*

$\qquad(pks') \notin (\mathfrak{s}^{sP\bot} \cup \mathfrak{s}^{s\bot P})$. *(i.e. no chosen* $\bar{\vee}$*-clause in the s-instantiation is a* $(\bot, P)$ *or* $(P, \bot)$*)*

$\qquad\quad\forall(s'_1,\ s'_2 \parallel s_1,\ s_2,\ (pks')) \in R^-_S.\ \forall(v'_1,\ v'_2 \parallel v_1,\ v_2, \tau,\ (pk')^{vm}) \in R^-_V.$

$\qquad\qquad(k'_1 s'_1 v'_1 = \top \Rightarrow k_2 s_2 v_2 = \top)\ \wedge$

$\qquad\qquad(k'_2 s'_2 v'_2 = \top \Rightarrow k_1 s_1 v_1 = \top), \qquad \wedge$

$\qquad\forall(pks') \in (pk')^{\mathbf{S}}, (pks') \in \mathfrak{s}^{sP\bot}$. *(i.e. a chosen* $\bar{\vee}$*-clause in the s-instantiation is a* $(P, \bot)$*)*

$\qquad\quad\forall(s'_1,\ s'_2 \parallel s_1,\ s_2,\ (pks')) \in R^-_S.$

$\qquad\quad\forall(v'_1,\ v'_2 \parallel v_1,\ v_2,\ \tau,\ (pk')^{vm}) \in R^-_V.$

$\qquad\qquad k'_1 s'_1 v'_1 = \bot, \qquad \wedge,$

$\qquad\forall(pks') \in (pk')^{\mathbf{S}}, (pks') \in \mathfrak{s}^{s\bot P}$. *(i.e. a chosen* $\bar{\vee}$*-clause in the s-instantiation is a* $(\bot, P)$*)*

$\qquad\quad\forall(s'_1,\ s'_2 \parallel s_1,\ s_2,\ (pks')) \in R^-_S.$

$\qquad\quad\forall(v'_1,\ v'_2 \parallel v_1,\ v_2,\ \tau,\ (pk')^{vm}) \in R^-_V.$

$\qquad\qquad k'_2 s'_2 v'_2 = \bot\ \},$

$F(R^-, R^+)_M = \{(m'_1,\ m'_2 \parallel m_1,\ m_2,\ T\tau,\ p) \mid p \in \mathfrak{p}^{vm}\ \wedge$

$\qquad m'_1 \sqsubseteq m_1\ \wedge\ m'_2 \sqsubseteq m_2\ \wedge$

$\qquad\forall p' \blacktriangleright p.\forall(pk') \in p'^{\mathbf{K}}$ *(the set of k-instantiations of* p' *i.e. choices of a* $\bar{\wedge}$*-clause in each local parameter).*

$\qquad\forall(pks') \in (pk')^{\mathbf{S}}$ *(the set of s-instantiations of* (pk') *i.e. choices of* $\bar{\vee}$*-clause from each chosen* $\bar{\wedge}$*-clause in* (pk') *),*

$\qquad(pks') \notin (\mathfrak{s}^{sP\bot} \cup \mathfrak{s}^{s\bot P})$. *(i.e. no chosen* $\bar{\vee}$*-clause is a* $(\bot, P)$ *or* $(P, \bot)$*)*

$\qquad\quad\forall(k'_1,\ k'_2 \parallel k_1,\ k_2,\ (x:\tau)^\top,\ (pk')) \in R^-_K.\ \forall(S'_1,\ S'_2 \parallel S_1,\ S_2,\ (pks')) \in R^-_S\ .$

$\qquad\qquad(m'_1 k'_1 S'_1 = \top \Rightarrow m_2 k_2 S_2 = \top)\ \wedge$

$\qquad\qquad(m'_2 k'_2 S'_2 = \top \Rightarrow m_1 k_1 S_1 = \top)\ \wedge$

$\qquad\forall(pks') \in (pk')^{\mathbf{S}}, (pks') \in \mathfrak{s}^{sP\bot}$. *(i.e. a chosen* $\bar{\vee}$*-clause is a* $(P, \bot)$*)*

$\qquad\quad\forall(k'_1,\ k'_2 \parallel k_1,\ k_2,\ (x:\tau)^\top,\ (pk')) \in R^-_K.$

$\qquad\quad\forall(S'_1,\ S'_2 \parallel S_1,\ S_2,\ (pks')) \in R^-_S.$

$\qquad\qquad m'_1 k'_1 S'_1 = \bot)\ \wedge$

$\qquad\forall(pks') \in (pk')^{\mathbf{S}}, (pks') \in \mathfrak{s}^{s\bot P}$. *(i.e. a chosen* $\bar{\vee}$*-clause is a* $(\bot, P)$*)*

$\qquad\quad\forall(k'_1,\ k'_2 \parallel k_1,\ k_2,\ (x:\tau)^\top,\ (pk')) \in R^-_K.$

$\qquad\quad\forall(S'_1,\ S'_2 \parallel S_1,\ S_2,\ (pks')) \in R^-_S.$

$\qquad\qquad m'_2 k'_2 S'_2 = \bot\ \}$

$F(R^-, R^+)_S = \{(\bot,\ \bot \parallel S_1,\ S_2,\ (pks)) \mid (pks) \in \mathfrak{p}^s$ *the set of all s-parameters* $\} \cup$

$\{(S'_1,\ S'_2 \parallel\ S_1,\ S_2,\ (pks) \mid (pks) = \{(r_1|q_1|Q_1), \ldots, (r_n|q_n|Q_n)\} \in \mathfrak{p}^s$
(q_i is a choice of $\bar{\wedge}$-clause in r_i, and Q_i is a choice of $\bar{\vee}$-clause in q_i) $\wedge$

$S'_1 \sqsubseteq S_1 \neq \bot \quad \wedge\ S'_2 \sqsubseteq S_2 \neq \bot\ \wedge$

$\mathcal{A}_1^{(pks)}(S_1) \cap \pi_1(Z^{(pks)}) = \emptyset\ \wedge\ \mathcal{A}_2^{(pks)}(S_2) \cap \pi_2(Z^{(pks)}) = \emptyset$
(in each side are the visible locations in $Z^{(pks)}$ disjoint from the known hidden locations given by the accessibility maps $\mathcal{A}_1^{(pks)}, \mathcal{A}_2^{(pks)}$) $\wedge$

$\forall i \neq j.\ \mathring{A}_1^{ri}(S_1) \cap \mathring{A}_1^{rj}(S_1) = \emptyset\ \wedge\ \mathring{A}_2^{ri}(S_2) \cap \mathring{A}_2^{rj}(S_2) = \emptyset$
($\mathring{A}_1^{ri}, \mathring{A}_2^{ri}$ accessibility maps are the most inclusive for the local parameter r_i. In each side is every location, visible, hidden or reserved, belonging to a local parameter outside the areas owned by any different local parameter) $\wedge$

$\forall (l_1, l_2, \tau) \in Z^{(pks)}.(S'_1(l_1),\ S'_2(l_2) \parallel S_1(l_1),\ S_2(l_2),\ \tau,\ (pks)^{vm}) \in R_V^+$
(All visible locations hold related values) $\wedge$

$\forall i \in 1..n.$ *if* $Q_i = (P_i, \bot)$ *then* $S_1 \in P_i$, *if* $Q_i = (\bot, P_i)$ *then* $S_2 \in P_i$,
if $Q_i = (P_i, LL_i)$ *then* $(S_1, S_2) \in P_i\ \wedge$
$\forall (l_1, l_2, \tau) \in LL_i.\ (S'_1(l_1),\ S'_2(l_2) \parallel S_1(l_1),\ S_2(l_2),\ \tau,\ (pks)^{vm}) \in R_V^+$
(The states belong to all simple state relations and predicates in the chosen $\bar{\vee}$-clauses. Corresponding LL location-sets hold related values) $\}$

It holds that

- For all $(v'_1,\ v'_2 \parallel v_1,\ v_2,\ \tau,\ p) \in F(R^-, R^+)_V$ it holds that $(v'_1 = v'_2 = \bot)$ or $(v_1 \neq \bot \wedge v_2 \neq \bot)$.
- For all $(s'_1,\ s'_2 \parallel s_1,\ s_2,\ pks) \in F(R^-, R^+)_S$ it holds that $(s'_1 = s'_2 = \bot)$ or $(s_1 \neq \bot \wedge s_2 \neq \bot)$.
- $\forall k_1, k_2 \in \mathbb{K}.\forall (x:\tau)^\top.\forall pk \in \mathfrak{p}^k$ it holds that $(\bot, \bot \parallel k_1, k_2, (x:\tau)^\top, pk) \in F(R^-, R^+)_K$.
- $\forall m_1, m_2 \in \mathbb{M}.\forall T\tau.\forall p \in \mathfrak{p}^{vm}$ it holds that $(\bot, \bot \parallel m_1, m_2, T\tau, p) \in F(R^-, R^+)_M$.

We need to show that the action of F with the new definitions preserves admissibility, downwards closure and parameter weakening.

Lemma 34. *The action of F preserves downwards closure.*

For all $R^+, R^- \in \mathcal{R}(\mathbb{D})$.
If R^+ is downwards closed, then $F(R^-, R^+)$ is downwards closed.

Proof

- $F(R^-, R^+)_K$: Follows from $k'' \sqsubseteq k' \Rightarrow \forall s, v.\ k''sv \sqsubseteq k'sv$, (independant of R^+). This is enough also with the new definition of ∇.
- $F(R^-, R^+)_M$: Follows from $m'' \sqsubseteq m' \Rightarrow \forall k, s.\ m''ks \sqsubseteq m'ks$, (independant of R^+). This is enough also with the new definition of ∇.
- $F(R^-, R^+)_V$: The proof is as the proof we have seen before, so it is omittet.
- $F(R^-, R^+)_S$: follows from downwards closure of R_V^+. Assume $(s'_1, s'_2 \parallel s_1, s_2, p) \in F(R^-, R^+)_S$, and $s'_1 \neq \bot\ \vee\ s'_2 \neq \bot$. Let $s''_1 \sqsubseteq s'_1 \wedge s''_2 \sqsubseteq s'_2$. Then $\forall l.\ s''_1 l \sqsubseteq s'_1 l \wedge s''_2 l \sqsubseteq s'_2 l$. We want to show $(s''_1, s''_2 \parallel s_1, s_2, p) \in F(R^-, R^+)_S$. The requirements concerning disjointness as well as requirements about belongings to finitary state relations and finitary state predicates are only stated on s_1, s_2 and hence follow from the assumptions. Requirements concerning that stored values are related follow from assumptions together with downwards closure of R_V^+.

□

Lemma 35. *The action of F on relations preserves admissibility.*

For all $R^+, R^- \in \mathcal{R}(\mathbb{D})$. If R^+ is admissible, then $F(R^-, R^+)$ is admissible.

Proof

Assume R^+ is admissible, we want to show $F(R^-, R^+)$ is admissible for all $R^- \in \mathcal{R}(\mathbb{D})$. By definition each of the four projections of $F(R^-, R^+)$ includes $(\bot, \bot \parallel (type), d_1, d_2, p)$ for all $(type), d_1, d_2, p$. To show that $F(R^-, R^+)$ is admissible it suffices to show for each of the four projections, that it is closed under least upper bounds of finitely supported chains of the form $(d_1^i, d_2^i \parallel (type), d_1, d_2, p)_{i\in\omega}$ where $type, d_1, d_2, p$ are constant.

- $F(R^-, R^+)_S =$

  $\{(\bot,\ \bot \parallel S_1,\ S_2,\ (pks)) \mid (pks) \in \mathfrak{p}^s$ the set of all s-parameters $\} \cup$
  $\{(S_1',\ S_2' \parallel\ S_1,\ S_2,\ (pks) \mid (pks) = \{(r_1|q_1|Q_1), \ldots, (r_n|q_n|Q_n)\} \in \mathfrak{p}^s\ \wedge$
  $S_1' \sqsubseteq S_1 \neq \bot\ \ \wedge\ S_2' \sqsubseteq S_2 \neq \bot\ \wedge$
  $\mathcal{A}_1^{(pks)}(S_1) \cap \pi_1(Z^{(pks)}) = \emptyset\ \wedge\ \mathcal{A}_2^{(pks)}(S_2) \cap \pi_2(Z^{(pks)}) = \emptyset\ \wedge$
  $\forall i \neq j.\ \mathring{A}_1^{ri}(S_1) \cap \mathring{A}_1^{rj}(S_1) = \emptyset\ \wedge\ \mathring{A}_2^{ri}(S_2) \cap \mathring{A}_2^{rj}(S_2) = \emptyset\ \wedge$
  $\forall (l_1, l_2, \tau) \in Z^{(pks)}.(S_1'(l_1),\ S_2'(l_2) \parallel S_1(l_1),\ S_2(l_2),\ \tau,\ (pks)^{vm}) \in R_V^+\ \wedge$
  $\forall i \in 1..n.$ if $Q_i = (P_i, \bot)$ then $S_1 \in P_i$, if $Q_i = (\bot, P_i)$ then $S_2 \in P_i$,
  if $Q_i = (P_i, LL_i)$ then $(S_1, S_2) \in P_i\ \wedge$
  $\forall (l_1, l_2, \tau) \in LL_i.\ (S_1'(l_1),\ S_2'(l_2) \parallel S_1(l_1),\ S_2(l_2),\ \tau,\ (pks)^{vm}) \in R_V^+\ \}$

  Assume a finitely supported chain $(S_1^i, S_2^i \parallel S_1, S_2, p)_{i\in\omega}$ in $F(R^-, R^+)_S$, we will show its least upper bound is in $F(R^-, R^+)_S$. If the chain is constantly $(\bot, \bot, \parallel S_1, S_2, p)$ we are done.
  Else it holds that $S_1 \neq \bot\ \wedge\ S_2 \neq \bot$ and $\forall i.\ S_1^i \sqsubseteq S_1\ \wedge\ S_2^i \sqsubseteq S_2$. Then it holds that $\bigsqcup S_1^i \sqsubseteq S_1\ \wedge\ \bigsqcup S_2^i \sqsubseteq S_2$.
  $\mathcal{A}_1^{(pks)}(S_1) \cap \pi_1(Z^{(pks)}) = \emptyset\ \wedge\ \mathcal{A}_2^{(pks)}(S_2) \cap \pi_2(Z^{(pks)}) = \emptyset$ and
  $\forall k \neq j.\ \mathring{A}_1^{rk}(S_1) \cap \mathring{A}_1^{rj}(S_1) = \emptyset\ \wedge\ \mathring{A}_2^{rk}(S_2) \cap \mathring{A}_2^{rj}(S_2) = \emptyset$ as in each step.
  When $(S_1^i, S_2^i \parallel S_1, S_2, p)_{i\in\omega}$ is a chain, then $\forall l.\ (S_1^i l)_{i\in\omega}$ is a chain and $(S_2^i l)_{i\in\omega}$ is a chain.
  Since R_V^+ is admissible and $\forall i.\forall (l_1, l_2, \tau) \in Z^{(pks)}.(S_1^i l_1\ S_2^i l_2 \parallel S_1 l_1,\ S_2 l_2,\ \tau, (pks)^{vm}) \in R_V^+$ and these are chains in R_V^+, then also $\forall (l_1, l_2, \tau) \in Z^{(pks)}.\ \bigsqcup((S_1^i)l_1, (S_2^i)l_2 \parallel S_1 l_1,\ S_2 l_2,\ \tau, (pks)^{vm}) = ((\bigsqcup S_1^i)l_1,\ (\bigsqcup S_2^i)l_2 \parallel S_1 l_1,\ S_2 l_2,\ \tau, (pks)^{vm}) \in R_V^+$.
  $\forall k \in 1..n.$ if $Q_k = (P_k, \bot)$ then $S_1 \in P_k$, if $Q_k = (\bot, P_k)$ then $S_2 \in P_k$,
  if $Q_k = (P_k, LL_k)$ then $(S_1, S_2) \in P_k$ holds as in each step.
  Since $\forall i.\ \forall k \in 1..n.$ if $Q_k = (P_k, LL_k)$ then $\forall (l_1, l_2, \tau) \in LL_k.\ (S_1^i(l_1),\ S_2^i(l_2) \parallel S_1(l_1),\ S_2(l_2),\ \tau,$ $(pks)^{vm}) \in R_V^+$ and these are chains in the admissible relation R_V^+, then also $\forall k \in 1..n.$ if $Q_k = (P_k, LL_k)$ then $\forall (l_1, l_2, \tau) \in LL_k.\ ((\bigsqcup S_1^i)l_1,\ (\bigsqcup S_2^i)l_2 \parallel S_1 l_1,\ S_2 l_2,\ \tau,\ (pks)^{vm}) \in R_V^+$.
  We conclude that $F(R^-, R^+)_S$ is closed under least upper bounds of chains.

- $F(R^-, R^+)_M =$

  $\{(\bot,\ \bot \parallel m_1,\ m_2,\ T\tau,\ p) \mid p \in \mathfrak{p}^{vm}$ the set of all vm-parameters $\} \cup$
  $\{(m_1',\ m_2' \parallel m_1,\ m_2,\ T\tau,\ p) \mid p \in \mathfrak{p}^{vm}\ \wedge$
  $m_1' \sqsubseteq m_1\ \wedge\ m_2' \sqsubseteq m_2\ \wedge$
  $\forall p' \blacktriangleright p. \forall (pk') \in p'^{\mathbf{K}}.$
  $\forall (pks') \in (pk')^{\mathbf{S}}, (pks') \notin (\mathfrak{s}^{sP\bot} \cup \mathfrak{s}^{s\bot P})$. (i.e. no chosen $\bar{\vee}$-clause is a $(\bot, P)$ or $(P, \bot)$)
  $\forall (k_1',\ k_2' \parallel k_1,\ k_2,\ (x:\tau)^\top,\ (pk')) \in R_K^-.\ \forall (S_1',\ S_2' \parallel S_1,\ S_2,\ (pks')) \in R_S^-$.
  $(m_1' k_1' S_1' = \top \Rightarrow m_2 k_2 S_2 = \top)\ \wedge$
  $(m_2' k_2' S_2' = \top \Rightarrow m_1 k_1 S_1 = \top)\ \wedge$

$\forall(pks') \in (pk')^{\mathbf{S}}, (pks') \in \mathfrak{s}^{sP\perp}$. (i.e. a chosen $\bar{\nabla}$-clause is a $(P, \perp)$)
$\quad\forall(k_1',\ k_2' \parallel k_1,\ k_2,\ (x:\tau)^\top,\ (pk')) \in R_K^-.\ \forall(S_1',\ S_2' \parallel S_1,\ S_2,\ (pks')) \in R_S^-$.
$\qquad m_1'k_1'S_1' = \perp) \wedge$
$\forall(pks') \in (pk')^{\mathbf{S}}, (pks') \in \mathfrak{s}^{s\perp P}$. (i.e. a chosen $\bar{\nabla}$-clause is a $(\perp, P)$)
$\quad\forall(k_1',\ k_2' \parallel k_1,\ k_2,\ (x:\tau)^\top,\ (pk')) \in R_K^-.\ \forall(S_1',\ S_2' \parallel S_1,\ S_2,\ (pks')) \in R_S^-$.
$\qquad m_2'k_2'S_2' = \perp\ \}$

Assume a finitely supported chain $(m_1^i, m_2^i \parallel m_1, m_2, T\tau, p)_{i\in\omega}$ in $F(R^-, R^+)_M$, we will show its least upper bound is in $F(R^-, R^+)_M$. If the chain is constant $(\perp, \perp \parallel m_1,\ m_2, T\tau, p)$ we are done. Else, since $\forall i.\ m_1^i \sqsubseteq m_1\ \wedge\ m_2^i \sqsubseteq m_2$, then also $\bigsqcup m_1^i \sqsubseteq m_1\ \wedge\ \bigsqcup m_2^i \sqsubseteq m_2$.

Assume $p' \blacktriangleright p, (pk') \in p'^{\mathbf{k}},\ (k_1', k_2' \parallel\ k_1,\ k_2,\ (x:\tau)^\top,\ (pk)') \in R_K^-$.
Assume $(pks') \in (pk')^{\mathbf{s}}, (pks') \notin (\mathfrak{s}^{sP\perp} \cup \mathfrak{s}^{s\perp P})$ and $(S_1', S_2' \parallel\ S_1,\ S_2,\ (pks')) \in R_S^-$.
Since m_1^i and m_2^i are chains, then also $m_1^ik_1'S_1'$ and $m_2^ik_2'S_2'$ are chains in $\mathbb{O}$. If $\forall i.m_1^ik_1'S_1' = \perp$ then also $(\bigsqcup m_1^i)k_1'S_1' = \perp$ and the implication $(\bigsqcup m_1^i)k_1'S_1' = \top \Rightarrow m_2k_2S_2 = \top$ holds trivially. Else it must be the case that $\exists j.\forall i \geq j.\ m_1^ik_1'S_1' = \top$. This implies both that $m_2k_2S_2 = \top$ and $(\bigsqcup m_1^i)k_1'S_1' = \top$, so the implication $(\bigsqcup m_1^i)k_1'S_1' = \top \Rightarrow m_2k_2S_2 = \top$ holds. The proof that $(\bigsqcup m_2^i)k_2'S_2' = \top \Rightarrow m_1k_1S_1 = \top$ holds is similar.
Assume $(pks') \in (pk')^{\mathbf{s}}, (pks') \in \mathfrak{s}^{sP\perp}$ and $(S_1',\ S_2' \parallel S_1,\ S_2,\ (pks')) \in R_S^-$, then in each step $m_1^ik_1'S_1' = \perp$. And so also $\bigsqcup_i(m_1^ik_1'S_1') = \bigsqcup_i(m_1^i)k_1'S_1' = \perp$.
Assume $(pks') \in (pk')^{\mathbf{s}}, (pks') \in \mathfrak{s}^{s\perp P}$ and $(S_1',\ S_2' \parallel S_1,\ S_2,\ (pks')) \in R_S^-$, then in each step $m_2^ik_2'S_2' = \perp$. And so also $\bigsqcup_i(m_2^ik_2'S_2') = \bigsqcup_i(m_2^i)k_2'S_2' = \perp$.
We conclude that $F(R^-, R^+)_M$ is closed under lubs of chains.

- $F(R^-, R^+)_K =$
  $\{(\perp,\ \perp \parallel k_1,\ k_2,\ (x:\tau)^\top,\ (pk)) \mid (pk) \in \mathfrak{p}^k$ the set of all k-parameters $\} \cup$
  $\{(k_1',\ k_2' \parallel k_1,\ k_2,\ (x:\tau)^\top,\ (pk)) \mid (pk) \in \mathfrak{p}^k\ \wedge$
  $\quad k_1' \sqsubseteq k_1\ \wedge\ k_2' \sqsubseteq k_2\ \wedge\ \forall(pk') \rhd (pk)$.
  $\qquad$(that is (pk) extended only with *ordinary* local k-parameters or local extensions)
  $\quad\forall(pks') \in (pk')^{\mathbf{S}}$ (the set of s-instantiations of p' i.e. choices of $\bar{\nabla}$-clauses),
  $\quad(pks') \notin (\mathfrak{s}^{sP\perp} \cup \mathfrak{s}^{s\perp P})$. (i.e. no chosen $\bar{\nabla}$-clause in the s-instantiation is a $(\perp, P)$ or $(P, \perp)$)
  $\qquad\forall(s_1',\ s_2' \parallel s_1,\ s_2,\ (pks')) \in R_S^-.\ \forall(v_1',\ v_2' \parallel v_1,\ v_2, \tau,\ (pk')^{vm}) \in R_V^-$.
  $\qquad\quad(k_1's_1'v_1' = \top \Rightarrow k_2s_2v_2 = \top)\ \wedge$
  $\qquad\quad(k_2's_2'v_2' = \top \Rightarrow k_1s_1v_1 = \top), \qquad \wedge$
  $\quad\forall(pks') \in (pk')^{\mathbf{S}}, (pks') \in \mathfrak{s}^{sP\perp}$. (i.e. a chosen $\bar{\nabla}$-clause in the s-instantiation is a $(P, \perp)$)
  $\qquad\forall(s_1',\ s_2' \parallel s_1,\ s_2,\ (pks')) \in R_S^-.\ \forall(v_1',\ v_2' \parallel v_1,\ v_2,\ \tau,\ (pk')^{vm}) \in R_V^-$.
  $\qquad\quad k_1's_1'v_1' = \perp, \qquad \wedge,$
  $\quad\forall(pks') \in (pk')^{\mathbf{S}}, (pks') \in \mathfrak{s}^{s\perp P}$. (i.e. a chosen $\bar{\nabla}$-clause in the s-instantiation is a $(\perp, P)$)
  $\qquad\forall(s_1',\ s_2' \parallel s_1,\ s_2,\ (pks')) \in R_S^-.\ \forall(v_1',\ v_2' \parallel v_1,\ v_2,\ \tau,\ (pk')^{vm}) \in R_V^-$.
  $\qquad\quad k_2's_2'v_2' = \perp\ \}$,

Assume a finitely supported chain $(k_1^i, k_2^i \parallel k_1, k_2, (x:\tau)^\top, (pk))_{i\in\omega}$ in $F(R^-, R^+)_K$, we will show its least upper bound is in $F(R^-, R^+)_K$. If the chain is constant $(\perp, \perp \parallel k_1,\ k_2, (x : \tau)^\top, (pk))$ we are done. Else, since $\forall i.\ k_1^i \sqsubseteq k_1\ \wedge\ k_2^i \sqsubseteq k_2$, then also $\bigsqcup k_1^i \sqsubseteq k_1\ \wedge\ \bigsqcup k_2^i \sqsubseteq k_2$.
Assume $(pk') \rhd (pk),\ (v_1', v_2' \parallel v_1,\ v_2,\ \tau,\ (pk')^{vm}) \in R_V^-$.
Assume $(pks') \in (pk')^{\mathbf{S}}, (pks') \notin (\mathfrak{s}^{sP\perp} \cup \mathfrak{s}^{s\perp P})$ and $(S_1',\ S_2' \parallel S_1,\ S_2,\ (pks')) \in R_S^-$.
Since k_1^i and k_2^i are chains, then also $k_1^iS_1'v_1'$ and $k_2^iS_2'v_2'$ are chains in $\mathbb{O}$. If $\forall i.k_1^iS_1'v_1' = \perp$ then also $(\bigsqcup k_1^i)S_1'v_1' = \perp$ and the implication $(\bigsqcup k_1^i)S_1'v_1' = \top \Rightarrow k_2S_2v_2 = \top$ holds trivially. Else it must be the case that $\exists j.\forall i \geq j.\ k_1^iS_1'v_1' = \top$. This implies both that $k_2S_2v_2 = \top$ and $(\bigsqcup k_1^i)S_1'v_1' = \top$, so the implication $(\bigsqcup k_1^i)S_1'v_1' = \top \Rightarrow k_2S_2v_2 = \top$ holds. The proof that $(\bigsqcup k_2^i)S_2'v_2' = \top \Rightarrow k_1S_1v_1 = \top$ holds is similar.
Assume $(pks') \in (pk')^{\mathbf{S}}, (pks') \in \mathfrak{s}^{sP\perp}$ and $(S_1',\ S_2' \parallel S_1,\ S_2,\ (pks')) \in R_S^-$, then in each step $k_1^iS_1'v_1' = \perp$. So also $\bigsqcup_i k_1^iS_1'v_1' = \perp$. Assume $(pks') \in (pk')^{\mathbf{S}}, (pks') \in \mathfrak{s}^{s\perp P}$ and $(S_1',\ S_2' \parallel S_1,\ S_2,\ (pks')) \in R_S^-$, then in each step $k_2^iS_2'v_2' = \perp$. So also $\bigsqcup_i k_2^iS_2'v_2' = \perp$.

We conclude that $F(R^-, R^+)_K$ is closed under lubs of chains.

- $F(R^-, R^+)_V$: The proof is as the proof we have seen before, so it is omitted.

We conclude that if R^+ is admissible, so is $F(R^-, R^+)$ for any relation R^-.

□

Lemma 36. *The action of F preserves parameter weakening.*

For all $R^+, R^- \in \mathcal{R}(\mathbb{D})$.
If R^+ is parameter weakened, then $F(R^-, R^+)$ is parameter weakened.

Proof

– $F(R^-, R^+)_K$: Let $(pk_1), (pk_0) \in \mathfrak{p}^k,\ (pk_1) \rhd (pk_0)$.
For all k_1, k_2, all $(x : \tau)^\top$ continuation type, all p vm-parameter it holds that $(\perp, \perp \parallel k_1, k_2, (x : \tau)^\top, p) \in F(R^-, R^+)_K$.
Assume $(k_1', k_2') \neq (\perp, \perp)$ and $(k_1', k_2' \parallel k_1, k_2, (x : \tau)^\top, (pk_0)) \in F(R^-, R^+)_K$. We want to show $(k_1', k_2' \parallel k_1, k_2, (x : \tau)^\top, (pk_1)) \in F(R^-, R^+)_K$. This follows from $(pk_1') \rhd (pk_1)$ and $(pk_1) \rhd (pk_0)$ implies $(pk_1') \rhd (pk_0)$, (independant of R^+).

– $F(R^-, R^+)_M$: Let $p_1, p_0 \in \mathfrak{p}^{vm},\ p_1 \blacktriangleright p_0$.
For all m_1, m_2, all $T\tau$ computation type, all p vm-parameter it holds that $(\perp, \perp \parallel m_1, m_2, T\tau, p) \in F(R^-, R^+)_M$.
Assume $(m_1', m_2') \neq (\perp, \perp)$ and $(m_1', m_2' \parallel m_1, m_2, T\tau, p_0) \in F(R^-, R^+)_M$. We want to show $(m_1', m_2' \parallel m_1, m_2, T\tau, p_1) \in F(R^-, R^+)_M$. This follows from $p_1' \blacktriangleright p_1$ and $p_1 \blacktriangleright p_0$ implies $p_1' \blacktriangleright p_0$, (independant of R^+).

– $F(R^-, R^+)_V$: The proof is as we have seen before, so we omit it.

We conclude that the action of F on relations on $\mathbb{D}$ preserves parameter weakening. □

As the action of F on relations on $\mathbb{D}$ preserves admissibility, downwards closure and parameter weakening it follows that

Lemma 37. *The action of F preserves adm^+.*

For all $R^+, R^- \in \mathcal{R}(\mathbb{D})$.
If $R^+ \in \mathcal{R}_{adm^+}(\mathbb{D})$, then $F(R^-, R^+) \in \mathcal{R}_{adm^+}(\mathbb{D})$.

Lemma 38. *The action of F on functions preserves the relation $(\_ , id) : \_ \subset \_$.*

$\forall R^+, S^+, R^-, S^- \in \mathcal{R}(\mathbb{D}).\ \forall f^+, f^- : \mathbb{D} \multimap \mathbb{D}.$

If $(f^-, id_{\mathbb{D}}) : S^- \subset R^-$ and $(f^+, id_{\mathbb{D}}) : R^+ \subset S^+$ then
$(F(f^-, f^+), F(id_{\mathbb{D}}, id_{\mathbb{D}})) = (F(f^-, f^+), id_{F(\mathbb{D},\mathbb{D})}) : F(R^-, R^+) \subset F(S^-, S^+)$.

Corollary 7. *Monotonicity.*
$\forall R^+, S^+, R^-, S^- \in \mathcal{R}(\mathbb{D})$. *If $S^- \subset R^-$ and $R^+ \subset S^+$ then $F(R^-, R^+) \subset F(S^-, S^+)$.*

The corollary follows from the lemma with $f^+ = f^- = id_{\mathbb{D}}$.

Proof of lemma 38
Let $R^+, S^+, R^-, S^- \in \mathcal{R}(\mathbb{D})$, and
let $f^+, f^- : \mathbb{D} \multimap \mathbb{D}$. $f^- = (f_v^-, f_k^-, f_m^-, f_s^-),\ f^+ = (f_v^+, f_k^+, f_m^+, f_s^+)$.

Assume $(f^-, id_{\mathbb{D}}) : S^- \subset R^-$ and $(f^+, id_{\mathbb{D}}) : R^+ \subset S^+$.
We aim to show $(F(f^-, f^+), id_{F(\mathbb{D},\mathbb{D})}) : F(R^-, R^+) \subset F(S^-, S^+)$.

$(f^-, id_{\mathbb{D}}) : S^- \subset R^- \;\wedge\; (f^+, id_{\mathbb{D}}) : R^+ \subset S^+$ implies $f^- \sqsubseteq id_{\mathbb{D}} \;\wedge\; f^+ \sqsubseteq id_{\mathbb{D}}$.
By the functorial properties of F we then have $F(f^-, f^+) \sqsubseteq F(id_{\mathbb{D}}, id_{\mathbb{D}}) = id_{F(\mathbb{D},\mathbb{D})}$.

Let $F(f^-, f^+) = h = (h_v, h_k, h_m, h_s)$ and let for now $id_F = id_{F(\mathbb{D},\mathbb{D})}$.
We need to show that $s \in F(R^-, R^+) \Rightarrow ((h, id_F)s) \in F(S^-, S^+)$.

- $F(\mathbb{D},\mathbb{D})_S$
Assume $(s'_1, s'_2 \parallel s_1, s_2, (pks)) \in F(R^-, R^+)_S$, $(pks) = \{(r_1|q_1|Q_1), \ldots, (r_n|q_n|Q_n)\} \in \mathfrak{p}^s$. We aim to show $(h_s s'_1, h_s s'_2 \parallel id_s s_1, id_s s_2, (pks)) \in F(S^-, S^+)$.

  If $s'_1 = \bot = s'_2$ then since h_s is strict so $(h_s s'_1, h_s s'_2 \parallel id_s s_1, id_s s_2, p) = (\bot, \bot \parallel s_1, s_2, (pks)) \in F(S^-, S^+)_S$.
  Else $(h_s s'_1, h_s s'_2 \parallel id_s s_1, id_s s_2, (pks)) = (\lambda l. f^+_v(s'_1 l), \lambda l. f^+_v(s'_2 l) \parallel s_1, s_2, (pks))$ and
  $(s'_1, s'_2 \parallel s_1, s_2, p) \in F(R^-, R^+)_S$ so
  (a) $s'_1 \sqsubseteq s_1 \neq \bot \;\wedge\; s'_2 \sqsubseteq s_2 \neq \bot$
  (b) $\mathcal{A}_1^{(pks)}(s_1) \cap \pi_1(Z^{(pks)}) = \emptyset \;\wedge\; \mathcal{A}_2^{(pks)}(s_2) \cap \pi_2(Z^{(pks)}) = \emptyset \;\wedge$
  $\forall i \neq j.\ \mathring{A}_1^{ri}(s_1) \cap \mathring{A}_1^{rj}(s_1) = \emptyset \;\wedge\; \mathring{A}_2^{ri}(s_2) \cap \mathring{A}_2^{rj}(s_2) = \emptyset$
  (c) $\forall (l_1, l_2, \tau) \in Z^{(pks)}.(s'_1(l_1),\ s'_2(l_2) \parallel s_1(l_1),\ s_2(l_2),\ \tau,\ (pks)^{vm}) \in R^+_V$
  (d) $\forall i \in 1..n.$ if $Q_i = (P_i, \bot)$ then $s_1 \in P_i$, if $Q_i = (\bot, P_i)$ then $s_2 \in P_i$ if $Q_i = (P_i, LL_i)$ then $(s_1, s_2) \in P_i$
  (e) if $Q_i = (P_i, LL_i)$ then $\forall (l_1, l_2, \tau) \in LL_i.\ (s'_1(l_1),\ s'_2(l_2) \parallel s_1(l_1),\ s_2(l_2),\ \tau,\ (pks)^{vm}) \in R^+_V$

  and we need to show that if $h_s(s'_1) \neq \bot \;\vee\; h_s(s'_2) \neq \bot$ then
  1. $h_s(s'_1) \sqsubseteq s_1 \neq \bot \;\wedge\; h_s(s'_2) \sqsubseteq s_2 \neq \bot$
  2. $\mathcal{A}_1^{(pks)}(s_1) \cap \pi_1(Z^{(pks)}) = \emptyset \;\wedge\; \mathcal{A}_2^{(pks)}(s_2) \cap \pi_2(Z^{(pks)}) = \emptyset \;\wedge$
  $\forall i \neq j.\ \mathring{A}_1^{ri}(s_1) \cap \mathring{A}_1^{rj}(s_1) = \emptyset \;\wedge\; \mathring{A}_2^{ri}(s_2) \cap \mathring{A}_2^{rj}(s_2) = \emptyset$
  3. $\forall (l_1, l_2, \tau) \in Z^{(pks)}.((h_s s'_1)(l_1),\ (h_s s'_2)(l_2) \parallel s_1(l_1),\ s_2(l_2),\ \tau,\ (pks)^{vm}) =$
  $(f^+_v(s'_1 l_1), f^+_v(s'_2 l_2) \parallel s_1(l_1), s_2(l_2),\ \tau, (pks)^{vm}) \in S^+_V$
  4. $\forall i \in 1..n.$ if $Q_i = (P_i, \bot)$ then $s_1 \in P_i$, if $Q_i = (\bot, P_i)$ then $s_2 \in P_i$ if $Q_i = (P_i, LL_i)$ then $(s_1, s_2) \in P_i$
  5. if $Q_i = (P_i, LL_i)$ then $\forall (l_1, l_2, \tau) \in LL_i.\ ((h_s s'_1)(l_1),\ (h_s s'_2)(l_2) \parallel s_1(l_1),\ s_2(l_2),\ \tau,\ (pks)^{vm}) =$
  $(f^+_v(s'_1 l_1), f^+_v(s'_2 l_2) \parallel s_1(l_1), s_2(l_2),\ \tau, (pks)^{vm}) \in S^+_V$
  1. Follows from (a) together with $h \sqsubseteq id$.
  2. Follows from (b) directly.
  3. Follows from (c) together with the assumption $(f^+, id_{\mathbb{D}}) : R^+ \subset S^+$.
  4. Follows from (d) directly.
  5. Follows from (e) together with $(f^+, id_{\mathbb{D}}) : R^+ \subset S^+$.

  We conclude that $(h_s s'_1, h_s s'_2 \parallel id_s s_1, id_s s_2, (pks)) \in F(S^-, S^+)_S$.

- $F(\mathbb{D},\mathbb{D})_M$
Assume $(m'_1, m'_2 \parallel m_1, m_2, T\tau, p) \in F(R^-, R^+)_M$.
We aim to show $(h_m m'_1, h_m m'_2 \parallel id_3 m_1, id_3 m_2, T\tau, p) =$
$(\lambda k.\lambda S.m'_1(f^-_k k)(f^-_s S),\ \ \lambda k.\lambda S.m'_2(f^-_k k)(f^-_s S) \parallel m_1,\ m_2,\ T\tau, p)\ \in F(S^-, S^+)_M$.

  If $m'_1 = \bot = m'_2$ then since h_m is strict , so $(h_m m'_1, h_m m'_2 \parallel id_3 m_1, id_3 m_2, T\tau, p) = (\bot, \bot \parallel m_1, m_2, T\tau, p) \in F(S^-, S^+)_M$.
  Else, let $p' \blacktriangleright p$, $(pk') \in p'^{\mathbf{K}}$, $(k'_1, k'_2 \parallel k_1, k_2, (x : \tau)^\top, (pk')) \in S^-_K$,
  Let $(pks') \in (pk')^{\mathbf{S}}$, $(s'_1, s'_2 \parallel s_1, s_2, (pks')) \in S^-_S$.

Assume $(pks') \notin (\mathfrak{s}^{sP\perp} \cup \mathfrak{s}^{s\perp P})$. We want to show
$(h_m m_1')k_1's_1' = \top \Rightarrow m_2k_2s_2 = \top$ and $(h_m m_2')k_2's_2' = \top \Rightarrow m_1k_1s_1 = \top$, or equivalently
$m_1'(f_k^- k_1')(f_s^- s_1') = \top \Rightarrow m_2k_2s_2 = \top$ and $m_2'(f_k^- k_2')(f_s^- s_2') = \top \Rightarrow m_1k_1s_1 = \top$.
Since $(f^-, id_{\mathbb{D}}) : S^- \subset R^-$ it holds that $(f_k^- k_1', f_k^- k_2' \parallel k_1, k_2, (x:\tau)^\top, (pk')) \in R_K^-$ and $(f_s^- s_1', f_s^- s_2' \parallel s_1, s_2, (pks')) \in R_S^-$. Then since $(m_1', m_2' \parallel m_1, m_2, T\tau, p) \in F(R^-, R^+)$ we have that $m_1'(f_k^- k_1')(f_s^- s_1') = \top \Rightarrow m_2k_2s_2$ and $m_2'(f_k^- k_2')(f_s^- s_2') = \top \Rightarrow m_1k_1s_1$.

Assume $(pks') \in \mathfrak{s}^{sP\perp}$. We want to show $(h_m m_1')k_1's_1' = m_1'(f_k^- k_1')(f_s^- s_1') = \perp$. Since $(f^-, id_{\mathbb{D}}) : S^- \subset R^-$ it holds that $(f_k^- k_1', f_k^- k_2' \parallel k_1, k_2, (x:\tau)^\top, (pk')) \in R_K^-$ and $(f_s^- s_1', f_s^- s_2' \parallel s_1, s_2, (pks')) \in R_S^-$. Then since $(m_1', m_2' \parallel m_1, m_2, T\tau, p) \in F(R^-, R^+)$ we have that $m_1'(f_k^- k_1')(f_s^- s_1') = \perp$.
Assume $(pks') \in \mathfrak{s}^{s\perp P}$ We want to show $(h_m m_2')k_2's_2' = m_2'(f_k^- k_2')(f_s^- s_2') = \perp$. Since $(f^-, id_{\mathbb{D}}) : S^- \subset R^-$ it holds that $(f_k^- k_1', f_k^- k_2' \parallel k_1, k_2, (x:\tau)^\top, (pk')) \in R_K^-$ and $(f_s^- s_1', f_s^- s_2' \parallel s_1, s_2, (pks')) \in R_S^-$. Then since $(m_1', m_2' \parallel m_1, m_2, T\tau, p) \in F(R^-, R^+)$ we have that $m_2'(f_k^- k_2')(f_s^- s_2') = \perp$

We also need to show
$h_m(m_1') = \lambda k.\lambda S.m_1'(f_k^- k)(f_s^- S) \sqsubseteq m_1 \wedge h_m(m_2') = \lambda k.\lambda S.m_2'(f_k^- k)(f_s^- S) \sqsubseteq m_2$. This follows from $m_1' \sqsubseteq m_1 \;\wedge\; m_2' \sqsubseteq m_2$ and $f^- \sqsubseteq id_{\mathbb{D}}$.

We conclude $(h_m m_1', h_m m_2' \parallel id_3 m_1, id_3 m_2, T\tau, p) \in F(S^-, S^+)_M$.

- $F(\mathbb{D}, \mathbb{D})_K$
  Assume $(k_1', k_2' \parallel k_1, k_2, (x:\tau)^\top, (pk)) \in F(R^-, R^+)_K$.
  We aim to show $(h_k k_1', h_k k_2' \parallel id_2 k_1, id_2 k_2, (x:\tau)^\top, (pk)) =$
  $(\lambda S.\lambda v.k_1'(f_s^- S)(f_v^- v),\ \lambda S.\lambda v.k_2'(f_s^- S)(f_v^- v) \parallel k_1,\ k_2,\ (x:\tau)^\top,\ (pk)) \in F(S^-, S^+)_K$.
  This follows by similar arguments as for $(m_1', m_2' \parallel m_1, m_2, T\tau, p)$ above, so we omit the proof.

- $F(\mathbb{D}, \mathbb{D})_V$
  Assume $(v_1', v_2' \parallel v_1, v_2, \tau, p) \in F(R^-, R^+)_V$.
  As before by strictness of h_v, if $(\perp, \perp \parallel v_1, v_2, \tau, p) \in F(R^-, R^+)$ then $(h_v\perp, h_v\perp \parallel v_1, v_2, \tau, p) = (\perp, \perp \parallel v_1, v_2, \tau, p) \in F(S^-, S^+)$.

  Else $v_1' \sqsubseteq v_1 \neq \perp \;\wedge\; v_2' \sqsubseteq v_2 \neq \perp$ and v_1', v_2', v_1, v_2 fulfill the type constructor determined properties required for $(v_1',\ v_2' \parallel v_1,\ v_2, \tau, p) \in \tilde{F}(R^-, R^+)_V$. And we need to show that if $h_v(v_1') \neq \perp \;\vee\; h_v(v_2') \neq \perp$ then $h_v(v_1') \sqsubseteq v_1 \;\wedge\; h_v(v_2') \sqsubseteq v_2$ and $h_v(v_1'),\ h_v(v_2'),\ v_1,\ v_2$ fulfill the type determined properties required for $\tilde{F}(S^-, S^+)_V$. We have $h_v(v_1') \sqsubseteq v_1 \neq \perp \wedge h_v(v_2') \sqsubseteq v_2 \neq \perp$ follows from the similar property in the assumption and $h_v \sqsubseteq id_{\mathbb{D}}$.

  For the rest we argue by cases of type constructors.
  - Assume $(v_1', v_2' \parallel v_1, v_2, unit, p) \in F(R^-, R^+)$ and $(v_1' \neq \perp \vee v_2' \neq \perp)$
    then $h_v v_1' = v_1' \sqsubseteq v_1 = in_{\mathbf{1}}\lfloor * \rfloor$ and $h_v v_2' = v_2' \sqsubseteq v_2 = in_{\mathbf{1}}\lfloor * \rfloor$, and
    $(h_v v_1', h_v v_2' \parallel v_1, v_2, unit, p) \in F(S^-, S^+)$.
  - Assume $(v_1', v_2' \parallel v_1, v_2, int, p) \in F(R^-, R^+)$ and $(v_1' \neq \perp \vee v_2' \neq \perp)$
    then $\exists n$ such that $h_v v_1' = v_1' \sqsubseteq v_1 = in_{\mathbb{Z}} n$ and $h_v v_2' = v_2' \sqsubseteq v_2 = in_{\mathbb{Z}} n$, and we have
    $(h_v v_1', h_v v_2' \parallel v_1, v_2, int, p) \in F(S^-, S^+)$.
  - Assume $(v_1', v_2' \parallel v_1, v_2, \sigma ref, p) \in F(R^-, R^+)$ and $(v_1' \neq \perp \vee v_2' \neq \perp)$
    then $\exists l_1, l_2$ such that $h_v v_1' = v_1' \sqsubseteq v_1 = in_{\mathbb{L}} l_1$, $h_v v_2' = v_2' \sqsubseteq v_2 = in_{\mathbb{L}} l_2$, $(l_1, l_2, \sigma) \in Z^p$, and we have $(h_v v_1', h_v v_2' \parallel v_1, v_2, \sigma ref, p) \in F(S^-, S^+)$.
  - Assume $(v_1', v_2' \parallel v_1, v_2, \tau_a + \tau_b, p) \in F(R^-, R^+)$ and $(v_1' \neq \perp \vee v_2' \neq \perp)$ then
    $\exists d_1', d_2' \in \mathbb{V}, d_1, d_2 \in \mathbb{V}_\downarrow.\ v_1 = in_\oplus in_i d_1 \neq \perp \wedge v_2 = in_\oplus in_i d_2 \neq \perp \;\wedge$
    $(v_1' = d_1' = \perp \vee v_1' = in_\oplus in_i d_1' \neq \perp) \wedge (v_2' = d_2' = \perp \vee v_2' = in_\oplus in_i d_2' \neq \perp) \;\wedge$
    $(d_1', d_2' \parallel d_1, d_2, \tau_i, p) \in R_V^+$.

$(v_1' = d_1' = \bot \Rightarrow h_v v_1' = f_v^+ d_1' = \bot) \wedge ((v_1' = in_\oplus in_i d_1' \wedge f_v^+ d_1' = \bot) \Rightarrow h_v v_1' = f_v^+ d_1' = \bot) \wedge$
$((v_1' = in_\oplus in_i d_1' \wedge f_v^+ d_1' \neq \bot) \Rightarrow h_v v_1' = in_\oplus in_i f_v^+ d_1' \neq \bot$ and
$(v_2' = d_2' = \bot \Rightarrow h_v v_2' = f_v^+ d_2' = \bot) \wedge ((v_2' = in_\oplus in_i d_2' \wedge f_v^+ d_2' = \bot) \Rightarrow h_v v_2' = f_v^+ d_2' = \bot) \wedge$
$((v_2' = in_\oplus in_i d_2' \wedge f_v^+ d_2' \neq \bot) \Rightarrow h_v v_2' = in_\oplus in_i f_v^+ d_2' \neq \bot$.
Since $(f^+, id_{\mathbb{D}}) : R^+ \subset S^+$ and $(d_1', d_2' \parallel d_1, d_2, \tau_i, p) \in R_V^+$ it follows that
$(f_v^+ d_1', f_v^+ d_2' \parallel d_1, d_2, \tau_i, p) \in S_V^+$. It holds that $(h_v v_1' = \bot = f_v^+ d_1' \vee h_v v_1' = in_\oplus in_i f_v^+ d_1' \neq \bot)$ and $(h_v v_2' = \bot = f_v^+ d_2' \vee h_v v_2' = in_\oplus in_i f_v^+ d_2' \neq \bot)$.
So $(h_v v_1', h_v v_2' \parallel v_1, v_2, \tau_a + \tau_b, p) \in F(S^-, S^+)$

- Assume $(v_1', v_2' \parallel v_1, v_2, \tau_a \times \tau_b, p) \in F(R^-, R^+)$ and $(v_1' \neq \bot \vee v_2' \neq \bot)$. By similar arguments as for sum-typed it follows that $(h_v v_1', h_v v_2' \parallel v_1, v_2, \tau_a \times \tau_b, p) \in F(S^-, S^+)$.

- Assume $(v_1', v_2' \parallel v_1, v_2, \mu\alpha.\tau, p) \in F(R^-, R^+)$ and $(v_1' \neq \bot \vee v_2' \neq \bot)$ then $\exists d_1', d_2' \in \mathbb{V}, d_1, d_2 \in \mathbb{V}_\downarrow$. $(v_1' = d_1' = \bot \vee v_1' = in_\mu d_1' \neq \bot) \wedge (v_2' = d_2' = \bot \vee v_2' = in_\mu d_2' \neq \bot) \wedge\ v_1 = in_\mu d_1 \neq \bot \wedge v_2 = in_\mu d_2 \neq \bot$ and $(d_1', d_2' \parallel d_1, d_2, \tau[\mu\alpha.\tau/\alpha], p) \in R_V^+$.
  $(v_1' = d_1' = \bot \Rightarrow h_v v_1' = \bot)\ \wedge\ (v_1' = in_\mu d_1' \Rightarrow h_v v_1' = in_\mu f_v^+ d_1' \sqsupseteq \bot)$ and $(v_2' = d_2' = \bot \Rightarrow h_v v_2' = \bot)\ \wedge\ (v_2' = in_\mu d_2' \Rightarrow h_v v_2' = in_\mu f_v^+ d_2' \sqsupseteq \bot)$.
  Since $(f^+, id_{\mathbb{D}}) : R^+ \subset S^+$ and $(d_1', d_2' \parallel d_1, d_2, \tau[\mu\alpha.\tau/\alpha], p) \in R_V^+$ it follows that
  $(f_v^+ d_1', f_v^+ d_2' \parallel d_1, d_2, \tau[\mu\alpha.\tau/\alpha], p) \in S_V^+$.
  Then $(h_v v_1', h_v v_2' \parallel v_1, v_2, \mu\alpha.\tau, p) \in F(S^-, S^+)$

- Assume $(v_1', v_2' \parallel v_1, v_2, \forall\alpha.T\tau, p) \in F(R^-, R^+)$ and $(v_1' \neq \bot \vee v_2' \neq \bot)$ then
  $\exists m_1', m_1, m_2', m_2 \in D_M^+$. $v_1 = in_\forall \lfloor m_1 \rfloor \wedge v_2 = in_\forall \lfloor m_2 \rfloor\ \wedge$
  $((v_1' = \bot \wedge m_1' = \bot) \vee (v_1' = in_\forall \lfloor m_1' \rfloor) \wedge ((v_2' = \bot \wedge m_2' = \bot) \vee (v_2' = in_\forall \lfloor m_2' \rfloor)\ \wedge$
  $\forall\sigma$ with $\_ \vdash \sigma : type.$ $(m_1', m_2' \parallel m_1, m_2, T\tau[\sigma/\alpha], p) \in R_M^+$.
  $(v_1' = \bot \wedge m_1' = \bot) \Rightarrow (h_v v_1' = \bot \wedge f_m^+ m_1' = \bot)$ and $v_1' = in_\forall \lfloor m_1' \rfloor \Rightarrow h_v v_1' = in_\forall \lfloor f_m^+ m_1' \rfloor$,
  $(v_2' = \bot \wedge m_2' = \bot) \Rightarrow (h_v v_2' = \bot \wedge f_m^+ m_2' = \bot)$ and $v_2' = in_\forall \lfloor m_2' \rfloor \Rightarrow h_v v_2' = in_\forall \lfloor f_m^+ m_2' \rfloor$.
  Since $(f^+, id_{\mathbb{D}}) : R^+ \subset S^+$ we have $\forall\sigma$ with $\_ \vdash \sigma : type.$ $(f_m^+ m_1', f_m^+ m_2' \parallel m_1, m_2, T\tau[\sigma/\alpha], p) \in S_3^+$. Hence $(h_v v_1', h_v v_2' \parallel v_1, v_2, \forall\alpha.T\tau, p) \in F(S^-, S^+)$.

- Assume $(v_1', v_2' \parallel v_1, v_2, \tau \rightarrow T\tau', p) \in F(R^-, R^+)$ and $(v_1' \neq \bot \vee v_2' \neq \bot)$
  then $\exists d_1', d_2', d_1, d_2 \in (\mathbb{V} \multimap \mathbb{M})$. $v_1 = in_\multimap \lfloor d_1 \rfloor \wedge v_2 = in_\multimap \lfloor d_2 \rfloor\ \wedge$
  $((v_1' = \bot \wedge d_1' = \bot) \vee v_1' = in_\multimap \lfloor d_1' \rfloor) \wedge ((v_2' = \bot \wedge d_2' = \bot) \vee v_2' = in_\multimap \lfloor d_2' \rfloor) \wedge$
  $\forall p' \blacktriangleright p, \forall(\hat{w}_1'\ , \hat{w}_2' \parallel \hat{w}_1,\ \hat{w}_2,\ \tau,\ p') \in R_V^-$. $(d_1' \hat{w}_1',\ d_2' \hat{w}_2' \parallel d_1 \hat{w}_1,\ d_2 \hat{w}_2,\ T\tau',\ p') \in R_M^+$.

  We need to show $(h_v v_1', h_v v_2' \parallel v_1, v_2, \tau \rightarrow T\tau', p) \in F(S^-, S^+)$.
  $((v_1' = \bot \wedge d_1' = \bot) \Rightarrow (h_v v_1' = \bot \wedge (\lambda w. f_m^+(d_1'(f_v^- w))) = \bot)\ \wedge$
  $(v_1' = in_\multimap \lfloor d_1' \rfloor) \Rightarrow h_v v_1' = in_\multimap \lfloor (\lambda w. f_m^+(d_1'(f_v^- w))) \rfloor)\ \wedge$
  $((v_2' = \bot \wedge d_2' = \bot) \Rightarrow (h_v v_2' = \bot \wedge (\lambda w. f_m^+(d_2'(f_v^- w))) = \bot)\ \wedge$
  $(v_2' = in_\multimap \lfloor d_2' \rfloor) \Rightarrow h_v v_2' = in_\multimap \lfloor (\lambda w. f_m^+(d_2'(f_v^- w))) \rfloor)$.
  Let $p' \blacktriangleright p,\ (w_1', w_2' \parallel w_1, w_2, \tau, p') \in S_V^-$.
  Since $(f^-, id_{\mathbb{D}}) : S^- \subset R^-$ we have that $(f_v^- w_1', f_v^- w_2' \parallel w_1, w_2, \tau, p') \in R_V^-$, and then $(d_1'(f_v^- w_1'), d_2'(f_v^- w_2') \parallel d_1 w_1, d_2 w_2, T\tau', p') \in R_M^+$.
  Since $(f^+, id_{\mathbb{D}}) : R^+ \subset S^+$ then $(f_m^+(d_1'(f_v^- w_1')),\ f_m^+(d_2'(f_v^- w_2')) \parallel d_1 w_1,\ d_2 w_2,\ T\tau', p') \in S_M^+$. So $(h_v v_1', h_v v_2' \parallel v_1, v_2, \tau \rightarrow T\tau', p) \in F(S^-, S^+)$.

We conclude that in all cases $(h_v v_1', h_v v_2' \parallel v_1, v_2, \tau, p) \in F(S^-, S^+)_M$.

Then we conclude that the action of $F(-,+)$ on functions in $\mathbb{D} \multimap \mathbb{D}$ preserves $(\_, id) : \_ \subset \_$.

$\square$

Theorem 6. *There exists an* adm$^+$ *relation* $\nabla = (\nabla_V, \nabla_K, \nabla_M, \nabla_S) \in \mathcal{R}_{adm^+}(\mathbb{D})$ *satisfying the equations in definition 50 and*
$(i, i) : F(\nabla, \nabla) \subset \nabla\ \wedge\ (i^{-1}, i^{-1}) : \nabla \subset F(\nabla, \nabla)$.

With the previous lemmas 37 and 38 in hand the proof of the existence of the minimal invariant relation ∇ (theorem 6) is done in the same way as we did before. Proof omitted.

9.2 Properties of the invariant relation $\boldsymbol{\nabla = (\nabla_V, \nabla_K, \nabla_M, \nabla_S) \in \mathcal{R}_{adm^+}(\mathbb{D})}$

∇ is admissible , downwards closed and parameter-weakened, and we have
$(i, i) : F(\nabla, \nabla) \subset \nabla \;\wedge\; (i^{-1}, i^{-1}) : \nabla \subset F(\nabla, \nabla)$.

Let $F(\nabla, \nabla) = (F(\nabla, \nabla)_V, F(\nabla, \nabla)_K, F(\nabla, \nabla)_M, F(\nabla, \nabla)_S)$, then further by definition of the action of F on relations it holds that:

$$\begin{aligned}
F(R^-, R^+)_V = \{&(\bot,\ \bot \parallel v_1,\ v_2,\ \tau,\ p) \mid p \in \mathfrak{p}^{vm} \text{ the set of all vm-parameters}, v_1, v_2 \in F(\mathbb{D}, \mathbb{D})_V\} \cup \\
&\{(v_1',\ v_2' \parallel v_1,\ v_2,\ \tau,\ p) \mid p \in \mathfrak{p}^{vm} \;\wedge\; v_1', v_2', v_1, v_2 \in F(\mathbb{D}, \mathbb{D})_V \;\wedge \\
&\quad v_1' \sqsubseteq v_1 \neq \bot \;\wedge\; v_2' \sqsubseteq v_2 \neq \bot \;\wedge \\
&\quad (v_1',\ v_2' \parallel v_1,\ v_2, \tau,\ p) \in \diamond\ \ \}
\end{aligned}$$

where

$$\begin{aligned}
\diamond \quad = \{&(v_1',\ v_2' \parallel in_{\mathbf{1}}\lfloor * \rfloor,\ in_{\mathbf{1}}\lfloor * \rfloor,\ unit,\ p)\ \} \cup \\
\{&(v_1',\ v_2' \parallel in_{\mathbb{Z}}\lfloor n \rfloor,\ in_{\mathbb{Z}}\lfloor n \rfloor,\ int,\ p) \mid n \in \mathbb{Z}\ \} \cup \\
\{&(v_1',\ v_2' \parallel in_{\mathbb{L}}\lfloor l_1 \rfloor,\ in_{\mathbb{L}}\lfloor l_2 \rfloor,\ \tau\ ref,\ p) \mid (l_1, l_2, \tau) \in Z_p\ \ \} \cup
\end{aligned}$$

$$\begin{aligned}
\{&(v_1',\ v_2' \parallel in_{\oplus} in_i d_1,\ in_{\oplus} in_i d_2,\ \tau_1 + \tau_2,\ p) \mid d_1, d_2 \in (\mathbb{D}_V)_{\downarrow} \;\wedge\; \exists d_1', d_2' \in \mathbb{D}_V. \\
&\quad ((v_1' = \bot \wedge d_1' = \bot) \vee v_1' = in_{\oplus} in_i d_1' \neq \bot) \wedge ((v_2' = \bot \wedge d_2' = \bot) \vee v_2' = in_{\oplus} in_i d_2' \neq \bot) \wedge \\
&\quad (d_1',\ d_2' \parallel d_1,\ d_2,\ \tau_i,\ p) \in \nabla_V,\ i \in \{1, 2\}\ \} \cup
\end{aligned}$$

$$\begin{aligned}
\{&(v_1',\ v_2' \parallel in_{\otimes}(d_{1a}, d_{1b}),\ in_{\otimes}(d_{2a}, d_{2b}),\ \tau_a \times \tau_b,\ p) \mid d_{1a}, d_{1b}, d_{2a}, d_{2b} \in (\mathbb{D}_V)_{\downarrow} \;\wedge \\
&\quad \exists d_{1a}', d_{1b}', d_{2a}', d_{2b}' \in \mathbb{D}_V. \\
&\quad (d_{1a}',\ d_{2a}' \parallel d_{1a},\ d_{2a},\ \tau_a,\ p) \in \nabla_V \;\wedge\; (d_{1b}',\ d_{2b}' \parallel d_{1b},\ d_{2b},\ \tau_b,\ p) \in \nabla_V \;\wedge \\
&\quad ((v_1' = \bot \wedge (d_{1a}' = \bot \vee d_{1b}' = \bot)) \;\vee\; (v_1' = in_{\otimes}(d_{1a}', d_{1b}') \neq \bot)) \;\wedge \\
&\quad ((v_2' = \bot \wedge (d_{2a}' = \bot \vee d_{2b}' = \bot)) \;\vee\; (v_2' = in_{\otimes}(d_{2a}', d_{2b}') \neq \bot))\} \cup
\end{aligned}$$

$$\begin{aligned}
\{&(v_1',\ v_2' \parallel in_{\mu} d_1,\ in_{\mu} d_2,\ \mu\alpha.\tau,\ p) \mid d_1, d_2 \in (\mathbb{D}_V)_{\downarrow} \;\wedge\; \exists d_1', d_2' \in \mathbb{D}_V. \\
&\quad ((v_1' = \bot \wedge d_1' = \bot) \vee v_1' = in_{\mu} d_1' \neq \bot) \wedge ((v_2' = \bot \wedge d_2' = \bot) \vee v_2' = in_{\mu} d_2' \neq \bot) \wedge \\
&\quad (d_1',\ d_2' \parallel d_1,\ d_2,\ \tau[\mu\alpha.\tau/\alpha],\ p) \in \nabla_V\ \} \cup
\end{aligned}$$

$$\begin{aligned}
\{&(v_1',\ v_2' \parallel in_{\forall}\lfloor d_1 \rfloor,\ in_{\forall}\lfloor d_2 \rfloor,\ \forall\alpha.T\tau,\ p) \mid d_1, d_2 \in \mathbb{D}_M \;\wedge\; \exists d_1', d_2' \in \mathbb{D}_M. \\
&\quad ((v_1' = \bot \wedge d_1' = \bot) \;\vee\; (v_1' = in_{\forall}\lfloor d_1' \rfloor \neq \bot \;\wedge\; d_1' \sqsupseteq \bot)) \;\wedge \\
&\quad ((v_2' = \bot \wedge d_2' = \bot) \;\vee\; (v_2' = in_{\forall}\lfloor d_2' \rfloor \neq \bot \;\wedge\; d_2' \sqsupseteq \bot)) \;\wedge \\
&\quad \forall\sigma\ with\ \_ \vdash \sigma : type.\ (d_1',\ d_2' \parallel d_1,\ d_2,\ T\tau[\sigma/\alpha],\ p) \in \nabla_M\ \} \cup
\end{aligned}$$

$$\begin{aligned}
\{&(v_1',\ v_2' \parallel in_{\multimap}\lfloor d_1 \rfloor,\ in_{\multimap}\lfloor d_2 \rfloor,\ \tau \to \mathrm{T}\tau',\ p) \mid d_1, d_2 \in (\mathbb{D}_V \multimap \mathbb{D}_M) \;\wedge\; \exists d_1', d_2' \in (\mathbb{D}_V \multimap \mathbb{D}_M). \\
&\quad ((v_1' = \bot \wedge d_1' = \bot) \;\vee\; (v_1' = in_{\multimap}\lfloor d_1' \rfloor \neq \bot \;\wedge\; d_1' \sqsupseteq \bot)) \;\wedge \\
&\quad ((v_2' = \bot \wedge d_2' = \bot) \;\vee\; (v_2' = in_{\multimap}\lfloor d_2' \rfloor \neq \bot \;\wedge\; d_2' \sqsupseteq \bot)) \;\wedge \\
&\quad \forall p' \blacktriangleright p, \forall(w_1',\ w_2' \parallel w_1,\ w_2, \tau,\ p') \in \nabla_V.\ (\ d_1'w_1',\ d_2'w_2' \parallel d_1 w_1,\ d_1 w_2,\ T\tau',\ p') \in \nabla_M\ \}
\end{aligned}$$

$$\begin{aligned}
F(\nabla, \nabla)_K = \{&(k_1',\ k_2' \parallel k_1,\ k_2,\ (x : \tau)^{\top},\ (pk)) \mid (pk) \in \mathfrak{p}^k \;\wedge \\
&\quad k_1' \sqsubseteq k_1 \;\wedge\; k_2' \sqsubseteq k_2 \;\wedge\; \forall(pk') \rhd (pk).
\end{aligned}$$

(that is (pk) extended only with *ordinary* local k-parameters or local extensions)

$\forall(pks') \in (pk')^{\mathbf{S}}$ (the set of s-instantiations of p' i.e. choices of $\bar{\nabla}$-clauses),
$(pks') \notin (\mathfrak{s}^{sP\bot} \cup \mathfrak{s}^{s\bot P})$. (i.e. no chosen $\bar{\nabla}$-clause in the s-instantiation is a $(\bot, P)$ or $(P, \bot)$)

$$\begin{aligned}
&\forall(s_1',\ s_2' \parallel s_1,\ s_2,\ (pks')) \in \nabla_S.\ \forall(v_1',\ v_2' \parallel v_1,\ v_2, \tau,\ (pk')^{vm}) \in \nabla_V. \\
&\quad (k_1's_1'v_1' = \top \Rightarrow k_2 s_2 v_2 = \top) \;\wedge \\
&\quad (k_2's_2'v_2' = \top \Rightarrow k_1 s_1 v_1 = \top), \qquad \wedge
\end{aligned}$$

$\forall (pks') \in (pk')^{\mathbf{S}}, (pks') \in \mathfrak{s}^{sP\perp}$. (i.e. a chosen $\bar{\vee}$-clause in the s-instantiation is a $(P, \perp)$)
$\quad \forall (s_1',\ s_2' \parallel s_1,\ s_2,\ (pks')) \in \nabla_S.\ \forall (v_1',\ v_2' \parallel v_1,\ v_2,\ \tau,\ (pk')^{vm}) \in \nabla_V.$
$\qquad k_1' s_1' v_1' = \perp, \qquad \wedge,$

$\forall (pks') \in (pk')^{\mathbf{S}}, (pks') \in \mathfrak{s}^{s\perp P}$. (i.e. a chosen $\bar{\vee}$-clause in the s-instantiation is a $(\perp, P)$)
$\quad \forall (s_1',\ s_2' \parallel s_1,\ s_2,\ (pks')) \in \nabla_S.\ \forall (v_1',\ v_2' \parallel v_1,\ v_2,\ \tau,\ (pk')^{vm}) \in \nabla_V.$
$\qquad k_2' s_2' v_2' = \perp\ \}$,

$F(\nabla, \nabla)_M = \{(m_1',\ m_2' \parallel m_1,\ m_2,\ T\tau,\ p) \mid p \in \mathfrak{p}^{vm}\ \wedge$
$m_1' \sqsubseteq m_1\ \wedge\ m_2' \sqsubseteq m_2\ \wedge$
$\forall p' \blacktriangleright p. \forall (pk') \in p'^{\mathbf{K}}$ (the set of k-instantiations of p' i.e. choices of a $\bar{\wedge}$-clause in each local parameter).

$\forall (pks') \in (pk')^{\mathbf{S}}$ (the set of s-instantiations of (pk') i.e. choices of $\bar{\vee}$-clause from each chosen $\bar{\wedge}$-clause in (pk')),
$(pks') \notin (\mathfrak{s}^{sP\perp} \cup \mathfrak{s}^{s\perp P})$. (i.e. no chosen $\bar{\vee}$-clause is a $(\perp, P)$ or $(P, \perp)$)
$\quad \forall (k_1',\ k_2' \parallel k_1,\ k_2,\ (x:\tau)^{\top},\ (pk')) \in \nabla_K.\ \forall (S_1',\ S_2' \parallel S_1,\ S_2,\ (pks')) \in \nabla_S$.
$\qquad (m_1' k_1' S_1' = \top \Rightarrow m_2 k_2 S_2 = \top)\ \wedge$
$\qquad (m_2' k_2' S_2' = \top \Rightarrow m_1 k_1 S_1 = \top)\ \wedge$

$\forall (pks') \in (pk')^{\mathbf{S}}, (pks') \in \mathfrak{s}^{sP\perp}$. (i.e. a chosen $\bar{\vee}$-clause is a $(P, \perp)$)
$\quad \forall (k_1',\ k_2' \parallel k_1,\ k_2,\ (x:\tau)^{\top},\ (pk')) \in \nabla_K.\ \forall (S_1',\ S_2' \parallel S_1,\ S_2,\ (pks')) \in \nabla_S.$
$\qquad m_1' k_1' S_1' = \perp)\ \wedge$

$\forall (pks') \in (pk')^{\mathbf{S}}, (pks') \in \mathfrak{s}^{s\perp P}$. (i.e. a chosen $\bar{\vee}$-clause is a $(\perp, P)$)
$\quad \forall (k_1',\ k_2' \parallel k_1,\ k_2,\ (x:\tau)^{\top},\ (pk')) \in \nabla_K.\ \forall (S_1',\ S_2' \parallel S_1,\ S_2,\ (pks')) \in \nabla_S.$
$\qquad m_2' k_2' S_2' = \perp\ \}$

$F(\nabla, \nabla)_S = \{(\perp,\ \perp \parallel S_1,\ S_2,\ (pks)) \mid (pks) \in \mathfrak{p}^s$ the set of all s-parameters $\} \cup$
$\{(S_1',\ S_2' \parallel\ S_1,\ S_2,\ (pks) \mid (pks) = \{(r_1|q_1|Q_1), \ldots, (r_n|q_n|Q_n)\} \in \mathfrak{p}^s$
(q_i is a choice of $\bar{\wedge}$-clause in r_i, and Q_i is a choice of $\bar{\vee}$-clause in q_i) $\wedge$

$S_1' \sqsubseteq S_1 \neq \perp\ \ \wedge\ S_2' \sqsubseteq S_2 \neq \perp\ \wedge$

$\mathcal{A}_1^{(pks)}(S_1) \cap \pi_1(Z^{(pks)}) = \emptyset\ \wedge\ \mathcal{A}_2^{(pks)}(S_2) \cap \pi_2(Z^{(pks)}) = \emptyset$
(in each side are the visible locations in $Z^{(pks)}$ disjoint from the known hidden locations given by the accessibility maps $\mathcal{A}_1^{(pks)}, \mathcal{A}_2^{(pks)}$) $\quad \wedge$

$\forall i \neq j.\ \mathring{A}_1^{ri}(S_1) \cap \mathring{A}_1^{rj}(S_1) = \emptyset\ \wedge\ \mathring{A}_2^{ri}(S_2) \cap \mathring{A}_2^{rj}(S_2) = \emptyset$
($\mathring{A}_1^{ri}, \mathring{A}_2^{ri}$ accessibility maps are the most inclusive for the local parameter r_i. In each side is every location, visible, hidden or reserved, belonging to a local parameter outside the areas owned by any different local parameter) $\quad \wedge$

$\forall (l_1, l_2, \tau) \in Z^{(pks)}.(S_1'(l_1),\ S_2'(l_2) \parallel S_1(l_1),\ S_2(l_2),\ \tau,\ (pks)^{vm}) \in \nabla_V$
(All visible locations hold related values) $\quad \wedge$

$\forall i \in 1..n.$ if $Q_i = (P_i, \perp)$ then $S_1 \in P_i$, if $Q_i = (\perp, P_i)$ then $S_2 \in P_i$,
if $Q_i = (P_i, LL_i)$ then $(S_1, S_2) \in P_i\ \wedge$
$\quad \forall (l_1, l_2, \tau) \in LL_i.\ (S_1'(l_1),\ S_2'(l_2) \parallel S_1(l_1),\ S_2(l_2),\ \tau,\ (pks)^{vm}) \in \nabla_V$
(The states belong to all simple state relations and predicates in the chosen $\bar{\vee}$-clauses. Corresponding LL location-sets hold related values) $\}$

– For all $(v_1',\ v_2' \parallel v_1,\ v_2,\ \tau,\ p) \in \nabla_V$ it holds that $(v_1' = v_2' = \bot)$ or $(v_1 \neq \bot \wedge v_2 \neq \bot)$.
– For all $(s_1',\ s_2' \parallel s_1,\ s_2,\ \tau,\ p) \in \nabla_V$ it holds that $(s_1' = s_2' = \bot)$ or $(s_1 \neq \bot \wedge s_2 \neq \bot)$.
– $\forall k_1, k_2 \in \mathbb{K}.\forall (x:\tau)^\top.\forall pk \in \mathfrak{p}^k.(\bot, \bot \parallel k_1, k_2, (x:\tau)^\top, pk) \in \nabla_K$.
– $\forall m_1, m_2 \in \mathbb{M}.\forall T\tau.\forall p \in \mathfrak{p}^{vm}.(\bot, \bot \parallel m_1, m_2, T\tau, p) \in \nabla_M$.

Again we have a special limited parameter weakening for states concerned with removal of $\bar{\wedge}$-clauses. If states s_1', s_2', s_1, s_2 are related under an s-parameter pks and pks' is a s-parameter derived from pks by removal of some $\bar{\wedge}$-clauses but such that those instantiated in pks are not touched, then s_1', s_2', s_1, s_2 will also be related under pks'.

Lemma 39. *Let $p = \{r_i\}$ be a vm-parameter, and let $pks = \{(r_i|q_i|Q_i)\} \in p^{\mathbf{S}}$.*
Assume $(s_1', s_2' \parallel s_1, s_2, pks) \in \nabla_S$.
If $\forall i.r_i \supseteq r_i'$ and $pks' = \{(r_i'|q_i|Q_i)\}$,
then $p' = (pks')^{vm} = \{r_i'\}$ and $(s_1', s_2' \parallel s_1, s_2, pks') \in \nabla_S$.

Proof By definition of $\blacktriangleright$ it holds that $p' = (pks')^{vm} = \{r_i'\} \blacktriangleright p$.
The approximation properties $s_1' \sqsubseteq s_1$ and $s_2' \sqsubseteq s_2$ are not changed. By assumption

(a) $\mathcal{A}_1^{(pks)}(s_1) \cap \pi_1(Z^{(pks)}) = \emptyset\ \wedge\ \mathcal{A}_2^{(pks)}(s_2) \cap \pi_2(Z^{(pks)}) = \emptyset$ and
(b) $\forall i \neq j.\ \mathring{A}_1^{ri}(s_1) \cap \mathring{A}_1^{rj}(s_1) = \emptyset\ \wedge\ \mathring{A}_2^{ri}(s_2) \cap \mathring{A}_2^{rj}(s_2) = \emptyset$ and
(c) $\forall i \in 1..n.$ if $Q_i = (P_i, \bot)$ then $s_1 \in P_i$, if $Q_i = (\bot, P_i)$ then $s_2 \in P_i$,
if $Q_i = (P_i, LL_i)$ then $(s_1, s_2) \in P_i\ \wedge$
$\forall (l_1, l_2, \tau) \in LL_i.\ (s_1'(l_1),\ s_2'(l_2) \parallel s_1(l_1),\ Ss_2(l_2),\ \tau,\ (pks)^{vm}) \in \nabla_V$ and
(d) $\forall (l_1, l_2, \tau) \in Z^{(pks)}.(s_1'(l_1),\ s_2'(l_2) \parallel s_1(l_1),\ s_2(l_2),\ \tau,\ (pks)^{vm}) \in \nabla_V$

We need to prove

1. $\mathcal{A}_1^{(pks')}(s_1) \cap \pi_1(Z^{(pks')}) = \emptyset\ \wedge\ \mathcal{A}_2^{(pks')}(s_2) \cap \pi_2(Z^{(pks')}) = \emptyset$ and
2. $\forall i \neq j.\ \mathring{A}_1^{ri'}(s_1) \cap \mathring{A}_1^{rj'}(s_1) = \emptyset\ \wedge\ \mathring{A}_2^{ri'}(s_2) \cap \mathring{A}_2^{rj'}(s_2) = \emptyset$ and
3. $\forall i \in 1..n.$ if $Q_i = (P_i, \bot)$ then $s_1 \in P_i$, if $Q_i = (\bot, P_i)$ then $s_2 \in P_i$,
if $Q_i = (P_i, LL_i)$ then $(s_1, s_2) \in P_i\ \wedge$
$\forall (l_1, l_2, \tau) \in LL_i.\ (s_1'(l_1),\ s_2'(l_2) \parallel s_1(l_1),\ Ss_2(l_2),\ \tau,\ (pks')^{vm}) \in \nabla_V$ and
4. $\forall (l_1, l_2, \tau) \in Z^{(pks')}.(s_1'(l_1),\ s_2'(l_2) \parallel s_1(l_1),\ s_2(l_2),\ \tau,\ (pks')^{vm}) \in \nabla_V$

1. follows from (a). $Z^{(pks')} = \biguplus Z^{q_i}$ is defined on the basis of the instantiated q_i's in (pks'). These are the same as in (pks), so $Z^{(pks')} = Z^{(pks)}$. Also $\mathcal{A}_1^{(pks')}, \mathcal{A}_2^{(pks')}, \mathcal{A}_1^{(pks)}, \mathcal{A}_2^{(pks)}$ are defined on the basis of the instantiated q_i's.
2. follows from (b) together with $\forall i.\ r_i' \subseteq r_i \Rightarrow (\mathring{A}_1^{ri'}(s_1) \subseteq \mathring{A}_1^{ri}(s_1) \wedge \mathring{A}_2^{ri'}(s_2) \subseteq \mathring{A}_2^{ri}(s_2))$.
3. There are no changes to finitary state predicates and finitary state relations, so also the LL's will be the same. As $(pks')^{vm} \blacktriangleright (pks)^{vm}$ then by weakening for values and (c) all values stored in LL's are still related.
4. follows from (d) and $Z^{(pks')} = Z^{(pks)}$ and $(pks')^{vm} \blacktriangleright (pks)^{vm}$ together with parameter weakening for stored values.

□

9.3 Relating denotations of open terms

We define a parameterized binary relation $\nabla^{\Xi\Gamma}$ between denotations of open expressions formally as it was before, but now it is based on the new relation ∇.

Definition 51. *The relation $\nabla^{\Xi\Gamma}$*

Let $p \in \mathfrak{o}^{vm}$ with associated Z^p and with $Z_1^p = Z_2^p = \Delta$

$\nabla_V^{\Xi\Gamma}$ *For* $\Xi = \alpha_1 \ldots \alpha_m,\ \Gamma = x_1 : \tau_1, \ldots, x_n : \tau_n$ *and* $\Delta; \Xi; \Gamma \vdash V_1 : \tau$ *and* $\Delta; \Xi; \Gamma \vdash V_2 : \tau$ *let* $v_1 = [\![\Delta; \Xi; \Gamma \vdash V_1]\!]$ *and* $v_2 = [\![\Delta; \Xi; \Gamma \vdash V_2]\!]$*, define*

$$(v_1, v_2, \tau, p) \in \nabla_V^{\Xi\Gamma} \ \overset{def}{\Longleftrightarrow}$$

$\forall \overline{\sigma_j}$ *with* $\_ \vdash \sigma_j : type,\ j = 1, .., m.\ \forall p' \blacktriangleright p.$
$\forall (v'_{11}, v'_{21} \parallel v_{11}, v_{21}, \tau_1[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V, \ldots, (v'_{1n}, v'_{2n} \parallel v_{1n}, v_{2n}, \tau_n[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V,$ *with*
$\rho'_1 = v'_{11} \otimes \ldots \otimes v'_{1n} \in [\![\Gamma]\!],\ \rho_1 = v_{11} \otimes \ldots \otimes v_{1n} \in [\![\Gamma]\!],$
$\rho'_2 = v'_{21} \otimes \ldots \otimes v'_{2n} \in [\![\Gamma]\!],\ \rho_2 = v_{21} \otimes \ldots \otimes v_{2n} \in [\![\Gamma]\!].$

it holds that $(v_1(\rho'_1), v_2(\rho'_2) \parallel v_1(\rho_1), v_2(\rho_2), \tau[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V.$

$\nabla_M^{\Xi\Gamma}$ *For* $\Xi = \alpha_1 \ldots \alpha_m,\ \Gamma = x_1 : \tau_1, \ldots, x_n : \tau_n$ *and* $\Delta; \Xi; \Gamma \vdash M_1 : T\tau$ *and* $\Delta; \Xi; \Gamma \vdash M_2 : T\tau$ *let* $m_1 = [\![\Delta; \Xi; \Gamma \vdash M_1]\!]$ *and* $m_2 = [\![\Delta; \Xi; \Gamma \vdash M_2]\!]$*, define*

$$(m_1, m_2, T\tau, p) \in \nabla_M^{\Xi\Gamma} \ \overset{def}{\Longleftrightarrow}$$

$\forall \overline{\sigma_j}$ *with* $\_ \vdash \sigma_j : type,\ j = 1, .., m.\ \forall p' \blacktriangleright p.$
$\forall (v'_{11}, v'_{21} \parallel v_{11}, v_{21}, \tau_1[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V, \ldots, (v'_{1n}, v'_{2n} \parallel v_{1n}, v_{2n}, \tau_n[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V,$ *with*
$\rho'_1 = v'_{11} \otimes \ldots \otimes v'_{1n} \in [\![\Gamma]\!],\ \rho_1 = v_{11} \otimes \ldots \otimes v_{1n} \in [\![\Gamma]\!],$
$\rho'_2 = v'_{21} \otimes \ldots \otimes v'_{2n} \in [\![\Gamma]\!],\ \rho_2 = v_{21} \otimes \ldots \otimes v_{2n} \in [\![\Gamma]\!].$

it holds that $(m_1(\rho'_{1i}), m_2(\rho'_{2i}) \parallel m_1(\rho_{1i}), m_2(\rho_{2i}), T\tau[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M.$

Proposition 5. *Typing rules preserve* $\nabla^{\Xi\Gamma}$ *relation of denotations. for all* $p \in \mathfrak{o}^{vm}$.

For typing rules with no premisses it holds that for all $p \in \mathfrak{o}^{vm}$ *with* $Z_1^p = Z_2^p = \Delta$

if $\dfrac{}{\Delta; \Xi; \Gamma \vdash V : \tau}$ *and* $v = [\![\Delta; \Xi; \Gamma \vdash V : \tau]\!]$ *then* $(v, v, \tau, p) \in \nabla_V^{\Xi\Gamma}$

For typing rules with j *premisses it holds that for all* $p \in \mathfrak{o}^{vm}$ *with* $Z_1^p = Z_2^p = \Delta$

if $\dfrac{\Delta; \Xi_1; \Gamma_1 \vdash G_{11} : \gamma_1 \ \ \ldots\ldots \ \ \Delta; \Xi_j; \Gamma_j \vdash G_{j1} : \gamma_j}{\Delta; \Xi; \Gamma \vdash G_1 : \gamma}$ *and by the same typing rule*

$\dfrac{\Delta; \Xi_1; \Gamma_1 \vdash G_{12} : \gamma_1 \ \ \ldots\ldots \ \ \Delta; \Xi_j; \Gamma_j \vdash G_{j2} : \gamma_j}{\Delta; \Xi; \Gamma \vdash G_2 : \gamma}$ *and for the well typed terms*

$d_{11} = [\![\Delta; \Xi_1; \Gamma_1 \vdash G_{11} : \gamma_1]\!],\ d_{12} = [\![\Delta; \Xi_1; \Gamma_1 \vdash G_{12} : \gamma_1]\!], \ldots,$
$d_{j1} = [\![\Delta; \Xi_j; \Gamma_j \vdash G_{j1} : \gamma_j]\!],\ d_{j2} = [\![\Delta; \Xi_j; \Gamma_j \vdash G_{j2} : \gamma_j]\!],$ *and*
$\forall i \in 1 \ldots j,\ (d_{i1}, d_{i2}, \gamma_i, p) \in \nabla_X^{\Xi_i\Gamma_i}$ *(where in each case* X *is the relevant* $X \in \{V, M\}$*).*
And $d_1 = [\![\Delta; \Xi; \Gamma \vdash G_1 : \gamma]\!],\ d_2 = [\![\Delta; \Xi; \Gamma \vdash G_2 : \gamma']\!].$

then it holds that $(d_1, d_2, \gamma, p) \in \nabla_X^{\Xi\Gamma}$

Proof
Let $p \in \mathfrak{o}^{vm}$ be a parameter such that $Z^p = \{(l_1, l_1, \sigma_1) \ldots (l_n, l_n, \sigma_n)\}$ where $\Delta = Z_1^p = Z_2^p$.

For $\Xi = \alpha_1 \ldots \alpha_m$, $\Gamma = x_1 : \tau_1, \ldots, x_n : \tau_n$ with $\Xi \vdash \Gamma$ and
arbitrary $\sigma_1 \ldots \sigma_m$ with $\_ \vdash \sigma_k : type$,

arbitrary $p' \blacktriangleright p$, *arbitrary* $(v'_{1i}, v'_{2i} \parallel v_{1i}, v_{2i}, \tau_i[\overline{\sigma_k/\alpha_k}], p') \in \nabla_V,\ i = 1, .., n.$

Let $\rho_1' = v_{11}' \otimes \ldots \otimes v_{1n}' \in [\![\Gamma]\!]$. Let $\rho_1 = v_{11} \otimes \ldots \otimes v_{1n} \in [\![\Gamma]\!]$.
Let $\rho_2' = v_{21}' \otimes \ldots \otimes v_{2n}' \in [\![\Gamma]\!]$. Let $\rho_2 = v_{21} \otimes \ldots \otimes v_{2n}' \in [\![\Gamma]\!]$.

As $\forall i.\ v_{1i}' \sqsubseteq v_{1i} \wedge v_{2i}' \sqsubseteq v_{2i}$ and $\big((v_{1i}' = v_{2i}' = \bot) \vee (v_{1i} \neq \bot \wedge v_{2i} \neq \bot)\big)$, then it holds that
$\rho_1' \sqsubseteq \rho_1 \wedge \rho_2' \sqsubseteq \rho_2$
$(\rho_1' = \rho_2' = \bot)\ \vee (\rho_1 \neq \bot \wedge \rho_2 \neq \bot)$
$(\rho_1 = \bot \vee \rho_2 = \bot) \Rightarrow (\rho_1' = \bot \wedge \rho_2' = \bot)$
$(\rho_1' \neq \bot \vee \rho_2' \neq \bot) \Rightarrow (\rho_1 \neq \bot \wedge \rho_2 \neq \bot)$.

We will now show for each typing rule that relatedness of denotations is preserved. Most of the proofs are very much like the ones we have seen before. No rule will need a new local parameter, which is not ordinary. The new extended parameters have a special treatment for state-instantiations to $(P, \bot)$ and $(\bot, P)$. In proofs the required termination behavior comes through by assumption about relatedness of the continuations and states we apply to. Here we present proofs for selected rules. The isomorphism i is sometimes omitted.

val: $$\frac{\Delta; \Xi; \Gamma \vdash V : \tau}{\Delta; \Xi; \Gamma \vdash val\ V : T\tau}$$

Let $v_1 = [\![\Delta; \Xi; \Gamma \vdash V_1 : \tau]\!]$ and $v_2 = [\![\Delta; \Xi; \Gamma \vdash V_2 : \tau]\!]$. Assume $(v_1, v_2, \tau, p) \in \nabla_V^{\Xi\Gamma}$.
This assumption implies $(v_1(\rho_1'), v_2(\rho_2') \parallel v_1(\rho_1), v_2(\rho_2), \tau[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$

Let $m_1 = [\![\Delta; \Xi; \Gamma \vdash val\ V_1 : T\tau]\!]$ and $m_2 = [\![\Delta; \Xi; \Gamma \vdash val\ V_2 : T\tau]\!]$.
We need to show $(m_1, m_2, T\tau, p) \in \nabla_M^{\Xi\Gamma}$, that is we want to show
$(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T\tau[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$, or
$(i^{-1}(m_1(\rho_1')), i^{-1}(m_2(\rho_2')) \parallel i^{-1}(m_1(\rho_1)), i^{-1}(m_2(\rho_2)), T\tau[\overline{\sigma_j/\alpha_j}], p') \in F(\nabla, \nabla)_M$.

$i^{-1}(m_1(\rho_1')) = \lambda k.\lambda S.i^{-1}(k)S(v_1(\rho_1'))$, $i^{-1}(m_2(\rho_2')) = \lambda k.\lambda S.i^{-1}(k)S(v_2(\rho_2'))$,
$i^{-1}(m_1(\rho_1)) = \lambda k.\lambda S.i^{-1}(k)S(v_1(\rho_1))$, $i^{-1}(m_2(\rho_2)) = \lambda k.\lambda S.i^{-1}(k)S(v_2(\rho_2))$.

If $v_1(\rho_1') = v_2(\rho_2') = \bot$ then $m_1(\rho_1')$ and $m_2(\rho_2')$ will both be the constant $\bot$ function in $\mathbb{M}$, and hence $(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T\tau[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$.

Else, let $p'' \blacktriangleright p'$, $(pk'') \in p''^{\mathbf{K}}$, $(pks'') \in pk''^{\mathbf{S}}$ and let $(k_1', k_2' \parallel k_1, k_2, (x : \tau[\overline{\sigma_j/\alpha_j}])^\top, (pk'')) \in \nabla_K, (S_1', S_2' \parallel S_1, S_2, (pks'')) \in \nabla_S$.

We need to show
$(pks'') \in \mathfrak{p}^s \setminus (\mathfrak{s}^{sP\bot} \cup \mathfrak{s}^{s\bot P}) \Longrightarrow$
$\quad (m_1(\rho_1'))k_1'S_1' = \top \Rightarrow (m_2(\rho_2))k_2S_2 = \top\ \wedge\ (m_2(\rho_2'))k_2'S_2' = \top \Rightarrow (m_1(\rho_1))k_1S_1 = \top$
$(pks'') \in \mathfrak{s}^{sP\bot} \Longrightarrow (m_1(\rho_1'))k_1'S_1' = \bot$
$(pks'') \in \mathfrak{s}^{s\bot P} \Longrightarrow (m_2(\rho_2'))k_2'S_2' = \bot$

By assumption and weakening $(v_1(\rho_1'), v_2(\rho_2') \parallel v_1(\rho_1), v_2(\rho_2), \tau[\overline{\sigma_j/\alpha_j}], p'') \in \nabla_V$.
$i^{-1}(m_1(\rho_1'))k_1'S_1' = i^{-1}(k_1')S_1'v_1(\rho_1')$, $i^{-1}(m_2(\rho_2'))k_2'S_2' = i^{-1}(k_2')S_2'v_2(\rho_2')$,
$i^{-1}(m_1(\rho_1))k_1S_1 = i^{-1}(k_1)S_1v_1(\rho_1)$, $i^{-1}(m_2(\rho_2))k_2S_2 = i^{-1}(k_2)S_2v_2(\rho_2)$.

So it follows that since k's, S's and v's are correspondingly related so
$(pks'') \in \mathfrak{p}^s \setminus (\mathfrak{s}^{sP\bot} \cup \mathfrak{s}^{s\bot P}) \Longrightarrow$
$\quad (m_1(\rho_1'))k_1'S_1' = (k_1')S_1'v_1(\rho_1') = \top \Rightarrow (k_2)S_2v_2(\rho_2) = (m_2(\rho_2))k_2S_2 = \top\ \wedge$
$\quad (m_2(\rho_2'))k_2'S_2' = (k_2')S_2'v_2(\rho_2') = \top \Rightarrow (k_1)S_1v_1(\rho_1) = (m_1(\rho_1))k_1S_1 = \top$
$(pks'') \in \mathfrak{s}^{sP\bot} \Longrightarrow (k_1')S_1'v_1(\rho_1') = (m_1(\rho_1'))k_1'S_1' = \bot$
$(pks'') \in \mathfrak{s}^{s\bot P} \Longrightarrow (k_2')S_2'v_2(\rho_2') = (m_2(\rho_2'))k_2'S_2' = \bot$

Hence $(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T\tau[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$.
We conclude $(m_1, m_2, T\tau, p) \in \nabla_M^{\Gamma}$.

deref: $\dfrac{\Delta; \Xi; \Gamma \vdash V : \tau\ ref}{\Delta; \Xi; \Gamma \vdash !V : T\tau}$

Let $v_1 = [\![\Delta; \Xi; \Gamma \vdash V_1 : \tau\ ref]\!]$ and $v_2 = [\![\Delta; \Xi; \Gamma \vdash V_2 : \tau\ ref]\!]$.
Assume $(v_1, v_2, \tau\ ref, p) \in \nabla_V^{\Xi\Gamma}$. So $(v_1(\rho_1'), v_2(\rho_2') \parallel v_1(\rho_1), v_2(\rho_2), \tau[\overline{\sigma_j/\alpha_j}]\ ref, p') \in \nabla_V$
This implies that either $v_1(\rho_1') = v_2(\rho_2') = \bot$ or $\exists(l_1, l_2, \tau[\overline{\sigma_j/\alpha_j}]) \in Z^{p'}.\ v_1(\rho_1') \sqsubseteq v_1(\rho_1) = i(in_{\mathbb{L}} l_1) \wedge v_2(\rho_2') \sqsubseteq v_2(\rho_2) = i(in_{\mathbb{L}} l_2)$.

Let $m_1 = [\![\Delta; \Xi; \Gamma \vdash !V_1]\!]$ and $m_2 = [\![\Delta; \Xi; \Gamma \vdash !V_2]\!]$.
We need to show $(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T\tau[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$.
If $v_1(\rho_1') = v_2(\rho_2') = \bot$ then $m_1(\rho_1') = m_2(\rho_2') = \bot$ and
$(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2'), v_2(\rho_2), T\tau[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$. Else,
$i^{-1}(m_1(\rho_1')) \sqsubseteq \lambda k.\lambda S.(i^{-1}k)S(Sl_1)$, $i^{-1}(m_1(\rho_1)) = \lambda k.\lambda S.(i^{-1}k)S(Sl_1)$,
$i^{-1}(m_2(\rho_2')) \sqsubseteq \lambda k.\lambda S.(i^{-1}k)S(Sl_2)$, $i^{-1}(m_2(\rho_2)) = \lambda k.\lambda S.(i^{-1}k)S(Sl_2)$.

Let $p'' \blacktriangleright p', (pk'') \in p''^{\mathbf{K}}, (pks'') \in (pk'')^{\mathbf{S}}, (k_1', k_2' \parallel k_1, k_2, (x : \tau[\overline{\sigma_j/\alpha_j}])^{\top}, (pk'')) \in \nabla_K$, $(S_1', S_2' \parallel S_1, S_2, (pks'')) \in \nabla_S$.
If $S_1' = S_2' = \bot$ we get required termination properties trivially. Else since $(l_1, l_2, \tau[\overline{\sigma_j/\alpha_j}]) \in Z^{p'}$ also $(l_1, l_2, \tau[\overline{\sigma_j/\alpha_j}]) \in Z^{p''}$, then $(S_1', S_2' \parallel S_1, S_2, (pks'')) \in \nabla_S \Rightarrow (S_1'l_1, S_2'l_2 \parallel S_1 l_1, S_2 l_2, \tau, p'') \in \nabla_V$. So continuations states and values are correspondingly related. It follows that
$(pks'') \in \mathfrak{p}^s \setminus (\mathfrak{s}^{sP\bot} \cup \mathfrak{s}^{s\bot P}) \Longrightarrow$
$\quad (m_1(\rho_1'))k_1'S_1' \sqsubseteq (k_1')S_1'(S_1'l_1) = \top \Rightarrow (k_2)S_2 v_2(S_2 l_2) = (m_2(\rho_2))k_2 S_2 = \top \wedge$
$\quad (m_2(\rho_2'))k_2'S_2' \sqsubseteq (k_2')S_2'(S_2'l_2) = \top \Rightarrow (k_1)S_1(S_1 l_1) = (m_1(\rho_1))k_1 S_1 = \top$
$(pks'') \in \mathfrak{s}^{sP\bot} \Longrightarrow (k_1')S_1'(S_1'l_1) = (m_1(\rho_1'))k_1'S_1' = \bot$.
$(pks'') \in \mathfrak{s}^{s\bot P} \Longrightarrow (k_2')S_2'(S_2'l_2) = (m_2(\rho_2'))k_2'S_2' = \bot$

Hence $(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T\tau[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$.
We conclude $(m_1, m_2, T\tau, p) \in \nabla_M^{\Xi\Gamma}$.

alloc: $\dfrac{\Delta; \Xi; \Gamma \vdash V : \tau}{\Delta; \Xi; \Gamma \vdash ref\ V : T(\tau\ ref)}$

Let $v_1 = [\![\Delta; \Xi; \Gamma \vdash V_1 : \tau]\!]$ and $v_2 = [\![\Delta; \Xi; \Gamma \vdash V_2 : \tau]\!]$. Assume $(v_1, v_2, \tau, p) \in \nabla_V^{\Xi\Gamma}$.
Let $m_1 = [\![\Delta; \Xi; \Gamma \vdash ref\ V_1 : T(\tau\ ref)]\!]$ and $m_2 = [\![\Delta; \Xi; \Gamma \vdash ref\ V_2 : T(\tau\ ref)]\!]$.
We need to show $(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T(\tau[\overline{\sigma_j/\alpha_j}]\ ref), p') \in \nabla_M$.

If $(m_1(\rho_1')) = \bot \wedge (m_2(\rho_2')) = \bot$ then $(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T(\tau[\overline{\sigma_j/\alpha_j}]\ ref), p') \in \nabla_M$. Else $\rho_1' = \bot \Rightarrow (m_1(\rho_1')) = \bot$, $\rho_1 = \bot \Rightarrow (m_1(\rho_1)) = \bot$, $\rho_2' = \bot \Rightarrow (m_1(\rho_2')) = \bot$, $\rho_2 = \bot \Rightarrow (m_1(\rho_2)) = \bot$.

$\rho_1' \neq \bot \Rightarrow i^{-1}(m_1(\rho_1')) = \lambda k.\lambda S.(i^{-1}k)(S[l_1' \mapsto (v_1(\rho_1'))])(i \circ in_{\mathbb{L}} l_1')$
$\quad$ for some/any $l_1' \notin supp(\lambda l'.(i^{-1}k)(S[l' \mapsto (v_1(\rho_1'))](i \circ in_{\mathbb{L}} l'))$.
$\rho_1 \neq \bot \Rightarrow i^{-1}(m_1(\rho_1)) = i^{-1}(m_1(\rho_1)) = \lambda k.\lambda S.(i^{-1}k)(S[l_1 \mapsto (v_1(\rho_1))])(i \circ in_{\mathbb{L}} l_1)$
$\quad$ for some/any $l_1 \notin supp(\lambda l'.(i^{-1}k)(S[l' \mapsto (v_1(\rho_1))](i \circ in_{\mathbb{L}} l'))$.
$\rho_2' \neq \bot \Rightarrow i^{-1}(m_2(\rho_2')) = i^{-1}(m_2(\rho_2')) = \lambda k.\lambda S.(i^{-1}k)(S[l_2' \mapsto (v_2(\rho_2'))])(i \circ in_{\mathbb{L}} l_2')$
$\quad$ for some/any $l_2' \notin supp(\lambda l'.(i^{-1}k)(S[l' \mapsto (v_2(\rho_2'))](i \circ in_{\mathbb{L}} l'))$.
$\rho_2 \neq \bot \Rightarrow i^{-1}(m_2(\rho_2)) = i^{-1}(m_2(\rho_2)) = \lambda k.\lambda S.(i^{-1}k)(S[l_2 \mapsto (v_2(\rho_2))])(i \circ in_{\mathbb{L}} l_2)$
$\quad$ for some/any $l_2 \notin supp(\lambda l'.(i^{-1}k)(S[l' \mapsto (v_2(\rho_2))](i \circ in_{\mathbb{L}} l'))$.

Let arbitrarily $p'' \blacktriangleright p', (pk'') \in p''^{\mathbf{K}}, (pks'') \in pk''^{\mathbf{S}}, (k_1', k_2' \parallel k_1, k_2, (x : \tau[\overline{\sigma_j/\alpha_j}]\ ref)^{\top}, (pk'')) \in \nabla_K, (S_1', S_2' \parallel S_1, S_2, (pks'')) \in \nabla_S$.

If $\rho_1' = \rho_2' = \bot$ or $S_1' = S_2' = \bot$ then $(m_1(\rho_1'))k_1'S_1' = (m_1(\rho_2'))k_2'S_2' = \bot$. Else

Let $l_0 \notin (\mathring{A}_1^{p''}(S_1) \cup \mathring{A}_2^{p''}(S_2) \cup$
$supp(\lambda l'.(i^{-1}k)(S[l' \mapsto (v_1(\rho_1'))(i \circ in_{\mathbb{L}} l')) \ \cup \ supp(\lambda l'.(i^{-1}k)(S[l' \mapsto (v_1(\rho_1))(i \circ in_{\mathbb{L}} l')) \cup$
$supp(\lambda l'.(i^{-1}k)(S[l' \mapsto (v_2(\rho_2'))(i \circ in_{\mathbb{L}} l'))) \ \cup \ supp(\lambda l'.(i^{-1}k)(S[l' \mapsto (v_2(\rho_2))(i \circ in_{\mathbb{L}} l'))).$
Then
$i^{-1}(m_1(\rho_1'))k_1'S_1' \sqsubseteq (i^{-1}k_1')(S_1'[l_0 \mapsto (v_1(\rho_1'))])(i \circ in_{\mathbb{L}} l_0),$
$i^{-1}(m_1(\rho_1))k_1S_1 = (i^{-1}k_1)(S_1[l_0 \mapsto (v_1(\rho_1))])(i \circ in_{\mathbb{L}} l_0),$
$i^{-1}(m_2(\rho_2'))k_2'S_2' \sqsubseteq (i^{-1}k_2')(S_2'[l_0 \mapsto (v_2(\rho_2'))])(i \circ in_{\mathbb{L}} l_0),$
$i^{-1}(m_2(\rho_2))k_2S_2 = (i^{-1}k_2)(S_2[l_0 \mapsto (v_2(\rho_2))])(i \circ in_{\mathbb{L}} l_0).$
We need to show that
$(pks'') \in \mathfrak{p}^s \setminus (\mathfrak{s}^{sP\perp} \cup \mathfrak{s}^{s\perp P}) \implies$
$\quad (m_1(\rho_1'))k_1'S_1' = \top \Rightarrow (m_2(\rho_2))k_2S_2 = \top \ \wedge \ (m_2(\rho_2'))k_2'S_2' = \top \Rightarrow (m_1(\rho_1))k_1S_1 = \top$
$(pks'') \in \mathfrak{s}^{sP\perp} \implies (m_1(\rho_1'))k_1'S_1' = \perp$
$(pks'') \in \mathfrak{s}^{s\perp P} \implies (m_2(\rho_2'))k_2'S_2' = \perp$

We define a parameter (pks^3) that extends (pks'') by adding $(l_0, l_0, \tau[\overline{\sigma_j/\alpha_j}])$ to the visible locations. Let $r = ((T, \emptyset_{LL}), A_\emptyset, A_\emptyset, \{(l_0, l_0, \tau[\overline{\sigma_j/\alpha_j}])\})$.
Let $(pks^3) = (pks'') \uplus \{(r|r|(T, \emptyset_{LL}))\}$ so $(pk^3) = (pks^3)^k = (pk)'' \uplus \{(r|r)\}$, $p^3 = (pks^3)^{vm} = p'' \uplus \{r\}$ and $p^3 \rhd p''$, $(pk^3) \rhd (pk'')$.
Then $Z^{pks^3} = Z^{pk^3} = Z^{pks''} \uplus \{(l_0, l_0, \tau[\overline{\sigma_j/\alpha_j}])\}$, $Z^{p^3} = Z^{p''} \uplus \{(l_0, l_0, \tau[\overline{\sigma_j/\alpha_j}])\}$

It holds that
$(i \circ in_{\mathbb{L}} l_0, i \circ in_{\mathbb{L}} l_0, i \circ in_{\mathbb{L}} l_0, i \circ in_{\mathbb{L}} l_0, \tau[\overline{\sigma_j/\alpha_j}] \ ref, p^3) \in \nabla_V$ and
$(pks^3) \in \mathfrak{p}^s \setminus (\mathfrak{s}^{sP\perp} \cup \mathfrak{s}^{s\perp P}) \Leftrightarrow (pks'') \in \mathfrak{p}^s \setminus (\mathfrak{s}^{sP\perp} \cup \mathfrak{s}^{s\perp P})$
$(pks^3) \in \mathfrak{s}^{sP\perp} \Leftrightarrow (pks'') \in \mathfrak{s}^{sP\perp}$,
$(pks^3) \in \mathfrak{s}^{s\perp P} \Leftrightarrow (pks'') \in \mathfrak{s}^{s\perp P}$.
To prove the required termination approximation we want to show
$(S_1'[l_0 \mapsto v_1(\rho_1')], S_2'[l_0 \mapsto v_2(\rho_2')] \parallel S_1[l_0 \mapsto v_1(\rho_1)], S_2[l_0 \mapsto v_2(\rho_2)], (pks^3)) \in \nabla_S$.

By assumption $(S_1', S_2' \parallel S_1, S_2, (pks'')) \in \nabla_S$. The states have only been changed outside the areas for the most consuming accessibility maps $\mathring{A}_1^{p''}, \mathring{A}_2^{p''}$ involved in p''. Hence the updated states $(S_1[l_0 \mapsto (v_1(\rho_1))], \ S_2[l_0 \mapsto (v_2(\rho_2))])$ still belong to the same simple state predicates and relations involved in p''. The associated location lists hold values related in p'' by weakening these are also related in $p^3 \rhd p''$. $Z^{pks^3} = Z^{pks''} \uplus \{(l_0, l_0, \tau)\}$. Since the original states were related and by weakening the locations in $Z^{pks''}$ hold values related in p^3. By assumption and weakening also $(v_1(\rho_1'), v_2(\rho_2') \parallel v_1(\rho_1), v_2(\rho_2), \tau[\overline{\sigma_j/\alpha_j}], p^3) \in \nabla_V$.

In more detail we have that
$(pks^3) = \{(r_1|\hat{q}_1|Q_1), \ldots, (r_n|\hat{q}_n|Q_n), (r_{n+1}|\hat{q}_{n+1}|Q_{n+1})\}$ where $r_{n+1} = r$ with trivial instantiations, and so relatedness of the updated states in pks^3 requires

1. $S_1'[l_0 \mapsto v_1(\rho_1')] \sqsubseteq S_1[l_0 \mapsto v_1(\rho_1)] \neq \perp \ \wedge \ S_2'[l_0 \mapsto v_2(\rho_2')] \sqsubseteq S_2[l_0 \mapsto v_2(\rho_2)] \neq \perp$
2. $\mathcal{A}_1^{pks^3}(S_1[l_0 \mapsto v_1(\rho_1)]) \cap \pi_1(Z^{pks^3}) = \emptyset \ \wedge \ \mathcal{A}_2^{pks^3}(S_2[l_0 \mapsto v_2(\rho_2)]) \cap \pi_2(Z^{pks^3}) = \emptyset$
3. $\forall i \neq j \in 1 \ldots$ n+1. $\mathring{A}_1^{ri}(S_1[l_0 \mapsto v_1(\rho_1)]) \cap \mathring{A}_1^{rj}(S_1[l_0 \mapsto v_1(\rho_1)]) = \emptyset \ \wedge \ \mathring{A}_2^{ri}(S_2[l_0 \mapsto v_2(\rho_2)]) \cap \mathring{A}_2^{rj}(S_2[l_0 \mapsto v_2(\rho_2)]) = \emptyset$
4. $\forall (l_1, l_2, \sigma) \in Z^{pks^3}.(S_1'[l_0 \mapsto v_1(\rho_1')](l_1), \ S_2'[l_0 \mapsto v_2(\rho_2')](l_2) \parallel S_1[l_0 \mapsto v_1(\rho_1)](l_1), \ S_2[l_0 \mapsto v_2(\rho_2)](l_2), \ \sigma, \ p^3) \in \nabla_V$
5. $\forall i \in 1..$n+1. if $Q_{ij_i} = (P_i, LL_i)$ then $(S_1[l_0 \mapsto v_1(\rho_1)], S_2[l_0 \mapsto v_2(\rho_2)]) \in P_i \ \wedge$
$\forall (l_1, l_2, \sigma) \in LL_i.\ (S_1'[l_0 \mapsto v_1(\rho_1')](l_1), \ S_2'[l_0 \mapsto v_2(\rho_2')](l_2) \parallel S_1[l_0 \mapsto v_1(\rho_1)](l_1), \ S_2[l_0 \mapsto v_2(\rho_2)](l_2), \ \sigma, \ p^3) \in \nabla_V$

1. Follows from: assumptions $S_1' \sqsubseteq S_1 \neq \bot \ \wedge \ S_2' \sqsubseteq S_2 \neq \bot$ and $\rho_1' \sqsubseteq \rho_1 \ \wedge \ \rho_2' \sqsubseteq \rho_2$

2. Follows from: $r = ((T, \emptyset_{LL}), A_\emptyset, A_\emptyset, \{(l_0, l_0, \tau)\})$, so
$\pi_1(Z^{pks^3}) = \pi_1(Z^{pks''}) \uplus \{l_0\}$, $\mathcal{A}_1^{pks^3}(S_1[l_0 \mapsto v_1(\rho_1)]) = \mathcal{A}_1^{pks''}(S_1),\ l_0 \notin \mathcal{A}_1^{pks''}(S_1)$ and
$\pi_2(Z^{pks^3}) = \pi_2(Z^{pks''}) \uplus \{l_0\}$, $\mathcal{A}_2^{pks^3}(S_2[l_0 \mapsto v_2(\rho_2)]) = \mathcal{A}_2^{pks''}(S_2),\ l_0 \notin \mathcal{A}_2^{pks''}(S_2)$

3. Follows from: $\mathring{A}_1^{r_{n+1}}(S_1[l_0 \mapsto v_1(\rho_1)]) = \mathring{A}_2^{r_{n+1}}(S_2[l_0 \mapsto v_2(\rho_2)]) = \{l_0\}$ and
$\forall i \neq j \in 1 \ldots n.\ \mathring{A}_1^{ri}(S_1) \cap \mathring{A}_1^{rj}(S_1) = \emptyset \ \wedge \ \mathring{A}_2^{ri}(S_2) \cap \mathring{A}_2^{rj}(S_2) = \emptyset$ and
$\forall i \in 1 \ldots n.\ \mathring{A}_1^{ri}(S_1[l_0 \mapsto v_1(\rho_1)]) = \mathring{A}_1^{ri}(S_1), l_0 \notin \mathring{A}_1^{ri}(S_1) \ \wedge \ \mathring{A}_2^{ri}(S_2[l_0 \mapsto v_2(\rho_2)]) = \mathring{A}_2^{ri}(S_2), l_0 \notin \mathring{A}_2^{ri}(S_2)$

4. Follows from: $Z^{pks^3} = Z^{pks''} \uplus \{(l_0, l_0, \tau[\overline{\sigma_j/\alpha_j}])\}, \forall (l_1, l_2, \sigma) \in Z^{pks''}.(S_1'(l_1),\ S_2'(l_2) \parallel S_1(l_1),\ S_2(l_2),\ \sigma,\ p'') \in \nabla_V$, weakening and the assumption $(v_1, v_2, \tau, p) \in \nabla_V^{\Xi \Gamma}$.

5. Follows from: Simple state relations and predicates only depend on the areas given by the accessibility maps and $Q_{n+1} = (T, \emptyset_{LL})$ and $\forall i \in 1..n.\ l_0 \notin A_1^{q_i}(S_1), l_0 \notin A_2^{q_i}(S_2)$ and if $Q_i = (P_i, LL_i)$ then $(S_1, S_2) \in P_i \wedge \forall (l_1, l_2, \sigma) \in LL_i.\ (S_1'(l_1), S_2'(l_2) \parallel S_1(l_1), S_2(l_2), \sigma, p^{vm}) \in \nabla_V$.

We have
$(S_1'[l_0 \mapsto (v_1(\rho_1'))]), S_1[l_0 \mapsto (v_1(\rho_1))]), S_2'[l_0 \mapsto (v_2(\rho_2'))]), S_2[l_0 \mapsto (v_2(\rho_2))]), pks^3) \in \nabla_S$.

So we have (pk'') related continuations applied to (pks^3) related states and p^3 related values. We saw that this is a $\triangleright$ extension and $(pks^3) \in \mathfrak{p}^{sPLL} \Leftrightarrow (pks'') \in \mathfrak{p}^{sPLL}$, $(pks^3) \in \mathfrak{s}^{sP\bot} \Leftrightarrow (pks'') \in \mathfrak{s}^{sP\bot}$, $(pks^3) \in \mathfrak{s}^{s\bot P} \Leftrightarrow (pks'') \in \mathfrak{s}^{s\bot P}$. So we get the required termination properties.
$i^{-1}(m_1(\rho_1'))k_1'S_1' \sqsubseteq (i^{-1}k_1')(S_1'[l_0 \mapsto (v_1(\rho_1'))]])(i \circ in_{\mathbb{L}} l_0)$,
$i^{-1}(m_1(\rho_1))k_1S_1 = (i^{-1}k_1)(S_1[l_0 \mapsto (v_1(\rho_1))]])(i \circ in_{\mathbb{L}} l_0)$,
$i^{-1}(m_2(\rho_2'))k_2'S_2' \sqsubseteq (i^{-1}k_2')(S_2'[l_0 \mapsto (v_2(\rho_2'))]])(i \circ in_{\mathbb{L}} l_0)$,
$i^{-1}(m_2(\rho_2))k_2S_2 = (i^{-1}k_2)(S_2[l_0 \mapsto (v_2(\rho_2))]])(i \circ in_{\mathbb{L}} l_0)$.
Hence $(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T(\tau[\overline{\sigma_j/\alpha_j}]\ ref), p') \in \nabla_M$ and so
$(m_1, m_2, T(\tau\ ref), p) \in \nabla_M^{\Xi \Gamma}$.

let: $$\frac{\Delta; \Xi; \Gamma \vdash M_a : T\tau_a \qquad \Delta; \Xi; \Gamma, x : \tau_a \vdash M_b : T\tau_b}{\Delta; \Xi; \Gamma \vdash let\ x \Leftarrow M_a\ in\ M_b : T\tau_b}$$

Let $m_{1a} = [\![\Delta; \Xi; \Gamma \vdash M_{1a} : T\tau_a]\!]$, $m_{2a} = [\![\Delta; \Xi; \Gamma \vdash M_{2a} : T\tau_a]\!]$, $m_{1b} = [\![\Delta; \Xi; \Gamma, x : \tau_a \vdash M_{1b} : T\tau_b]\!]$, $m_{2b} = [\![\Delta; \Xi; \Gamma, x : \tau_a \vdash M_{2b} : T\tau_b]\!]$.
Assume $(m_{1a}, m_{2a}, T\tau_a, p) \in \nabla_M^{\Xi \Gamma}$, and assume $(m_{1b}, m_{2b}, T\tau_b, p) \in \nabla_M^{\Xi(\Gamma, x:\tau_a)}$,
then $(m_{1a}(\rho_1'), m_{2a}(\rho_2') \parallel m_{1a}(\rho_1), m_{2a}(\rho_2), T\tau_a[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$, and
for any $(v_{1x}', v_{2x}' \parallel v_{1x}, v_{2x}, \tau_a[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$ it holds that
$(m_{1b}(\rho_1' \otimes v_{1x}'), m_{2b}(\rho_2' \otimes v_{2x}') \parallel m_{1b}(\rho_1 \otimes v_{1x}), m_{2b}(\rho_2 \otimes v_{2x}), T\tau_b[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$.

Let $m_1 = [\![\Delta; \Xi; \Gamma \vdash let\ x \Leftarrow M_{1a}\ in\ M_{1b} : T\tau_b]\!]$ and
let $m_2 = [\![\Delta; \Xi; \Gamma \vdash let\ x \Leftarrow M_{2a}\ in\ M_{2b} : T\tau_b]\!]$.
We need to show $(m_1, m_2, T\tau_b, p) \in \nabla_M^{\Xi \Gamma}$ that is
$(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T\tau_b[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$.
If $\rho_1' = \rho_2' = \bot$ then $m_1(\rho_1') = m_2(\rho_2') = \bot$ and we are done. Else we have
$i^{-1}(m_1(\rho_1')) = \lambda k.\lambda S.\ (i^{-1}(m_{1a}(\rho_1')))(\lambda S_0.\lambda d_x.(\ i^{-1}(m_{1b}(\rho_1' \otimes d_x))kS_0)S$,
$i^{-1}(m_1(\rho_1)) = \lambda k.\lambda S.\ (i^{-1}(m_{1a}(\rho_1)))(\lambda S_0.\lambda d_x.(\ i^{-1}(m_{1b}(\rho_1 \otimes d_x))kS_0)S$,
$i^{-1}(m_2(\rho_2')) = \lambda k.\lambda S.\ (i^{-1}(m_{2a}(\rho_2')))(\lambda S_0.\lambda d_x.(\ i^{-1}(m_{2b}(\rho_2' \otimes d_x))kS_0)S$,
$i^{-1}(m_2(\rho_2)) = \lambda k.\lambda S.\ (i^{-1}(m_{2a}(\rho_2)))(\lambda S_0.\lambda d_x.(\ i^{-1}(m_{2b}(\rho_2 \otimes d_x))kS_0)S$.

Let $p'' \blacktriangleright p'$, $(pk'') \in p''^{\mathbf{k}}$, $(pks'') \in pk''^{\mathbf{s}}$, $(k_1', k_2' \parallel k_1, k_2, (x : \tau_b[\overline{\sigma_j/\alpha_j}])^\top, (pk'')) \in \nabla_K$, $(S_1', S_2' \parallel S_1, S_2, (pks'')) \in \nabla_S$.
$(i^{-1}(m_1(\rho_1')))k_1'S_1' = (i^{-1}(m_{1a}(\rho_1')))(\lambda S_0.\lambda d_x.(\ i^{-1}(m_{1b}(\rho_1' \otimes d_x))k_1'S_0)S_1'$,
$(i^{-1}(m_1(\rho_1)))k_1S_1 = (i^{-1}(m_{1a}(\rho_1)))(\lambda S_0.\lambda d_x.(\ i^{-1}(m_{1b}(\rho_1 \otimes d_x))k_1S_0)S_1$,
$(i^{-1}(m_2(\rho_2')))k_2'S_2' = (i^{-1}(m_{2a}(\rho_2')))(\lambda S_0.\lambda d_x.(\ i^{-1}(m_{2b}(\rho_2' \otimes d_x))k_2'S_0)S_2'$,
$(i^{-1}(m_2(\rho_2)))k_2S_2 = (i^{-1}(m_{2a}(\rho_2)))(\lambda S_0.\lambda d_x.(\ i^{-1}(m_{2b}(\rho_2 \otimes d_x))k_2S_0)S_2$.

By assupmtion $(m_{1a}(\rho_1'), m_{2a}(\rho_2') \parallel m_{1a}(\rho_1), m_{2a}(\rho_2), T\tau_a[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$ and the states are not changed $(S_1', S_2' \parallel S_1, S_2, (pks'')) \in \nabla_S$.
So the state parameter is the same and so to prove the required termination-approximations we want to show that the continuations are related under (pk'')
$((\lambda S_0.\lambda d_x.\ (i^{-1}(m_{1b}(\rho_1' \otimes d_x))k_1'S_0), (\lambda S_0.\lambda d_x.\ (i^{-1}(m_{2b}(\rho_2' \otimes d_x))k_2'S_0) \parallel$
$(\lambda S_0.\lambda d_x.\ (i^{-1}(m_{1b}(\rho_1 \otimes d_x))k_1S_0), (\lambda S_0.\lambda d_x.\ (i^{-1}(m_{2b}(\rho_2 \otimes d_x))k_2S_0), (x : \tau_a[\overline{\sigma_j/\alpha_j}])^\top, (pk'')) \in F(\nabla, \nabla)_K$.

Let $(pk^3) \rhd (pk'')$, $(pks^3) \in (pk^3)^{\mathbf{S}}$, $p^3 = (pk^3)^{vm}$ so $p^3 \rhd p''$.
Let $(S_{10}', S_{20}' \parallel S_{10}, S_{20}, (pks^3)) \in \nabla_S$, $(d_1', d_2' \parallel d_1, d_2, \tau_a[\overline{\sigma_j/\alpha_j}], p^3) \in \nabla_V$.
$(\lambda S_0.\lambda d_x.(i^{-1}(m_{1b}(\rho_1' \otimes d_x))k_1'S_0))S_{10}'d_1' = (i^{-1}(m_{1b}(\rho_1' \otimes d_1')))k_1'S_{10}'$,
$(\lambda S_0.\lambda d_x.\ (i^{-1}(m_{1b}(\rho_1 \otimes d_x))k_1S_0)S_{10}d_1 = (i^{-1}(m_{1b}(\rho_1 \otimes d_1)))k_1S_{10}$,
$(\lambda S_0.\lambda d_x.\ (i^{-1}(m_{2b}(\rho_2' \otimes d_x))k_2'S_0)S_{20}'d_2' = (i^{-1}(m_{2b}(\rho_2' \otimes d_2')))k_2'S_{20}'$,
$(\lambda S_0.\lambda d_0.\ (i^{-1}(m_{2b}(\rho_2 \otimes d_x))k_2S_0)S_{20}d_2 = (i^{-1}(m_{2b}(\rho_2 \otimes d_2)))k_2S_{20}$.

We have $p^3 \rhd p'' \blacktriangleright p' \rhd p$. So $p^3 \blacktriangleright p$. $(d_1', d_2' \parallel d_1, d_2, \tau_a[\overline{\sigma_j/\alpha_j}], p^3) \in \nabla_V$,
In the $\rho's$ we have $\forall i.(v_{1i}', v_{2i}' \parallel v_{1i}, v_{2i}, \tau_i[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$, it follows by weakening that $\forall i.(v_{1i}', v_{2i}' \parallel v_{1i}, v_{2i}, \tau_i[\overline{\sigma_j/\alpha_j}], p^3) \in \nabla_V$. Also $(m_{1b}, m_{2b}, T\tau_b[\overline{\sigma_j/\alpha_j}], p) \in \nabla_M^{\Xi(\Gamma, x:\tau_a)}$.

Then $((m_{1b}(\rho_1' \otimes d_1')), (m_{2b}(\rho_2' \otimes d_2')) \parallel (m_{1b}(\rho_1 \otimes d_1)), (m_{2b}(\rho_2 \otimes d_2)), T\tau_b[\overline{\sigma_j/\alpha_j}], p^3) \in \nabla_M$.
By assumption $(k_1', k_2' \parallel k_1, k_2, (x : \tau_b[\overline{\sigma_j/\alpha_j}])^\top, (pk'')) \in \nabla_K$. Since $(pk^3) \rhd (pk'')$ then by parameter weakening $(k_1', k_2' \parallel k_1, k_2, (x : \tau_b[\overline{\sigma_j/\alpha_j}])^\top, (pk^3)) \in \nabla_K$.
Since also $(S_{10}', S_{20}' \parallel S_{10}, S_{20}, (pks^3)) \in \nabla_S$ we can conclude that we get the required termination properties. So the continuations are related and hence
$(m_1, m_2, T\tau_b, p) \in \nabla_M^{\Xi\Gamma}$.

□

For store type Δ let id_Δ be the match of finite store types $\{(l, l, \tau) | l : \tau \in \Delta\}$.

Theorem 7. *Fundamental Theorem*
For all ordinary parameters p with $Z^p = id_\Delta$.

If $\Delta; \Xi; \Gamma \vdash V : \tau$ *then* $([\![\Delta; \Xi; \Gamma \vdash V : \tau]\!], [\![\Delta; \Xi; \Gamma \vdash V : \tau]\!], \tau, p) \in \nabla_V^{\Xi\Gamma}$
If $\Delta; \Xi; \Gamma \vdash M : T\tau$ *then* $([\![\Delta; \Xi; \Gamma \vdash M : T\tau]\!], [\![\Delta; \Xi; \Gamma \vdash M : T\tau]\!], T\tau, p) \in \nabla_M^{\Xi\Gamma}$

Proof by induction over typing derivations using proposition 5.

The next lemma has to do with four-tuples which are really two pairs.

Lemma 40.
Let $p \in \mathfrak{o}^{vm}$

$(m_1, m_2, T\tau, p) \in \nabla_M^{\emptyset\emptyset} \Rightarrow$

$\forall p' \rhd p. \forall (pk') \in p'^{\mathbf{K}}. \forall (pks') \in (pk')^{\mathbf{S}}.$
$\forall (k_1, k_2 \parallel k_1, k_2, (x : \tau)^\top, (pk')) \in \nabla_K. \forall (S_1, S_2 \parallel S_1, S_2, (pks')) \in \nabla_S.$
$(i^{-1}m_1)k_1S_1 = \top \iff (i^{-1}m_2)k_2S_2 = \top.$

Proof

Recall that for any ordinary parameter *all* parameter parts have the form (P, LL), and $(p' \rhd p \wedge p \in \mathfrak{o}^{vm}) \Longrightarrow (p' \in \mathfrak{o}^{vm} \ \wedge \ p' \blacktriangleright p)$.
$(m_1, m_2, T\tau, p) \in \nabla_M^{\emptyset\emptyset} \ \Rightarrow$
$(m_1, m_2 \parallel m_1, m_2, T\tau, p) \in \nabla_M \Rightarrow$
$(i^{-1}(m_1), i^{-1}(m_2) \parallel i^{-1}(m_1), i^{-1}(m_2), T\tau, p) \in F(\nabla, \nabla)_M$.

It follows from the definition of $F(\nabla, \nabla)_M$ that $\forall p' \rhd p. \forall (pk') \in p'^{\mathbf{K}}. \forall (pks') \in (pk')^{\mathbf{S}}$.
$\forall (k_1, k_2 \parallel k_1, k_2, (x : \tau)^\top, (pk')) \in \nabla_K. \forall (S_1, S_2 \parallel S_1, S_2, (pks')) \in \nabla_S$.
$(i^{-1}(m_1))k_1 S_1 = \top \Rightarrow (i^{-1}(m_2))k_2 S_2 = \top$ and
$(i^{-1}(m_1))k_1 S_1 = \top \Leftarrow (i^{-1}(m_2))k_2 S_2 = \top$

□

Recall T_Δ is the vm-parameter $\{(T, A_\emptyset, A_\emptyset, id_\Delta)\}$ and T_Δ^K, T_Δ^S the trivial instantiations.

Lemma 41.

1. $\forall \_ \vdash \tau : type. \ \forall (pk) \in \mathfrak{o}^k$ *with* $Z^{pk} = id_\Delta$. $(\llbracket \Delta \vdash val\ x : (x : \tau)^\top \rrbracket, \llbracket \Delta \vdash val\ x : (x : \tau)^\top \rrbracket \parallel \llbracket \Delta \vdash val\ x : (x : \tau)^\top \rrbracket, \llbracket \Delta \vdash val\ x : (x : \tau)^\top \rrbracket, (x : \tau)^\top, (pk)) \in \nabla_K$

2. $\forall \Sigma : \Delta. \ \forall S \in \llbracket \Sigma : \Delta \rrbracket. \ (S, S \parallel S, S, T_\Delta^S) \in \nabla_S$

Proof (We have omitted the isomorphisms i, i^{-1}).

1. Assume $\_ \vdash \tau : type$.
$\llbracket \Delta \vdash val\ x : (x : \tau)^\top \rrbracket =$
$\lambda S. \lambda d. \llbracket \Delta; ; x : \tau \vdash val\ x : T\tau \rrbracket [x \mapsto d]((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S =$
$\lambda S. \lambda d. (\lambda k_0. \lambda S_0. k_0 S_0 \llbracket \Delta; ; x : \tau \vdash x : \tau \rrbracket [x \mapsto d])((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S =$
$\lambda S. \lambda d. (\lambda k_0. \lambda S_0. k_0 S_0 d)((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S =$
$\lambda S. \lambda d. ((\lambda S'.(\lambda d'.\top)_\bot)_\bot)Sd.$

   Let $(pk') \rhd (pk), (pks') \in (pk')^{\mathbf{S}}, (S_1', S_2' \parallel S_1, S_2, (pks')) \in \nabla_S, (v_1', v_2' \parallel v_1, v_2, \tau, (pk')^{vm}) \in \nabla_V$.
By definition of ∇ it holds that either $(v_1' = v_2' = \bot)$ or $(v_1 \neq \bot \wedge v_2 \neq \bot)$ and either $(S_1' = S_2' = \bot)$ or $(S_1 \neq \bot \wedge S_2 \neq \bot)$
$\llbracket \Delta \vdash val\ x : (x : \tau)^\top \rrbracket S_1' v_1' = ((\lambda S'.(\lambda d'.\top)_\bot)_\bot) S_1' v_1'$
$\llbracket \Delta \vdash val\ x : (x : \tau)^\top \rrbracket S_1 v_1 = ((\lambda S'.(\lambda d'.\top)_\bot)_\bot) S_1 v_1$
$\llbracket \Delta \vdash val\ x : (x : \tau)^\top \rrbracket S_2' v_2' = ((\lambda S'.(\lambda d'.\top)_\bot)_\bot) S_2' v_2'$
$\llbracket \Delta \vdash val\ x : (x : \tau)^\top \rrbracket S_2 v_2 = ((\lambda S'.(\lambda d'.\top)_\bot)_\bot) S_2 v_2$
We have $(pk') \rhd (pk) \in \mathfrak{o}^k$implies $(pk') \in \mathfrak{o}^k$. So we need to show
$((\lambda S'.(\lambda d'.\top)_\bot)_\bot) S_1' v_1' = \top \ \Rightarrow \ ((\lambda S'.(\lambda d'.\top)_\bot)_\bot) S_2 v_2 = \top$ and
$((\lambda S'.(\lambda d'.\top)_\bot)_\bot) S_2' v_2' = \top \ \Rightarrow \ ((\lambda S'.(\lambda d'.\top)_\bot)_\bot) S_1 v_1 = \top$.
Assume $((\lambda S'.(\lambda d'.\top)_\bot)_\bot) S_1' v_1' = \top$. Then $v_1' \neq \bot$ and $S_1' \neq \bot$. $v_1' \neq \bot \Rightarrow (v_1 \neq \bot \ \wedge \ v_2 \neq \bot)$, and $S_1' \neq \bot \Rightarrow (S_1 \neq \bot \ \wedge \ S_2 \neq \bot)$. This implies that $((\lambda S'.(\lambda d'.\top)_\bot)_\bot) S_2 v_2 = \top$.
The other direction is proved similarly.

2. We show $S \in \llbracket \Sigma : \Delta \rrbracket$ implies $(S, S \parallel S, S, T_\Delta^S) \in \nabla_S$.
It holds that $S \sqsubseteq S \neq \bot$ and $\forall l \in dom(\Delta)$. $Sl = \llbracket \Delta; ; \vdash \Sigma(l) : \Delta(l) \rrbracket$.
Then, by the previous theorem 7, $\forall l \in dom(\Delta).(Sl, Sl, Sl, Sl, \Delta l, T_\Delta) \in \nabla_V$.
It holds that $dom(\Delta) \cap A_\emptyset(S) = \emptyset$. Also $(S, S) \in T$.

□

Theorem 8. *Contextual equivalence*
For all $\Delta, \Xi, \gamma, \tau$, for all value- or computation terms G_1, G_2,
for all contexts $C[\_\,] : (\Delta; \Xi; \Gamma \vdash \gamma) \Rightarrow (\Delta; ; \vdash T\tau)$ (let $j \in V, M$)

If $\Delta;\Xi;\Gamma \vdash G_1 : \gamma$ *and* $\Delta;\Xi;\Gamma \vdash G_2 : \gamma$ *and*
$([\![\Delta;\Xi;\Gamma \vdash G_1 : \gamma]\!], [\![\Delta;\Xi;\Gamma \vdash G_2 : \gamma]\!], \gamma, T_\Delta) \in \nabla_j^{\Xi\Gamma}$ *then*

$\forall \Sigma : \Delta.\ \ (\Sigma, let\ x \Leftarrow C[G_1]\ in\ val\ x \downarrow \Longleftrightarrow \Sigma, let\ x \Leftarrow C[G_2]\ in\ val\ x \downarrow)$

Proof

By induction over the structure of $C[\_\ ]$ and using that typing rules preserve relatedness in $\nabla^{\Xi\Gamma}$and fundamental theorem 7 it holds that
$([\![\Delta;\Xi;\Gamma \vdash G_1 : \gamma]\!], [\![\Delta;\Xi;\Gamma \vdash G_2 : \gamma]\!], \gamma, T_\Delta) \in \nabla_j^{\Xi\Gamma} \Longrightarrow$
$([\![\Delta;;\vdash C[G_1] : T\tau]\!], [\![\Delta;;\vdash C[G_2] : T\tau]\!], T\tau, T_\Delta) \in \nabla_M^{\emptyset\emptyset}$.
This by definition of relatedness in $\nabla_M^{\emptyset\emptyset}$ implies
$([\![\Delta;;\vdash C[G_1] : T\tau]\!], [\![\Delta;;\vdash C[G_2] : T\tau]\!] \parallel [\![\Delta;;\vdash C[G_1] : T\tau]\!], [\![\Delta;;\vdash C[G_2] : T\tau]\!], T\tau, T_\Delta) \in \nabla_M$.

By the previous lemma 41
$([\![\Delta \vdash val\ x : (x : \tau)^\top]\!], [\![\Delta \vdash val\ x : (x : \tau)^\top]\!] \parallel [\![\Delta \vdash val\ x : (x : \tau)^\top]\!], [\![\Delta \vdash val\ x : (x : \tau)^\top]\!],$
$(x : \tau)^\top, T_\Delta) \in \nabla_K$
and
$\forall S \in [\![\Sigma : \Delta]\!].\ (S, S \parallel S, S, T_\Delta) \in \nabla_S$.

By the definition of ∇ and since T_Δ is ordinary then it holds that
$\forall S \in [\![\Sigma : \Delta]\!]$
$(i^{-1}[\![\Delta;\vdash C[M_1]]\!]\{\})[\![\Delta;\vdash val\ x : (x)^\top]\!]S = \top \Longleftrightarrow$
$(i^{-1}[\![\Delta;\vdash C[M_2]]\!]\{\})[\![\Delta;\vdash val\ x : (x)^\top]\!]S = \top$.

By soundness and adequacy of the denotational semantics this implies
$\Sigma, let\ x \Leftarrow C[M_1]\ in\ val\ x\ \downarrow \Longleftrightarrow \Sigma, let\ x \Leftarrow C[M_2]\ in\ val\ x\ \downarrow$.

□

10 Example using full definition of parameters

10.1 Example. Divergence at different steps

The programs M and N below are open computations with one free variable g of function type $(unit \to Tunit) \to T\tau$. The programs were presented in the introduction. In the following we sometimes omit injections functions and for instance write $*$ for $in_1\lfloor * \rfloor$.

```
M:  g (rec f_M(a:unit)= f_M())

N:  let x ⇐ ref 0 in
      let y ⇐ ref 0 in
        let z ⇐ g (rec f_N(a:unit)= if (!x = 0) then (y := 1) else f_N()) in
            if (!y ≠ 0) then ⟨(rec f(a:unit)= f()) ()⟩ else (x := 1; y := 1)
```

We want to show that the denotations of M and N are related in the vm-parameter
$p = T_\emptyset \in \mathfrak{o}^{vm}$, that is

$([\![;;g \vdash M]\!], [\![;;g \vdash N]\!], T\tau, p) \in \nabla_M^{\emptyset g}$

Let $p_1 \blacktriangleright p$, $(\hat{g}'_1, \hat{g}'_2 \parallel \hat{g}_1, \hat{g}_2, ((unit \to Tunit) \to T\tau), p_1) \in \nabla_V$, so we want to prove that $(\llbracket ; ; g \vdash M \rrbracket(g \mapsto \hat{g}'_1), \llbracket ; ; g \vdash N \rrbracket(g \mapsto \hat{g}'_2) \parallel \llbracket ; ; g \vdash M \rrbracket(g \mapsto \hat{g}_1), \llbracket ; ; g \vdash N \rrbracket(g \mapsto \hat{g}_2), T\tau, p_1) \in \nabla_M$.
Let $p_2 \blacktriangleright p_1$, $(pk_2) \in p_2^{\mathbf{K}}, (pks_2) \in (pk_2)^{\mathbf{S}}$,
$(k'_1, k'_2 \parallel k_1, k_2, (z : unit)^{\top}, (pk_2)) \in \nabla_K$, $(S'_1, S'_2 \parallel S_1, S_2, (pks_2)) \in \nabla_S$.

If $\hat{g}'_1 = \hat{g}'_2 = \bot$ then $\llbracket M \rrbracket \hat{g}'_1 = \llbracket N \rrbracket \hat{g}'_2 = \bot$ and we are done, else exists $g'_1, g'_2, g_1, g_2 \in (\mathbb{V} \multimap \mathbb{M})$ such that $\hat{g}_1 = in_{\multimap}\lfloor g_1 \rfloor$, $\hat{g}_2 = in_{\multimap}\lfloor g_2 \rfloor$ and $(\hat{g}'_1 = \bot \wedge g'_1 = \bot) \vee \hat{g}'_1 = in_{\multimap}\lfloor g'_1 \rfloor$ and $(\hat{g}'_2 = \bot \wedge g'_2 = \bot) \vee \hat{g}'_2 = in_{\multimap}\lfloor g'_2 \rfloor$. Then
$\llbracket M \rrbracket \hat{g}_1 k_1 S_1 = \big(g_1(\llbracket ; ; \vdash recf_M(a : unit) = f_M() \rrbracket)\big) k_1 S_1$.
$\llbracket M \rrbracket \hat{g}'_1 k'_1 S'_1 = \big(g'_1(\llbracket ; ; \vdash recf_M(a : unit) = f_M() \rrbracket)\big) k'_1 S'_1$.
$\llbracket N \rrbracket \hat{g}_2 k_2 S_2 = \big(g_2(\llbracket ; ; x, y \vdash recf_N(a : unit) = if\ (!x = 0)\ then\ (y := 1)\ else\ f_N() \rrbracket(x \mapsto l_x, y \mapsto l_y))\big)\ \ (\lambda S^0 \lambda d^0. \llbracket x, y \vdash\ \ if\ (!y \neq 0)\ then f_N()\ else\ (x := 1; y := 1) \rrbracket(x \mapsto l_x, y \mapsto l_y) k_2 S^0)$
$S_2[l_x \mapsto 0, l_y \mapsto 0]$
$\llbracket N \rrbracket \hat{g}'_2 k'_2 S'_2 = \big(g'_2(\llbracket ; ; x, y \vdash recf_N(a : unit) = if\ (!x = 0)\ then\ (y := 1)\ else\ f_N() \rrbracket(x \mapsto l_x, y \mapsto l_y))\big)\ \ (\lambda S^0 \lambda d^0. \llbracket x, y \vdash\ \ if\ (!y \neq 0)\ then f_N()\ else\ (x := 1; y := 1) \rrbracket(x \mapsto l_x, y \mapsto l_y) k'_2 S^0)$
$S'_2[l_x \mapsto 0, l_y \mapsto 0]$

where l_x, l_y are fresh.

Define local parameter r_3 with constant accessibility maps and empty match of finite store types $\emptyset_Z$. $r_3 = (q^o \bar{\wedge} (q^s \prec q^b))$ where $q^o = ((\exists n. S_2 l_x = n \neq 0), \emptyset_Z, A_\emptyset, A_{\{l_x, l_y\}})$ and $(q^s \prec q^b) = \big(\big(((S_2 l_x = 0 \wedge S_2 l_y = 0), \emptyset_Z, A_\emptyset, A_{\{l_x, l_y\}}) \prec ((\bot, \langle S_2 l_x = 0 \wedge \exists m. S_2 l_y = m \neq 0 \rangle)\big), \emptyset_Z, A_\emptyset, A_{\{l_x, l_y\}}\big)$

Let $p_3 = p_2 \cup \{r_3\}$. Then $p_3 \blacktriangleright p_2$.

First we want to show that the g's are applied to functions related in p_3. We show that the functions give related computations when applied to related arguments. To show,
$(\llbracket ; ; \vdash recf_M(a : unit) = f_M() \rrbracket$,
$\llbracket ; ; x, y \vdash recf_N(a : unit) = if\ (!x = 0)\ then\ (y := 1)\ else\ f_N() \rrbracket(x \mapsto l_x, y \mapsto l_y) \parallel$
$\llbracket ; ; \vdash recf_M(a : unit) = f_M() \rrbracket$,
$\llbracket ; ; x, y \vdash recf_N(a : unit) = if\ (!x = 0)\ then\ (y := 1)\ else\ f_N() \rrbracket(x \mapsto l_x, y \mapsto l_y)$,
$unit \to Tunit, p_3) \in \nabla_V$.

Let $p_4 \blacktriangleright p_3$. $(v'_1, v'_2 \parallel v_1, v_2, unit, p_4) \in \nabla_V$. This requires that either $(v'_1 = v'_2 = \bot)$ or $(v_1 = v_2 = * \ \wedge\ v'_1, v'_2 \in \{\bot, *\})$.

By denotations of function values we get
$\llbracket ; ; \vdash recf_M(a : unit) = f_M() \rrbracket = in_{\multimap}\lfloor \lambda a. \lambda k. \lambda S. \bot_{\mathbb{O}} \rfloor = in_{\multimap}\lfloor \lambda a. \bot_{\mathbb{M}} \rfloor$ and
$\llbracket x, y \vdash recf_N(a : unit) = if\ (!x = 0)\ then\ (y := 1)\ else\ f_N() \rrbracket(x \mapsto l_x, y \mapsto l_y) =$
$in_{\multimap}\lfloor \lambda a. \lambda k. \lambda S.\ if\ (S(l_x) = 0)\ then\ (k(S[l_y \mapsto 1])*)\ else\ \bot \rfloor$.

We then want to show
$(\lambda k. \lambda S. \bot,\ \lambda k. \lambda S.\ if\ (S(l_x) = 0)\ then\ (k(S[l_y \mapsto 1])*)\ else\ \bot \parallel$
$\lambda k. \lambda S. \bot,\ \lambda k. \lambda S.\ if\ (S(l_x) = 0)\ then\ (k(S[l_y \mapsto 1])*)\ else\ \bot,\ Tunit,\ p_4) \in \nabla_M$.

Let $p_5 \blacktriangleright p_4$, then because r_3 has two $\bar{\wedge}$-clauses and one with a local extension there are these possibilities for the local parameter associated with the area for p_3: $r_3 \in p_5$, $(q^o \bar{\wedge} q^b) \in p_5$, $q^o \in p_5$, $(q^s \prec q^b) \in p_5$ or $q^b \in p_5$. We will consider the different possibilities and different possibilities of instantiations. We see here that there are a number of cases, but the example will also show that things come through by assumptions on continuations. A more tricky situation is later when we need to show that continuations which are partly build up by our original programs are related.

Let $(pk_5) \in p_5^{\mathbf{K}}, (pks_5) \in (pk_5)^{\mathbf{S}}$. Assume
$(K_1', K_2' \parallel K_1, K_2, (z : unit)^{\top}, (pk)_5) \in \nabla_K,\ (s_1', s_2' \parallel s_1, s_2, (pks_5)) \in \nabla_S$.

We will consider the different possible cases, some of them together.
If $K_1' = K_2' = \bot$ or $s_1' = s_2' = \bot$ then
$(\lambda k.\lambda S.\bot)K_1's_1' = (\lambda k.\lambda S.\ if\ (S(l_x) = 0)\ then\ (k(S[l_y \mapsto 1])*)\ else\ \bot)K_2's_2' = \bot$ and we are done.
So assume $K_1' \neq \bot \vee K_2' \neq \bot$ and $s_1' \neq \bot \vee s_2' \neq \bot$.

For all cases we have
$(\lambda k.\lambda S.\bot)K_1's_1' = (\lambda k.\lambda S.\bot)K_1 s_1 = \bot$. So we want to prove that in the other side the primed application gives $\bot$.

If the original states were related in $\mathfrak{s}^{sP\bot}$ then we now have what we want, so assume that $(pks_5) \notin \mathfrak{s}^{sP\bot}$.

Cases:
1) $q^o \in p_5$ or $r_3 \in p_5$ with k-instantiation to q^o or $(q^o \bar{\wedge} q^b) \in p_5$ with k-instantiation to q^o. Only one possible instantiation for states so $(S_2 l_x = n \neq 0)$ and then $S_2' l_x \in \{n, \bot\}$:
$(\lambda k.\lambda S.\ if\ (S(l_x) = 0)\ then\ (k(S[l_y \mapsto 1])*)\ else\ \bot)K_2 s_2 = \bot$
$(\lambda k.\lambda S.\ if\ (S(l_x) = 0)\ then\ (k(S[l_y \mapsto 1])*)\ else\ \bot)K_2' s_2' = \bot$

2) $(q^b) \in p_5$ or $(q^o \bar{\wedge} q^b) \in p_5$ with k-instantiation to q^b. Then states are related in $(\bot, \langle S_2 l_x = 0 \wedge S_2 l_y = m \neq 0\rangle)$ and then $s_2' l_x \in \{0, \bot\}$ and $s_2' l_y \in \{m, \bot\}$:
If $s_2' = \bot$ or $s_2' l_x = \bot$ then $(\lambda k.\lambda S.\ if\ (S(l_x) = 0)\ then\ (k(S[l_y \mapsto 1])*)\ else\ \bot)K_2' s_2' = \bot$,
If $s_2' \neq \bot$ and $s_2' l_x = 0$ then
$(\lambda k.\lambda S.\ if\ (S(l_x) = 0)\ then\ (k(S[l_y \mapsto 1])*)\ else\ \bot)K_2' s_2' = K_2'(s_2'[l_y \mapsto 1])* = \bot$
$(\lambda k.\lambda S.\ if\ (S(l_x) = 0)\ then\ (k(S[l_y \mapsto 1])*)\ else\ \bot)K_2 s_2 = K_2(s_2[l_y \mapsto 1])*$
Here we have used: The updated states are related in the same instantiation as before.
K_1', K_2', K_1, K_2 are related by assumption. Also * values are related. So it follows that $K_2'(s_2'[l_y \mapsto 1])* = \bot$.

3) $(q^s \prec q^b) \in p_5$ or $r_3 \in p_5$ with k-instantiation to $(q^s \prec q^b)$. Then states are related in $(S_2 l_x = 0 \wedge S_2 l_y = 0)$:
If $s_2' = \bot$ or $s_2 l_x = \bot$ then $(\lambda k.\lambda S.\ if\ (S(l_x) = 0)\ then\ (k(S[l_y \mapsto 1])*)\ else\ \bot)K_2' s_2' = \bot$.
Else $(\lambda k.\lambda S.\ if\ (S(l_x) = 0)\ then\ (k(S[l_y \mapsto 1])*)\ else\ \bot)K_2' s_2' = K_2'(s_2'[l_y \mapsto 1])* = \bot$
$(\lambda k.\lambda S.\ if\ (S(l_x) = 0)\ then\ (k(S[l_y \mapsto 1])*)\ else\ \bot)K_2 s_2 = K_2(s_2[l_y \mapsto 1])*$
Here we have used: The updated states $s_1', s_1, s_2'[l_y \mapsto 1], s_2[l_y \mapsto 1]$ are related in the local extension q^b (follows from parameter weakening for stored values). K_1', K_2', K_1, K_2 are related by assumption. Also * values are related. So it follows that $K_2'(s_2'[l_y \mapsto 1])* = \bot$.

We conclude that the functions are related
$(in_{\multimap}\lfloor \lambda a.\lambda k.\lambda S.\bot \rfloor,\ in_{\multimap}\lfloor \lambda a.\lambda k.\lambda S.\ if\ (S(l_x) = 0)\ then\ (k(S[l_y \mapsto 1])*)\ else\ \bot \rfloor \parallel$
$in_{\multimap}\lfloor \lambda a.\lambda k.\lambda S.\bot \rfloor,\ in_{\multimap}\lfloor \lambda a.\lambda k.\lambda S.\ if\ (S(l_x) = 0)\ then\ (k(S[l_y \mapsto 1])*)\ else\ \bot \rfloor,$
$unit \to Tunit, p_3) \in \nabla_V$.

By assumptions of relatedness of g's this implies that we have related computations
$(g_1'(in_{\multimap}\lfloor \lambda.\bot \rfloor),\ g_2'(in_{\multimap}\lfloor \lambda a.\lambda k.\lambda S.\ if\ (S(l_x) = 0)\ then\ (k(S[l_y \mapsto 1])*)\ else\ \bot \rfloor) \parallel$
$g_1(in_{\multimap}\lfloor \lambda.\bot \rfloor),\ g_2(in_{\multimap}\lfloor \lambda a.\lambda k.\lambda S.\ if\ (S(l_x) = 0)\ then\ (k(S[l_y \mapsto 1])*)\ else\ \bot \rfloor),$
$Tunit, p_3) \in \nabla_M$.

We now want to show that these computations are applied to continuations and states related in an extension. In this case, actually in instantiations of the same parameter.

Let $(pk_3) \in p_3^{\mathbf{K}}$ and $(pk_3) \blacktriangleright (pk_2)$ with the r_3 k-instantiation
$((S_2 l_x = 0 \wedge S_2 l_y = 0) \prec (\bot, (S_2 l_x = 0 \wedge S_2 l_y \neq 0))$.
Let $(pks_3) \in (pk_3)^{\mathbf{S}}$, $(pks_3) \supseteq (pks_2)$ it then has the r_3 s-instantiation $(S_2 l_x = 0 \wedge S_2 l_y = 0)$.

By assumption $(S_1', S_2' \parallel S_1, S_2, (pks_2)) \in \nabla_S$ then by the definition of p^3 and parameter weakening for stored values so $(S_1', S_2'[l_x \mapsto 0, l_y \mapsto 0] \parallel S_1, S_2[l_x \mapsto 0, l_y \mapsto 0], (pks_3)) \in \nabla_S$.

We will show that the continuations are related:
$(k_1', (\lambda S^0 \lambda d^0.[\![x, y \vdash \text{if } (!y \neq 0) \text{ then } \langle(\text{rec f(a)= f())()}\rangle \text{ else } (x := 1; y := 1)]\!](x \mapsto l_x, y \mapsto l_y) k_2' S^0) ||$
$k_1, (\lambda S^0 \lambda d^0.[\![x, y \vdash \text{if } (!y \neq 0) \text{ then } \langle(\text{rec f(a)= f()) ()}\rangle \text{ else } (x := 1; y := 1)]\!](x \mapsto l_x, y \mapsto l_y) k_2 S^0)$,
$(x : unit)^{\top}$, $(pk_3)) \in \nabla_K$.

Relatedness of continuations require that they "behave well" for all states and values in $\rhd$-extensions, that is local extensions and additional ordinary local parameters in disjoint areas. It is important that removal of $\bar{\wedge}$-clauses is *not* a $\rhd$-extension. We look at the cases without and with local extension.

Let $(pk_6) \rhd (pk_3)$ with $r_3 \in (pk_6)$ and r_3 k-instantiation
$((S_2 l_x = 0 \wedge S_2 l_y = 0) \prec (\bot, (S_2 l_x = 0 \wedge S_2 l_y \neq 0))$. Let $(pks_6) \in (pk_6)^{\mathbf{S}}$, there is only one possible r_3 instantiation $(S_2 l_x = 0 \wedge S_2 l_y = 0)$. Let $p_6 = (pk_6)^{vm}$ so also $p_6 \rhd p_3$.

Let $(pk_6^b) \rhd (pk_3)$ with $(q^o \bar{\wedge} q^b) \in (pk_6^b)$ with k-instantiation $(\bot, (S_2 l_x = 0 \wedge S_2 l_y \neq 0)$.
Let $p_6^b = (pk_6^b)^{vm}$ so also $p_6^b \rhd p_3$. Let $(pks_6^b) \in (pk_6^b)^{\mathbf{S}}$ it has the s-instantiation
$(\bot, (S_2 l_x = 0 \wedge S_2 l_y \neq 0))$.

First, let $(S_1^{a\prime}, S_2^{a\prime} \parallel S_1^a, S_2^a, (pks_6)) \in \nabla_S$.
We have $(d_1', d_2' \parallel *, *, unit, p_6) \in \nabla_V$ when $d_1', d_2' \in \{\bot, *\}$.

We apply the continuations to states and values and get
$k_1 S_1^a *$
$k_1' S_1^{a\prime} d_1'$

$(\lambda S^0 \lambda d^0.[\![x, y \vdash \text{if } (!y \neq 0) \text{ then } \langle(\text{rec f(a)= f())()}\rangle \text{ else } (x := 1; y := 1)]\!](x \mapsto l_x, y \mapsto l_y) k_2 S^0) S_2^a *$
$= k_2(S_2^a[l_x \mapsto 1, l_y \mapsto 1]) *$
$(\lambda S^0 \lambda d^0.[\![x, y \vdash \text{if } (!y \neq 0) \text{ then } \langle(\text{rec f(a)= f())()}\rangle \text{ else } (x := 1; y := 1)]\!](x \mapsto l_x, y \mapsto l_y) k_2' S^0) S_2^{a\prime} d_2'$
$\in \{\bot, k_2'(S_2^{a\prime}[l_x \mapsto 1, l_y \mapsto 1]) *\}$

In the following reasoning we see some of the properties of the extended parameters in use. Now the requirement $(\exists n. S_2 l_x = n \neq 0)$ is in the conjunct q^o in r_3. Let
$r_3' = \{q^o\} = \{((\exists n. S_2 l_x = n \neq 0, \emptyset), Z^{p3}, A_{\emptyset}, A_{l_x, l_y})\}$ and let p_3' be p_3 with r_3 replaced by r_3' and p_6' be p_6 with r_3 replaced by r_3' and correspondingly define (pk_6') and (pks_6'). By definition $p_3 = p_2 \cup \{r_3\}$ and so $p_3' = p_2 \cup \{r_3'\}$ and as r_3' is ordinary then $p_3' \rhd p_2$. We have $p_6' \in \mathfrak{sub}(p_6)$ so $p_6' \blacktriangleright p_6$. As $p_6 \rhd p_3$ also $p_6' \rhd p_3'$. So it holds that $p_6' \rhd p_3' \rhd p_2$ by transitivity $p_6' \rhd p_2$. Using parameter weakening for the stored values, we see that the updated states are related in the parameter (pks_6') where r_3 is replaced by r_3'. Also *'s are related under p_6'. By assumption $(k_1', k_1, k_2', k_2, (z : unit)^{\top}, (pk_2)) \in \nabla_K$. So termination approximation is ensured.

Next, let $(S_1^{b\prime}, S_2^{b\prime} \parallel S_1^b, S_2^b, (pks_6^b)) \in \nabla_S$

$(\lambda S^0 \lambda d^0.[\![x, y \vdash \text{if } (!y \neq 0) \text{ then } \langle(\text{rec f(a)= f())()}\rangle \text{ else } (x := 1; y := 1)]\!](x \mapsto l_x, y \mapsto l_y) k_2 S^0) S_2^b *$
$= \bot$ and then also for the primed case.

We conclude that
$(k_1', (\lambda S^0 \lambda d^0.[\![x, y \vdash \text{if } (!y \neq 0) \text{ then } \langle(\text{rec f(a)= f())()}\rangle \text{ else } (x := 1; y := 1)]\!](x \mapsto l_x, y \mapsto l_y) k_2' S^0) ||$

k_1, $(\lambda S^0 \lambda d^0.[\![x, y \vdash \text{if } (!y \neq 0) \text{ then } \langle(\text{rec f(a)= f())()}\rangle \text{ else } (x := 1; y := 1)]\!](x \mapsto l_x, y \mapsto l_y)k_2S^0)$, $(x : unit)^\top, (pk)_3) \in \nabla_K$.

We can then conclude that $([\![g \vdash M]\!], [\![g \vdash N]\!], type,) \in \nabla_M^{\emptyset g}$.

11 Conclusion

Certainty of equivalence of programs occur in many connections as an important and challenging problem. Especially in the presence of recursive types and general references the reasoning becomes complicated. We have develop a proof method for contextual equivalence for recursive and polymorphic types and general references. The method uses a parameterized logical relation on top of a denotational semantics in a recursive domain. Parameters for states express properties of two states. Parameters have been specified to utilize correspondence between computations and knowledge of the initial steps of continuations they are applied to. Behind the definition of parameters and the orders between them lies an analysis of how computations and continuation interact in the presence of state. We believe that the parameters often can express in a natural way hypotheses of why two programs are expected to be equivalent. Testing a hypothesis then become a rather automatic analysis. We hope in the future to combine the method developed here with a relationally parametric interpretation of polymorphic types. We also intend to make careful comparison between our proof method and methods based on sets of bisimulations.

A Appendix

A.1 Proof of Soundness, lemma 4

Lemma 42. *Soundness*

$$\text{If } \Delta;;\vdash M : T\tau, \qquad \Delta \vdash K : (x:\tau)^\top, \qquad \Sigma : \Delta \quad \text{and} \quad S \in [\![\Sigma : \Delta]\!] \quad \text{then}$$

$$\Sigma, let\ x \Leftarrow M\ in\ K \downarrow \quad \text{implies} \quad i_3^{-1}([\![\Delta;;\vdash M\ ]\!]\{\})[\![\Delta \vdash K : (x:\tau)^\top]\!]S = \top$$

Proof by induction over termination judgements.
(In the following we omit the isomorphism i and sometimes also injection functions)

Proof
Assume $\Sigma : \Delta$ and $S \in [\![\Sigma : \Delta]\!]$. Then $S \neq \bot$.
Assume in each case termination by the stated rule.

- $$\frac{}{\Sigma, let\ x \Leftarrow val\ V\ in\ val\ x \downarrow}$$

  Assume $\Delta;;\vdash val\ V : T\tau$. This requires $\Delta;;\vdash V : \tau$. By lemma 2 then $[\![\Delta;;\vdash V : \tau]\!]\{\} \neq \bot$.
  $[\![\Delta;;\vdash val\ V : T\tau]\!]\{\}[\![\Delta \vdash val\ x : (x:\tau)^\top]\!]S =$
  $[\![\Delta \vdash val\ x : (x:\tau)^\top]\!]S([\![\Delta;;\vdash V : \tau]\!]\{\}) =$
  $[\![\Delta;: x:\tau \vdash val\ x : T\tau]\!]\{x \mapsto [\![\Delta;;\vdash V : \tau]\!]\{\}\}((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S =$
  $((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S[\![\Delta;;\vdash V : \tau]\!]\{\} = \top.$

- $$\frac{\Sigma, let\ y \Leftarrow M[V/x]\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow val\ V\ in\ (let\ y \Leftarrow M\ in\ K) \downarrow}$$

  We assume $\Delta;;\vdash val\ V : T\tau$ and $\Delta \vdash let\ y \Leftarrow M\ in\ K : (x:\tau)^\top$.
  This requires $\Delta \vdash K : (y:\tau')^\top$ and $\Delta;;\vdash V : \tau$ and $\Delta;;x:\tau \vdash M : T\tau'$. Then also $\Delta;;\vdash M[V/x] : T\tau'$. So we may assume by induction $[\![\Delta;;\vdash M[V/x] : T\tau']\!]\{\}[\![\Delta \vdash K : (y:\tau')^\top]\!]S = \top$. We want to show that $[\![\Delta;;\vdash val\ V : T\tau]\!]\{\}[\![\Delta \vdash (let\ y \Leftarrow M\ in\ K) : (x:\tau)^\top]\!]S = \top$.
  $[\![\Delta;;\vdash val\ V : T\tau]\!]\{\}[\![\Delta \vdash (let\ y \Leftarrow M\ in\ K) : (x:\tau)^\top]\!]S =$
  $[\![\Delta;;\vdash val\ V : T\tau]\!]\{\}(\lambda S'.\lambda d'.[\![\Delta;;x:\tau \vdash let\ y \Leftarrow M\ in\ K : T\tau'']\!]\{x \mapsto d'\}((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S')S =$
  $(\lambda S'.\lambda d'.[\![\Delta;;x:\tau \vdash let\ y \Leftarrow M\ in\ K : T\tau'']\!]\{x \mapsto d'\}((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S')S[\![\Delta;;\vdash V : \tau]\!]\{\} =$
  $[\![\Delta;;x:\tau \vdash let\ y \Leftarrow M\ in\ K : T\tau'']\!]\{x \mapsto [\![\Delta;;\vdash V : \tau]\!]\{\}\}((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S =$
  $[\![\Delta;;\vdash let\ y \Leftarrow M[V/x]\ in\ K : T\tau'']\!]\{\}((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S =$
  $[\![\Delta;;\vdash M[V/x]]\!]\{\}(\lambda S^0.\lambda d^0.[\![\Delta;;y:\tau' \vdash K : T\tau'']\!]\{y \mapsto d^0\}((\lambda S'.(\lambda d'.\top)_\bot)_\bot)S^0)S =$
  $[\![\Delta;;\vdash M[V/x]]\!]\{\}[\![\Delta \vdash K : (y:\tau')^\top]\!]S = \top$
  where we have used the inductive assumption for the last equality.

- $$\frac{\Sigma, let\ y \Leftarrow M_1\ in\ (let\ x \Leftarrow M_2\ in\ K) \downarrow}{\Sigma, let\ x \Leftarrow (let\ y \Leftarrow M_1\ in\ M_2)\ in\ K \downarrow}$$

  We assume $\Delta;;\vdash (let\ y \Leftarrow M_1\ in\ M_2) : T\tau$ and $\Delta \vdash K : (x:\tau)^\top$. This requires $\Delta;;\vdash M_1 : T\tau'$ and $\Delta;;y:\tau' \vdash M_2 : T\tau$. Then also $\Delta \vdash let\ x \Leftarrow M_2\ in\ K : (y:\tau')^\top$. So by induction we may assume $[\![\Delta;;\vdash M_1 : T\tau']\!]\{\}[\![\Delta \vdash let\ x \Leftarrow M_2\ in\ K : (y:\tau')^\top]\!]S = \top$.
  We want to show that $[\![\Delta;;\vdash let\ y \Leftarrow M_1\ in\ M_2 : T\tau]\!]\{\}[\![\Delta \vdash K : (x:\tau)^\top]\!]S = \top$.
  We derive this: $[\![\Delta;;\vdash let\ y \Leftarrow M_1\ in\ M_2 : T\tau]\!]\{\}[\![\Delta \vdash K : (x:\tau)^\top]\!]S =$
  $[\![\Delta;;\vdash M_1 : T\tau']\!]\{\}(\lambda S'.\lambda d'.[\![\Delta;;y:\tau' \vdash M_2 : T\tau]\!]\{y \mapsto d'\}[\![\Delta \vdash K : (x:\tau)^\top]\!]S')S =$
  $[\![\Delta;;\vdash M_1 : T\tau']\!]\{\}[\![\Delta;;\vdash let\ x \Leftarrow M_2\ in\ K : (y:\tau')^\top]\!]S = \top$

- $$\frac{\Sigma, let\ x \Leftarrow M[V/y, (rec\ f(y:\tau) = M)/f]\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow ((rec\ f(y:\tau) = M)V)\ in\ K \downarrow}$$

We assume $\Delta;;\vdash (rec\ f(y:\tau) = M)V : T\tau'$ and $\Delta \vdash K : (x:\tau')^\top$. This requires $\Delta;;\vdash rec\ f(y:\tau) = M : \tau \to T\tau'$ and $\Delta;;\vdash V : \tau$, and also $\Delta;;f:\tau \to T\tau', y:\tau \vdash M : T\tau'$. Then also $\Delta;;\vdash M[V/y, (rec\ f(y:\tau) = M)/f] : T\tau'$ and we may inductively assume $[\![\Delta;;\vdash M[V/y, (rec\ f(y:\tau) = M)/f] : T\tau']\!]\{\}[\![\Delta \vdash K : (x:\tau')^\top]\!]S = \top$, and also by lemma 3 $[\![\Delta;;y:\tau, f:\tau \to T\tau' \vdash M : T\tau']\!]\{y \mapsto [\![\Delta;;\vdash V : \tau]\!]\{\}, f \mapsto [\![\Delta;\vdash rec\ f(y:\tau) = M : T\tau']\!]\{\}\}[\![\Delta \vdash K : (x:\tau')^\top]\!]S = \top$.
We want to show that $[\![\Delta;;\vdash (rec\ f(y:\tau) = M)V : T\tau']\!]\{\}[\![\Delta \vdash K : (x:\tau')^\top]\!]S = \top$.
$[\![\Delta;;\vdash (rec\ f(y:\tau) = M)V : T\tau']\!]\{\}[\![\Delta;\vdash K : (x:\tau')^\top]\!]^{\mathcal{K}}S =$
$([\![\Delta;;\vdash rec\ f(y:\tau) = M : \tau \to T\tau']\!]\{\}[\![\Delta;;\vdash V : \tau]\!]\{\})[\![\Delta;\vdash K : (x:\tau')^\top]\!]S =$
$[\![\Delta;;y:\tau, f:\tau \to T\tau' \vdash M : T\tau']\!]\{y \mapsto [\![\Delta;;\vdash V : \tau]\!]\{\}, f \mapsto [\![\Delta;;\vdash rec\ f(y:\tau) = M : T\tau']\!]\{\}\}[\![\Delta;\vdash K : (x:\tau')^\top]\!]S = \top$

- $$\frac{\Sigma, let\ x \Leftarrow val\ V_1\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow \pi_1(V_1, V_2)\ in\ K \downarrow} \qquad \frac{\Sigma, let\ x \Leftarrow val\ V_2\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow \pi_2(V_1, V_2)\ in\ K \downarrow}$$ Similar.

We assume $\Delta;;\vdash \pi_1(V_1, V_2) : T\tau_1$ and $\Delta \vdash K : (x:\tau_1)^\top$. This requires $\Delta;;\vdash (V_1, V_2) : \tau_1 \times \tau_2$, and also $\Delta;;\vdash V_1 : \tau_1$ and then $\Delta;;\vdash val\ V_1 : T\tau_1$. So by induction we assume $[\![\Delta;;\vdash val\ V_1 : T\tau_1]\!]\{\}[\![\Delta \vdash K : (x:\tau_1)^\top]\!]S = \top$. Then $[\![\Delta \vdash K : (x:\tau_1)^\top]\!]S[\![\Delta;;\vdash V_1 : \tau_1]\!]\{\} = \top$.
We want to show that $[\![\Delta;;\vdash \pi_1(V_1, V_2) : T\tau_1]\!]\{\}[\![\Delta \vdash K : (x:\tau_1)^\top]\!]S = \top$.
$[\![\Delta;;\vdash \pi_1(V_1, V_2) : T\tau_1]\!]\{\}[\![\Delta \vdash K : (x:\tau_1)^\top]\!]S = [\![\Delta \vdash K : (x:\tau_1)^\top]\!]S[\![\Delta;;\vdash V_1 : \tau_1]\!]\{\} = \top$.

- $$\frac{\Sigma, let\ x \Leftarrow M_1[V/x_1]\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow case\ in_1 V\ of\ in_1 x_1 \Rightarrow M_1; in_2 x_2 \Rightarrow M_2\ in\ K \downarrow}$$

We assume $\Delta;;\vdash case\ in_1 V\ of\ in_1 x_1 \Rightarrow M_1; in_2 x_2 \Rightarrow M_2 : T\tau'$ and $\Delta \vdash K : (x:\tau')^\top$. This requires $\Delta;;\vdash in_1 V : \tau_1 + \tau_2$ and $\Delta;;\vdash V : \tau_1$ and $\Delta;;x_1 : \tau_1 \vdash M_1 : T\tau'$ and $\Delta;;x_2 : \tau_2 :\vdash M_2 : T\tau'$. Then $\Delta;;\vdash M_1[V/x_1] : T\tau'$ and by induction we assume $[\![\Delta;;\vdash M_1[V/x_1] : T\tau']\!]\{\}[\![\Delta \vdash K : (x:\tau')^\top]\!]S = \top$. By lemma 3 also $[\![\Delta;;\vdash M_1[V/x_1] : T\tau']\!]\{\} = [\![\Delta;;x_1 : \tau_1 \vdash M_1 : T\tau']\!]\{x_1 \mapsto [\![\Delta;;\vdash V : \tau_1]\!]\{\}\}$. We want to show that $[\![\Delta;;\vdash case\ in_1 V\ of\ in_1 x_1 \Rightarrow M_1; in_2 x_2 \Rightarrow M_2 : T\tau']\!]\{\}[\![\Delta \vdash K : (x:\tau')^\top]\!]S = \top$. That is $[\![\Delta;;x_1 : \tau_1 \vdash M_1 : T\tau']\!]\{x_1 \mapsto [\![\Delta;;: V\tau_1]\!]\{\}\}[\![\Delta \vdash K : (x:\tau')^\top]\!]S = \top$. This holds by inductive assumption.

- $$\frac{\Sigma, let\ x \Leftarrow M_2[V/x_2]\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow case\ in_2 V\ of\ in_1 x_1 \Rightarrow M_1; in_2 x_2 \Rightarrow M_2\ in\ K \downarrow}$$ Similar.

- $$\frac{\Sigma, let\ x \Leftarrow val\ true\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow \underline{l} = \underline{l}\ in\ K \downarrow}$$

We assume $\Delta;;\vdash \underline{l} = \underline{l} : T(unit + unit)$ and $\Delta \vdash K : (x : unit + unit)^\top$. It also holds that $\Delta;;\vdash val\ true : T(unit + unit)$, so by induction we assume $[\![\Delta;;\vdash val\ true : T(unit + unit)]\!]\{\}[\![\Delta \vdash K : (x : unit + unit)^\top]\!]S = \top$ and so $[\![\Delta \vdash K : (x : unit + unit)^\top]\!]S(in_\oplus in_1(in_{\mathbf{1}}*)) = \top$.
We want to show that $[\![\Delta;;\vdash l = l : T(unit + unit)]\!]\{\}[\![\Delta \vdash K : (x : (unit + unit))^\top]\!]S = \top$. This holds $[\![\Delta;;\vdash l = l : T(unit + unit)]\!]\{\}[\![\Delta \vdash K : (x : (unit + unit))^\top]\!]S = \lambda k.\lambda s.ks(in_\oplus in_1(in_{\mathbf{1}}*))[\![\Delta \vdash K : (x : (unit + unit))^\top]\!]S = \top$.

- $$\frac{\Sigma, let\ x \Leftarrow val\ false\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow \underline{l} = \underline{l'}\ in\ K \downarrow}\ (l \neq l')$$

We assume $l \neq l'$ and $\Delta;;\vdash \underline{l} = \underline{l'} : T(unit + unit)$ and $\Delta \vdash K.(x : unit + unit)$. It also holds that $\Delta;;\vdash val\ false : T(unit + unit)$, so by induction we assume $[\![\Delta;;\vdash val\ false : T(unit+unit)]\!]\{\}[\![\Delta \vdash K : (x : unit + unit)^{\top}]\!]S = \top$.
We want to show that $[\![\Delta;;\vdash l = l' : T(unit+unit)]\!]\{\}[\![\Delta \vdash K : (x : (unit+unit))^{\top}]\!]S = \top$.
We have for $l \neq l'$ that $[\![\Delta;;\vdash l = l' : T(unit + unit)]\!]\{\} = \lambda k.\lambda s.ks(in_{\oplus}in_2(in_{\mathbf{1}}*)) = [\![\Delta;;\vdash val\ false : T(unit + unit)]\!]\{\}$

- $$\frac{\Sigma[l \mapsto V], let\ x \Leftarrow val()\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow l := V\ in\ K \downarrow}$$

We assume $\Delta;;\vdash l := V : Tunit$ and $\Delta \vdash K : (x : unit)^{\top}$. This requires $l \in dom(\Delta)$ and $\Delta;;\vdash l : \Delta(l)\ ref$ and $\Delta;;\vdash V : \Delta(l)$. So it holds that $\Sigma : \Delta$ implies $\Sigma[l \mapsto V] : \Delta$.

Let $S' = S[l \mapsto [\![\Delta;;\vdash V : \tau]\!]\{\}]$, first we want to show $S' \in [\![\Sigma[l \mapsto V] : \Delta]\!]$, this requires $\forall l^0 \in dom(\Delta).\ S'(l^0) = [\![\Delta;;\vdash \Sigma[l \mapsto V](l^0) : \Delta(l^0)]\!]\{\}$. By assumptions $S \in [\![\Sigma : \Delta]\!]$ so $\forall l^0 \in dom(\Delta).\ S(l^0) = [\![\Delta;;\vdash \Sigma(l^0) : \Delta(l^0)]\!]\{\}$. Let $l^0 \neq l$ then $\Sigma[l \mapsto V](l^0) = \Sigma(l^0)$ and $S'(l^0) = S(l^0)$. For l we get $\Sigma[l \mapsto V](l) = V$ and so $\Delta;;\vdash \Sigma[l \mapsto V](l) : \Delta(l)$. Now $[\![\Delta;;\vdash l := V : Tunit]\!]\{\}[\![\Delta \vdash K : (x : unit)^{\top}]\!]S = [\![\Delta \vdash K : (x : unit)^{\top}]\!](S[l \mapsto [\![\Delta;;\vdash V : \tau]\!]\{\}])* = \top$. The last equality comes by inductive assumptions.

- $$\frac{\Sigma(l) = V \qquad \Sigma, let\ x \Leftarrow val\ V\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow !l\ in\ K \downarrow}$$

We assume $\Delta;;\vdash !l : T\tau$ and $\Delta \vdash K : (x : \tau)^{\top}$. This requires $\Delta;;\vdash l : \tau\ ref$ and $\Delta l = \tau$. When $S \in [\![\Sigma : \Delta]\!]$ then it holds that $Sl = [\![\Delta;;\vdash \Sigma l : \Delta l]\!]\{\}$ that is $Sl = [\![\Delta;;\vdash V : \tau]\!]\{\}$ and $\Delta;;\vdash V : \tau$ and so $\Delta;;\vdash val\ V : T\tau$. By induction we assume $[\![\Delta;;\vdash val\ V : T\tau]\!]\{\}[\![\Delta \vdash K : (x : \tau)^{\top}]\!]S = \top$ so $[\![\Delta \vdash K : (x : \tau)^{\top}]\!]S[\![\Delta;;\vdash V : \tau]\!]\{\} = \top$. We want to show that $[\![\Delta;;\vdash !l : T\tau]\!]\{\}[\![\Delta \vdash K : (x : \tau)^{\top}]\!]S = \top$. Since $S \in [\![\Sigma : \Delta]\!]$ then $[\![\Delta;;\vdash !l : T\tau]\!]\{\}[\![\Delta \vdash K : (x : \tau)^{\top}]\!]S = [\![\Delta \vdash K : (x : \tau)^{\top}]\!]S(Sl) = [\![\Delta \vdash K : (x : \tau)^{\top}]\!]S[\![\Delta;;\vdash V : \tau]\!]\{\} = \top$.

- $$\frac{\Sigma[l \mapsto V], let\ x \Leftarrow val\ l\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow ref\ V\ in\ K \downarrow}\ (l \notin locs(\Sigma) \cup locs(K) \cup locs(V))$$

We assume $\Delta;;\vdash ref\ V : T\tau\ ref$ and $\Delta \vdash K : (x : \tau\ ref)^{\top}$. This requires $\Delta;;\vdash V : \tau$.
We want to show that $[\![\Delta;\vdash refV : T\tau\ ref]\!]\{\}[\![\Delta;\vdash K : (x : \tau\ ref)^{\top}]\!]S = \top$.
Let $l \notin locs(\Sigma) \cup locs(K) \cup locs(V)$ and let $S' = S[l \mapsto [\![\Delta;;\vdash V : \tau]\!]\{\}]$. Then also $l \notin supp(\lambda l'.[\![\Delta \vdash K : (x : \tau\ ref)]\!](S[l' \mapsto [\![\Delta;;\vdash V : \tau]\!]\{\}])l')$ and $l \notin dom(\Delta) \subseteq locs(\Sigma)$. By assumption $S \in [\![\Sigma : \Delta]\!]$ so $\forall l^0 \in dom(\Delta).S(l^0) = [\![\Delta;;\vdash \Sigma(l^0) : \Delta(l^0)]\!]\{\}$. Let $l^0 \in dom(\Delta)$ then $S'(l^0) = S(l^0)$ and $\Sigma[l \mapsto V](l^0) = \Sigma(l^0)$ and $\Delta \uplus \{l : \tau\})(l^0) = \Sigma(l^0)$. For l we have $S'(l) = [\![\Delta;;\vdash V : \tau]\!]\{\}$ and $\Sigma[l \mapsto V](l) = V$ and $(\Delta \uplus \{l \mapsto \tau\})(l) = \tau$. So it holds that $S' \in [\![\Sigma[l \mapsto V] : \Delta \uplus \{l : \tau\}]\!]$.
Then $[\![\Delta;;\vdash refV : T\tau\ ref]\!]\{\}[\![\Delta \vdash K : (x : \tau\ ref)^{\top}]\!]S =$
$[\![\Delta;\vdash K : (x : \tau\ ref)^{\top}]\!](S[l \mapsto [\![\Delta;\vdash V : \tau]\!]\{\}])l =$
$[\![\Delta;\vdash val\ l : T\tau\ ref]\!]\{\}[\![\Delta;\vdash K : (x : \tau\ ref)^{\top}]\!]S' = \top$.
Where we have used induction for the last equality.

- $$\frac{\Sigma, let\ x \Leftarrow val\ V\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow unfold(fold\ V)\ in\ K \downarrow}$$

We assume $\Delta;;\vdash unfold(fold\ V) : T(\tau[\mu\alpha.\tau/\alpha])$ and $\Delta \vdash K : (x : \tau[\mu\alpha.\tau/\alpha])^{\top}$. This requires $\Delta;;\vdash foldV : \mu\alpha.\tau$ and $\Delta;;\vdash V : \tau[\mu\alpha.\tau/\alpha]$.

We want to show $[\![\Delta;;\vdash unfold(fold\ V) : T(\tau[\mu\alpha.\tau/\alpha])]\!]\{\}[\![\Delta \vdash K : (x : \tau[\mu\alpha.\tau/\alpha])^{\top}]\!]S = \top$.

By induction we assume $[\![\Delta;;\vdash val\ V : T(\tau[\mu\alpha.\tau/\alpha])]\!]\{\}[\![\Delta \vdash K : (x : \tau[\mu\alpha.\tau/\alpha])^{\top}]\!]S = \top$. Then $[\![\Delta \vdash K : (x : \tau[\mu\alpha.\tau/\alpha])^{\top}]\!]S[\![\Delta;\vdash V : \tau[\mu\alpha.\tau/\alpha]]\!]\{\} = \top$.

We use $[\![\Delta;;\vdash fold\ V : \mu\alpha.\tau]\!]\{\} = in_{\mu}([\![\Delta;;\vdash V : \tau[\mu\alpha.\tau/\alpha]]\!]\{\})$.
$[\![\Delta;;\vdash unfold(fold\ V) : T(\tau[\mu\alpha.\tau/\alpha])]\!]\{\}[\![\Delta \vdash K : (x : \tau[\mu\alpha.\tau/\alpha])^{\top}]\!]S =$
$[\![\Delta \vdash K : (x : \tau[\mu\alpha.\tau/\alpha])^{\top}]\!]S[\![\Delta;;\vdash V : \tau[\mu\alpha.\tau/\alpha]]\!]\{\} = \top$

- $\dfrac{\Sigma, let\ x \Leftarrow M\ in\ K \downarrow}{\Sigma, let\ x \Leftarrow (\Lambda\alpha.M)\tau'\ in\ K \downarrow}$

We assume $\Delta;;\vdash (\Lambda\alpha.M)\tau' : T(\tau[\tau'/\alpha])$ and $\Delta \vdash K : (x : \tau[\tau'/\alpha])^{\top}$. This requires $\Delta;;\vdash \Lambda\alpha.M : \forall\alpha.T\tau$ and $\vdash \tau' : type$. $\Delta;;\vdash \Lambda\alpha.M : \forall\alpha.T\tau$ again requires $\Delta;\alpha;\vdash M : T\tau$ and so $\Delta;;\vdash M : T\tau[\tau'/\alpha]$. We want to show $[\![\Delta;;\vdash (\Lambda\alpha.M)\tau' : T\tau[\tau'/\alpha]]\!]\{\}[\![\Delta \vdash K : (x : \tau[\tau'/\alpha])^{\top}]\!]S = \top$. By induction we assume $[\![\Delta;;\vdash M : T\tau[\tau'/\alpha]]\!]\{\}[\![\Delta \vdash K : (x : \tau[\tau'/\alpha])^{\top}]\!]S = \top$.
We have $[\![\Delta;;\vdash \Lambda\alpha.M : \forall\alpha.T\tau]\!]\{\} = in_{\forall}\lfloor[\![\Delta;\alpha;\vdash M : T\tau]\!]\{\}\rfloor$, so $[\![\Delta;;\vdash (\Lambda\alpha.M)\tau' : T\tau[\tau'/\alpha]]\!]\{\}[\![\Delta \vdash K : (x : \tau[\tau'/\alpha])^{\top}]\!]S = [\![\Delta;\alpha;\vdash M : T\tau]\!]\{\}[\![\Delta \vdash K : (x : \tau[\tau'/\alpha])^{\top}]\!]S$. As when $\Delta;\alpha;\vdash M : T\tau$ then $[\![\Delta;\vdash M : T\tau[\tau'/\alpha]]\!]\{\} = [\![\Delta;\alpha;\vdash M : T\tau]\!]\{\}$ then $[\![\Delta;;\vdash \Lambda\alpha : M : \forall\alpha.T\tau]\!]\{\} = in_{\forall}\lfloor[\![\Delta;;\vdash M : T\tau[\tau'/\alpha]]\!]\{\}\rfloor$. From this follows $\top = [\![\Delta;;\Gamma \vdash M : T\tau[\tau'/\alpha]]\!]\{\}[\![\Delta \vdash K : (x : \tau[\tau'/\alpha])^{\top}]\!]S = [\![\Delta;;\vdash (\Lambda\alpha.M)\tau'\ ]\!]\{\}[\![\Delta \vdash K : (x : \tau[\tau'/\alpha])^{\top}]\!]S$.

Some rules about arithmetics omitted.

□

A.2 Proof of lemma 12. Typing rules preserve the $\mathbb{R}_{\Gamma}$ relation of a term and its denotation

.

We often omit the isomorphism i and sometimes also injection functions into $F(\mathbb{D},\mathbb{D})_V$.

Proof $\Xi = \alpha_1,\ldots,\alpha_m$ and $\Gamma = x_1 : \tau_1,\ldots,x_n : \tau_n$ with Γ well typed by Ξ.
To shorten explanations in proofs we let *arbitrarily* $\sigma_1\ldots\sigma_m$ be closed value types, and $\Delta' \supseteq \Delta$ and $(v_i,\ \Delta';;\vdash V_i : \tau_i\overline{[\sigma_j/\alpha_j]}) \in \mathbb{R}_V,\ i \in 1,..,n$ and $\rho = v_1 \otimes \ldots \otimes v_n$. The $\rho = \bot$ cases are trivial by strictness, so we also assume $\rho \neq \bot$.

(id) No premisses, so we need to show $([\![\Delta;\Xi;\Gamma \vdash x_i : \tau_i]\!],\ \Delta;\Xi;\Gamma \vdash x_i : \tau_i) \in \mathbb{R}_{\Xi\Gamma\vdash V}$. This requires that in all cases $([\![\Delta;\Xi;\Gamma \vdash x_i : \tau_i]\!](\rho),\ \Delta';;\vdash x_i[V_i/x_i] : \tau_i\overline{[\sigma_j/\alpha_j]}) \in \mathbb{R}_V$.
If $\rho = \bot$ then $[\![\Delta;\Xi;\Gamma \vdash x_i : \tau_i]\!](\rho) = \bot$ and $(\bot,\ \Delta';;\vdash x_i[V_i/x_i] : \tau_i\overline{[\sigma_j/\alpha_j]}) \in \mathbb{R}_V$.
If $\rho \neq \bot$ then $[\![\Delta;\Xi;\vdash x_i : \tau_j]\!](\rho) = \rho(x_i) = v_i$, and we have $(v_i,\ \Delta';;\vdash V_i : \tau_i\overline{[\sigma_j/\alpha_j]}) \in \mathbb{R}_V$.

(unit) To show $([\![\Delta;\Xi;\Gamma \vdash () : unit]\!](\rho),\ \Delta;\Xi;\Gamma \vdash () : unit) \in \mathbb{R}_{\Xi\Gamma\vdash V}$. This is (when $\rho \neq \bot$) that $(in_1\lfloor * \rfloor,\ \Delta';;\vdash () : unit) \in \mathbb{R}_V$. Holds by definition.

(int) To show $([\![\Delta;\Xi;\Gamma \vdash n : int]\!](\rho),\ \Delta;\Xi;\Gamma \vdash n : int) \in \mathbb{R}_{\Xi\Gamma\vdash V}$. This is that $(in_{\mathbb{Z}}\lfloor n\rfloor,\ \Delta';;\vdash n : int) \in \mathbb{R}_V$. Holds by definition.

(loc) To show $([\![\Delta;\Xi;\Gamma \vdash l : \Delta(l)\ ref]\!](\rho),\ \Delta;\Xi;\Gamma \vdash l : \Delta(l)\ ref) \in \mathbb{R}_{\Xi\Gamma\vdash V}$. This is that $(in_{\mathbb{L}}\lfloor l\rfloor,\ \Delta';;\vdash l : \Delta(l)[\sigma_j/\alpha_j]\ ref) = (in_{\mathbb{L}}\lfloor l\rfloor,\ \Delta';;\vdash l : \Delta(l)\ ref) \in \mathbb{R}_V$. Follows from $\Delta' \supseteq \Delta$.

(+ intro) Assume $(\llbracket \Delta;\Xi;\Gamma \vdash V : \tau_j \rrbracket,\ \Delta;\Xi;\Gamma \vdash V : \tau_j) \in \mathbb{R}_{\Xi\Gamma\vdash V}$ (j is 1 or 2).
To show $(\llbracket \Delta;\Xi;\Gamma \vdash in_j V : \tau_1 + \tau_2 \rrbracket,\ \Delta;\Xi;\Gamma \vdash in_j V : \tau_1 + \tau_2) \in \mathbb{R}_{\Xi\Gamma\vdash V}$.
This requires $(\llbracket \Delta;\Xi;\Gamma \vdash in_j V : \tau_1 + \tau_2 \rrbracket(\rho),\ \Delta';;\vdash in_j V[\overline{V_i/x_i}] : (\tau_1[\overline{\sigma_j/\alpha_j}] + \tau_2[\overline{\sigma_j/\alpha_j}])) \in \mathbb{R}$.
From the assumtion we get $(\llbracket \Delta;\Xi;\Gamma \vdash V : \tau_j \rrbracket(\rho),\ \Delta';;\vdash V[\overline{V_i/x_i}] : \tau_j[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_V$. Then from the definition of $\mathbb{R}$ since $- \vdash \tau_2[\overline{\sigma_j/\alpha_j}]$ it holds that $(in_\oplus in_j \llbracket \Delta;\Xi;\Gamma \vdash V : \tau_j \rrbracket(\rho),\ \Delta';;\vdash V[\overline{V_i/x_i}] : (\tau_1[\overline{\sigma_j/\alpha_j}] + \tau_2[\overline{\sigma_j/\alpha_j}])) \in \mathbb{R}_V$. And we have $\llbracket \Delta;\Xi;\Gamma \vdash V : \tau_1 + \tau_2 \rrbracket(\rho) = i \circ in_\oplus(in_j \llbracket \Delta;\Xi;\Gamma \vdash V : \tau_j \rrbracket(\rho))$.

(× intro) Assume $(\llbracket \Delta;\Xi;\Gamma \vdash W_j : \tau_j \rrbracket,\ \Delta;\Xi;\Gamma \vdash W_j : \tau_j) \in \mathbb{R}_{\Xi\Gamma\vdash V}$ (for $j = 1$ and $j = 2$) so $(\llbracket \Delta;\Xi;\Gamma \vdash W_j : \tau_j \rrbracket(\rho),\ \Delta';;\vdash W_j[\overline{V_i/x_i}] : \tau_j[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_V$. Then from the definition of $\mathbb{R}$ it holds that $(in_\otimes(\llbracket \Delta;\Xi;\Gamma \vdash W_1 : \tau_1 \rrbracket(\rho), \llbracket \Delta;\Xi;\Gamma \vdash W_2 : \tau_2 \rrbracket(\rho)),\ \Delta';\vdash (W_1,W_2)[\overline{V_i/x_i}] : (\tau_1[\overline{\sigma_j/\alpha_j}] \times \tau_2[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_V$. And we have $\llbracket \Delta;\Xi;\Gamma \vdash (W_1,W_2) : \tau_1 + \tau_2 \rrbracket(\rho) = i \circ in_\otimes(\llbracket \Delta;\Xi;\Gamma \vdash W_1 : \tau_1 \rrbracket(\rho), \llbracket \Delta;\Xi;\Gamma \vdash W_2 : \tau_2 \rrbracket(\rho))$.

(fold) Assume $(\llbracket \Delta;\Xi;\Gamma \vdash V : \tau[\mu\alpha.\tau/\alpha] \rrbracket,\ \Delta;\Xi;\Gamma \vdash V : \tau[\mu\alpha.\tau/\alpha]) \in \mathbb{R}_{\Xi\Gamma\vdash V}$. We may assume $\alpha \notin \Xi$. To show $(\llbracket \Delta;\Xi;\Gamma \vdash fold\ V : \mu\alpha.\tau \rrbracket,\ \Delta;\Xi;\Gamma \vdash fold\ V : \mu\alpha.\tau) \in \mathbb{R}_{\Xi\Gamma\vdash V}$ or $(\llbracket \Delta;\Xi;\Gamma \vdash fold\ V : \mu\alpha.\tau \rrbracket(\rho),\ \Delta;;\vdash fold\ V[\overline{V_i/x_i}] : \mu\alpha.\tau[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_V$. So from the assumption we have $(\llbracket \Delta;\Xi;\Gamma \vdash V : \tau[\mu\alpha.\tau/\alpha] \rrbracket(\rho),\ \Delta';;\vdash V[\overline{V_i/x_i}] : \tau[\mu\alpha.\tau/\alpha][\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_V$, that is $(\llbracket \Delta;\Xi;\Gamma \vdash V : \tau[\mu\alpha.\tau/\alpha] \rrbracket(\rho),\ \Delta';;\vdash V[\overline{V_i/x_i}] : (\tau[\overline{\sigma_j/\alpha_j}])[\mu\alpha.(\tau[\overline{\sigma_j/\alpha_j}])/\alpha][\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_V$. Then from the definition of $\mathbb{R}$ it holds that $(in_\mu \llbracket \Delta;\Xi;\Gamma \vdash V : \tau[\mu\alpha.\tau/\alpha] \rrbracket(\rho),\ \Delta';\vdash fold\ V[\overline{V_i/x_i}] : \mu\alpha.(\tau[\overline{\sigma_j/\alpha_j}])) \in \mathbb{R}_V$. And we have $\llbracket \Delta;\Xi;\Gamma \vdash fold\ V : \mu\alpha.\tau \rrbracket(\rho) = i \circ in_\mu(\llbracket \Delta;\Xi;\Gamma \vdash V : \tau[\mu\alpha.\tau/\alpha] \rrbracket(\rho)$.

(rec) Let $\Gamma' = \Gamma, x : \tau, f : \tau \to T\tau'$. Assume $(\llbracket \Delta;\Xi;\Gamma' \vdash M : T\tau' \rrbracket,\ \Delta;\Xi;\Gamma' \vdash M : T\tau') \in \mathbb{R}_{\Xi\Gamma'\vdash M}$.
To show that $(\llbracket \Delta;\Xi;\Gamma \vdash rec\ f(x:\tau) = M : (\tau \to T\tau') \rrbracket,\ \Delta;\Xi;\Gamma \vdash rec\ f(x:\tau) = M : (\tau \to T\tau')) \in \mathbb{R}_{\Xi\Gamma\vdash M}$. That is $(\llbracket \Delta;\Xi;\Gamma \vdash rec\ f(x:\tau) = M : (\tau \to T\tau') \rrbracket(\rho),\ \Delta';;\vdash rec\ f(x : \tau[\overline{\sigma_j/\alpha_j}]) = M[\overline{V_i/x_i}] : (\tau[\overline{\sigma_j/\alpha_j}] \to T\tau'[\overline{\sigma_j/\alpha_j}])) \in \mathbb{R}_M$.
$\llbracket \Delta;\Xi;\Gamma \vdash rec\ f(x:\tau) = M : (\tau \to T\tau') \rrbracket(\rho) = i \circ in_{\multimap} \lfloor fix(\lambda f' \in (\mathbb{V} \multimap \mathbb{M}).(\lambda x' \in \mathbb{V}.\llbracket \Delta;\Xi;\Gamma' \vdash M : T\tau' \rrbracket(\rho \otimes f \mapsto i \circ in_{\multimap}\lfloor f' \rfloor \otimes x \mapsto x'))) \rfloor = i \circ in_{\multimap} \lfloor \bigsqcup_{n\in\omega} g_n \rfloor$ where $g_n \in (\mathbb{V} \multimap \mathbb{M})$, $g_0 = \bot_{\mathbb{V}\multimap\mathbb{M}}$ and $g_{n+1} = \lambda x_0 \in \mathbb{V}.\llbracket \Delta;\Xi;\Gamma' \vdash M : T\tau' \rrbracket(\rho \otimes f \mapsto i \circ in_{\multimap}\lfloor g_n \rfloor \otimes x \mapsto x_0)$.
We will show by induction that for each n $(i \circ in_{\multimap}\lfloor g_n \rfloor,\ \Delta';;\vdash rec\ f(x : \tau[\overline{\sigma_j/\alpha_j}]) = M[\overline{V_i/x_i}] : (\tau[\overline{\sigma_j/\alpha_j}] \to T\tau'[\overline{\sigma_j/\alpha_j}])) \in \mathbb{R}_V$. As $\bigsqcup_{n\in\omega}(i \circ in_{\multimap}\lfloor g_n \rfloor) = i \circ in_{\multimap} \lfloor \bigsqcup_{n\in\omega} g_n \rfloor$ it then it follows that since $\mathbb{R}$ is admissible then also $(\llbracket \Delta;\Xi;\Gamma \vdash rec\ f(x:\tau) = M : (\tau \to T\tau') \rrbracket(\rho),\ \Delta';;\vdash rec\ f(x : \tau[\overline{\sigma_j/\alpha_j}]) = M[\overline{V_i/x_i}] : (\tau[\overline{\sigma_j/\alpha_j}] \to T\tau'[\overline{\sigma_j/\alpha_j}])) \in \mathbb{R}_V$.
Application of g_0 gives $\bot_{\mathbb{M}}$ which is related to any typing judgement for a computation. Hence $(i \circ in_{\multimap}\lfloor g_0 \rfloor,\ \Delta';;\vdash rec\ f(x : \tau[\overline{\sigma_j/\alpha_j}]) = M[\overline{V_i/x_i}] : (\tau[\overline{\sigma_j/\alpha_j}] \to T\tau'[\overline{\sigma_j/\alpha_j}])) \in \mathbb{R}_V$. Now assume inductively that $(i \circ in_{\multimap}\lfloor g_n \rfloor,\ \Delta';;\vdash rec\ f(x : \tau[\overline{\sigma_j/\alpha_j}]) = M[\overline{V_i/x_i}] : (\tau[\overline{\sigma_j/\alpha_j}] \to T\tau'[\overline{\sigma_j/\alpha_j}])) \in \mathbb{R}_V$. We want to show that this implies $(i \circ in_{\multimap}\lfloor g_{n+1} \rfloor,\ \Delta';;\vdash rec\ f(x : \tau[\overline{\sigma_j/\alpha_j}]) = M[\overline{V_i/x_i}] : (\tau[\overline{\sigma_j/\alpha_j}] \to T\tau'[\overline{\sigma_j/\alpha_j}])) \in \mathbb{R}_V$.
Let closed $\Delta'' \supseteq \Delta', (v,\ \Delta'';;\vdash V : \tau[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_V$. We want to show that $(g_{n+1}v,\ \Delta'';;\vdash (rec\ f(x : \tau[\overline{\sigma_j/\alpha_j}]) = M[\overline{V_i/x_i}])V : T\tau'[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_M$.
Let closed $\Delta''' \supseteq \Delta''$, $(k,\ \Delta''' \vdash K : (y : \tau')^\top) \in \mathbb{R}_K$, $(s,\ \Sigma : \Delta''') \in \mathbb{R}_s$. Assume $g_{n+1}vks = \top$. We need to show $\Sigma,\ let\ y \Leftarrow (rec\ f(x : \tau[\overline{\sigma_j/\alpha_j}]) = M[\overline{V_i/x_i}])V\ in\ K \downarrow$. By judgement of termination it suffices to show $\Sigma,\ let\ y \Leftarrow M[\overline{V_i/x_i}, V/x, (rec\ f(x : \tau[\overline{\sigma_j/\alpha_j}]) = M[\overline{V_i/x_i}])/f]\ in\ K \downarrow$.
This follows from $(g_{n+1}v,\ \Delta';;\vdash M[\overline{V_i/x_i}, V/x, rec\ f(x:\tau) = M/f] : T\tau'[\overline{\sigma_j/\alpha_j}]) = (\llbracket \Delta;\Xi;\Gamma' \vdash M : T\tau' \rrbracket(\rho \otimes f \mapsto i \circ in_{\multimap}\lfloor g_n \rfloor \otimes x \mapsto v),\ \Delta';;\vdash M[\overline{V_i/x_i}, V/x, rec\ f(x:\tau) = M/f] : T\tau'[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_M$. This again follows from the definition of $\mathbb{R}_{\Xi\Gamma}$ together with these assumptions : $(\llbracket \Delta;\Xi;\Gamma' \vdash M : T\tau' \rrbracket,\ \Delta;\Xi;\Gamma' \vdash M : T\tau') \in \mathbb{R}_{\Xi\Gamma'\vdash M}$ and $(i \circ in_{\multimap}\lfloor g_n \rfloor,\ \Delta';;\vdash rec\ f(x : \tau[\overline{\sigma_j/\alpha_j}]) = M[\overline{V_i/x_i}] : (\tau[\overline{\sigma_j/\alpha_j}] \to T\tau'[\overline{\sigma_j/\alpha_j}])) \in \mathbb{R}_V$ and $(v,\ \Delta'';;\vdash V : \tau[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_V$ where we use weakening for ρ and f.

So it holds that $\forall n \in \omega.(i \circ in_{\multimap}\lfloor g_n \rfloor,\ \Delta';;\vdash rec\ f(x : \tau[\overline{\sigma_j/\alpha_j}]) = M[\overline{V_i/x_i}] : (\tau[\overline{\sigma_j/\alpha_j}] \to T\tau'[\overline{\sigma_j/\alpha_j}])) \in \mathbb{R}_V$, and then by admissibility $([\![rec\ f(x : \tau[\overline{\sigma_j/\alpha_j}]) = M[\overline{V_i/x_i}] : (\tau[\overline{\sigma_j/\alpha_j}] \to T\tau'[\overline{\sigma_j/\alpha_j}])]\!]\rho,\ \Delta';;\vdash rec\ f(x : \tau[\overline{\sigma_j/\alpha_j}]) = M[\overline{V_i/x_i}] : (\tau[\overline{\sigma_j/\alpha_j}] \to T\tau'[\overline{\sigma_j/\alpha_j}])) \in \mathbb{R}_V$. We conclude that $([\![\Delta;\Xi;\Gamma \vdash rec\ f(x : \tau) = M : \tau \to T\tau']\!],\ \Delta;\Xi;\Gamma \vdash rec\ f(x : \tau) = M : \tau \to T\tau') \in \mathbb{R}_{\Xi\Gamma\vdash V}$.

(val) Assume $([\![\Delta;\Xi;\Gamma \vdash V : \tau]\!],\ \Delta;\Xi;\Gamma \vdash V : \tau) \in \mathbb{R}_{\Xi\Gamma\vdash V}$.
We want to show that $([\![\Delta;\Xi;\Gamma \vdash val\ V : T\tau]\!],\ \Delta;\Xi;\Gamma \vdash val\ V : T\tau) \in \mathbb{R}_{\Xi\Gamma\vdash M}$. This requires $([\![\Delta;\Xi;\Gamma \vdash val\ V : T\tau]\!](\rho),\ \Delta';;\vdash val\ V[\overline{V_i/x_i}] : T\tau[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_M$, and this again requires $\forall$ closed $\Delta'' \supseteq \Delta'.\forall(k, \Delta'' \vdash K : (x : \tau)^\top) \in \mathbb{R}_K.\forall(s, \Sigma : \Delta'') \in \mathbb{R}_S.[\![\Delta;\Xi;\Gamma \vdash val\ V : T\tau]\!](\rho)ks = \top \Rightarrow \Sigma,\ let\ x \Leftarrow val\ V\ in\ K \downarrow$. Let closed $\Delta'' \supseteq \Delta', (k, \Delta'' \vdash K : (x : \tau)^\top) \in \mathbb{R}_K, (s, \Sigma : \Delta'') \in \mathbb{R}_S$. $[\![\Delta;\Xi;\Gamma \vdash val\ V : T\tau]\!](\rho)ks = ks[\![\Delta;\Xi;\Gamma \vdash V : \tau]\!](\rho)$.
We now have by assuptions, weakening and definition of relatedness for continuationstyped that $kS[\![\Delta;\Xi;\Gamma \vdash V : \tau]\!](\rho) = \top \Longrightarrow \Sigma,\ let\ x \Leftarrow val\ V\ in\ K \downarrow$.

(Λ) Assume $([\![\Delta;\Xi,\alpha;\Gamma \vdash M : T\tau]\!],\ \Delta;\Xi,\alpha;\Gamma \vdash M : T\tau) \in \mathbb{R}_{\Xi\alpha\Gamma\vdash M}$ and $\Xi \vdash \Gamma$. Then because $\Xi \vdash \Gamma$ it holds that α does not occur in the types τ_i in Γ. Also, by the assumption, for all instantiations with closed value types $\sigma_1 \ldots \sigma_m, \sigma_\alpha$ and $\Delta' \supseteq \Delta$ and ρ then $([\![\Delta;\Xi,\alpha;\Gamma \vdash M : T\tau]\!](\rho),\ \Delta';;\vdash M[\overline{V_i/x_i}] : T(\tau[\overline{\sigma_j/\alpha_j}, \sigma_\alpha/\alpha]) \in \mathbb{R}_M$.
We want to show $[\![\Delta;\Xi;\Gamma \vdash \Lambda\alpha.M : \forall\alpha.T\tau]\!],\ \Delta;\Xi;\Gamma \vdash \Lambda\alpha.M : \forall\alpha.T\tau) \in \mathbb{R}_{\Xi\Gamma\vdash V}$. This requires $\forall\sigma_1 \ldots \sigma_m$ and ρ it holds that $[\![\Delta;\Xi;\Gamma \vdash \Lambda\alpha.M : \forall\alpha.T\tau]\!](\rho),\ \Delta';;\vdash \Lambda\alpha.M[\overline{V_i/x_i}] : \forall\alpha.T(\tau[\overline{\sigma_j/\alpha_j}])) \in \mathbb{R}_V$.
We have that $[\![\Delta;\Xi;\Gamma \vdash \Lambda\alpha.M : \forall\alpha.T\tau]\!](\rho) = in_\forall\lfloor[\![\Delta;\Xi;\Gamma \vdash M : T\tau]\!](\rho)\rfloor$.
We need to show that $\exists\Delta_0 \subseteq \Delta'.\forall$ *closed* σ_α. $([\![\Delta;\Xi,\alpha;\Gamma \vdash M : T\tau]\!](\rho),\ \Delta_0;;\vdash M[\overline{V_i/x_i}] : \tau[\overline{\sigma_j/\alpha_j}, \sigma_\alpha/\alpha]) \in \mathbb{R}_M$. With $\Delta_0 = \Delta'$ this holds by assumption.

(alloc) Assume $([\![\Delta;\Xi;\Gamma \vdash V : \tau]\!],\ \Delta;\Xi;\Gamma \vdash V : \tau) \in \mathbb{R}_{\Xi\Gamma\vdash V}$, so $([\![\Delta;\Xi;\Gamma \vdash V : \tau]\!](\rho),\ \Delta;;\vdash V[\overline{V_i/x_i}] : \tau[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_{\Xi\Gamma\vdash V}$.
We want to show that $([\![\Delta;\Xi;\Gamma \vdash ref\ V : T(\tau\ ref)]\!],\ \Delta;\Xi;\Gamma \vdash ref\ V : T(\tau\ ref)) \in \mathbb{R}_{\Gamma\vdash M}$. This requires $([\![\Delta;\Xi;\Gamma \vdash ref\ V : T(\tau\ ref)]\!](\rho),\ \Delta;;\vdash ref\ V[\overline{V_i/x_i}] : T(\tau[\overline{\sigma_j/\alpha_j}]\ ref)) \in \mathbb{R}_M$, and this again requires that $\forall$ closed $\Delta'' \supseteq \Delta'.\forall(k,\ \Delta'' \vdash K : (x : \tau[\overline{\sigma_j/\alpha_j}])^\top) \in \mathbb{R}_K.\forall(s,\ \Sigma : \Delta'') \in \mathbb{R}_S.[\![\Delta;\Xi;\Gamma \vdash ref\ V : T(\tau\ ref)]\!](\rho)ks = \top \Rightarrow \Sigma, let\ x \Leftarrow ref\ V[\overline{V_i/x_i}]\ in\ K \downarrow$.

Let closed $\Delta'' \supseteq \Delta', (k, \Delta'';\vdash K : (x : \tau[\overline{\sigma_j/\alpha_j}])^\top) \in \mathbb{R}_K, (s, \Sigma : \Delta'') \in \mathbb{R}_S$.
$[\![\Delta;\Xi;\Gamma \vdash ref\ V : T(\tau\ ref)]\!](\rho)ks = k(S[l \mapsto [\![\Delta;\Xi;\Gamma \vdash V : \tau]\!](\rho)])in_{\mathbb{L}}l$
for some/any $l \notin supp(\lambda l'.k(S[l' \mapsto [\![\Delta;\Xi;\Gamma \vdash V : \tau]\!](\rho)])in_{\mathbb{L}}l')$. Choose $l_1 \notin supp(\lambda l'.k(S[l' \mapsto [\![\Delta;\Xi;\Gamma \vdash V : \tau]\!](\rho)])in_{\mathbb{L}}l') \cup dom(\Delta'') \cup locs(\Sigma) \cup locs(K) \cup locs(V)$.
We have $(in_{\mathbb{L}}l_1,\ ;(\Delta'' \cup \{l_1 \mapsto \tau[\overline{\sigma_j/\alpha_j}]\});\vdash l_1 : \tau[\overline{\sigma_j/\alpha_j}]\ ref) \in \mathbb{R}_V$ and by weakening for the stored values $(S[l_1 \mapsto [\![\Delta;\Xi;\Gamma \vdash V : \tau]\!](\rho)],\ \Sigma[l_1 \mapsto V[\overline{V_i/x_i}]] : \Delta'' \cup \{l_1 \mapsto \tau[\overline{\sigma_j/\alpha_j}]\}) \in \mathbb{R}_S$.
Hence $[\![\Delta;\Xi;\Gamma \vdash ref\ V : T(\tau\ ref)]\!](\rho)ks = k(S[l_1 \mapsto [\![\Delta;\Xi;\Gamma \vdash V : \tau]\!](\rho)])in_{\mathbb{L}}l_1 = \top \Longrightarrow$
$\Sigma[l_1 \mapsto V[\overline{V_i/x_i}]],\ let\ x \Leftarrow val\ l_1\ in\ K \downarrow$. By the rules for judgement of termination this implies that $\Sigma,\ let\ x \Leftarrow ref\ V[\overline{V_i/x_i}]]\ in\ K \downarrow$.

(deref) Assume $([\![\Delta;\Xi;\Gamma \vdash V : \tau\ ref]\!],\ \Delta;\Xi;\Gamma \vdash V : \tau\ ref) \in \mathbb{R}_{\Xi\Gamma\vdash V}$. Then $([\![\Delta;\Xi;\Gamma \vdash V : \tau\ ref]\!](\rho),\ \Delta;;\vdash V[\overline{V_i/x_i}] : \tau[\overline{\sigma_j/\alpha_j}]\ ref) \in \mathbb{R}_V$. So $\rho \neq \bot$ then exists $l \in dom(\Delta')$ with $V[\overline{V_i/x_i}] = l$ and $\Delta' l = \tau[\overline{\sigma_j/\alpha_j}]$. When $[\![\Delta;\Xi;\Gamma \vdash V : \tau\ ref]\!](\rho) = in_{\mathbb{L}}\lfloor l \rfloor$.
We want to show that $([\![\Delta;\Xi;\Gamma \vdash !V : T\tau]\!],\ \Delta;\Xi;\Gamma \vdash !V : T\tau) \in \mathbb{R}_{\Gamma\vdash M}$, that is $([\![\Delta;\Xi;\Gamma \vdash !V : T\tau]\!](\rho),\ \Delta';\vdash !V[\overline{V_i/x_i}] : T\tau[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_M$. This again requires $\forall$ closed $\Delta'' \supseteq \Delta'.\forall(k,\ \Delta'' \vdash K : (x : \tau[\overline{\sigma_j/\alpha_j}])^\top) \in \mathbb{R}_K.\forall(s,\ \Sigma : \Delta'') \in \mathbb{R}_S.[\![\Delta;\Xi;\Gamma \vdash !V : T\tau]\!](\rho)ks = \top \Rightarrow \Sigma, let\ x \Leftarrow\ !V[\overline{V_i/x_i}]\ in\ K \downarrow$.
Let $\Delta'' \supseteq \Delta', (k, \Delta'';\vdash K : (x : \tau[\overline{\sigma_j/\alpha_j}])^\top) \in \mathbb{R}_K, (s, \Sigma : \Delta'') \in \mathbb{R}_S$.
Either $[\![\Delta;\Xi;\Gamma \vdash !V : T\tau]\!](\rho)ks = ks(sl)$ or else $[\![\Delta;\Gamma \vdash !V : T\tau]\!](\rho) = \bot$.

Since by assumption $(s, \Sigma : \Delta'') \in \mathbb{R}_S$ and $l \in dom(\Delta') \subseteq dom(\Delta'')$ then it holds that $(sl, \Delta''; \vdash \Sigma l : \tau[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_V$. Since $(k, \Delta''; \vdash K : (x : \tau[\overline{\sigma_j/\alpha_j}])^{\top}) \in \mathbb{R}_K$ it follows that
$ks(sl) = \top \Longrightarrow (\Sigma,\ let\ x \Leftarrow val(\Sigma l)\ in\ K \downarrow)$.
By judgement of termination then also $(\Sigma,\ let\ x \Leftarrow !l\ in\ K \downarrow)$

(assign) Assume $([\![\Delta; \Xi; \Gamma \vdash V_a : \tau\ ref]\!],\ \Delta; \Xi; \Gamma \vdash V_a : \tau\ ref) \in \mathbb{R}_{\Xi\Gamma\vdash V}$ and $([\![\Delta; \Xi; \Gamma \vdash V_b : \tau]\!],\ \Delta; \Xi; \Gamma \vdash V_b : \tau) \in \mathbb{R}_{\Xi\Gamma\vdash V}$. Then $([\![\Delta; \Xi; \Gamma \vdash V_a : \tau\ ref]\!](\rho),\ \Delta';; \vdash V_a[\overline{V_i/x_i}] : \tau[\overline{\sigma_j/\alpha_j}]\ ref) \in \mathbb{R}_V$ and $([\![\Delta; \Xi; \Gamma \vdash V_b : \tau]\!](\rho),\ \Delta';; \vdash V_b[\overline{V_i/x_i}] : \tau[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_V$. This again implies that when $\rho \neq \bot$ then exists $l \in dom(\Delta').V_a[\overline{V_i/x_i}] = l$ and $\Delta' l = \tau[\overline{\sigma_j/\alpha_j}]$ and $[\![\Delta; \Xi; \Gamma \vdash V_a : \tau\ ref]\!](\rho) = in_{\mathbb{L}}\lfloor l \rfloor$.
We want to show that $([\![\Delta; \Xi; \Gamma \vdash V_1 := V_2 : Tunit]\!],\ \Delta; \Xi; \Gamma \vdash V_1 := V_2 : Tunit) \in \mathbb{R}_{\Xi\Gamma\vdash M}$, that is $([\![\Delta; \Xi; \Gamma \vdash V_1 := V_2 : Tunit]\!](\rho),\ \Delta'; \vdash (V_1 := V_2)[\overline{V_i/x_i}] : Tunit) \in \mathbb{R}_M$.
Let $\Delta'' \supseteq \Delta', (k, \Delta''; \vdash K : (x : unit)^{\top}) \in \mathbb{R}_K, (s, \Sigma : \Delta'') \in \mathbb{R}_S$.
Assume $[\![\Delta; \Xi; \Gamma \vdash V_1 := V_2 : Tunit]\!](\rho)ks = \top$. We have $[\![\Delta; \Xi; \Gamma \vdash V_1 := V_2 : Tunit]\!](\rho)ks = k(s[l \mapsto [\![\Delta; \Xi; \Gamma \vdash V_2 : \tau]\!](\rho) in_{\mathbf{1}}\lfloor * \rfloor$.
Since by assumption $(s, \Sigma : \Delta'') \in \mathbb{R}_S$ and by weakening $([\![\Delta; \Xi; \Gamma \vdash V_2 : \tau]\!](\rho),\ \Delta''; \vdash V_2[\overline{V_i/x_i}] : \tau[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_{\Gamma\vdash V}$ and $l \in dom(\Delta') \subseteq dom(\Delta'')$ then it holds that $(s[l \mapsto [\![\Delta; \Xi; \Gamma \vdash V_2 : \tau]\!](\rho)], \Sigma[l \mapsto V_2[\overline{V_i/x_i}]] : \Delta'') \in \mathbb{R}_S$. Since $(k, \Delta''; \vdash K : (x : \tau)^{\top}) \in \mathbb{R}_K$ and $(in_{\mathbf{1}}*, \Delta''; \vdash () : unit) \in \mathbb{R}_V$ it follows that
$k(s[l \mapsto [\![\Delta; \Gamma \vdash V_2 : \tau]\!](\rho)])(in_{\mathbf{1}}\lfloor * \rfloor) = \top \Longrightarrow (\Sigma[l \mapsto V_2[\overline{V_i/x_i}]],\ let\ x \Leftarrow val()\ in\ K \downarrow)$.
By judgement of termination then also $(\Sigma,\ let\ x \Leftarrow l := V_2[\overline{V_i/x_i}]\ in\ K \downarrow)$

(app) Assume $([\![\Delta; \Xi; \Gamma \vdash V_1 : \tau \to T\tau']\!],\ \Delta; \Xi; \Gamma \vdash V_1 : \tau \to T\tau') \in \mathbb{R}_{\Xi\Gamma\vdash V}$ and $([\![\Delta; \Xi; \Gamma \vdash V_2 : \tau]\!],\ \Delta; \Xi; \Gamma \vdash V_2 : \tau) \in \mathbb{R}_{\Xi\Gamma\vdash V}$, so $([\![\Delta; \Xi; \Gamma \vdash V_1 : \tau \to T\tau']\!](\rho),\ \Delta'; \vdash V_1[\overline{V_i/x_i}] : \tau[\overline{\sigma_j/\alpha_j}] \to T\tau'[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_V$ and $([\![\Delta; \Xi; \Gamma \vdash V_2 : \tau]\!](\rho),\ \Delta'; \vdash V_2[\overline{V_i/x_i}] : \tau[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_V$.
We want to show that $([\![\Delta; \Xi; \Gamma \vdash V_1 V_2 : T\tau']\!],\ \Delta; \Xi; \Gamma \vdash V_1 V_2 : T\tau') \in \mathbb{R}_{\Xi\Gamma\vdash M}$, that is $([\![\Delta; \Xi; \Gamma \vdash V_1 V_2 : T\tau']\!](\rho),\ \Delta'; \vdash (V_1 V_2)[\overline{V_i/x_i}] : T\tau'[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_M$. Here $(V_1 V_2)[\overline{V_i/x_i}] = (V_1[\overline{V_i/x_i}])(V_2[\overline{V_i/x_i}])$. Let closed $\Delta'' \supseteq \Delta'$, $(k,\ \Delta'' \vdash K : (y : \tau')^{\top}) \in \mathbb{R}_K$, $(s,\ \Sigma : \Delta'') \in \mathbb{R}_S$.
Assume $[\![\Delta; \Xi; \Gamma \vdash V_1 V_2 : T\tau']\!](\rho)ks = \top$. To show $\Sigma,\ let\ y \Leftarrow (V_1[\overline{V_i/x_i}])(V_2[\overline{V_i/x_i}])\ in\ K \downarrow$.
Since $\Delta; \Xi; \Gamma \vdash V_1 : \tau \to T\tau'$ then it must hold that there exists f, M such that $V_1 = rec\ f(x : \tau) = M$ and it holds that $\Delta; \Xi; \Gamma, x : \tau, f : \tau \to T\tau' \vdash M : T\tau'$. The assumption $[\![\Delta; \Xi; \Gamma \vdash V_1 V_2 : T\tau']\!](\rho)ks = \top$ implies by the definition of the denotational semantics that there exists $v_f,\ v_2$ s.t. $[\![\Delta; \Xi; \Gamma \vdash V_1 : \tau \to T\tau']\!](\rho)ks = i \circ in_{\multimap}\lfloor v_f \rfloor \neq \bot$ and $[\![\Delta; \Xi; \Gamma \vdash V_2 : \tau]\!](\rho) = v_2 \neq \bot$ and $[\![\Delta; \Xi; \Gamma V_1 V_2 : T\tau']\!]\rho = d_f(v_2)$. By IH $(i \circ in_{\multimap}\lfloor v_f \rfloor,\ \Delta'; \vdash V_1[\overline{V_i/x_i}] : \tau[\overline{\sigma_j/\alpha_j}] \to T\tau'[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_V$ and $(v_2,\ \Delta'; \vdash V_2[\overline{V_i/x_i}] : \tau[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_V$. By definition of $\mathbb{R}_V$ then $(v_f v_2,\ \Delta' \vdash (V_1 V_2)[\overline{V_i/x_i}] : T\tau'[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_M$ and so $([\![\Delta; \Xi; \Gamma \vdash V_1 V_2 : T\tau']\!],\ \Delta; \Xi; \Gamma \vdash V_1 V_2 : T\tau') \in \mathbb{R}_{\Xi\gamma\vdash V}$.

(Λapp) Assume $([\![\Delta; \Xi; \Gamma \vdash V : \forall\alpha.T\tau]\!],\ \Delta; \Xi; \Gamma \vdash V : \forall\alpha.T\tau) \in \mathbb{R}_{\Xi\Gamma\vdash V}$.
Then $\forall\sigma_1 \ldots \sigma_m, \Delta' \supseteq \Delta$ and $(v_i,\ \Delta';; \vdash V_i : \tau_i[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_V$ with $\rho = \bigotimes_i v_i$ it holds that $([\![\Delta; \Xi; \Gamma \vdash V : \forall\alpha.T\tau]\!](\rho),\ \Delta';; \vdash V[\overline{V_i/x_i}] : \forall\alpha.T\tau[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_V$. When $[\![\Delta; \Xi; \Gamma \vdash V : \forall\alpha.T\tau]\!](\rho) \neq \bot$ then exists $d \in \mathbb{M}$. $[\![\Delta; \Xi; \Gamma \vdash V]\!](\rho) = in_{\forall}\lfloor d \rfloor$ and exists M s.t. $V[\overline{V_i/x_i}] = \Lambda\alpha.M$. Also $\exists\Delta_0 \subseteq \Delta.\forall\tau^0\ with\ {}_{-} \vdash \tau^0 : type.\ (d,\ \Delta_0;; \vdash M : T\tau[\overline{\sigma_j/\alpha_j}, \tau^0/\alpha]) \in \mathbb{R}_M$.

We want to show $([\![\Delta; \Xi; \Gamma \vdash V\tau' : T\tau[\tau'/\alpha]]\!],\ \Delta; \Xi; \Gamma \vdash V\tau' : T\tau[\tau'/\alpha]) \in \mathbb{R}_{\Xi\Gamma\vdash M}$.
This requires $\forall\sigma_1 \ldots \sigma_m, \Delta' \supseteq \Delta$ and $(v_i,\ \Delta';; \vdash V_i : \tau_i[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_V$ with $\rho = \bigotimes_i v_i$ it holds that $([\![\Delta; \Xi; \Gamma \vdash V\tau' : T\tau[\tau'/\alpha]]\!](\rho),\ \Delta';; \vdash V[\overline{V_i/x_i}] : (T\tau[\tau'/\alpha])[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_M$.
Now let $\Delta'' \supseteq \Delta'$, $(k,\ \Delta'' \vdash K : (x : \tau[\tau'/\alpha])[\overline{\sigma_j/\alpha_j}])^{\top}) \in \mathbb{R}_K$, $(S,\ \Sigma : \Delta'') \in \mathbb{R}_S$, and assume $[\![\Delta; \Xi; \Gamma \vdash V\tau' : T\tau[\tau'/\alpha]]\!](\rho)kS = \top$. $[\![\Delta; \Xi; \Gamma \vdash V\tau' : T\tau[\tau'/\alpha]]\!](\rho)kS = kSd$.
We need to show $\Sigma,\ let\ x \Leftarrow \Lambda\alpha.M[\overline{V_i/x_i}]\tau'\ in\ K\ \downarrow$, by the termination rules it suffices to show $\Sigma,\ let\ x \Leftarrow M[\overline{V_i/x_i}]\ in\ K\ \downarrow$. This follows from the assumptions.

(unfold) Assume $(\llbracket \Delta;\Xi;\Gamma \vdash V : \mu\alpha.\tau \rrbracket,\ \Delta;\Xi;\Gamma \vdash V : \mu\alpha.\tau) \in \mathbb{R}_{\Xi\Gamma\vdash V}$, so $(\llbracket \Delta;\Xi;\Gamma \vdash V : \mu\alpha.\tau \rrbracket(\rho),\ \Delta';;\vdash V[\overline{V_i/x_i}] : (\mu\alpha.\tau)[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_V$. By the definition of $\mathbb{R}_V$ this can only be the case if either $\llbracket \Delta;\Xi;\Gamma \vdash V : \mu\alpha.\tau \rrbracket(\rho) = \bot$ or else there exist d, W such that $\llbracket \Delta;\Xi;\Gamma \vdash V : \mu\alpha.\tau \rrbracket(\rho) = in_\mu d$ and $V[\overline{V_i/x_i}] = fold\ W$ and further $(d,\ \Delta';\vdash W : (\tau[\mu\alpha.\tau/\alpha])[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_V$. We want to show $(\llbracket \Delta;\Xi;\Gamma \vdash unfold\ V : T(\tau[\mu\alpha.\tau/\alpha])\rrbracket(\rho),\ \Delta';\vdash unfold\ V[\overline{V_i/x_i}] : T(\tau[\mu\alpha.\tau/\alpha])[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_M$. Let $\Delta'' \supseteq \Delta'$, $(k,\ \Delta'';\vdash K : (x : (\tau[\mu\alpha.\tau/\alpha])[\overline{\sigma_j/\alpha_j}])^\top) \in \mathbb{R}_K$, $(S,\ \Sigma : \Delta'') \in \mathbb{R}_S$. It holds that either $\llbracket \Delta;\Xi;\Gamma \vdash unfold\ V : T(\tau[\mu\alpha.\tau/\alpha])\rrbracket(\rho) = \bot$, or else $\llbracket \Delta;\Xi;\Gamma \vdash V : \mu\alpha.\tau \rrbracket(\rho) = i_1 \circ in_\mu d$. In the first case we have $(\bot,\ \Delta';\vdash unfold\ V[\overline{V_i/x_i}] : T(\tau[\mu\alpha.\tau/\alpha])[\overline{\sigma_j/\alpha_j}]\ ) \in \mathbb{R}_M$. In the second case we want to show that $\llbracket \Delta;\Xi;\Gamma \vdash unfold\ V : T(\tau[\mu\alpha.\tau/\alpha])\rrbracket(\rho)kS = \top \implies (\Sigma,\ let\ x \Leftarrow unfold\ V[\overline{V_i/x_i}]\ in\ K \downarrow)$. By the rules for judgements of termination it holds that $\Sigma,\ let\ x \Leftarrow val\ W\ in\ K \downarrow$ implies $\Sigma,\ let\ x \Leftarrow unfold(fold\ W)\ in\ K \downarrow$ which is the same as $\Sigma,\ let\ x \Leftarrow unfold\ V[\overline{V_i/x_i}]\ in\ K \downarrow$. So it suffices to show that $\llbracket \Delta;\Xi;\Gamma \vdash unfold\ V : T(\tau[\mu\alpha.\tau/\alpha])\rrbracket(\rho)kS = \top \implies (\Sigma,\ let\ x \Leftarrow val\ W\ in\ K \downarrow)$.
$\llbracket \Delta;\Xi;\Gamma \vdash unfold\ V : T(\tau[\mu\alpha.\tau/\alpha])\rrbracket(\rho)kS = kSd$.
Since $(k,\ \Delta'';\vdash K : (x : (\tau[\mu\alpha.\tau/\alpha])[\overline{\sigma_j/\alpha_j}])^\top) \in \mathbb{R}_K$, $(S,\ \Sigma : \Delta'') \in \mathbb{R}_S$ and $(d,\ \Delta';\vdash W : (\tau[\mu\alpha.\tau/\alpha])[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_V$ this implies $\Sigma,\ let\ x \Leftarrow val\ W\ in\ K \downarrow$.

(let) Assume $(\llbracket \Delta;\Xi;\Gamma \vdash M_1 : T\tau_1 \rrbracket,\ \Delta;\Xi;\Gamma \vdash M_1 : T\tau_1) \in \mathbb{R}_{\Xi\Gamma\vdash M}$ and $(\llbracket \Delta;\Xi;\Gamma, x:\tau_1 \vdash M_2 : T\tau_2 \rrbracket,\ \Delta;\Xi;\Gamma, x:\tau_1 \vdash M_2 : T\tau_2) \in \mathbb{R}_{\Xi\Gamma,x:\tau_1\vdash M}$.
So $(\llbracket \Delta;\Xi;\Gamma \vdash M_1 : T\tau_1 \rrbracket(\rho),\ \Delta';\vdash M_1[V_i/x_i] : T\tau_1[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_M$. We want to show that $(\llbracket \Delta;\Xi;\Gamma \vdash let\ x \Leftarrow M_1\ in\ M_2 : T\tau_2 \rrbracket,\ \Delta;\Xi;\Gamma \vdash let\ x \Leftarrow M_1\ in\ M_2 : T\tau_2) \in \mathbb{R}_{\Xi\Gamma\vdash M}$, that is $(\llbracket \Delta;\Xi;\Gamma \vdash let\ x \Leftarrow M_1\ in\ M_2 : T\tau_2 \rrbracket(\rho),\ \Delta';\vdash let\ x \Leftarrow M_1[\overline{V_i/x_i}]\ in\ M_2[\overline{V_i/x_i}] : T\tau_2[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_M$. Let $\Delta'' \supseteq \Delta'$, $(k, \Delta'';\vdash K : (y : \tau_2[\overline{\sigma_j/\alpha_j}])^\top) \in \mathbb{R}_K$, $(S,\ \Sigma : \Delta'') \in \mathbb{R}_S$. Then we want to show that $\big(\llbracket \Delta;\Xi;\Gamma \vdash let\ x \Leftarrow M_1\ in\ M_2 : T\tau_2 \rrbracket(\rho)kS = \top\big) \Rightarrow \big(\Sigma,\ let\ y \Leftarrow (let\ x \Leftarrow M_1[\overline{V_i/x_i}]\ in\ M_2[\overline{V_i/x_i}])\ in\ K) \downarrow\big)$. By rules for judgement of termination it suffices to show that $\Sigma,\ let\ x \Leftarrow M_1[\overline{V_i/x_i}]\ in\ (let\ y \Leftarrow M_2[\overline{V_i/x_i}]\ in\ K) \downarrow$.
Assume $\llbracket \Delta;\Xi;\Gamma \vdash let\ x \Leftarrow M_1\ in\ M_2 : T\tau_2 \rrbracket(\rho)kS = \top$. Then $\llbracket \Delta;\Xi;\Gamma \vdash M_1 : T\tau_1 \rrbracket(\rho)$ $(\lambda S^0.\lambda d^0.\llbracket \Delta;\Xi;\Gamma, x:\tau_1 \vdash M_2 : T\tau_2 \rrbracket(\rho, d^0)kS^0)S = \top$. We have that $(S,\ \Sigma : \Delta'') \in \mathbb{R}_S$ and by weakening $(\llbracket \Delta;\Xi;\Gamma \vdash M_1 : T\tau_1 \rrbracket(\rho),\ \Delta'';\vdash M_1[V_i/x_i] : T\tau_1[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_M$. So it is enough to show $(\lambda S^0.\lambda d^0.\llbracket \Delta;\Xi;\Gamma, x:\tau_1 \vdash M_2 : T\tau_2 \rrbracket(\rho, d^0)kS^0),\ \Delta'';\vdash (let\ y \Leftarrow M_2[\overline{V_i/x_i}]\ in\ K) : (x : \tau_1[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_K$. Let $\Delta^2 \supseteq \Delta''$, $(s,\ \Sigma^2 : \Delta^2) \in \mathbb{R}_S$ and $(w,\ \Delta^2;\vdash W : \tau_1[\overline{\sigma_j/\alpha_j}]) \in \mathbb{R}_V$ then $(\lambda S^0.\lambda d^0.\llbracket \Delta;\Xi;\Gamma, x:\tau_1 \vdash M_2 : T\tau_2 \rrbracket(\rho, d^0)kS^0)sw = \llbracket \Delta;\Xi;\Gamma, x:\tau_1 \vdash M_2 : T\tau_2 \rrbracket(\rho, w)ks$. Assume this equals $\top$, then we want to show that $\Sigma^2,\ let\ y \Leftarrow M_2[\overline{V_i/x_i}, W/x]\ in\ K \downarrow$. This follows using weakening from the assumption $(\llbracket \Delta;\Xi;\Gamma, x:\tau_1 \vdash M_2 : T\tau_2 \rrbracket,\ \Delta;\Xi;\Gamma, x:\tau_1 \vdash M_2 : T\tau_2) \in \mathbb{R}_{\Xi\Gamma,x:\tau_1\vdash M}$.

We have omitted the proof for the rules (+ elim), (× elim), (eq), (arith) and (iszero). □

A.3 Proof of lemma 19

Lemma: The action of F preserves downwards closure.

For all $R^+, R^- \in \mathcal{R}(\mathbb{D})$.
If R^+ is downwards closed, then $F(R^-, R^+)$ is downwards closed.

Proof

- $F(R^-, R^+)_K$: Follows from $k'' \sqsubseteq k' \Rightarrow \forall s, v.\ k''sv \sqsubseteq k'sv$, (independant of R^+).
- $F(R^-, R^+)_M$: Follows from $m'' \sqsubseteq m' \Rightarrow \forall k, s.\ m''ks \sqsubseteq m'ks$, (independant of R^+).
- $F(R^-, R^+)_V$: The proof is by cases of the constuctor for the type parameter:

- (unit),(int),(σ ref): immidiate.
- $(+)$,$(\times)$,(μ): follows from downwards closure of R_V^+. We show the proof for sum types, the other proofs are similar.
  $(+)$: Assume $(v_1', v_2' \parallel v_1, v_2, \tau_1 + \tau_2, p) \in F(R^-, R^+)_V$, and $v_1' \neq \bot \ \vee \ v_2' \neq \bot$. Then exists d_1', d_2', d_1, d_2 such that $(d_1', d_2' \parallel d_1, d_2, \tau_j, p) \in R_V^+$ and $v_1 = in_\oplus in_j(d_1) \neq \bot,\ v_2 = in_\oplus in_j(d_2) \neq \bot$ and $v_1' = d_1' = \bot \vee v_1' = in_\oplus in_j(d_1') \neq \bot,\ v_2' = d_2' = \bot \vee v_2' = in_\oplus in_j(d_2') \neq \bot$. Let $v_1'' \sqsubseteq v_1' \ \wedge \ v_2'' \sqsubseteq v_2'$. Then it holds that $v_1'' = \bot \vee \exists e_1''.v_1'' = in_\oplus in_j(e_1'') \neq \bot \wedge e_1'' \sqsubseteq d_1'$ and $v_2'' = \bot \vee \exists e_2''.v_1'' = in_\oplus in_j(e_2'') \neq \bot \wedge e_2'' \sqsubseteq d_2'$. Let d_1'', d_2'' be given by $if\ v_1'' = \bot\ then\ d_1'' = \bot \wedge if\ v_1'' = in_\oplus in_j(e_1'')\ then\ d_1'' = e_1''$ and $if\ v_2'' = \bot\ then\ d_2'' = \bot \wedge if\ v_2'' = in_\oplus in_j(e_2'')\ then\ d_2'' = e_2''$. Then $d_1'' \sqsubseteq d_1'$ and $d_2'' \sqsubseteq d_2'$. By downwards closure of R_V^+ then $(d_1'', d_2'' \parallel d_1, d_2, \tau_j, p) \in R_V^+$. And so $(v_1'', v_2'' \parallel v_1, v_2, \tau_1 + \tau_2, p) \in F(R^-, R^+)_V$.
- $(\forall)$: follows from downwards closure of R_M^+. Assume $(v_1', v_2' \parallel v_1, v_2, \forall\alpha.T\tau, p) \in F(R^-, R^+)_V$, and $v_1' \neq \bot \ \vee \ v_2' \neq \bot$. Then exists d_1', d_2', d_1, d_2 such that $v_1 = in_\forall\lfloor d_1 \rfloor$ and $v_2 = in_\forall\lfloor d_2 \rfloor$ and $\forall \vdash \sigma : type.(d_1', d_2' \parallel d_1, d_2, T\tau_j[\sigma/\alpha], p) \in R_M^+$. Also $(v_1' = \bot \vee d_1' = \bot) \vee v_1' = in_\forall\lfloor d_1' \rfloor$ and $(v_2' = \bot \vee d_2' = \bot) \vee v_2' = in_\forall\lfloor d_2' \rfloor$. Let $v_1'' \sqsubseteq v_1' \ \wedge \ v_2'' \sqsubseteq v_2'$. Then it holds that $v_1'' = \bot \vee \exists e_1''.v_1'' = in_\forall\lfloor e_1'' \rfloor \wedge e_1'' \sqsubseteq d_1'$ and $v_2'' = \bot \vee \exists e_2''.v_2'' = in_\forall\lfloor e_2'' \rfloor \wedge e_2'' \sqsubseteq d_2'$. Let d_1'', d_2'' be given by $if\ v_1'' = \bot\ then\ d_1'' = \bot \wedge if\ v_1'' = in_\forall\lfloor e_1'' \rfloor\ then\ d_1'' = e_1''$ and $if\ v_2'' = \bot\ then\ d_2'' = \bot \wedge if\ v_2'' = in_\forall\lfloor e_2'' \rfloor\ then\ d_2'' = e_2''$. Then $d_1'' \sqsubseteq d_1'$ and $d_2'' \sqsubseteq d_2'$. By downwards closure of R_M^+ then $\forall \vdash \sigma : type.(d_1'', d_2'' \parallel d_1, d_2, T\tau_j[\sigma/\alpha], p) \in R_M^+$. And so $(v_1'', v_2'' \parallel v_1, v_2, \forall\alpha.T\tau, p) \in F(R^-, R^+)_V$.
- $(\rightarrow)$: follows from downwards closure of R_M^+. Assume $(v_1', v_2' \parallel v_1, v_2, \tau \rightarrow T\tau', p) \in F(R^-, R^+)_V$, and $v_1' \neq \bot \ \vee \ v_2' \neq \bot$. Then exists d_1', d_2', d_1, d_2 such that $v_1 = in_\multimap\lfloor d_1 \rfloor$ and $v_2 = in_\multimap\lfloor d_2 \rfloor$ and $(v_1' = \bot \wedge d_1' = \bot) \vee v_1' = in_\multimap\lfloor d_1' \rfloor$ and $(v_2' = \bot \wedge d_2' = \bot) \vee v_2' = in_\multimap\lfloor d_2' \rfloor$. Also $\forall p' \blacktriangleright p.\ \forall.(w_1', w_2' \parallel w_1, w_2, \tau, p') \in R_V^-.(d_1'w_1', d_2'w_2' \parallel d_1w_1, d_2w_2, T\tau', p') \in R_M^+$. Let $v_1'' \sqsubseteq v_1' \ \wedge \ v_2'' \sqsubseteq v_2'$. Then it holds that $v_1'' = \bot \vee \exists e_1''.v_1'' = in_\multimap\lfloor e_1'' \rfloor \wedge e_1'' \sqsubseteq d_1'$ and $v_2'' = \bot \vee \exists e_2''.v_2'' = in_\multimap\lfloor e_2'' \rfloor \wedge e_2'' \sqsubseteq d_2'$. Let d_1'', d_2'' be given by $if\ v_1'' = \bot\ then\ d_1'' = \bot \wedge if\ v_1'' = in_\multimap\lfloor e_1'' \rfloor\ then\ d_1'' = e_1''$ and $if\ v_2'' = \bot\ then\ d_2'' = \bot \wedge if\ v_2'' = in_\multimap\lfloor e_2'' \rfloor\ then\ d_2'' = e_2''$. Then $d_1'' \sqsubseteq d_1'$ and $d_2'' \sqsubseteq d_2'$ and so $\forall u_1, u_2.d_1''u_1 \sqsubseteq d_1'u_1 \wedge d_2''u_2 \sqsubseteq d_2'u_2$. By downwards closure of R_M^+ then $\forall p' \blacktriangleright p.\ \forall.(w_1', w_2' \parallel w_1, w_2, \tau, p') \in R_V^-.(d_1''w_1', d_2''w_2' \parallel d_1w_1, d_2w_2, T\tau', p') \in R_M^+$. So $(v_1'', v_2'' \parallel v_1, v_2, \tau \rightarrow T\tau', p) \in F(R^-, R^+)_V$.

– $F(R^-, R^+)_S$: follows from downwards closure of R_V^+. Assume $(s_1', s_2' \parallel s_1, s_2, p) \in F(R^-, R^+)_S$, and $s_1' \neq \bot \ \vee \ s_2' \neq \bot$. Let $s_1'' \sqsubseteq s_1' \wedge s_2'' \sqsubseteq s_2'$. Then $\forall l.\ s_1''l \sqsubseteq s_1'l \wedge s_2''l \sqsubseteq s_2'l$. We want to show $(s_1'', s_2'' \parallel s_1, s_2, p) \in F(R^-, R^+)_S$. The requirements conserning disjointness as well as requirements about belongings to finitary state relations and finitary state predicates are only stated on s_1, s_2 and hence follow from the assumptions. Requirements conserning that stored values are related follow from assumptions together with downwards closure of R_V^+.

$\square$

A.4 Proof of lemma 20

Lemma: The action of F on adm$^+$relations preserves admissibility.

For all $R^+, R^- \in \mathcal{R}(\mathbb{D})$. If R^+ is adm$^+$, then $F(R^-, R^+)$ is admissible.

Proof Assume R^+ is adm$^+$, we want to show $F(R^-, R^+)$ is admissible for all $R^- \in \mathcal{R}(\mathbb{D})$. By definition each of the four projections of $F(R^-, R^+)$ includes $(\bot, \bot \parallel (type), d_1, d_2, p)$ for all $(type), d_1, d_2, p$. To show that $F(R^-, R^+)$ is admissible it suffices to show for each of the four projections, that it is closed under least upper bounds of finitely supported chains of the form $(d_1^i, d_2^i \parallel (type), d_1, d_2, p)_{i\in\omega}$ where $type, d_1, d_2, p$ are constant.

- $F(R^-, R^+)_S =$

$$
\begin{array}{l}
\{(\bot,\ \bot \parallel S_1,\ S_2,\ (pks)) \mid (pks) \in \mathfrak{p}^s \text{ the set of all s-parameters } \} \cup \\
\{(S'_1,\ S'_2 \parallel\ S_1,\ S_2,\ (pks) \mid (pks) = \{(r_1|q_1|Q_1), \ldots, (r_n|q_n|Q_n)\} \in \mathfrak{p}^s \ \wedge \\
\quad S'_1 \sqsubseteq S_1 \neq \bot \ \ \wedge\ S'_2 \sqsubseteq S_2 \neq \bot\ \wedge \\
\quad \mathcal{A}_1^{(pks)}(S_1) \cap \pi_1(Z^{(pks)}) = \emptyset\ \wedge\ \mathcal{A}_2^{(pks)}(S_2) \cap \pi_2(Z^{(pks)}) = \emptyset\ \wedge \\
\quad \forall i \neq j.\ \mathring{A}_1^{ri}(S_1) \cap \mathring{A}_1^{rj}(S_1) = \emptyset\ \wedge\ \mathring{A}_2^{ri}(S_2) \cap \mathring{A}_2^{rj}(S_2) = \emptyset\ \wedge \\
\quad \forall (l_1, l_2, \tau) \in Z^{(pks)}.(S'_1(l_1),\ S'_2(l_2) \parallel S_1(l_1),\ S_2(l_2),\ \tau,\ (pks)^{vm}) \in R_V^+\ \wedge \\
\quad \forall i \in 1..n.\ \text{if } Q_i = (P_i, LL_i) \text{ then } (S_1, S_2) \in P_i\ \wedge \\
\qquad \forall (l_1, l_2, \tau) \in LL_i.\ (S'_1(l_1),\ S'_2(l_2) \parallel S_1(l_1),\ S_2(l_2),\ \tau,\ (pks)^{vm}) \in R_V^+ \ \}
\end{array}
$$

Assume a finitely supported chain $(S_1^i, S_2^i \parallel S_1, S_2, p)_{i\in\omega}$ in $F(R^-, R^+)_S$, we will show its least upper bound is in $F(R^-, R^+)_S$. If the chain is constantly $(\bot, \bot,\ \parallel S_1, S_2, p)$ we are done.

Else it holds that $S_1 \neq \bot\ \wedge\ S_2 \neq \bot$ and $\forall i.\ S_1^i \sqsubseteq S_1\ \wedge\ S_2^i \sqsubseteq S_2$. Then it holds that $\bigsqcup S_1^i \sqsubseteq S_1\ \wedge\ \bigsqcup S_2^i \sqsubseteq S_2$.

$\mathcal{A}_1^{(pks)}(S_1) \cap \pi_1(Z^{(pks)}) = \emptyset\ \wedge\ \mathcal{A}_2^{(pks)}(S_2) \cap \pi_2(Z^{(pks)}) = \emptyset$ and

$\forall k \neq j.\ \mathring{A}_1^{rk}(S_1) \cap \mathring{A}_1^{rj}(S_1) = \emptyset\ \wedge\ \mathring{A}_2^{rk}(S_2) \cap \mathring{A}_2^{rj}(S_2) = \emptyset$ as in each step.

When $(S_1^i, S_2^i \parallel S_1, S_2, p)_{i\in\omega}$ is a chain, then $\forall l.\ (S_1^i l)_{i\in\omega}$ is a chain and $(S_2^i l)_{i\in\omega}$ is a chain. Since R_V^+ is admissible and $\forall i.\forall (l_1, l_2, \tau) \in Z^{(pks)}.(S_1^i l_1\ S_2^i l_2 \parallel S_1 l_1,\ S_2 l_2,\ \tau, (pks)^{vm}) \in R_V^+$ and these are chains in R_V^+, then also $\forall (l_1, l_2, \tau) \in Z^{(pks)}.\ \bigsqcup((S_1^i)l_1, (S_2^i)l_2 \parallel S_1 l_1,\ S_2 l_2,\ \tau, (pks)^{vm}) = ((\bigsqcup S_1^i)l_1,\ (\bigsqcup S_2^i)l_2 \parallel S_1 l_1,\ S_2 l_2,\ \tau, (pks)^{vm}) \in R_V^+$.

$\forall k \in 1..n.$ if $Q_k = (P_k, LL_k)$ then $(S_1, S_2) \in P_k$ holds as in each step.

Since $\forall i.\ \forall k \in 1..n.$ if $Q_k = (P_k, LL_k)$ then $\forall (l_1, l_2, \tau) \in LL_k.\ (S_1^i(l_1),\ S_2^i(l_2) \parallel S_1(l_1),\ S_2(l_2),\ \tau,\ (pks)^{vm}) \in R_V^+$ and these are chains in the admissible relation R_V^+, then also $\forall k \in 1..n.$ if $Q_k = (P_k, LL_k)$ then $\forall (l_1, l_2, \tau) \in LL_k.\ ((\bigsqcup S_1^i)l_1,\ (\bigsqcup S_2^i)l_2 \parallel S_1 l_1,\ S_2 l_2,\ \tau,\ (pks)^{vm}) \in R_V^+$.

We conclude that $F(R^-, R^+)_S$ is closed under least upper bounds of chains.

- $F(R^-, R^+)_M =$

$$
\begin{array}{l}
\{(m'_1,\ m'_2 \parallel m_1,\ m_2,\ T\tau,\ p) \mid p \in \mathfrak{p}^{vm}\ \wedge \\
\quad m'_1 \sqsubseteq m_1\ \wedge\ m'_2 \sqsubseteq m_2\ \wedge \\
\quad \forall p' \blacktriangleright p. \forall (pk') \in p'^{\mathbf{K}}. \\
\quad \forall (pks') \in (pk')^{\mathbf{S}}. \\
\qquad \forall (k'_1,\ k'_2 \parallel k_1,\ k_2,\ (x:\tau)^\top,\ (pk')) \in R_K^-.\ \forall (S'_1,\ S'_2 \parallel S_1,\ S_2,\ (pks')) \in R_S^-\ . \\
\qquad\quad (m'_1 k'_1 S'_1 = \top \Rightarrow m_2 k_2 S_2 = \top)\ \wedge \\
\qquad\quad (m'_2 k'_2 S'_2 = \top \Rightarrow m_1 k_1 S_1 = \top)\ \}
\end{array}
$$

Assume a finitely supported chain $(m_1^i, m_2^i \parallel m_1, m_2, T\tau, p)_{i\in\omega}$ in $F(R^-, R^+)_M$, we will show its least upper bound is in $F(R^-, R^+)_M$. If the chain is constant $(\bot, \bot \parallel m_1,\ m_2, T\tau, p)$ we are done. Else, since $\forall i.\ m_1^i \sqsubseteq m_1\ \wedge\ m_2^i \sqsubseteq m_2$, then also $\bigsqcup m_1^i \sqsubseteq m_1\ \wedge\ \bigsqcup m_2^i \sqsubseteq m_2$.

Let $p' \blacktriangleright p, (pk') \in p'^{\mathbf{K}},\ (k'_1, k'_2 \parallel\ k_1,\ k_2,\ (x:\tau)^\top,\ (pk)') \in R_K^-$.

Let $(pks') \in (pk')^{\mathbf{S}}$ and $(S'_1, S'_2 \parallel\ S_1,\ S_2,\ (pks')) \in R_S^-$.

Since m_1^i and m_2^i are chains, then also $m_1^i k'_1 S'_1$ and $m_2^i k'_2 S'_2$ are chains in $\mathbb{O}$. If $\forall i. m_1^i k'_1 S'_1 = \bot$ then also $(\bigsqcup m_1^i)k'_1 S'_1 = \bot$ and the implication $(\bigsqcup m_1^i)k'_1 S'_1 = \top \Rightarrow m_2 k_2 S_2 = \top$ holds trivially. Else it must be the case that $\exists j.\forall i \geq j.\ m_1^i k'_1 S'_1 = \top$. This implies both that $m_2 k_2 S_2 = \top$ and $(\bigsqcup m_1^i)k'_1 S'_1 = \top$, so the implication $(\bigsqcup m_1^i)k'_1 S'_1 = \top \Rightarrow m_2 k_2 S_2 = \top$ holds. The proof that $(\bigsqcup m_2^i)k'_2 S'_2 = \top \Rightarrow m_1 k_1 S_1 = \top$ holds is similar.

We conclude that $F(R^-, R^+)_M$ is closed under lubs of chains.

- $F(R^-, R^+)_K =$

$\{(k_1',\ k_2' \parallel k_1,\ k_2,\ (x:\tau)^\top,\ (pk)) \mid (pk) \in \mathfrak{p}^k\ \wedge$
$k_1' \sqsubseteq k_1\ \wedge\ k_2' \sqsubseteq k_2\ \wedge\ \ \forall (pk') \rhd (pk).$
(that is (pk) extended only with *ordinary* local k-parameters or local extensions)
$\forall (pks') \in (pk')^{\mathbf{S}}$ (the set of s-instantiations of p' i.e. choices of $\bar{\vee}$-clauses),
$\forall (s_1',\ s_2' \parallel s_1,\ s_2,\ (pks')) \in R_S^-.\ \forall (v_1',\ v_2' \parallel v_1,\ v_2, \tau,\ (pk')^{vm}) \in R_V^-.$
$(k_1's_1'v_1' = \top \Rightarrow k_2s_2v_2 = \top)\ \wedge$
$(k_2's_2'v_2' = \top \Rightarrow k_1s_1v_1 = \top)\ \}$

Assume a finitely supported chain $(k_1^i, k_2^i \parallel k_1, k_2, (x:\tau)^\top, (pk))_{i\in\omega}$ in $F(R^-,R^+)_K$, we will show its least upper bound is in $F(R^-,R^+)_K$. If the chain is constant $(\bot, \bot \parallel k_1,\ k_2, (x:\tau)^\top, (pk))$ we are done. Else, since $\forall i.\ k_1^i \sqsubseteq k_1\ \wedge\ k_2^i \sqsubseteq k_2$, then also $\bigsqcup k_1^i \sqsubseteq k_1\ \wedge\ \bigsqcup k_2^i \sqsubseteq k_2$.
Let $(pk') \rhd (pk)$, $(v_1', v_2' \parallel v_1,\ v_2,\ \tau,\ (pk')^{vm}) \in R_V^-$.
Let $(pks') \in (pk')^{\mathbf{S}}$ and $(S_1',\ S_2' \parallel S_1,\ S_2,\ (pks')) \in R_S^-$.
Since k_1^i and k_2^i are chains, then also $k_1^iS_1'v_1'$ and $k_2^iS_2'v_2'$ are chains in $\mathbb{O}$. If $\forall i. k_1^iS_1'v_1' = \bot$ then also $(\bigsqcup k_1^i)S_1'v_1' = \bot$ and the implication $(\bigsqcup k_1^i)S_1'v_1' = \top \Rightarrow k_2S_2v_2 = \top$ holds trivially. Else it must be the case that $\exists j.\forall i \geq j.\ k_1^iS_1'v_1' = \top$. This implies both that $k_2S_2v_2 = \top$ and $(\bigsqcup k_1^i)S_1'v_1' = \top$, so the implication $(\bigsqcup k_1^i)S_1'v_1' = \top \Rightarrow k_2S_2v_2 = \top$ holds. The proof that $(\bigsqcup k_2^i)S_2'v_2' = \top \Rightarrow k_1S_1v_1 = \top$ holds is similar.
We conclude that $F(R^-,R^+)_K$ is closed under lubs of chains.

- $F(R^-,R^+)_V$:
Assume a chain in $F(R^-,R^+)_V$. If the chain is constantly $(\bot,\ \bot \parallel v_1,\ v_2,\ \tau,\ p)$ we are done. Else $v_1 \neq \bot\ \wedge\ v_2 \neq \bot\ \wedge\ \forall i.\ v_1^i \sqsubseteq v_1\ \wedge\ v_2^i \sqsubseteq v_2$. If follows that $\bigsqcup v_1^i \sqsubseteq v_1\ \wedge\ \bigsqcup v_2^i \sqsubseteq v_2$.
Also from some point onwards $v_1^i \neq \bot \vee v_2^i \neq \bot$ and $(v_1^i,\ v_2^i \parallel v_1,\ v_2, \tau, p) \in \tilde{F}(R^-,R^+)$and a chain will have a type parameter. The proof proceeds over cases of the type constructor for the type parameter.
  - Chains of type Unit, Int and references are constant from some point onwards.
  - Chains of sum-types, product-types and recursive μ-types can be proved to be closed under least upper bounds via the admissibility of R_V^+. We show this for chains of sum-type, the other proofs are similar.

    A chain of sum-type which is not constantly $(\bot,\ \bot \parallel v_1,\ v_2,\ \tau_a+\tau_b,\ p)$ must from some point onwards $(\forall i \geq j)$ have the form
    $(v_1^i,\ v_2^i \parallel in_\oplus(in_x(d_1)),\ in_\oplus(in_x(d_2)),\ \tau_a+\tau_b,\ p)$, $x \in \{a,b\}$, where $d_1, d_2 \in \mathbb{V}_\downarrow$ and $\forall i \geq j.\exists d_1^i,\ d_2^i \in \mathbb{V}.\ (v_1^i = d_1^i = \bot \vee v_1^i = in_\oplus(in_x(d_1^i)) \neq \bot) \wedge (v_2^i = d_2^i = \bot \vee v_2^i = in_\oplus(in_x(d_2^i)) \neq \bot)\ \wedge\ (d_1^i \sqsubseteq d_1^{i+1} \wedge d_2^i \sqsubseteq d_2^{i+1})\ \wedge\ (d_1^i,\ d_2^i \parallel d_1,\ d_2,\ \tau_x,\ p) \in R_V^+$, and this is a chain in R_V^+. Then $\bigsqcup d_1^i, \bigsqcup d_2^i \in \mathbb{V}$. Since R^+ is admissible also $(\bigsqcup d_1^i,\ \bigsqcup d_2^i \parallel d_1,\ d_2, \tau_x, p) \in R_V^+$. So, as $in_\oplus(in_x(\bigsqcup d^i)) = \bigsqcup in_\oplus(in_x(d^i)) =$ then
    $\bigsqcup (v_1^i,\ v_2^i \parallel in_\oplus(in_x(d_1)),\ in_\oplus(in_x(d_2)),\ \tau_a+\tau_b,\ p) \in F(R^-,R^+)_V$
  - A chain of $\forall$-type which is not constantly $(\bot,\ \bot \parallel v_1,\ v_2,\ \forall\alpha.\tau,\ p)$ must from some point onwards $(\forall i \geq j)$ have the form $(v_1^i,\ v_2^i \parallel in_\forall\lfloor d_1\rfloor,\ in_\forall\lfloor d_2\rfloor,\ \forall\alpha.T\tau,\ p)$, where $d_1, d_2 \in \mathbb{M}$ and $\forall i \geq j.\exists d_1^i,\ d_2^i \in \mathbb{M}.\ ((v_1^i = \bot \wedge d_1^i = \bot) \vee v_1^i = in_\forall\lfloor d_1^i\rfloor) \wedge ((v_2^i = \bot \wedge d_2^i = \bot) \vee v_2^i = in_\forall\lfloor d_2^i\rfloor)\ \wedge\ (d_1^i \sqsubseteq d_1^{i+1} \wedge d_2^i \sqsubseteq d_2^{i+1})\ \wedge\ \forall\sigma$ with $\vdash \sigma : type.\ (d_1^i, d_2^i \parallel d_1, d_2, T\tau[\sigma/\alpha], p) \in R_M^+$. Then since R^+ is admissible and each of these is a chain also $\forall\sigma$ with $\vdash \sigma : type.\ (\bigsqcup d_1^i, \bigsqcup d_2^i \parallel d_1, d_2, T\tau[\sigma/\alpha], p) \in R_M^+$. So (possibly using downwards closure) $\bigsqcup (v_1^i,\ v_2^i \parallel in_\forall\lfloor d_1\rfloor,\ in_\forall\lfloor d_2\rfloor,\ \forall\alpha.T\tau,\ p) \in F(R^-,R^+)_V$.
  - A chain of function type which is not constantly $(\bot, \bot \parallel v_1,\ v_2,\ \tau \to T\tau',\ p)$ must for all $i \geq j$ have the form $(v_1^i,\ v_2^i \parallel in_\multimap\lfloor d_1\rfloor,\ in_\multimap\lfloor d_2\rfloor,\ \tau \to T\tau',\ p)$, where $d_1, d_2 \in (\mathbb{V} \multimap \mathbb{M})$ and $\forall i \geq j.\exists d_1^i,\ d_2^i \in (\mathbb{V} \multimap \mathbb{M}).\ ((v_1^i = \bot \wedge d_1^i = \bot) \vee v_1^i = in_\multimap\lfloor d_1^i\rfloor) \wedge ((v_2^i = \bot \wedge = d_2^i = \bot) \vee v_2^i = in_\multimap\lfloor d_2^i\rfloor)\ \wedge\ (d_1^i \sqsubseteq d_1^{i+1} \wedge d_2^i \sqsubseteq d_2^{i+1})$ and $\forall p' \blacktriangleright p.\forall (v_1'\ , v_2' \parallel v_1,\ v_2,\ \tau,\ p') \in R_V^-$ it holds $\forall i \geq j.\ (d_1^iv_1',\ d_2^iv_2' \parallel d_1v_1,\ d_2v_2, T\tau',\ p') \in R_M^+$. Since R_M^+ is admissible

and in each case this is a chain, then also $(\bigsqcup(d_1^i v_1'),\ \bigsqcup(d_2^i v_2') \parallel d_1 v_1,\ d_2 v_2,\ T\tau',\ p') = ((\bigsqcup d_1^i)v_1', (\bigsqcup d_2^i)v_2' \parallel d_1 v_1,\ d_2 v_2,\ T\tau',\ p') \in R_M^+$. So (possibly using downwards closure) $\bigsqcup(v_1^i,\ v_2^i \parallel in_{\multimap}\lfloor d_1 \rfloor,\ in_{\multimap}\lfloor d_2 \rfloor,\ \tau \to \mathrm{T}\tau',\ p) \in F(R^-, R^+)_V$.

We conclude that $F(R^-, R^+)_V$ is closed under lubs of chains.

And, so we conclude that if R^+ is adm$^+$, then $F(R^-, R^+)$ is admissible for any relation R^-.

□

A.5 Proof of lemma 21

Lemma: The action of F preserves parameter weakening.

For all $R^+, R^- \in \mathcal{R}(\mathbb{D})$.
If R^+ is parameter weakened, then $F(R^-, R^+)$ is parameter weakened.

Proof

- $F(R^-, R^+)_K$: Let $(pk_1), (pk_0) \in \mathfrak{p}^k,\ (pk_1) \rhd (pk_0)$.

  For all k_1, k_2, all $(x:\tau)^\top$ continuation type, all p vm-parameter it holds that $(\bot, \bot \parallel k_1, k_2, (x:\tau)^\top, p) \in F(R^-, R^+)_K$.

  Assume $(k_1', k_2') \neq (\bot, \bot)$ and $(k_1', k_2' \parallel k_1, k_2, (x:\tau)^\top, (pk_0)) \in F(R^-, R^+)_K$. We want to show $(k_1', k_2' \parallel k_1, k_2, (x:\tau)^\top, (pk_1)) \in F(R^-, R^+)_K$. This follows from $(pk_1') \rhd (pk_1)$ and $(pk_1) \rhd (pk_0)$ implies $(pk_1') \rhd (pk_0)$, (independant of R^+).

- $F(R^-, R^+)_M$: Let $p_1, p_0 \in \mathfrak{p}^{vm},\ p_1 \blacktriangleright p_0$.

  For all m_1, m_2, all $T\tau$ computation type, all p vm-parameter it holds that $(\bot, \bot \parallel m_1, m_2, T\tau, p) \in F(R^-, R^+)_M$.

  Assume $(m_1', m_2') \neq (\bot, \bot)$ and $(m_1', m_2' \parallel m_1, m_2, T\tau, p_0) \in F(R^-, R^+)_M$. We want to show $(m_1', m_2' \parallel m_1, m_2, T\tau, p_1) \in F(R^-, R^+)_M$. This follows from $p_1' \blacktriangleright p_1$ and $p_1 \blacktriangleright p_0$ implies $p_1' \blacktriangleright p_0$, (independant of R^+).

- $F(R^-, R^+)_V$: Let $p_1, p_0 \in \mathfrak{p}^{vm},\ p_1 \blacktriangleright p_0$

  Forall v_1, v_2, all τ value-type, all p vm-parameter it holds that $(\bot, \bot \parallel v_1, v_2, \tau, p) \in F(R^-, R^+)_V$.

  Assume $(v_1', v_2') \neq (\bot, \bot)$ and $(v_1', v_2' \parallel v_1, v_2, \tau, p_0) \in F(R^-, R^+)_V$. The proof proceeds over type parameter cases:
  - (unit),(int): Immidiate.
  - $(\sigma\ ref)$: Follows from $p_1 \blacktriangleright p_0$ implies $Z^{p_1} \supseteq Z^{p_0}$.
  - $(+)$,$(\times)$,(μ): Follows from parameter-weakening of R_V^+. We show the proof for sum types, the other proofs are similar.
    $(+)$: Assume $(v_1', v_2' \parallel v_1, v_2, \tau_a + \tau_b, p_0) \in F(R^-, R^+)_V$, and $v_1' \neq \bot\ \vee\ v_2' \neq \bot$. Then $\exists d_1', d_2' \in \mathbb{V}, d_1, d_2 \in \mathbb{V}_\downarrow.\ v_1 = in_\oplus in_j d_1 \wedge v_2 = in_\oplus in_j d_2 \wedge (v_1' = d_1' = \bot \vee v_1' = in_\oplus in_j d_1' \neq \bot) \wedge (v_2' = d_2' = \bot \vee v_2' = in_\oplus in_j d_2' \neq \bot) \wedge (d_1', d_2' \parallel d_1, d_2, \tau_j, p_0) \in R_V^+$ where $j \in a, b$.
    Since R^+ is parameter weakened then $(d_1', d_2' \parallel d_1, d_2, \tau_j, p_1) \in R_V^+$, and so
    $(v_1', v_2' \parallel v_1, v_2, \tau_1 + \tau_2, p_1) \in F(R^-, R^+)_V$.

- ($\forall$): Follows from parameter-weakening of R^+_M.
  (+): Assume $(v'_1, v'_2 \parallel v_1, v_2, \forall\alpha.T\tau, p_0) \in F(R^-, R^+)_V$, and $v'_1 \neq \bot \;\vee\; v'_2 \neq \bot$. Then $\exists d'_1, d'_2, d_1, d_2 \in \mathbb{M}.\ v_1 = in_\forall\lfloor d_1\rfloor \wedge v_2 = in_\forall\lfloor d_2\rfloor \wedge ((v'_1 = \bot \wedge d'_1 = \bot) \vee v'_1 = in_\forall\lfloor d'_1\rfloor) \wedge ((v'_2 = \bot \wedge d'_2 = \bot) \vee v'_2 = in_\forall\lfloor d'_2\rfloor) \wedge \forall - \vdash \sigma : \mathit{type}.\ (d'_1, d'_2 \parallel d_1, d_2, T\tau[\sigma/\alpha], p_0) \in R^+_M$.
  Since R^+ is parameter weakened then $\forall - \vdash \sigma : \mathit{type}.\ (d'_1, d'_2 \parallel d_1, d_2, T\tau[\sigma/\alpha], p_1) \in R^+_M$, and so $(v'_1, v'_2 \parallel v_1, v_2, \forall\alpha.T\tau, p_1) \in F(R^-, R^+)_V$.

- ($\rightarrow$): Assume $(v'_1, v'_2 \parallel v_1, v_2, \tau \rightarrow T\tau', p_0) \in F(R^-, R^+)_V$, and $v'_1 \neq \bot \;\vee\; v'_2 \neq \bot$. Then $\exists d'_1, d'_2, d_1, d_2 \in (\mathbb{V} \multimap \mathbb{M}).\ v_1 = in_\multimap\lfloor d_1\rfloor \wedge v_2 = in_\multimap\lfloor d_2\rfloor \wedge ((v'_1 = \bot \wedge d'_1 = \bot) \vee v'_1 = in_\multimap\lfloor d'_1\rfloor) \wedge ((v'_2 = \bot \wedge d'_2 = \bot) \vee v'_2 = in_\multimap\lfloor d'_2\rfloor) \wedge \forall p'_0 \blacktriangleright p_0.\ \forall.(w'_1, w'_2 \parallel w_1, w_2, \tau, p'_0) \in R^-_V.\ (d'_1 w'_1, d'_2 w'_2 \parallel d_1 w_1, d_2 w_2, T\tau', p'_0) \in R^+_M$.
  We want to show $(v'_1, v'_2 \parallel v_1, v_2, \tau \rightarrow T\tau', p_1) \in F(R^-, R^+)_V$. This requires $\forall p'_1 \blacktriangleright p_1.\ \forall.(w'_1, w_1, w'_2, w_2, \tau, p'_1) \in R^-_V.\ (d'_1 w'_1, d'_2 w'_2 \parallel d_1 w_1, d_2 w_2, T\tau', p'_1) \in R^+_M$. Let $p'_1 \blacktriangleright p_1$. We have $p'_1 \blacktriangleright p_1 \;\wedge\; p_1 \blacktriangleright p_0 \Rightarrow p'_1 \blacktriangleright p_0$. So it follows from the assumpitions that $\forall.(w'_1, w'_2 \parallel w_1, w_2, \tau, p'_1) \in R^-_V.\ (d'_1 w'_1, d'_2 w'_2 \parallel d_1 w_1, d_2 w_2, T\tau', p'_1) \in R^+_M$, and hence $(v'_1, v'_2 \parallel v_1, v_2, \tau \rightarrow T\tau', p_1) \in F(R^-, R^+)_V$.

We conclude that the action of F on relations on $\mathbb{D}$ preserves parameter weakening. $\square$

A.6 Proof of lemma 22

Lemma: The action of F on functions $\mathbb{D} \multimap \mathbb{D}$ preserves the relation $(\_\,, id) : \_ \subset \_$.
$\forall R^+, S^+, R^-, S^- \in \mathcal{R}(\mathbb{D}).\ \forall f^+, f^- : \mathbb{D} \multimap \mathbb{D}$.

If $(f^-, id_\mathbb{D}) : S^- \subset R^-$ and $(f^+, id_\mathbb{D}) : R^+ \subset S^+$ then
$(F(f^-, f^+), F(id_\mathbb{D}, id_\mathbb{D})) = (F(f^-, f^+), id_{F(\mathbb{D},\mathbb{D})}) : F(R^-, R^+) \subset F(S^-, S^+)$.

Proof of lemma.
The remaining part of the proof:
Let $R^+, S^+, R^-, S^- \in \mathcal{R}(\mathbb{D})$, and
let $f^+, f^- : \mathbb{D} \multimap \mathbb{D}$. $f^- = (f^-_v, f^-_k, f^-_m, f^-_s)$, $f^+ = (f^+_v, f^+_k, f^+_m, f^+_s)$.

- $F(\mathbb{D}, \mathbb{D})_M$
  Assume $(m'_1, m'_2 \parallel m_1, m_2, T\tau, p) \in F(R^-, R^+)_M$.
  We aim to show $(h_m m'_1, h_m m'_2 \parallel m_1, m_2, T\tau, p) =$
  $(\lambda k.\lambda S.m'_1(f^-_k k)(f^-_s S),\ \ \lambda k.\lambda S.m'_2(f^-_k k)(f^-_s S) \parallel m_1,\ m_2,\ T\tau, p)\ \in F(S^-, S^+)_M$.

  If $m'_1 = \bot = m'_2$ then since h_m is strict , so $(h_m m'_1, h_m m'_2 \parallel m_1, m_2, T\tau, p) = (\bot, \bot \parallel m_1, m_2, T\tau, p) \in F(S^-, S^+)_M$.
  Else, let $p' \blacktriangleright p$, $(pk') \in p'^{\mathbf{K}}$, $(k'_1, k'_2 \parallel k_1, k_2, (x : \tau)^\top, (pk')) \in S^-_K$,
  Let $(pks') \in (pk')^{\mathbf{S}}$, $(s'_1, s'_2 \parallel s_1, s_2, (pks')) \in S^-_S$.

  We want to show
  $(h_m m'_1) k'_1 s'_1 = \top \Rightarrow m_2 k_2 s_2 = \top$ and $(h_m m'_2) k'_2 s'_2 = \top \Rightarrow m_1 k_1 s_1 = \top$, or equivalently
  $m'_1(f^-_k k'_1)(f^-_s s'_1) = \top \Rightarrow m_2 k_2 s_2 = \top$ and $m'_2(f^-_k k'_2)(f^-_s s'_2) = \top \Rightarrow m_1 k_1 s_1 = \top$.
  Since $(f^-, id_\mathbb{D}) : S^- \subset R^-$ it holds that $(f^-_k k'_1, f^-_k k'_2 \parallel k_1, k_2, (x : \tau)^\top, (pk')) \in R^-_K$ and $(f^-_s s'_1, f^-_s s'_2 \parallel s_1, s_2, (pks')) \in R^-_S$. Then since $(m'_1, m'_2 \parallel m_1, m_2, T\tau, p) \in F(R^-, R^+)$ we have that $m'_1(f^-_k k'_1)(f^-_s s'_1) = \top \Rightarrow m_2 k_2 s_2$ and $m'_2(f^-_k k'_2)(f^-_s s'_2) = \top \Rightarrow m_1 k_1 s_1$.

  We also need to show
  $h_m(m'_1) = \lambda k.\lambda S.m'_1(f^-_k k)(f^-_s S) \sqsubseteq m_1 \wedge h_m(m'_2) = \lambda k.\lambda S.m'_2(f^-_k k)(f^-_s S) \sqsubseteq m_2$. This follows from $m'_1 \sqsubseteq m_1 \;\wedge\; m'_2 \sqsubseteq m_2$ and $f^- \sqsubseteq id_\mathbb{D}$.

  We conclude $(h_m m'_1, h_m m'_2 \parallel id_3 m_1, id_3 m_2, T\tau, p) \in F(S^-, S^+)_M$.

- $F(\mathbb{D}, \mathbb{D})_K$
  Assume $(k_1', k_2' \parallel k_1, k_2, (x:\tau)^\top, (pk)) \in F(R^-, R^+)_K$.
  We aim to show $(h_k k_1', h_k k_2' \parallel id_2 k_1, id_2 k_2, (x:\tau)^\top, (pk)) =$
  $(\lambda S.\lambda v.k_1'(f_s^- S)(f_v^- v),\ \lambda S.\lambda v.k_2'(f_s^- S)(f_v^- v) \parallel k_1,\ k_2,\ (x:\tau)^\top,\ (pk)) \in F(S^-, S^+)_K$.
  This follows by similar arguments as for $(m_1', m_2' \parallel m_1, m_2, T\tau, p)$ above, so we omit the proof.

- $F(\mathbb{D}, \mathbb{D})_V$
  Assume $(v_1', v_2' \parallel v_1, v_2, \tau, p) \in F(R^-, R^+)_V$.
  As before by strictness of h_v, if $(\bot, \bot \parallel v_1, v_2, \tau, p) \in F(R^-, R^+)$ then $(h_v\bot, h_v\bot \parallel v_1, v_2, \tau, p) = (\bot, \bot \parallel v_1, v_2, \tau, p) \in F(S^-, S^+)$.

  Else $v_1' \sqsubseteq v_1 \neq \bot\ \wedge\ v_2' \sqsubseteq v_2 \neq \bot$ and v_1', v_2', v_1, v_2 fulfill the type parameter constructor determined properties required for $(v_1',\ v_2' \parallel v_1,\ v_2, \tau, p) \in \tilde{F}(R^-, R^+)_V$. And we need to show that if $h_v(v_1') \neq \bot\ \vee\ h_v(v_2') \neq \bot$ then $h_v(v_1') \sqsubseteq v_1\ \wedge\ h_v(v_2') \sqsubseteq v_2$ and $h_v(v_1'),\ h_v(v_2'),\ v_1,\ v_2$ fulfill the type determined properties required for $\tilde{F}(S^-, S^+)_V$. We have $h_v(v_1') \sqsubseteq v_1 \neq \bot\ \wedge\ h_v(v_2') \sqsubseteq v_2 \neq \bot$ follows from the similar property in the assumption and $h_v \sqsubseteq id_{\mathbb{D}}$.

  For the rest we argue by type constructor cases.

  - Assume $(v_1', v_2' \parallel v_1, v_2, unit, p) \in F(R^-, R^+)$ and $(v_1' \neq \bot \vee v_2' \neq \bot)$
    then $h_v v_1' = v_1' \sqsubseteq v_1 = in_{\mathbf{1}}\lfloor * \rfloor$ and $h_v v_2' = v_2' \sqsubseteq v_2 = in_{\mathbf{1}}\lfloor * \rfloor$, and
    $(h_v v_1', h_v v_2' \parallel v_1, v_2, unit, p) \in F(S^-, S^+)$.
  - Assume $(v_1', v_2' \parallel v_1, v_2, int, p) \in F(R^-, R^+)$ and $(v_1' \neq \bot \vee v_2' \neq \bot)$
    then $\exists n$ such that $h_v v_1' = v_1' \sqsubseteq v_1 = in_{\mathbb{Z}} n$ and $h_v v_2' = v_2' \sqsubseteq v_2 = in_{\mathbb{Z}} n$, and we have
    $(h_v v_1', h_v v_2' \parallel v_1, v_2, int, p) \in F(S^-, S^+)$.
  - Assume $(v_1', v_2' \parallel v_1, v_2, \sigma ref, p) \in F(R^-, R^+)$ and $(v_1' \neq \bot \vee v_2' \neq \bot)$
    then $\exists l_1, l_2$ such that $h_v v_1' = v_1' \sqsubseteq v_1 = in_{\mathbb{L}} l_1$, $h_v v_2' = v_2' \sqsubseteq v_2 = in_{\mathbb{L}} l_2$, $(l_1, l_2, \sigma) \in Z^p$, and we have $(h_v v_1', h_v v_2' \parallel v_1, v_2, \sigma ref, p) \in F(S^-, S^+)$.
  - Assume $(v_1', v_2' \parallel v_1, v_2, \tau_a + \tau_b, p) \in F(R^-, R^+)$ and $(v_1' \neq \bot \vee v_2' \neq \bot)$ then
    $\exists d_1', d_2' \in \mathbb{V}, d_1, d_2 \in \mathbb{V}_\downarrow.\ v_1 = in_\oplus in_i d_1 \neq \bot \wedge v_2 = in_\oplus in_i d_2 \neq \bot\ \wedge$
    $(v_1' = d_1' = \bot \vee v_1' = in_\oplus in_i d_1' \neq \bot) \wedge (v_2' = d_2' = \bot \vee v_2' = in_\oplus in_i d_2' \neq \bot)\ \wedge$
    $(d_1', d_2' \parallel d_1, d_2, \tau_i, p) \in R_V^+$.
    $(v_1' = d_1' = \bot \Rightarrow h_v v_1' = f_v^+ d_1' = \bot) \wedge ((v_1' = in_\oplus in_i d_1' \wedge f_v^+ d_1' = \bot) \Rightarrow h_v v_1' = f_v^+ d_1' = \bot) \wedge$
    $((v_1' = in_\oplus in_i d_1' \wedge f_v^+ d_1' \neq \bot) \Rightarrow h_v v_1' = in_\oplus in_i f_v^+ d_1' \neq \bot$ and
    $(v_2' = d_2' = \bot \Rightarrow h_v v_2' = f_v^+ d_2' = \bot) \wedge ((v_2' = in_\oplus in_i d_2' \wedge f_v^+ d_2' = \bot) \Rightarrow h_v v_2' = f_v^+ d_2' = \bot) \wedge$
    $((v_2' = in_\oplus in_i d_2' \wedge f_v^+ d_2' \neq \bot) \Rightarrow h_v v_2' = in_\oplus in_i f_v^+ d_2' \neq \bot$.
    Since $(f^+, id_{\mathbb{D}}) : R^+ \subset S^+$ and $(d_1', d_2' \parallel d_1, d_2, \tau_i, p) \in R_V^+$ it follows that
    $(f_v^+ d_1', f_v^+ d_2' \parallel d_1, d_2, \tau_i, p) \in S_V^+$. It holds that $(h_v v_1' = \bot = f_v^+ d_1' \vee h_v v_1' = in_\oplus in_i f_v^+ d_1' \neq \bot)$ and $(h_v v_2' = \bot = f_v^+ d_2' \vee h_v v_2' = in_\oplus in_i f_v^+ d_2' \neq \bot)$.
    So $(h_v v_1', h_v v_2' \parallel v_1, v_2, \tau_a + \tau_b, p) \in F(S^-, S^+)$
  - Assume $(v_1', v_2' \parallel v_1, v_2, \tau_a \times \tau_b, p) \in F(R^-, R^+)$ and $(v_1' \neq \bot \vee v_2' \neq \bot)$. By similar arguments as for sum-typed it follows that $(h_v v_1', h_v v_2' \parallel v_1, v_2, \tau_a \times \tau_b, p) \in F(S^-, S^+)$.
  - Assume $(v_1', v_2' \parallel v_1, v_2, \mu\alpha.\tau, p) \in F(R^-, R^+)$ and $(v_1' \neq \bot \vee v_2' \neq \bot)$ then $\exists d_1', d_2' \in \mathbb{V}, d_1, d_2 \in \mathbb{V}_\downarrow.\ (v_1' = d_1' = \bot \vee v_1' = in_\mu d_1' \neq \bot) \wedge (v_2' = d_2' = \bot \vee v_2' = in_\mu d_2' \neq \bot) \wedge\ v_1 = in_\mu d_1 \neq \bot \wedge v_2 = in_\mu d_2 \neq \bot$ and $(d_1', d_2' \parallel d_1, d_2, \tau[\mu\alpha.\tau/\alpha], p) \in R_V^+$.
    $(v_1' = d_1' = \bot \Rightarrow h_v v_1' = \bot)\ \wedge\ (v_1' = in_\mu d_1' \Rightarrow h_v v_1' = in_\mu f_v^+ d_1' \sqsupseteq \bot)$ and $(v_2' = d_2' = \bot \Rightarrow h_v v_2' = \bot)\ \wedge\ (v_2' = in_\mu d_2' \Rightarrow h_v v_2' = in_\mu f_v^+ d_2' \sqsupseteq \bot)$.
    Since $(f^+, id_{\mathbb{D}}) : R^+ \subset S^+$ and $(d_1', d_2' \parallel d_1, d_2, \tau[\mu\alpha.\tau/\alpha], p) \in R_V^+$ it follows that
    $(f_v^+ d_1', f_v^+ d_2' \parallel d_1, d_2, \tau[\mu\alpha.\tau/\alpha], p) \in S_V^+$.
    Then $(h_v v_1', h_v v_2' \parallel v_1, v_2, \mu\alpha.\tau, p) \in F(S^-, S^+)$

- Assume $(v_1', v_2' \parallel v_1, v_2, \forall\alpha.T\tau, p) \in F(R^-, R^+)$ and $(v_1' \neq \bot \vee v_2' \neq \bot)$ then
  $\exists m_1', m_1, m_2', m_2 \in D_M^+.\ v_1 = in_\forall\lfloor m_1\rfloor \wedge v_2 = in_\forall\lfloor m_2\rfloor\ \wedge$
  $((v_1' = \bot \wedge m_1' = \bot) \vee (v_1' = in_\forall\lfloor m_1'\rfloor) \wedge ((v_2' = \bot \wedge m_2' = \bot) \vee (v_2' = in_\forall\lfloor m_2'\rfloor)\ \wedge$
  $\forall\sigma$ with $\_ \vdash \sigma : type.\ (m_1', m_2' \parallel m_1, m_2, T\tau[\sigma/\alpha], p) \in R_M^+$.
  $(v_1' = \bot \wedge m_1' = \bot) \Rightarrow (h_v v_1' = \bot \wedge f_m^+ m_1' = \bot)$ and $v_1' = in_\forall\lfloor m_1'\rfloor \Rightarrow h_v v_1' = in_\forall\lfloor f_m^+ m_1'\rfloor$,
  $(v_2' = \bot \wedge m_2' = \bot) \Rightarrow (h_v v_2' = \bot \wedge f_m^+ m_2' = \bot)$ and $v_2' = in_\forall\lfloor m_2'\rfloor \Rightarrow h_v v_2' = in_\forall\lfloor f_m^+ m_2'\rfloor$.
  Since $(f^+, id_{\mathbb{D}}) : R^+ \subset S^+$ we have $\forall\sigma$ with $\_ \vdash \sigma : type.\ (f_m^+ m_1', f_m^+ m_2' \parallel m_1, m_2, T\tau[\sigma/\alpha], p) \in S_3^+$. Hence $(h_v v_1', h_v v_2' \parallel v_1, v_2, \forall\alpha.T\tau, p) \in F(S^-, S^+)$.
- Assume $(v_1', v_2' \parallel v_1, v_2, \tau \to T\tau', p) \in F(R^-, R^+)$ and $(v_1' \neq \bot \vee v_2' \neq \bot)$
  then $\exists d_1', d_2', d_1, d_2 \in (\mathbb{V} \multimap \mathbb{M}).\ v_1 = in_\multimap\lfloor d_1\rfloor \wedge v_2 = in_\multimap\lfloor d_2\rfloor\ \wedge$
  $((v_1' = \bot \wedge d_1' = \bot) \vee v_1' = in_\multimap\lfloor d_1'\rfloor) \wedge ((v_2' = \bot \wedge d_2' = \bot) \vee v_2' = in_\multimap\lfloor d_2'\rfloor) \wedge$
  $\forall p' \blacktriangleright p, \forall(\hat{w}_1'\ , \hat{w}_2' \parallel \hat{w}_1,\ \hat{w}_2,\ \tau,\ p') \in R_V^-.\ (d_1'\hat{w}_1',\ d_2'\hat{w}_2' \parallel d_1\hat{w}_1,\ d_2\hat{w}_2,\ T\tau',\ p') \in R_M^+$.

  We need to show $(h_v v_1', h_v v_2' \parallel v_1, v_2, \tau \to T\tau', p) \in F(S^-, S^+)$.
  $((v_1' = \bot \wedge d_1' = \bot) \Rightarrow (h_v v_1' = \bot \wedge (\lambda w. f_m^+(d_1'(f_v^- w))) = \bot)\ \wedge$
  $(v_1' = in_\multimap\lfloor d_1'\rfloor) \Rightarrow h_v v_1' = in_\multimap\lfloor(\lambda w. f_m^+(d_1'(f_v^- w)))\rfloor)\ \wedge$
  $((v_2' = \bot \wedge d_2' = \bot) \Rightarrow (h_v v_2' = \bot \wedge (\lambda w. f_m^+(d_2'(f_v^- w))) = \bot)\ \wedge$
  $(v_2' = in_\multimap\lfloor d_2'\rfloor) \Rightarrow h_v v_2' = in_\multimap\lfloor(\lambda w. f_m^+(d_2'(f_v^- w)))\rfloor)$.
  Let $p' \blacktriangleright p,\ (w_1', w_2' \parallel w_1, w_2, \tau, p') \in S_V^-$.
  Since $(f^-, id_{\mathbb{D}}) : S^- \subset R^-$ we have that $(f_v^- w_1', f_v^- w_2' \parallel w_1, w_2, \tau, p') \in R_V^-$, and then $(d_1'(f_v^- w_1'), d_2'(f_v^- w_2') \parallel d_1 w_1, d_2 w_2, T\tau', p') \in R_M^+$.
  Since $(f^+, id_{\mathbb{D}}) : R^+ \subset S^+$ then $(f_m^+(d_1'(f_v^- w_1')),\ f_m^+(d_2'(f_v^- w_2')) \parallel d_1 w_1,\ d_2 w_2,\ T\tau', p') \in S_M^+$. So $(h_v v_1', h_v v_2' \parallel v_1, v_2, \tau \to T\tau', p) \in F(S^-, S^+)$.

We conclude that in all cases $(h_v v_1', h_v v_2' \parallel v_1, v_2, \tau, p) \in F(S^-, S^+)_M$.

Then we conclude that the action of $F(-,+)$ on functions in $\mathbb{D} \multimap \mathbb{D}$ preserves $(\_, id) : \_ \subset \_$.

$\square$

A.7 Proof of Proposition 4

Proposition: Typing rules preserve $\nabla^{\Xi\Gamma}$ relation of denotations.

unit: $$\frac{}{\Delta; \Xi; \Gamma \vdash () : unit}$$

There are no assumptions, so we need to prove that it holds $([\![\Delta; \Xi; \Gamma \vdash () : unit]\!], [\![\Delta; \Xi; \Gamma \vdash () : unit]\!], unit, p) \in \nabla_V^{\Xi\Gamma}$ that is
$([\![\Delta; \Xi; \Gamma \vdash () : unit]\!](\rho_1'), [\![\Delta; \Xi; \Gamma \vdash () : unit]\!](\rho_2') \parallel$
$[\![\Delta; \Xi; \Gamma \vdash () : unit]\!](\rho_1), [\![\Delta; \Xi; \Gamma \vdash () : unit]\!](\rho_2)), unit, p') \in \nabla_V$.

If $\rho_1' = \bot \wedge \rho_2' = \bot$ then
$([\![\Delta; \Xi; \Gamma \vdash () : unit]\!](\rho_1'), [\![\Delta; \Xi; \Gamma \vdash () : unit]\!](\rho_2')),$
$[\![\Delta; \Xi; \Gamma \vdash () : unit]\!](\rho_1), [\![\Delta; \Xi; \Gamma \vdash () : unit]\!](\rho_2)) =$
$(\bot, \bot, [\![\Delta; \Xi; \Gamma \vdash () : unit]\!](\rho_1), [\![\Delta; \Xi; \Gamma \vdash () : unit]\!](\rho_2))$ and
$(\bot, \bot \parallel [\![\Delta; \Xi; \Gamma \vdash () : unit]\!](\rho_1), [\![\Delta; \Xi; \Gamma \vdash () : unit]\!](\rho_2), unit, p') \in \nabla_V$,

If $\rho_1' \neq \bot \vee \rho_2' \neq \bot$ and $\rho_1 \neq \bot \wedge \rho_2 \neq \bot$ then
$([\![\Delta; \Xi; \Gamma \vdash () : unit]\!](\rho_1'), [\![\Delta; \Xi; \Gamma \vdash () : unit]\!](\rho_2')),$
$[\![\Delta; \Xi; \Gamma \vdash () : unit]\!](\rho_1), [\![\Delta; \Xi; \Gamma \vdash () : unit]\!](\rho_2)) =$
$(d_1', d_2', i(in_{\mathbf{1}}*), i(in_{\mathbf{1}}*))$ where $d_1', d_2' \in \{\bot, i(in_{\mathbf{1}}*)\}$, and so
$(d_1', d_2' \parallel i(in_{\mathbf{1}}*), i(in_{\mathbf{1}}*), unit, p') \in \nabla_V$

Hence $([\![\Delta; \Gamma \vdash () : unit]\!], [\![\Delta; \Gamma \vdash () : unit]\!], unit, p) \in \nabla_V^{\Xi\Gamma}$.

int: $$\frac{}{\Delta;\Xi;\Gamma \vdash n : int}$$

We need to prove that it holds $([\![\Delta;\Xi;\Gamma \vdash n : int]\!], [\![\Delta;\Xi;\Gamma \vdash n : int]\!], int, p) \in \nabla_V^{\Xi\Gamma}$ that is $([\![\Delta;\Xi;\Gamma \vdash n : int]\!](\rho_1'), [\![\Delta;\Xi;\Gamma \vdash n : int]\!](\rho_2') \parallel$
$[\![\Delta;\Xi;\Gamma \vdash n : int]\!](\rho_1), [\![\Delta;\Xi;\Gamma \vdash n : int]\!](\rho_2)), int, p') \in \nabla_V$.

If $\rho_1' = \bot \wedge \rho_2' = \bot$ then
$([\![\Delta;\Xi;\Gamma \vdash n : int]\!](\rho_1'), [\![\Delta;\Xi;\Gamma \vdash n : int]\!](\rho_2'),$
$[\![\Delta;\Xi;\Gamma \vdash n : int]\!](\rho_1), [\![\Delta;\Xi;\Gamma \vdash n : int]\!](\rho_2)) =$
$(\bot, \bot, [\![\Delta;\Xi;\Gamma \vdash n : int]\!](\rho_1), [\![\Delta;\Xi;\Gamma \vdash n : int]\!](\rho_2))$ and
$(\bot, \bot \parallel [\![\Delta;\Xi;\Gamma \vdash n : int]\!](\rho_1), [\![\Delta;\Xi;\Gamma \vdash n : int]\!](\rho_2), int, p') \in \nabla_V$,

If $\rho_1' \neq \bot \vee \rho_2' \neq \bot$ and $\rho_1 \neq \bot \wedge \rho_2 \neq \bot$ then
$([\![\Delta;\Xi;\Gamma \vdash n : int]\!](\rho_1'), [\![\Delta;\Xi;\Gamma \vdash n : int]\!](\rho_2')),$
$[\![\Delta;\Xi;\Gamma \vdash n : int]\!](\rho_1), [\![\Delta;\Xi;\Gamma \vdash n : int]\!](\rho_2)) =$
$(d_1', d_2', i(in_{\mathbb{Z}}n), i(in_{\mathbb{Z}}n))$ where $d_1', d_2' \in \{\bot, i(in_{\mathbb{Z}}n)\}$ and so
$(d_1', d_2' \parallel i(in_{\mathbb{Z}}n), i(in_{\mathbb{Z}}n), int, p') \in \nabla_V$

Hence $([\![\Delta;\Gamma \vdash n : int]\!], [\![\Delta;\Gamma \vdash n : int]\!], unit, p) \in \nabla_V^{\Xi\Gamma}$.

eq: $$\frac{\Delta;\Xi;\Gamma \vdash V_a : \tau_a\ ref \qquad \Delta;\Xi;\Gamma \vdash V_b : \tau_b\ ref}{\Delta;\Xi;\Gamma \vdash V_a = V_b : T(unit + unit)}$$

Let $v_{1a} = [\![\Delta;\Xi;\Gamma \vdash V_{1a} : \tau_a\ ref]\!]$, $v_{2a} = [\![\Delta;\Xi;\Gamma \vdash V_{2a} : \tau_a\ ref]\!]$,
$v_{1b} = [\![\Delta;\Xi;\Gamma \vdash V_{1b} : \tau_b\ ref]\!]$, $v_{2b} = [\![\Delta;\Xi;\Gamma \vdash V_{2b} : \tau_b\ ref]\!]$.
Assume $(v_{1a}, v_{2a}, \tau_a\ ref, p) \in \nabla_V^{\Xi\Gamma}$ and $(v_{1b}, v_{2b}, \tau_b\ ref, p) \in \nabla_V^{\Xi\Gamma}$.

The assumption implies that either $v_{1a}(\rho_1') = v_{2a}(\rho_2') = \bot$ or $\exists (l_{1a}, l_{2a}, \tau_a[\overline{\sigma_j/\alpha_j}]) \in Z^{p'}$.
$i^{-1}(v_{1a}(\rho_1')) \sqsubseteq i^{-1}(v_{1a}(\rho_1)) = in_{\mathbb{L}} l_{1a}$.
$i^{-1}(v_{2a}(\rho_2')) \sqsubseteq i^{-1}(v_{2a}(\rho_2)) = in_{\mathbb{L}} l_{2a}$.
and either $v_{1b}(\rho_1') = v_{2b}(\rho_2') = \bot$ or $\exists (l_{1b}, l_{2b}, \tau_b[\overline{\sigma_j/\alpha_j}]) \in Z^{p'}$.
$i^{-1}(v_{1a}(\rho_1')) \sqsubseteq i^{-1}(v_{1a}(\rho_1)) = in_{\mathbb{L}} l_{1b}$.
$i^{-1}(v_{2a}(\rho_2')) \sqsubseteq i^{-1}(v_{2a}(\rho_2)) = in_{\mathbb{L}} l_{2b}$.
It follows from the definition of a match of finite store types that
$(l_{1a}, l_{2a}, \tau_a[\overline{\sigma_j/\alpha_j}]) \in Z^{p'} \wedge (l_{1b}, l_{2b}, \tau_b[\overline{\sigma_j/\alpha_j}]) \in Z^{p'}$ implies $l_{1a} = l_{1b} \Leftrightarrow l_{2a} = l_{2b}$.

Let $m_1 = [\![\Delta;\Xi;\Gamma \vdash V_{1a} = V_{1b} : T(unit+unit)]\!]$, $m_2 = [\![\Delta;\Xi;\Gamma \vdash V_{2a} = V_{2b} : T(unit+unit)]\!]$.
If $v_{1a}(\rho_1') = v_{2a}(\rho_2') = \bot$ or $v_{1b}(\rho_1') = v_{2b}(\rho_2') = \bot$ then $m_1(\rho_1') = m_2(\rho_2') = \bot$ and then $(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T(unit + unit), p') \in \nabla_M$.

Else if $l_{1a} = l_{2a}\ \wedge\ l_{1b} = l_{2b}$ then
$m_1(\rho_1') \sqsubseteq \lambda k.\lambda S.(i^{-1}k)S(i \circ in_{\oplus} \circ in_1(i(in_1*))$, $m_1(\rho_1) = \lambda k.\lambda S.(i^{-1}k)S(i \circ in_{\oplus} \circ in_1(i(in_1*))$,
$m_2(\rho_2') \sqsubseteq \lambda k.\lambda S.(i^{-1}k)S(i \circ in_{\oplus} \circ in_1(i(in_1*))$, $m_2(\rho_2) = \lambda k.\lambda S.(i^{-1}k)S(i \circ in_{\oplus} \circ in_1(i(in_1*))$.
and if
$l_{1a} \neq l_{2a}\ \wedge\ l_{1b} \neq l_{2b}$ then
$m_1(\rho_1') \sqsubseteq \lambda k.\lambda S.(i^{-1}k)S(i \circ in_{\oplus} \circ in_2(i(in_1*))$, $m_1(\rho_1) = \lambda k.\lambda S.(i^{-1}k)S(i \circ in_{\oplus} \circ in_2(i(in_1*))$,
$m_2(\rho_2') \sqsubseteq \lambda k.\lambda S.(i^{-1}k)S(i \circ in_{\oplus} \circ in_2(i(in_1*))$, $m_2(\rho_2) = \lambda k.\lambda S.(i^{-1}k)S(i \circ in_{\oplus} \circ in_2(i(in_1*))$.
Application of these computations to related continuations and related states, will always give that related continuations are applied to correspondingly related states and values. So possibly using downwards closure
$(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T(unit + unit), p') \in \nabla_M$.
We conclude $(m_1, m_2, T(unit + unit), p) \in \nabla_M^{\Xi\Gamma}$.

+ in: $$\frac{\Delta;\Xi;\Gamma \vdash V : \tau_j}{\Delta;\Xi;\Gamma \vdash in_j V : \tau_1 + \tau_2}\ (j \in \{1,2\})$$

Let $v_{10} = [\![\Delta;\Xi;\Gamma \vdash V_1 : \tau_j]\!]$, $v_{20} = [\![\Delta;\Xi;\Gamma \vdash V_2 : \tau_j]\!]$ and $j \in \{1,2\}$. Assume $(v_{10}, v_{20}, \tau_j, p) \in \nabla_V^{\Xi\Gamma}$.
Let $v_1 = [\![\Delta;\Xi;\Gamma \vdash in_j V_1 : \tau_1 + \tau_2]\!]$ and $v_2 = [\![\Delta;\Xi;\Gamma \vdash in_j V_2 : \tau_1 + \tau_2]\!]$.
$i^{-1}(v_1(\rho_1')) = in_+ in_j(v_{10}(\rho_1'))$, $i^{-1}(v_1(\rho_1)) = in_+ in_j(v_{10}(\rho_1))$,
$i^{-1}(v_2(\rho_2')) = in_+ in_j(v_{20}(\rho_2'))$, $i^{-1}(v_2(\rho_2)) = in_+ in_j(v_{20}(\rho_2))$,
where we accept notation $\perp = in_+ in_j(\perp)$.

Since by assumption $(v_{10}(\rho_1'), v_{20}(\rho_2') \parallel v_{10}(\rho_1), v_{20}(\rho_2), \tau_j[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$ then
$(v_1(\rho_1'), v_2(\rho_2') \parallel v_1(\rho_1), v_2(\rho_2), (\tau_1+\tau_2)[\overline{\sigma_j/\alpha_j}], p') \in \nabla_v$ hence $(v_1, v_2, \tau_1 + \tau_2, p) \in \nabla_V^{\Xi\Gamma}$.

+ el: $$\frac{\Delta;\Xi;\Gamma \vdash V : \tau_1 + \tau_2 \qquad \Delta;\Xi;\Gamma, x_1 : \tau_1 \vdash M_a : T\tau' \qquad \Delta;\Xi;\Gamma, x_2 : \tau_2 \vdash M_b : T\tau'}{\Delta;\Xi;\Gamma \vdash case\ V\ of\ in_1 X_1 \Leftarrow M_b; in_2 X_2 \Leftarrow M_b : T\tau'}$$

Let $v_1 = [\![\Delta;\Xi;\Gamma \vdash V_1 : \tau_1 + \tau_2]\!]$, $v_2 = [\![\Delta;\Xi;\Gamma \vdash V_2 : \tau_1 + \tau_2]\!]$.
Assume $(v_1, v_2, \tau_1 + \tau_2, p) \in \nabla_V^{\Xi\Gamma}$.
Let $m_{1a} = [\![\Delta;\Xi;\Gamma, x_1 : \tau_1 \vdash M_{1a} : T\tau']\!]$, $m_{2a} = [\![\Delta;\Xi;\Gamma, x_1 : \tau_1 \vdash M_{2a} : T\tau']\!]$.
Assume $(m_{1a}, m_{2a}, T\tau', p) \in \nabla_M^{\Xi\Gamma \cup \{x_1:\tau_1\}}$.
Let $m_{1b} = [\![\Delta;\Xi;\Gamma, x_2 : \tau_2 \vdash M_{1b} : T\tau']\!]$, $m_{2b} = [\![\Delta;\Xi;\Gamma, x_2 : \tau_2 \vdash M_{2b} : T\tau']\!]$.
Assume $(m_{1b}, m_{2b}, T\tau', p) \in \nabla_M^{\Xi\Gamma \cup \{x_2:\tau_2\}}$.

Let $m_1 = [\![\Delta;\Xi;\Gamma \vdash case\ V_1\ of\ in_1 X_1 \Leftarrow M_{1a}; in_2 X_2 \Leftarrow M_{1b} : T\tau']\!]$ and
$m_2 = [\![\Delta;\Xi;\Gamma \vdash case\ V_2\ of\ in_1 X_1 \Leftarrow M_{2a}; in_2 X_2 \Leftarrow M_{2b} : T\tau']\!]$.

$m_1(\rho_1') = \lambda k.\lambda S.$ case $v_1(\rho_1')$ of
  $in_\oplus(in_1 d_{1a}')$ then $m_{1a}(\rho_1' \otimes d_{1a}')kS$; $in_\oplus(in_2 d_{1b}')$ then $m_{1b}(\rho_1' \otimes d_{1b}')kS$ else $\perp$,
$m_1(\rho_1) = \lambda k.\lambda S.$ case $v_1(\rho_1)$ of
  $in_\oplus(in_1 d_{1a})$ then $m_{1a}(\rho_1 \otimes d_{1a})kS$; $in_\oplus(in_2 d_{1b})$ then $m_{1b}(\rho_1 \otimes d_{1b})kS$ else $\perp$,
$m_2(\rho_2') = \lambda k.\lambda S.$ case $v_2(\rho_2')$ of
  $in_\oplus(in_1 d_{2a}')$ then $m_{2a}(\rho_2' \otimes d_{2a}')kS$; $in_\oplus(in_2 d_{2b}')$ then $m_{2b}(\rho_2' \otimes d_{2b}')kS$ else $\perp$,
$m_2(\rho_2) = \lambda k.\lambda S.$ case $v_2(\rho_2)$ of
  $in_\oplus(in_1 d_{2a})$ then $m_{2a}(\rho_2 \otimes d_{2a})kS$; $in_\oplus(in_2 d_{2b})$ then $m_{2b}(\rho_2 \otimes d_{2b})kS$ else $\perp$.

Since by assumption $(v_1, v_2, \tau_1 + \tau_2, p) \in \nabla_V^{\Xi\Gamma}$ then there are three possibilities. Either $v_1(\rho_1') = v_2(\rho_2') = \perp$ and so $m_1(\rho_1') = m_2(\rho_2') = \perp$ and $(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T\tau'[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$.
Or $\exists d_{1a}', d_{2a}', d_{1a}, d_{2a}.\ (d_{1a}', d_{2a}' \parallel d_{1a}, d_{2a}, \tau_1[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$ and
$(v_1(\rho_1) = in \oplus (in_1 d_{1a}) \neq \perp) \wedge ((v_1(\rho_1') = \perp \wedge d_{1a}' = \perp \vee = (v_1(\rho_1') in \oplus (in_1 d_{1a}' \neq \perp))$,
$(v_2(\rho_2) = in \oplus (in_1 d_{2a}) \neq \perp) \wedge ((v_2(\rho_2') = \perp \wedge d_{2a}' = \perp \vee = (v_2(\rho_2') in \oplus (in_1 d_{2a}' \neq \perp))$.
In this case it follows from the assumption $(m_{1a}, m_{2a}, T\tau', p) \in \nabla_M^{\Xi\Gamma \cup \{x_1:\tau_1\}}$ possibly together with downwards closure that
$(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T\tau'[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$.
Or $\exists d_{1b}', d_{2b}', d_{1b}, d_{2b}.\ (d_{1b}', d_{2b}' \parallel d_{1b}, d_{2b}, \tau_2[\overline{\sigma_j/\alpha_j}], p') \in \nabla_V$ and
$(v_1(\rho_1) = in \oplus (in_2 d_{1b}) \neq \perp) \wedge ((v_1(\rho_1') = \perp \wedge d_{1b}' = \perp \vee = (v_1(\rho_1') in \oplus (in_1 d_{1b}' \neq \perp))$,
$(v_2(\rho_2) = in \oplus (in_1 d_{2b}) \neq \perp) \wedge ((v_2(\rho_2') = \perp \wedge d_{2b}' = \perp \vee = (v_2(\rho_2') in \oplus (in_1 d_{2b}' \neq \perp))$.
then it follows from the assumption $(m_{1b}, m_{2b}, T\tau', p) \in \nabla_M^{\Xi\Gamma \cup \{x_2:\tau_2\}}$ possibly together with downwards closure that
$(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T\tau'[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M$.
So we conclude $(m_1, m_2, T\tau', p) \in \nabla_M^{\Xi\Gamma}$.

$\times$ in: $\dfrac{\Delta;\Xi;\Gamma\vdash V_a:\tau_a \qquad \Delta;\Xi;\Gamma\vdash V_b:\tau_b}{\Delta;\Xi;\Gamma\vdash (V_a,V_b):\tau_a\times\tau_b}$

Let $v_{1a}=[\![\Delta;\Xi;\Gamma\vdash V_{1a}:\tau_a]\!]$, $v_{1b}=[\![\Delta;\Xi;\Gamma\vdash V_{1b}:\tau_b]\!]$,
$v_{2a}=[\![\Delta;\Xi;\Gamma\vdash V_{2a}:\tau_a]\!]$, $v_{2b}=[\![\Delta;\Xi;\Gamma\vdash V_{2b}:\tau_b]\!]$,
Assume $(v_{1a},v_{2a},\tau_a,p)\in\nabla_V^{\Xi\Gamma}$ and $(v_{1b},v_{2b},\tau_b,p)\in\nabla_V^{\Xi\Gamma}$.
Let $v_1=[\![\Delta;\Xi;\Gamma\vdash (V_{1a},V_{1b}):\tau_a\times\tau_b]\!]$, $v_2=[\![\Delta;\Xi;\Gamma\vdash (V_{2a},V_{2b}):\tau_a\times\tau_b]\!]$.

$i^{-1}(v_1(\rho_1'))=If\ (v_{1a}(\rho_1')=\bot\vee v_{1b}(\rho_1')=\bot)\ then\ \bot\ else\ in_\otimes(v_{1a}(\rho_1'),v_{1b}(\rho_1'))$,
$i^{-1}(v_1(\rho_1))=If\ (v_{1a}(\rho_1)=\bot\vee v_{1b}(\rho_1)=\bot)\ then\ \bot\ else\ in_\otimes(v_{1a}(\rho_1),v_{1b}(\rho_1))$,
$i^{-1}(v_2(\rho_2'))=If\ (v_{2a}(\rho_2')=\bot\vee v_{2b}(\rho_2')=\bot)\ then\ \bot\ else\ in_\otimes(v_{2a}(\rho_2'),v_{2b}(\rho_2'))$,
$i^{-1}(v_2(\rho_2))=If\ (v_{2a}(\rho_2)=\bot\vee v_{2b}(\rho_2)=\bot)\ then\ \bot\ else\ in_\otimes(v_{2a}(\rho_2),v_{2b}(\rho_2))$.

If $v_{1a}(\rho_1')=v_{2a}(\rho_2')=\bot$ or $v_{1b}(\rho_1')=v_{2b}(\rho_2')=\bot$ then $i^{-1}(v_1(\rho_1'))=i^{-1}(v_2(\rho_2'))=\bot$.
Else, since by assupmtion $(v_{10a}(\rho_1'),v_{20a},(\rho_2')\parallel v_{10a}(\rho_1),v_{20a}(\rho_2),\tau_1\overline{[\sigma_j/\alpha_j]},p')\in\nabla_V$ and
$(v_{10b}(\rho_1'),v_{20b}(\rho_2')\parallel v_{10b}(\rho_1),v_{20b}(\rho_2),\tau_2\overline{[\sigma_j/\alpha_j]},p')\in\nabla_V$
then $(v_1(\rho_1'),v_2(\rho_2')\parallel v_1(\rho_1),v_2(\rho_2),(\tau_1\times\tau_2)\overline{[\sigma_j/\alpha_j]},p')\in\nabla_V$ hence $(v_1,v_2,\tau_1\times\tau_2,p)\in\nabla_V^{\Xi\Gamma}$.

$\times$ el: $\dfrac{\Delta;\Xi;\Gamma\vdash V:\tau_1\times\tau_2}{\Delta;\Xi;\Gamma\vdash \pi_iV:T\tau_i}\ (i\in\{1,2\})$

Let $v_1=[\![\Delta;\Xi;\Gamma\vdash V_1:\tau_1\times\tau_2]\!]$, $v_2=[\![\Delta;\Xi;\Gamma\vdash V_2:\tau_1\times\tau_2]\!]$,
Assume $(v_1,v_2,\tau_1\times\tau_2,p)\in\nabla_V^{\Xi\Gamma}$.
Let $m_1=[\![\Delta;\Xi;\Gamma\vdash \pi_iV_1:T\tau_i]\!]$, $m_2=[\![\Delta;\Xi;\Gamma\vdash \pi_iV_2:T\tau_i]\!]$.
$m_1(\rho_1')=\lambda k.\lambda S.$ case $v_1(\rho_1')$ of $in_\otimes(d_{11}',d_{12}')\neq\bot$ then kSd_{1i}' else $\bot$,
$m_1(\rho_1)=\lambda k.\lambda S.$ case $v_1(\rho_1)$ of $in_\otimes(d_{11},d_{12})\neq\bot$ then kSd_{1i} else $\bot$,
$m_2(\rho_2')=\lambda k.\lambda S.$ case $v_2(\rho_2')$ of $in_\otimes(d_{21}',d_{22}')\neq\bot$ then kSd_{2i}' else $\bot$,
$m_2(\rho_2)=\lambda k.\lambda S.$ case $v_2(\rho_2)$ of $in_\otimes(d_{21},d_{22})\neq\bot$ then kSd_{2i} else $\bot$.

The assumption $(v_1,v_2,\tau_1\times\tau_2,p)\in\nabla_V^{\Xi\Gamma}$ implies that either $v_1(\rho_1')=v_2(\rho_2')=\bot$ or
$\exists(d_{11}',d_{12}'\parallel d_{11},d_{12},\tau_1\overline{[\sigma_j/\alpha_j]},p')\in\nabla_V,\ (d_{21}',d_{22}'\parallel d_{21},d_{22},\tau_2\overline{[\sigma_j/\alpha_j]},p')\in\nabla_V$ and
$v_1(\rho_1)=in\otimes(d_{11},d_{12})\neq\bot,\ (v_1(\rho_1')=\bot\wedge(d_{11}'=\bot\vee d_{12}'=\bot))\vee(v_1(\rho_1')=in\otimes(d_{11}',d_{12}')\neq\bot)$,
$v_2(\rho_2)=in\otimes(d_{21},d_{22})\neq\bot,\ (v_2(\rho_2')=\bot\wedge(d_{21}'=\bot\vee d_{22}'=\bot))\vee(v_2(\rho_2')=in\otimes(d_{21}',d_{22}')\neq\bot)$.
In all cases we get $(m_1(\rho_1'),m_2(\rho_2')\parallel m_1(\rho_1),m_2(\rho_2),T\tau_i\overline{[\sigma_j/\alpha_j]},p')\in\nabla_M$
so $(m_1,m_2,T\tau_i,p)\in\nabla^{\Xi\Gamma}$.

arith: $\dfrac{\Delta;\Xi;\Gamma\vdash V_a:int \qquad \Delta;\Xi;\Gamma\vdash V_b:int}{\Delta;\Xi;\Gamma\vdash V_a+V_b:Tint}$

Let $v_{1a}=[\![\Delta;\Xi;\Gamma\vdash V_{1a}:int]\!]$, $v_{2a}=[\![\Delta;\Xi;\Gamma\vdash V_{2a}:int]\!]$,
$v_{1b}=[\![\Delta;\Xi;\Gamma\vdash V_{1b}:int]\!]$, $v_{2b}=[\![\Delta;\Xi;\Gamma\vdash V_{2b}:int]\!]$.
Assume $(v_{1a},v_{2a},int,p)\in\nabla_V^{\Xi\Gamma}$ and $(v_{1b},v_{2b},int,p)\in\nabla_V^{\Xi\Gamma}$.

The assumption implies that either $v_{1a}(\rho_1')=v_{2a}(\rho_2')=\bot$ or
$\exists n_a\in\mathbb{Z}.\ i^{-1}(v_{1a}(\rho_1'))\sqsubseteq i^{-1}(v_{1a}(\rho_1))=in_{\mathbb{Z}}n_a$ and $i^{-1}(v_{2a}(\rho_2'))\sqsubseteq i^{-1}(v_{2a}(\rho_2))=in_{\mathbb{Z}}n_a$.
and either $v_{1b}(\rho_1')=v_{2b}(\rho_2')=\bot$ or
$\exists n_b\in\mathbb{Z}.\ i^{-1}(v_{1a}(\rho_1'))\sqsubseteq i^{-1}(v_{1a}(\rho_1))=in_{\mathbb{L}}n_b$ and $i^{-1}(v_{2a}(\rho_2'))\sqsubseteq i^{-1}(v_{2a}(\rho_2))=in_{\mathbb{L}}n_b$.
Let $m_1=[\![\Delta;\Xi;\Gamma\vdash V_{1a}+V_{1b}:Tint]\!]$, $m_2=[\![\Delta;\Xi;\Gamma\vdash V_{2a}+V_{2b}:Tint]\!]$.

If $v_{1a}(\rho_1')=v_{2a}(\rho_2')=\bot$ or $v_{1b}(\rho_1')=v_{2b}(\rho_2')=\bot$ then $m_1(\rho_1')=m_2(\rho_2')=\bot$ and then $(m_1(\rho_1'),m_2(\rho_2')\parallel m_1(\rho_1),m_2(\rho_2),T(unit+unit),p')\in\nabla_M$.

Else let $n = n_a + n_b$ then
$m_1(\rho_1') \sqsubseteq m_1(\rho_1) = \lambda k.\lambda S.(i^{-1}k)S(i \circ in_{\mathbb{Z}}n)$,
$m_2(\rho_2') \sqsubseteq m_2(\rho_2) = \lambda k.\lambda S.(i^{-1}k)S(i \circ in_{\mathbb{Z}}n)$.
Then
$(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), Tint[\overline{\sigma_j/\alpha_j}], p') \in \nabla_M =$
$(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), Tint, p') \in \nabla_M$
hence $(m_1, m_2, Tint, p) \in \nabla_M^{\Xi\Gamma}$.

zero: $$\frac{\Delta; \Xi; \Gamma \vdash V : int}{\Delta; \Xi; \Gamma \vdash iszero\ V : T(unit + unit)}$$

Let $v_1 = [\![\Delta; \Xi; \Gamma \vdash V_1 : int]\!]$ and $v_2 = [\![\Delta; \Xi; \Gamma \vdash V_2 : int]\!]$. Assume $(v_1, v_2, int, p) \in \nabla_V^{\Xi\Gamma}$.
This assumption implies $(v_1(\rho_1'), v_2(\rho_2') \parallel v_1(\rho_1), v_2(\rho_2), int, p') \in \nabla_V$,
and this again implies that either $v_1(\rho_1') = v_2(\rho_2') = \bot$ or
$\exists n \in \mathbb{Z}.\ v_1(\rho_1') \sqsubseteq v_1(\rho_1) = in_{\mathbb{Z}}n$ and $v_2(\rho_2') \sqsubseteq v_2(\rho_2) = in_{\mathbb{Z}}n$

Let $m_1 = [\![\Delta; \Xi; \Gamma \vdash iszero\ V_1 :: T(unit + unit)]\!]$ and $m_2 = [\![\Delta; \Xi; \Gamma \vdash iszero\ V_2 :: T(unit + unit)]\!]$. We need to show $(m_1, m_2, T(unit + unit), p) \in \nabla_M^{\Xi\Gamma}$, that is we want to show
$(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T(unit + unit), p') \in \nabla_M$, or
$(i^{-1}(m_1(\rho_1')), i^{-1}(m_2(\rho_2')) \parallel i^{-1}(m_1(\rho_1)), i^{-1}(m_2(\rho_2)), T(unit + unit), p') \in F(\nabla, \nabla)_M$.

If $v_1(\rho_1') = v_2(\rho_2') = \bot$ then $m_1(\rho_1')$ and $m_2(\rho_2')$ will both be the constant $\bot$ function in $\mathbb{M}$, and hence $(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T(unit + unit), p') \in \nabla_M$.
Else, if $n = 0$ then
$i^{-1}(m_1(\rho_1')) \sqsubseteq i^{-1}(m_1(\rho_1)) = \lambda k.\lambda S.i^{-1}(k)S(in_{\oplus} \circ in_1(i(in_1 *)))$,
$i^{-1}(m_2(\rho_2')) \sqsubseteq i^{-1}(m_2(\rho_2)) = \lambda k.\lambda S.i^{-1}(k)S(in_{\oplus} \circ in_1(i(in_1 *)))$.
If $n \neq 0$ then
$i^{-1}(m_1(\rho_1')) \sqsubseteq i^{-1}(m_1(\rho_1)) = \lambda k.\lambda S.i^{-1}(k)S(in_{\oplus} \circ in_2(i(in_1 *)))$,
$i^{-1}(m_2(\rho_2')) \sqsubseteq i^{-1}(m_2(\rho_2)) = \lambda k.\lambda S.i^{-1}(k)S(in_{\oplus} \circ in_2(i(in_1 *)))$.

Hence $(m_1(\rho_1'), m_2(\rho_2') \parallel m_1(\rho_1), m_2(\rho_2), T(unit + unit), p') \in \nabla_M$.
We conclude $(m_1, m_2, T(unit + unit), p) \in \nabla_M^{\Gamma}$.

CALL-BY-VALUE TERMINATION IN THE UNTYPED λ-CALCULUS

NEIL D. JONES AND NINA BOHR

Neil D. Jones, DIKU, University of Copenhagen, Denmark
E-mail address: neil@diku.dk

Nina Bohr, IT-University of Copenhagen, Denmark
E-mail address: ninab@itu.dk

Abstract. A fully-automated algorithm is developed able to show that evaluation of a given untyped λ-expression will terminate under CBV (call-by-value). The "size-change principle" from first-order programs is extended to arbitrary untyped λ-expressions in two steps. The first step suffices to show CBV termination of a single, stand-alone λ-expression. The second suffices to show CBV termination of any member of a regular set of λ-expressions, defined by a tree grammar. (A simple example is a minimum function, when applied to arbitrary Church numerals.) The algorithm is sound and proven so in this paper. The Halting Problem's undecidability implies that any sound algorithm is necessarily incomplete: some λ-expressions may in fact terminate under CBV evaluation, but not be recognised as terminating.

The intensional power of the termination algorithm is reasonably high. It certifies as terminating many interesting and useful general recursive algorithms including programs with mutual recursion and parameter exchanges, and Colson's "minimum" algorithm. Further, our type-free approach allows use of the Y combinator, and so can identify as terminating a substantial subset of PCF.

1. Introduction

The *size-change* analysis by Lee, Jones and Ben-Amram [32] can show termination of programs whose parameter values have a well-founded size order. The method is reasonably general, easily automated, and does not require human invention of lexical or other parameter orders. It applies to first-order functional programs. This paper applies similar ideas to termination of *higher-order* programs. For simplicity and generality we focus on the simplest such language, the λ-calculus.

Contribution of this paper. Article [23] (prepared for an invited conference lecture) showed how to lift the methods of [32] to show termination of closed λ-expressions. The current paper is a journal version of [23]. It extends [23] to deal not only with a single λ-expression in isolation, but with a regular set of λ-expressions generated by a finite tree grammar. For example, we can show that a λ-expression terminates when applied to Church numerals, even though it may fail to terminate on all possible arguments. This paper includes a number of examples showing its analytical power, including programs with primitive recursion, mutual recursion and parameter exchanges, and Colson's "minimum" algorithm. Further, examples show that our type-free approach allows free use of the Y combinator, and so can identify as terminating a substantial subset of PCF.

1.1. **Related work.** Jones [22] was an early paper on control-flow analysis of the untyped λ-calculus. Shivers' thesis and subsequent work [53, 54] on CFA (control flow analysis) developed this approach considerably further and applied it to the Scheme programming language. This line is closely related to the approximate semantics (static control graph) of Section 3.6 [22].

Termination of untyped programs. Papers based on [32] have used size-change graphs to find bounds on program running times (Frederiksen and Jones [16]); solved related problems, e.g., to ensure that partial evaluation will terminate (Glenstrup and Jones, Lee [24, 30]); and found more efficient (though less precise) algorithms (Lee [31]). Further, Lee's thesis [29] extends the first-order size-change method [32] to handle higher-order named combinator programs. It uses a different approach than ours, and appears to be less general.

We had anticipated from the start that our framework could naturally be extended to higher-order functional programs, e.g., functional subsets of Scheme or ML. This has since been confirmed by Sereni and Jones, first reported in [50]. Sereni's Ph.D. thesis [49] develops this direction in considerably more detail with full proofs, and also investigates problems with lazy (call-by-name) languages. Independently and a bit later, Giesl and coauthors have addressed the analysis of the lazy functional language Haskell [17].

Termination of typed λ-calculi. Quite a few people have written about termination based on types. Various subsets of the λ-calculus, in particular subsets typable by various disciplines, have been proven strongly normalising. Work in this direction includes pathbreaking results by Tait [56] and others concerning simple types, and Girard's System F [26]. Abel, Barthe and others have done newer type-based approaches to show termination of a λ-calculus extended with recursive data types [1, 2, 5].

Typed functional languages: Xi's Ph.D. research focused on tracing value flow via data types for termination verification in higher order programming languages [61], Wahlstedt has an approach to combine size-change termination analysis with constructive type theory [59, 60].

Term rewriting systems: The popular "dependency pair" method was developed by Arts and Giesl [4] for first-order programs in TRS form. This community has begun to study termination of higher order term rewriting systems, including research by Giesl et.al. [18, 17], Toyama [57] and others.

2. The call-by-value λ-calculus

First, we review relevant definitions and results for the call-by-value λ-calculus, and then provide an observable characterisation of the behavior of a nonterminating expression.

2.1. Classical semantics.

Definition 2.1. *Exp* is the set of all λ-expressions that can be formed by these syntax rules, where `@` is the *application operator* (sometimes omitted). We use the `teletype` font for λ-expressions.

$$\begin{array}{lll} \mathtt{e,\ P} & \mathtt{::=} & \mathtt{x} \mid \mathtt{e} \mathbin{@} \mathtt{e} \mid \lambda\mathtt{x.e} \\ \mathtt{x} & \mathtt{::=} & \text{Variable name} \end{array}$$

- The set of *free variables* $fv(\mathtt{e})$ is defined as usual: $fv(\mathtt{x}) = \{\mathtt{x}\}$, $fv(\mathtt{e@e'}) = fv(\mathtt{e}) \cup fv(\mathtt{e'})$ and $fv(\lambda\mathtt{x.e}) = fv(\mathtt{e}) \setminus \{\mathtt{x}\}$. A *closed* λ-expression `e` satisfies $fv(\mathtt{e}) = \emptyset$.
- A *program*, usually denoted by `P`, is any closed λ-expression.
- The set of *subexpressions* of a λ-expression `e` is denoted by $subexp(\mathtt{e})$.

The following is standard, e.g., [43]. Notation: β-reduction is done by substituting v for all free occurrences of $\mathtt{x}$ in $\mathtt{e}$, written $\mathtt{e}[v/\mathtt{x}]$, and renaming λ-bound variables if needed to avoid capture.

Definition 2.2. (Call-by-value semantics) The *call-by-value evaluation relation* is defined by the following inference rules, with judgement form $\mathtt{e} \Downarrow v$ where $\mathtt{e}$ is a closed λ-expression and $v \in \mathit{ValueS}$. *ValueS* (for "standard value") is the set of all abstractions $\lambda\mathtt{x}.\mathtt{e}$.

$$\text{(ValueS)}\ \frac{}{v \Downarrow v}\ (\text{If } v \in \mathit{ValueS}) \qquad \text{(ApplyS)}\ \frac{\mathtt{e}_1 \Downarrow \lambda\mathtt{x}.\mathtt{e}_0 \quad \mathtt{e}_2 \Downarrow v_2 \quad \mathtt{e}_0[v_2/\mathtt{x}] \Downarrow v}{\mathtt{e}_1 @ \mathtt{e}_2 \Downarrow v}$$

Lemma 2.3. (Determinism) *If* $\mathtt{e} \Downarrow v$ *and* $\mathtt{e} \Downarrow w$ *then* $v = w$.

2.2. **Nontermination is sequential.** A proof of $\mathtt{e} \Downarrow v$ is a finite object, and no such proof exists if the evaluation of $\mathtt{e}$ fails to terminate. Thus in order to be able to trace an arbitrary computation, terminating or not, we introduce a new "calls" relation $\mathtt{e} \to \mathtt{e}'$, in order to make nontermination visible.

The "calls" relation. The rationale is straightforward: $\mathtt{e} \to \mathtt{e}'$ if in order to deduce $\mathtt{e} \Downarrow v$ for some value v, it is necessary first to deduce $\mathtt{e}' \Downarrow u$ for some u, i.e., some inference rule has form $\frac{\ldots\ \mathtt{e}' \Downarrow ?\ \ldots}{\mathtt{e} \Downarrow ?}$. Applying this to Definition 2.2 gives the following.

Definition 2.4. (Evaluation and call semantics) The *evaluation and call relations* are defined by the following inference rules, where $\underset{r}{\to}, \underset{d}{\to}, \underset{c}{\to} \subseteq \mathit{Exp} \times \mathit{Exp}$[1].

$$\text{(Value)}\ \frac{}{v \Downarrow v}\ (\text{If } v \in \mathit{Value})$$

$$\text{(Operator)}\ \frac{}{\mathtt{e}_1 @ \mathtt{e}_2 \underset{r}{\to} \mathtt{e}_1} \qquad \text{(Operand)}\ \frac{\mathtt{e}_1 \Downarrow v_1}{\mathtt{e}_1 @ \mathtt{e}_2 \underset{d}{\to} \mathtt{e}_2}$$

$$\text{(Call}_0\text{)}\ \frac{\mathtt{e}_1 \Downarrow \lambda\mathtt{x}.\mathtt{e}_0 \quad \mathtt{e}_2 \Downarrow v_2}{\mathtt{e}_1 @ \mathtt{e}_2 \underset{c}{\to} \mathtt{e}_0[v_2/\mathtt{x}]} \qquad \text{(Apply}_0\text{)}\ \frac{\mathtt{e}_1 \Downarrow \lambda\mathtt{x}.\mathtt{e}_0 \quad \mathtt{e}_2 \Downarrow v_2 \quad \mathtt{e}_0[v_2/\mathtt{x}] \Downarrow v}{\mathtt{e}_1 @ \mathtt{e}_2 \Downarrow v}$$

For convenience we will sometimes combine the three into a single *call relation* $\to\ = \underset{r}{\to} \cup \underset{d}{\to} \cup \underset{c}{\to}$. As usual, we write $\to^+$ for the transitive closure of $\to$, and $\to^*$ for its reflexive transitive closure. We will sometimes write $s \Downarrow$ to mean $s \Downarrow v$ for some $v \in \mathit{ValueS}$, and write $s \not\Downarrow$ to mean there is no $v \in \mathit{ValueS}$ such that $s \Downarrow v$, i.e., if evaluation of s does not terminate.

A small improvement to the operational semantics. Note that rules (Call$_0$) and (Apply$_0$) from Definition 2.4 overlap: $\mathtt{e}_2 \Downarrow v_2$ appears in both, as does $\mathtt{e}_0[v_2/\mathtt{x}]$. Thus (Call$_0$) can be used as an intermediate step to simplify (Apply$_0$), giving a more orthogonal set of rules. Variations on the following combined set will be used in the rest of the paper:

[1]Naming: r, d in $\underset{r}{\to}, \underset{d}{\to}$ are the last letters of *operato***r** and *operan***d**, and c in $\underset{c}{\to}$ stands for "call".

Definition 2.5. (Combined evaluate and call rules, standard semantics)

$$(\text{Value})\ \frac{}{v \Downarrow v}\ (\text{If } v \in \textit{Value})$$

$$(\text{Operator})\ \frac{}{\mathtt{e}_1@\mathtt{e}_2 \underset{r}{\to} \mathtt{e}_1} \qquad (\text{Operand})\ \frac{\mathtt{e}_1 \Downarrow v_1}{\mathtt{e}_1@\mathtt{e}_2 \underset{d}{\to} \mathtt{e}_2}$$

$$(\text{Call})\ \frac{\mathtt{e}_1 \Downarrow \lambda\mathtt{x}.\mathtt{e}_0 \qquad \mathtt{e}_2 \Downarrow v_2}{\mathtt{e}_1@\mathtt{e}_2 \underset{c}{\to} \mathtt{e}_0[v_2/\mathtt{x}]} \qquad (\text{Apply})\ \frac{\mathtt{e}_1@\mathtt{e}_2 \underset{c}{\to} \mathtt{e}' \qquad \mathtt{e}' \Downarrow v}{\mathtt{e}_1@\mathtt{e}_2 \Downarrow v}$$

The *call tree* of program P is the smallest set of expressions *CT* containing P that is closed under $\to$. It is not necessarily finite.

Lemma 2.6. (*NIS, or* **N***ontermination* **I***s* **S***equential*) *Let* P *be a program. Then* P $\Downarrow$ *if and only if* CT *has no infinite call chain starting with* P*:*

$$\mathtt{P} = \mathtt{e}_0 \to \mathtt{e}_1 \to \mathtt{e}_2 \to \dots$$

Example: evaluation of expression $\Omega = (\lambda\mathtt{x}.\mathtt{x}@\mathtt{x})@(\lambda\mathtt{y}.\mathtt{y}@\mathtt{y})$ yields an infinite call chain:

$$\Omega = (\lambda\mathtt{x}.\mathtt{x}@\mathtt{x})@(\lambda\mathtt{y}.\mathtt{y}@\mathtt{y}) \to (\lambda\mathtt{y}.\mathtt{y}@\mathtt{y})@(\lambda\mathtt{y}.\mathtt{y}@\mathtt{y}) \to (\lambda\mathtt{y}.\mathtt{y}@\mathtt{y})@(\lambda\mathtt{y}.\mathtt{y}@\mathtt{y}) \to \dots$$

By the NIS Lemma all nonterminating computations give rise to *infinite linear call chains*. Such call chains need not, however, be repetitive as in this example, or even finite.

Informally $\mathtt{e}_0 \not\Downarrow$ implies existence of an infinite call chain as follows: Try to build, bottom-up and left-to-right, a proof tree for $\mathtt{e}_0 \Downarrow v$. Since call-by-value evaluation cannot "get stuck" this process will continue infinitely, leading to an infinite call chain. Figure 1 shows such a call tree with infinite path starting with $\mathtt{e}_0 \to \mathtt{e}_1 \to \mathtt{e}_2 \to \mathtt{e}_3 \to \dots$, where $\to\ =\ \underset{r}{\to} \cup \underset{d}{\to} \cup \underset{c}{\to}$. The Appendix contains a formal proof.

3. An approach to termination analysis

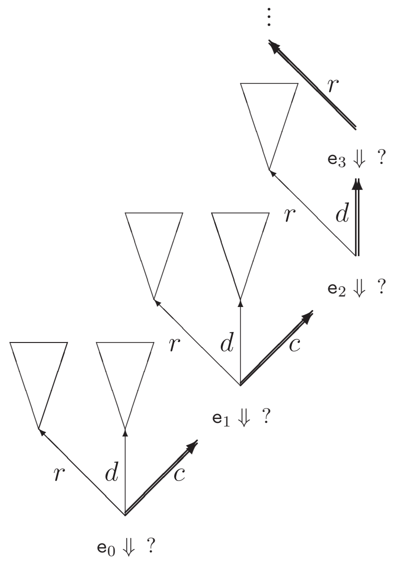
The "size-change termination" analysis of Lee, Jones and Ben-Amram [32] is based on several concepts, including:

(1) Identifying nontermination as caused by *infinitely long sequential state transitions.*
(2) A fixed set of *program control points.*
(3) *Observable decreases* in data value sizes.
(4) *Construction* of one size-change graph for each function call.
(5) Finding the program's *control flow graph*, and the call sequences that follow it.

The NIS Lemma establishes point 1. However, concepts 2, 3, 4 and 5 all seem a priori absent from the λ-calculus, except that an application must be a call; and even then, it is not a priori clear *which* function is being called. We will show, one step at a time, that all the concepts do in fact exist in call-by-value λ-calculus evaluation.

3.1. **An environment-based semantics.**

Program flow analysis usually requires evident program control points. An alternate environment-based formulation remedies their absence in the λ-calculus. The ideas were formalised by Plotkin [43], and have long been used in implementations of functional programming language such as SCHEME and ML.



Code: r = "Operato**r**", d = "Operan**d**", c = "**c**all".

Figure 1: Nontermination implies existence of an infinite call chain

Definition 3.1. (States, etc.) Define *State*, *Value*, *Env* to be the smallest sets such that

$$\begin{array}{lcll} \mathit{State} & = & \{ \quad \mathtt{e} : \rho & \mid \ \mathtt{e} \in \mathit{Exp}, \rho \in \mathit{Env} \text{ and } \ \mathit{dom}(\rho) \supseteq \mathit{fv}(\mathtt{e}) \quad \} \\ \mathit{Value} & = & \{ \quad \lambda\mathtt{x}.\mathtt{e} : \rho & \mid \ \lambda\mathtt{x}.\mathtt{e} : \rho \in \mathit{State} \quad \} \\ \mathit{Env} & = & \{ \quad \rho : X \to \mathit{Value} & \mid \ X \text{ is a finite set of variables} \quad \} \end{array}$$

Equality of states is defined by:

$$\mathtt{e}_1 : \rho_1 = \mathtt{e}_2 : \rho_2 \ \text{ holds if } \ \mathtt{e}_1 = \mathtt{e}_2 \text{ and } \rho_1(x) = \rho_2(x) \text{ for all } \mathtt{x} \in fv(\mathtt{e}_1)$$

The empty environment with domain $X = \emptyset$ is written $[]$. The environment-based evaluation judgement form is $s \Downarrow v$ where $s \in \mathit{State}, v \in \mathit{Value}$.

The Plotkin-style rules follow the pattern of Definition 2.1, except that substitution (β-reduction) $\mathtt{e}_0[v_2/\mathtt{x}]$ of the (CallS) rule is replaced by a "lazy substitution" that just updates the environment in the new (Call) rule. Further, variable values are fetched from the environment

Definition 3.2. (Environment-based evaluation semantics) The evaluation relation $\Downarrow$, is defined by the following inference rules.

$$(\text{ValueE}) \ \frac{}{v \Downarrow v} \ (\text{If } v \in \mathit{Value}) \qquad (\text{VarE}) \ \frac{}{\mathtt{x} : \rho \Downarrow \rho(\mathtt{x})}$$

$$(\text{ApplyE}_0)\quad \frac{\texttt{e}_1 : \rho \Downarrow \lambda\texttt{x.e}_0 : \rho_0 \qquad \texttt{e}_2 : \rho \Downarrow v_2 \qquad \texttt{e}_0 : \rho_0[\texttt{x} \mapsto v_2] \Downarrow v}{\texttt{e}_1\texttt{@e}_2 : \rho \Downarrow v}$$

3.2. **States are tree structures.** A state has form $s = \texttt{e} : \rho$ as in Definition 3.1 where ρ binds the free variables of e to values, which are themselves states. Consider, for two examples, these two states

$$s \;=\; \texttt{e}:\rho \;=\; \texttt{r@(r@a)}:[\texttt{r} \mapsto \texttt{succ} : [], \texttt{a} \mapsto \underline{2} : []]$$

$$s' \;=\; \texttt{e}':\rho' \;=\; \texttt{r@(r@a)}:[\texttt{r} \mapsto \lambda\texttt{a}\,.\,\texttt{r@(r@a)} : [\texttt{r} \mapsto \texttt{succ} : []], \texttt{a} \mapsto \underline{2} : []]$$

(written in our usual linear notation and using the standard Church numerals $\underline{0}, \underline{1}, \underline{2}, \ldots$. For brevity details of the successor function succ are omitted. It is straightforward to verify that $s \Downarrow \underline{4}$ and $s' \Downarrow \underline{6}$ by Definition 3.2.

More generally, each value bound in an environment is a state in turn, so in full detail a state's structure is a *finite tree*. (The levels of this tree represent variable bindings, not to be confused with the syntactic or subexpression tree structures from Figure 3.)

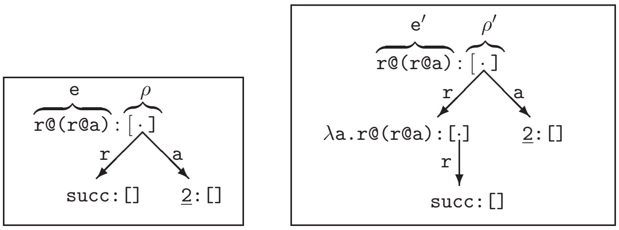


Figure 2: Structures of two states s, s'. Each state is a finite tree.

Figure 2 shows the structure of these two states, with abbreviations for Church numerals such as $\underline{2} = \lambda\texttt{s}\lambda\texttt{z}\,.\,\texttt{s@(s@z)}$.

3.3. **Nontermination made visible in an environment-based semantics.** Straightforwardly adapting the approach of Section 2.2. gives the following set of inference rules, variations on which will be used in the rest of the paper:

Definition 3.3. (Combined evaluate and call rules, environment semantics)

$$(\text{Value})\quad \frac{}{v \Downarrow v}\ (\text{If } v \in \textit{Value}) \qquad\qquad (\text{Var})\quad \frac{}{\texttt{x} : \rho \Downarrow \rho(\texttt{x})}$$

$$(\text{Operator})\quad \frac{}{\texttt{e}_1\texttt{@e}_2 : \rho \underset{r}{\rightarrow} \texttt{e}_1 : \rho} \qquad\qquad (\text{Operand})\quad \frac{\texttt{e}_1 : \rho \Downarrow v_1}{\texttt{e}_1\texttt{@e}_2 : \rho \underset{d}{\rightarrow} \texttt{e}_2 : \rho}$$

$$(\text{Call})\quad \frac{\texttt{e}_1 : \rho \Downarrow \lambda\texttt{x.e}_0 : \rho_0 \qquad \texttt{e}_2 : \rho \Downarrow v_2}{\texttt{e}_1\texttt{@e}_2 : \rho \underset{c}{\rightarrow} \texttt{e}_0 : \rho_0[\texttt{x} \mapsto v_2]} \qquad\qquad (\text{Apply})\quad \frac{\texttt{e}_1\texttt{@e}_2 : \rho \underset{c}{\rightarrow} \texttt{e}' : \rho' \qquad \texttt{e}' : \rho' \Downarrow v}{\texttt{e}_1\texttt{@e}_2 : \rho \Downarrow v}$$

The following is proven in the same way as Lemma 2.6.

Lemma 3.4. (*NIS, or* ***N****ontermination* ***I****s* ***S****equential*) *Let* $\mathtt{P}$ *be a program. Then* $\mathtt{P} : [] \Downarrow$ *if and only if CT has no infinite call chain staring with* $\mathtt{P} : []$ *(where* $\rightarrow = \underset{r}{\rightarrow} \cup \underset{d}{\rightarrow} \cup \underset{c}{\rightarrow}$ *):*

$$\mathtt{P} : [] = \mathtt{e}_0 : \rho_0 \rightarrow \mathtt{e}_1 : \rho_1 \rightarrow \mathtt{e}_2 : \rho_2 \rightarrow \ldots$$

Following the lines of Plotkin [43], the environment-based semantics is shown equivalent to the usual semantics in the sense that they have the same termination behaviour. Further, when evaluation terminates the computed values are related by function $F : States \rightarrow Exp$ defined by

$$F(\mathtt{e} : \rho) = \mathtt{e}[F(\rho(\mathtt{x}_1))/\mathtt{x}_1, ..., F(\rho(\mathtt{x}_k))/\mathtt{x}_k] \;\text{ where } \{\mathtt{x}_1, .., \mathtt{x}_k\} = fv(\mathtt{e}).$$

Lemma 3.5. $\mathtt{P} : [] \Downarrow v$ *(by Definition 3.2) implies* $\mathtt{P} \Downarrow F(v)$ *(by Definition 2.5), and* $\mathtt{P} \Downarrow w$ *implies there exists* v' *such that* $\mathtt{P} : [] \Downarrow v'$ *and* $F(v') = w$.

Example: evaluation of closed $\Omega = (\lambda\mathtt{x.x@x})\mathtt{@}(\lambda\mathtt{y.y@y})$ yields an infinite call chain:

$$\Omega : [] = (\lambda\mathtt{x.x@x})\mathtt{@}(\lambda\mathtt{y.y@y}) : [] \rightarrow \mathtt{x@x} : \rho_1 \rightarrow \mathtt{y@y} : \rho_2 \rightarrow \mathtt{y@y} : \rho_2 \rightarrow \mathtt{y@y} : \rho_2 \rightarrow \ldots$$

where $\rho_1 = [\mathtt{x} \mapsto \lambda\mathtt{y.y@y} : []]$ and $\rho_2 = [\mathtt{y} \mapsto \lambda\mathtt{y.y@y} : []]$.

3.4. **A control point is a subexpression of a λ-expression.** The following *subexpression property* does not hold for the classical rewriting λ-calculus semantics, but does hold for Plotkin-style environment semantics of Definition 3.2. It is central to our program analysis: A *control point* will be a subexpression of the program $\mathtt{P}$ being analysed, and our analyses will trace program information flow to and from subexpressions of $\mathtt{P}$.

Lemma 3.6. *If* $\mathtt{P} : [] \Downarrow \lambda\mathtt{x.e} : \rho$ *then* $\lambda\mathtt{x.e} \in subexp(\mathtt{P})$*.* [Recall Definition 2.1.]

This is proven as follows, using a more general inductive hypothesis.

Definition 3.7. The *expression support* of a given state s is $exp\_sup(s)$, defined by

$$exp\_sup(\mathtt{e} : \rho) = subexp(\mathtt{e}) \cup \bigcup_{\mathtt{x} \in fv(\mathtt{e})} exp\_sup(\rho(\mathtt{x}))$$

Lemma 3.8. (Subexpression property) *If* $s \Downarrow s'$ *or* $s \rightarrow s'$ *then* $exp\_sup(s) \supseteq exp\_sup(s')$*.*

Proof. This follows by induction on the proof of $s \Downarrow v$ or $s \rightarrow s'$. Lemma 3.6 is an immediate corollary.

Base cases: $s = \mathtt{x} : \rho$ and $s = \lambda\mathtt{x.e} : \rho$ are immediate. For rule (Call) suppose $\mathtt{e}_1 : \rho \Downarrow \lambda\mathtt{x.e}_0 : \rho_0$ and $\mathtt{e}_2 : \rho \Downarrow v_2$. By induction

$$exp\_sup(\mathtt{e}_1 : \rho) \supseteq exp\_sup(\lambda\mathtt{x.e}_0 : \rho_0) \quad\text{and}\quad exp\_sup(\mathtt{e}_2 : \rho) \supseteq exp\_sup(v_2)$$

Thus

$$\begin{array}{rcl} exp\_sup(\mathtt{e}_1\mathtt{@}\mathtt{e}_2 : \rho) & \supseteq & exp\_sup(\mathtt{e}_1 : \rho) \cup exp\_sup(\mathtt{e}_2 : \rho) \;\supseteq \\ exp\_sup(\lambda\mathtt{x.e}_0 : \rho_0) \cup exp\_sup(v_2) & \supseteq & exp\_sup(\mathtt{e}_0 : \rho_0[\mathtt{x} \mapsto v_2]) \end{array}$$

For rule (Apply) we have $exp\_sup(\mathtt{e}_1\mathtt{@}\mathtt{e}_2 : \rho) \supseteq exp\_sup(\mathtt{e}' : \rho') \supseteq exp\_sup(v)$. Cases (Operator), (Operand) are immediate. □

3.5. **Finitely describing a program's computation space.** A standard approach to program analysis is to trace data flow along the arcs of the program's *dynamic control graph* or DCG. In our case this is the call relation $\to$ of Definition 2.5. Unfortunately the DCG may be infinite, so for program analysis we will instead compute a safe finite approximation called the SCG, for *static control graph*.

Example 3.9. Figure 3 shows the combinator $\Omega = (\lambda\mathtt{x}.\mathtt{x@x})\mathtt{@}(\lambda\mathtt{y}.\mathtt{y@y})$ as a syntax tree whose subexpressions are labeled by numbers. To its right is the "calls" relation $\to$. It has an infinite call chain:

$$\Omega : [] \to \mathtt{x@x} : \rho_1 \to \mathtt{y@y} : \rho_2 \to \mathtt{y@y} : \rho_2 \to \mathtt{y@y} : \rho_2 \to \ldots$$

Using subexpression numbers, the loop is

$$\mathtt{1} : [] \to \mathtt{3} : \rho_1 \to \mathtt{7} : \rho_2 \to \mathtt{7} : \rho_2 \to \ldots$$

where $\rho_1 = [\mathtt{x} \mapsto \lambda\mathtt{y}.\mathtt{y@y} : []]$ and $\rho_2 = [\mathtt{y} \mapsto \lambda\mathtt{y}.\mathtt{y@y} : []]$. The set of states reachable from $\mathtt{P} : []$ is finite, so this computation is in fact a "repetitive loop." (It is also possible that a computation will reach infinitely many states that are all different.)

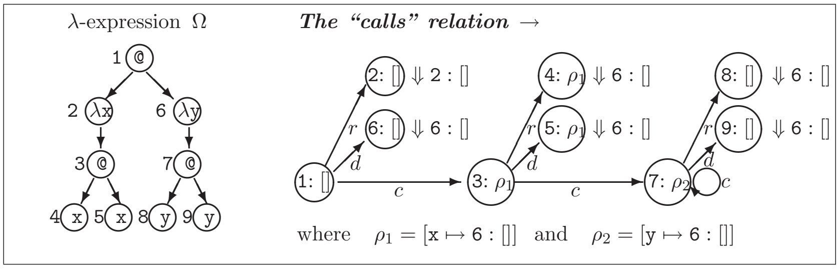


Figure 3: The DCG or dynamic control graph of a λ-expression

By the NIS Lemma 3.4, if $\mathtt{P} \not\Downarrow$ then there exists an infinite call chain

$$\mathtt{P} : [] = \mathtt{e}_0 : \rho_0 \to \mathtt{e}_1 : \rho_1 \to \mathtt{e}_2 : \rho_2 \to \ldots$$

By Lemma 3.8, $\mathtt{e}_i \in \mathit{subexp}(\mathtt{P})$ for each i. Our termination-detecting algorithm will focus on the *size relations between consecutive environments* ρ_i and ρ_{i+1} in this chain. Since $\mathit{subexp}(\mathtt{P})$ is a finite set, at least one subexpression $\mathtt{e}$ occurs infinitely often, so "self-loops" will be of particular interest.

Since all states have an expression component lying in a set of fixed size, and each expression in the environment also lies in this finite set, in an infinite state set $\mathcal{S}$ there will be states whose *environment depths* are arbitrarily large.

3.6. **Static control flow graphs for λ-expressions.** The end goal, given program $\mathtt{P}$, is implied by the NIS Lemma 3.4: correctly to assert the nonexistence of any infinite call chain starting at $\mathtt{P} : []$. By the Subexpression Lemma 3.8 an infinite call chain $\mathtt{e}_0 : \rho_0 \to \mathtt{e}_1 : \rho_1 \to \mathtt{e}_2 : \rho_2 \to \ldots$ can only contain finitely many different expression components $\mathtt{e}_i$. A static control flow graph (SCG for short) including all expression components can be obtained by abstract interpretation of the "Calls" and "Evaluates-to" relations (Cousot and Cousot [14]). Figure 4 shows a SCG for Ω.

An approximating SCG may be obtained by removing all environment components from Definition 3.3. To deal with the absence of environments the variable lookup rule

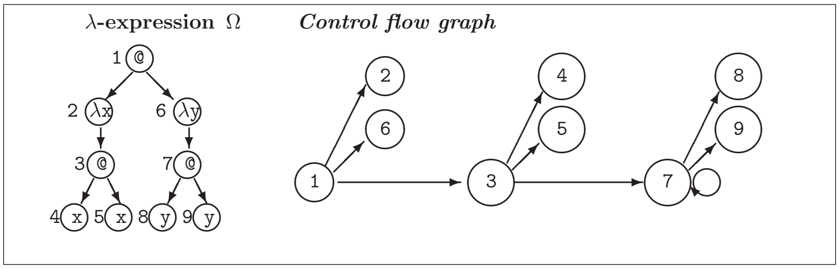


Figure 4: The SCG or static control graph of a λ-expression

is modified: If $\mathtt{e}_1\mathtt{@e}_2$ is *any* application in P such that $\mathtt{e}_1$ can evaluate to a value of form $\lambda\mathtt{x.e}$ and $\mathtt{e}_2$ can evaluate to value v_2, then v_2 is regarded as a possible value of x.

Although approximate, these rules have the virtue that there are only finitely many possible judgements $\mathtt{e} \to \mathtt{e}'$ and $\mathtt{e} \Downarrow \mathtt{e}'$. Consequently, the runtime behavior of program P may be (approximately) analysed by exhaustively applying these inference rules. A later section will extend the rules so they also generate size-change graphs.

Definition 3.10. (Approximate evaluation and call rules) The new judgement forms are $\mathtt{e} \Downarrow \mathtt{e}'$ and $\mathtt{e} \to \mathtt{e}'$. The inference rules are:

$$\text{(ValueA)}\ \frac{}{\lambda\mathtt{x.e} \Downarrow \lambda\mathtt{x.e}} \qquad \text{(VarA)}\ \frac{\mathtt{e}_1\mathtt{@e}_2 \in \mathit{subexp}(\mathtt{P}) \quad \mathtt{e}_1 \Downarrow \lambda\mathtt{x.e}_0 \quad \mathtt{e}_2 \Downarrow v_2}{\mathtt{x} \Downarrow v_2}$$

$$\text{(OperatorA)}\ \frac{}{\mathtt{e}_1\mathtt{@e}_2 \underset{r}{\to} \mathtt{e}_1} \qquad \text{(OperandA)}\ \frac{}{\mathtt{e}_1\mathtt{@e}_2 \underset{d}{\to} \mathtt{e}_2}$$

$$\text{(CallA)}\ \frac{\mathtt{e}_1 \Downarrow \lambda\mathtt{x.e}_0 \quad \mathtt{e}_2 \Downarrow v_2}{\mathtt{e}_1\mathtt{@e}_2 \underset{c}{\to} \mathtt{e}_0} \qquad \text{(ApplyA)}\ \frac{\mathtt{e}_1\mathtt{@e}_2 \underset{c}{\to} \mathtt{e}' \quad \mathtt{e}' \Downarrow v}{\mathtt{e}_1\mathtt{@e}_2 \Downarrow v}$$

The (VarA) rule refers globally to P, the program being analysed. The approximate evaluation is nondeterministic, since an expression may evaluate to more than one value.

Following is a central result: that all possible values obtained by the *actual evaluation* of Definition 3.3 are accounted for by the *approximate evaluation* of Definition 3.10.

Lemma 3.11.
If $\mathtt{P} : [] \to^* \mathtt{e} : \rho$ *and* $\mathtt{e} : \rho \Downarrow \mathtt{e}' : \rho'$, *then* $\mathtt{e} \Downarrow \mathtt{e}'$.
If $\mathtt{P} : [] \to^* \mathtt{e} : \rho$ *and* $\mathtt{e} : \rho \to \mathtt{e}' : \rho'$, *then* $\mathtt{e} \to \mathtt{e}'$.

Proof is in the Appendix.

4. A quick review of size-change analysis

Using the framework of [32], the relation between two states s_1 and s_2 in a call $s_1 \to s_2$ or an evaluation $s_1 \Downarrow s_2$ will be described by means of a size-change graph G.

Example 4.1. Let first-order functions f and g be defined by mutual recursion:

```
f(x,y)   = if x=0 then y else 1: g(x,y,y)
g(u,v,w) = if w=0 then 3:f(u-1,w) else 2:g(u,v-1,w+2)
```

Label the three function calls `1`, `2` and `3`. The "control flow graph" in Figure 5 shows the calling function and called function of each call, e.g., $\texttt{1} : \texttt{f} \to \texttt{g}$. Associate with each call a "size-change graph", e.g., G_1 for call `1`, that safely describes the data flow from the calling function's parameters to the called function's parameters. Symbol $\downarrow$ indicates a value decrease.

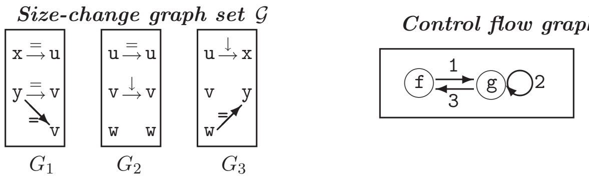

Size-change graph set $\mathcal{G}$
$\texttt{x} \overset{=}{\to} \texttt{u}$
$\texttt{y} \overset{=}{\to} \texttt{v}$
$=$
$\texttt{v}$
$\texttt{u} \overset{=}{\to} \texttt{u}$
$\texttt{v} \overset{\downarrow}{\to} \texttt{v}$
$\texttt{w}$ $\texttt{w}$
$\texttt{u} \overset{\downarrow}{\to} \texttt{x}$
$\texttt{v}$ $\texttt{y}$
$=$
$\texttt{w}$
G_1
G_2
G_3
Control flow graph
f
1
3
g
2

Figure 5: Call graph and size-change graphs for the example first-order program.

Termination reasoning: We show that *all infinite size-change graph sequences* $\mathcal{M} = g_1 g_2 \ldots \in \{G_1, G_2, G_3\}^\omega$ that follow the program's control flow *are impossible* (assuming that the data value set is well-founded):

Case 1: $\mathcal{M} \in \ldots (G_2)^\omega$ ends in infinitely many G_2's: This would imply that *variable* `v` *descends infinitely.*

Case 2: $\mathcal{M} \in \ldots (G_1 G_2^* G_3)^\omega$. This would imply that *variable* `u` *descends infinitely.*

Both cases are impossible; therefore a call of *any program function with any data will terminate.* *End of example.*

Definition 4.2.

(1) A *size-change graph* $A \overset{G}{\to} B$ consists of a *source* set A; a *target* set B; and a set of labeled[2] arcs $G \subseteq A \times \{=, \downarrow\} \times B$.

(2) The *identity* size-change graph for A is $A \overset{id_A}{\to} A$ where $id_A = \{\texttt{x} \overset{=}{\to} \texttt{x} \mid \texttt{x} \in A\}$.

(3) Size-change graphs $A \overset{G_1}{\to} B$ and $C \overset{G_2}{\to} D$ are *composible* if $B = C$. The *composition* of $A \overset{G_1}{\to} B$ and $B \overset{G_2}{\to} C$ is $A \overset{G_1;G_2}{\longrightarrow} C$ where

$$\begin{array}{rclcl}
G_1; G_2 & = & \{\texttt{x} \overset{\downarrow}{\to} \texttt{z} \mid & \downarrow & \in \{ r, s \mid \texttt{x} \overset{r}{\to} \texttt{y} \in G_1 \text{ and } \texttt{y} \overset{s}{\to} \texttt{z} \in G_2 \text{ for some } \texttt{y} \in B\} \; \} \\
& \cup & \{\texttt{x} \overset{=}{\to} \texttt{z} \mid & \{=\} & = \{ r, s \mid \texttt{x} \overset{r}{\to} \texttt{y} \in G_1 \text{ and } \texttt{y} \overset{s}{\to} \texttt{z} \in G_2 \text{ for some } \texttt{y} \in B\} \; \}
\end{array}$$

Lemma 4.3. *Composition is associative, and* $A \overset{G}{\to} B$ *implies* $id_A; G = G; id_B = G$.

Definition 4.4. A *multipath* $\mathcal{M}$ over a set $\mathcal{G}$ of size-change graphs is a finite or infinite composible sequence of graphs in $\mathcal{G}$. Define

$$\mathcal{G}^\omega = \{\mathcal{M} = G_0, G_1, \ldots \mid \text{graphs } G_i, G_{i+1} \text{ are composible for } i = 0, 1, 2, \ldots\}$$

Definition 4.5.

(1) A *thread* in a multipath $\mathcal{M} = G_0, G_1, G_2, \ldots$ is a sequence $t = a_j \overset{r_j}{\to} a_{j+1} \overset{r_{j+1}}{\to} \ldots$ such that $a_k \overset{r_k}{\to} a_{k+1} \in G_k$ for every $k \geq j$ (and each r_k is $=$ or $\downarrow$.)

(2) Thread t is of *infinite descent* if $r_k = \downarrow$ for infinitely many $k \geq j$.

[2]Arc label $\overline{\downarrow}$ signifying $\geq$ was used in [32] instead of $=$, but this makes no difference in our context.

Definition 4.6. The size-change condition.

A set $\mathcal{G}$ of size-change graphs satisfies the *size-change condition* if every infinite multipath $\mathcal{M} \in \mathcal{G}^\omega$ contains at least one thread of infinite descent.

Perhaps surprisingly, the size-change condition is decidable. Its worst-case complexity is shown to be complete for PSPACE in [32] (for first-order programs, in relation to the length of the program being analysed).

The example revisited. The program of Figure 5 has three size-change graphs, one for each of the calls $\mathtt{1} : \mathtt{f} \to \mathtt{g}, \mathtt{2} : \mathtt{g} \to \mathtt{g}, \mathtt{3} : \mathtt{g} \to \mathtt{f}$, so $\mathcal{G} = \{A \overset{G_1}{\to} B, B \overset{G_2}{\to} B, B \overset{G_3}{\to} A\}$ where $A = \{\mathtt{x}, \mathtt{y}\}$ and $B = \{\mathtt{u}, \mathtt{v}, \mathtt{w}\}$. (Note: the vertical layout of size-change graphs in Figure 5 is inessential; one could simply write $G_3 = \{\mathtt{u} \overset{\downarrow}{\to} \mathtt{x}, \mathtt{w} \overset{=}{\to} \mathtt{y}\}$.)

$\mathcal{G}$ satisfies the size-change condition, since *every infinite multipath has either a thread that decreases* $\mathtt{u}$ *infinitely, or a thread that decreases* $\mathtt{v}$ *infinitely.*

5. Tracing data size changes in call-by-value λ-calculus evaluation

The next focus is on size relations between consecutive environments in a call chain.

5.1. Size changes in a computation: a well-founded relation between states.

Definition 5.1.

(1) A *name path* is a finite string p of variable names, where the empty string is (as usual) written ϵ.

(2) The *graph basis* of a state $s = \mathtt{e} : \rho$ is the smallest set $gb(s)$ of name paths satisfying

$$gb(\mathtt{e} : \rho) = \{\epsilon\} \cup \{\mathtt{x}p \mid \mathtt{x} \in fv(\mathtt{e}) \text{ and } \mathtt{p} \in gb(\rho(\mathtt{x}))\}$$

By this definition, for the two states in the example above we have $gb(s) = \{\epsilon, \mathtt{r}, \mathtt{a}\}$ and $gb(s') = \{\epsilon, \mathtt{r}, \mathtt{rr}, \mathtt{a}\}$. Further, given a state s and a path $p \in gb(s)$, we can find the substate identified by name path p as follows:

Definition 5.2. The *valuation function* $\overline{s} : gb(s) \to State$ of a state s is defined by:

$$\overline{s}(\epsilon) = s \quad \text{and} \quad \overline{\mathtt{e} : \rho}(\mathtt{x}p) = \overline{\rho(\mathtt{x})}(p)$$

We need to develop a size ordering on states. This will be modeled by size-change arcs $\overset{=}{\to}$ and $\overset{\downarrow}{\to}$. The size relation we use is partly the "subtree" relation on closure values $\mathtt{e} : \rho$, and partly the "subexpression" relation on λ-expressions.

Definition 5.3.

(1) The *state support* of a state $\mathtt{e} : \rho$ is given by

$$support(\mathtt{e} : \rho) = \{\mathtt{e} : \rho\} \ \cup \bigcup_{\mathtt{x} \in fv(\mathtt{e})} support(\rho(\mathtt{x}))$$

(2) Relations $\succ_1$, $\succ_2$, $\succeq$ and $\succ$ on states are defined by:
- $s_1 \succ_1 s_2$ holds if $support(s_1) \ni s_2$ and $s_1 \neq s_2$;
- $s_1 \succ_2 s_2$ holds if $s_1 = \mathtt{e}_1 : \rho_1$ and $s_2 = \mathtt{e}_2 : \rho_2$, where $subexp(\mathtt{e}_1) \ni \mathtt{e}_2$ and $\mathtt{e}_1 \neq \mathtt{e}_2$ and $\forall \mathtt{x} \in fv(\mathtt{e}_2) . \rho_1(x) = \rho_2(x)$. Further,

- Relation $\succeq$ is defined to be the transitive closure of $\succ_1 \cup \succ_2 \cup =$.
- Finally, $s_1 \succ s_2$ if $s_1 \succeq s_2$ and $s_1 \neq s_2$

Lemma 5.4. *The relation $\succ \subseteq State \times State$ is well-founded.*

We prove that the relation $\succ$ on states is well-founded by proving that

$$\mathtt{e}_1 : \rho_1 \succ \mathtt{e}_2 : \rho_2 \text{ implies that } (H(\mathtt{e}_1 : \rho_1), L(\mathtt{e}_1)) >_{lex} (H(\mathtt{e}_2 : \rho_2), L(\mathtt{e}_2))$$

in the lexicographic order, where H gives the height of the environment and L gives the length of the expression. The proof is in the Appendix.

Lemma 5.5. *If $p \in gb(s)$ then $s \succeq_1 \overline{s}(p)$. If $p \in gb(s)$ and $p \neq \epsilon$ then $s \succ_1 \overline{s}(p)$.*

6. Size-change graphs that safely describe a program

6.1. **Safely describing state transitions.** We now define the arcs of the size-change graphs (recalling Definition 4.2):

Definition 6.1. A size-change graph G relating state s_1 to state s_2 has *source* $gb(s_1)$ and *target* $gb(s_2)$.

Definition 6.2. Let $s_1 = \mathtt{e}_1 : \rho_1$ and $s_2 = \mathtt{e}_2 : \rho_2$. Size-change graph $\mathtt{s}_1 \rightarrow \mathtt{s}_2, G$ is *safe*[3] for (s_1, s_2) if

$$p_1 \overset{=}{\rightarrow} p_2 \in G \text{ implies } \overline{s_1}(p_1) = \overline{s_2}(p_2) \text{ and } p_1 \overset{\downarrow}{\rightarrow} p_2 \in G \text{ implies } \overline{s_1}(p_1) \succ \overline{s_2}(p_2)$$

By $dom(G)$ we denote the subset of $source(G)$ from where arcs begin. By $codom(G)$ we denote the subset of $target(G)$ where arcs end. Notice that if a size-change graph G is safe for the states (s_1, s_2), then any subset size-change graph $G' \subset G$ with $source(G') = source(G)$ and $target(G') = target(G)$ is safe for (s_1, s_2).

Definition 6.3. A set $\mathcal{G}$ of size-change graphs is *safe for program* `P` if $\mathtt{P} : [] \rightarrow^* s_1 \rightarrow s_2$ implies some $G \in \mathcal{G}$ is safe for the pair (s_1, s_2).

Example 6.4. Figure 6 shows a graph set $\mathcal{G}$ that is safe for program $\Omega = (\lambda\mathtt{x}.\mathtt{x@x})(\lambda\mathtt{y}.\mathtt{y@y})$. For brevity, each subexpression of Ω is referred to by number in the diagram of $\mathcal{G}$. Subexpression $\mathtt{1} = \Omega$ has no free variables, so arcs from node `1` are labeled with size-change graphs $G_0 = \emptyset$.

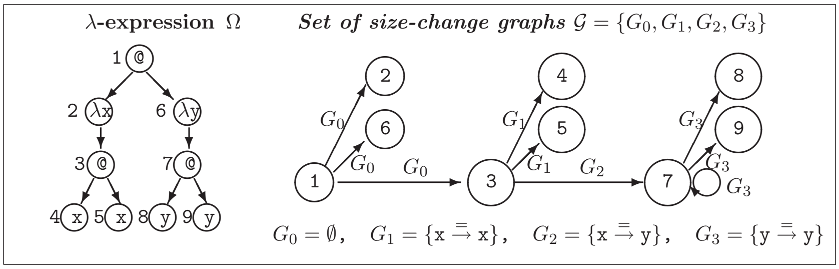

λ-expression Ω
Set of size-change graphs 𝒢 = {G0, G1, G2, G3}
1 @
2 λx
6 λy
3 @
7 @
4 x
5 x
8 y
9 y
G0
G1
G2
G3
G0 = ∅, G1 = {x =→ x}, G2 = {x =→ y}, G3 = {y =→ y}

Figure 6: A set of size-change graphs that safely describe Ω's nonterminating computation.

Theorem 6.5. *If $\mathcal{G}$ is safe for program* `P` *and satisfies the size-change condition, then call-by-value evaluation of* `P` *terminates.*

[3]The term "safe" comes from abstract interpretation [25]. An alternative would be "sound."

Proof. Suppose call-by-value-evaluation of P does not terminate. Then by Lemma 3.4 there is an infinite call chain

$$\mathtt{P} : [] = \mathtt{e}_0 : \rho_0 \to \mathtt{e}_1 : \rho_1 \to \mathtt{e}_2 : \rho_2 \to \dots$$

Letting $s_i = \mathtt{e}_i : \rho_i$, by safety of $\mathcal{G}$ (Definition 6.3), there is a size-change graph $G_i \in \mathcal{G}$ that safely describes each pair (s_i, s_{i+1}). By the size-change condition (Definition 4.6) the multipath $\mathcal{M} = G_0, G_1, \dots$ has an infinite thread $t = a_j \overset{r_j}{\to} a_{j+1} \overset{r_{j+1}}{\to} \dots$ such that $k \geq j$ implies $a_k \overset{r_k}{\to} a_{k+1} \in G_k$, and each r_k is $\downarrow$ or $=$, and there are infinitely many $r_k = \downarrow$. Consider the value sequence $\overline{s_j}(a_j), \overline{s_{j+1}}(a_{j+1}), \dots$. By safety of G_k (Definition 6.2) we have $\overline{s_k}(a_k) \succeq \overline{s_{k+1}}(a_{k+1})$ for every $k \geq j$, and infinitely many proper decreases $\overline{s_k}(a_k) \succ \overline{s_{k+1}}(a_{k+1})$. However this is impossible since by Lemma 5.4 the relation $\succ$ on *State* is well-founded.

Conclusion: call-by-value-evaluation of P terminates. □

The goal is partly achieved: We have found a sufficient condition on a set of size-change graphs to guarantee program termination. What we have not yet done is to find an algorithm to *construct* a size-change graph set $\mathcal{G}$ that is safe for P (The safety condition of Definition 6.3 is in general undecidable, so enumeration of all graphs won't work.) Our graph construction algorithm is developed in two stages:

- First, the exact evaluation and call relations are "instrumented" so as to produce safe size-change graphs during evaluation.
- Second, an extension of the abstract interpretation from Section 3.6 yields a *computable* over-approximation $\mathcal{G}$ that contains all graphs that can be built during exact evaluation.

6.2. Generating size-change graphs during a computation.

We now "instrument" the exact evaluation and call relations so as to produce safe size-change graphs during evaluation. In the definition of the size-change graphs x, y, z are variables, and p, q can be variables or ϵ, the empty path. Recall the valuation function for a state gives $\bar{s}(\epsilon) = s$, so in a sence ϵ is bound to the whole state.

Definition 6.6. (Evaluation and call with graph generation) The extended evaluation and call judgement forms are $\mathtt{e} : \rho \to \mathtt{e}' : \rho', G$ and $\mathtt{e} : \rho \Downarrow \mathtt{e}' : \rho', G$, where $source(G) = \text{fv}(\mathtt{e}) \cup \{\epsilon\}$ and $target(G) = \text{fv}(\mathtt{e}') \cup \{\epsilon\}$. The inference rules are:

$$\text{(ValueG)} \quad \frac{}{\lambda\mathtt{x.e} : \rho \Downarrow \lambda\mathtt{x.e} : \rho, id^{=}_{\lambda\mathtt{x.e}}}$$

$$\text{(OperatorG)} \quad \frac{}{\mathtt{e}_1@\mathtt{e}_2 : \rho \underset{r}{\to} \mathtt{e}_1 : \rho, id^{\downarrow}_{\mathtt{e}_1}} \qquad \text{(OperandG)} \quad \frac{\mathtt{e}_1 : \rho \Downarrow v_1}{\mathtt{e}_1@\mathtt{e}_2 : \rho \underset{d}{\to} \mathtt{e}_2 : \rho, id^{\downarrow}_{\mathtt{e}_2}}$$

$$\begin{array}{lll} id^{=}_{e} & \text{stands for} & \{\epsilon \overset{=}{\to} \epsilon\} \cup \{\mathtt{y} \overset{=}{\to} \mathtt{y} \mid \mathtt{y} \in fv(\mathtt{e})\} \\ id^{\downarrow}_{e} & \text{stands for} & \{\epsilon \overset{\downarrow}{\to} \epsilon\} \cup \{\mathtt{y} \overset{=}{\to} \mathtt{y} \mid \mathtt{y} \in fv(\mathtt{e})\} \end{array}$$

An arc $\mathtt{y} \overset{=}{\to} \mathtt{y}$ express that the state bound to the variable y is the same in both sides, before and after the evaluation or call.
The ϵ "represent" the whole state. In the (ValueG) rule the state $\lambda\mathtt{x.e} : \rho$ is the same in

both sides and so there is an arc $\epsilon \overset{=}{\to} \epsilon$. In the (OperatorG) and (OperandG) rules the state is smaller in the right hand side because we go to a strict subexpression and possibly also restrict the environment ρ accordingly. So there are $\epsilon \overset{\downarrow}{\to} \epsilon$ arcs.

$$\text{(VarG)} \qquad \frac{}{\mathtt{x} : \rho \Downarrow \rho(\mathtt{x}),\ \{\mathtt{x} \overset{\downarrow}{\to} \mathtt{y} \mid \mathtt{y} \in fv(\mathtt{e}')\ \} \cup \{\mathtt{x} \overset{=}{\to} \epsilon\}}\ (\rho(\mathtt{x}) = \mathtt{e}' : \rho')$$

In the (VarG) rule the state on the right side is $\rho(x)$. This is the state which x is bound to in the environment in the left hand side, therefore we have an arc $\mathtt{x} \overset{=}{\to} \epsilon$. Suppose $\rho(\mathtt{x}) = \mathtt{e}' : \rho'$ and $\mathtt{y} \in fv(\mathtt{e}')$. Then $\mathtt{y}$ is bound in ρ' and this binding is then a subtree of $\mathtt{e}' : \rho'$. So we have an arc $\mathtt{x} \overset{\downarrow}{\to} \mathtt{y}$.

$$\text{(CallG)} \qquad \frac{\mathtt{e}_1 : \rho \Downarrow \lambda\mathtt{x}.\mathtt{e}_0 : \rho_0, G_1 \qquad \mathtt{e}_2 : \rho \Downarrow v_2, G_2}{\mathtt{e}_1@\mathtt{e}_2 : \rho \underset{c}{\to} \mathtt{e}_0 : \rho_0[\mathtt{x} \mapsto v_2], G_1^{-\epsilon/\lambda x.e_0} \cup_{e_0} G_2^{\epsilon \mapsto \mathtt{x}}}$$

In the definition of the size-change graphs used in the (CallG) rule $\mathtt{x}$, $\mathtt{y}$, $\mathtt{z}$ are variables, and p, q can be variables or ϵ. In $\overset{r}{\to}$ the r can be either $\downarrow$ or $=$. The construction of the size-change graph associated with the call is explained below.

$G_1^{-\epsilon/\lambda x.e_0}$	stands for	cases
	$\mathtt{x} \in fv(\mathtt{e}_0)$	$: \{\mathtt{y} \overset{r}{\to} \mathtt{z} \mid \mathtt{y} \overset{r}{\to} \mathtt{z} \in G_1\} \cup \{\epsilon \overset{\downarrow}{\to} \mathtt{z} \mid \epsilon \overset{r}{\to} \mathtt{z} \in G_1\}$
	$\mathtt{x} \notin fv(\mathtt{e}_0)$	$: \{\mathtt{y} \overset{r}{\to} \mathtt{z} \mid \mathtt{y} \overset{r}{\to} \mathtt{z} \in G_1\} \cup \{\epsilon \overset{\downarrow}{\to} q \mid \epsilon \overset{r}{\to} q \in G_1\} \cup$ $\{p \overset{\downarrow}{\to} \epsilon \mid p \overset{r}{\to} \epsilon \in G_1\}$
$G_2^{\epsilon \mapsto \mathtt{x}}$	stands for	$\{\ \mathtt{y} \overset{r}{\to} \mathtt{x} \mid\ \mathtt{y} \overset{r}{\to} \epsilon \in G_2\} \cup \{\ \epsilon \overset{\downarrow}{\to} \mathtt{x} \mid\ \epsilon \overset{r}{\to} \epsilon \in G_2\ \}$
$G \cup_e G'$	stands for	the restriction of $G \cup G'$ such that the codomain $\subseteq fv(\mathtt{e}) \cup \{\epsilon\}$

First we consider how much information from G_1 we can preserve. We have that the whole state $\mathtt{e}_1@\mathtt{e}_2 : \rho$ in left hand side for the c-call is strictly larger than $\mathtt{e}_1 : \rho$. The variable $\mathtt{x}$ is not free in $\lambda\mathtt{x}.\mathtt{e}_0$ and so does not belong to the target of G_1. If a variable $\mathtt{z} \in \text{fv}(\lambda\mathtt{x}.\mathtt{e}_0)$ is bound in ρ_0 then it is bound to the same state in $\rho_0[\mathtt{x} \mapsto v_2]$. Therefore, if there is an arc $\mathtt{y} \overset{r}{\to} \mathtt{z}$ in G_1, then it also safely describes the c-call and can be preserved. Also, if there is an arc $\epsilon \overset{r}{\to} \mathtt{z}$ in G_1, then an arc $\epsilon \overset{\downarrow}{\to} \mathtt{z}$ describes the c-call. Further, if $\mathtt{x} \notin fv(\mathtt{e}_0)$ then $\mathtt{e}_0 : \rho_0[\mathtt{x} \mapsto v_2] = \mathtt{e}_0 : \rho_0$ and then $\lambda\mathtt{x}.\mathtt{e}_0 : \rho_0 \succ \mathtt{e}_0 : \rho_0[\mathtt{x} \mapsto v_2] = \mathtt{e}_0 : \rho_0$. In this case, if there is an arc $p \overset{r}{\to} \epsilon$ going to ϵ in G_1, then the arc $p \overset{\downarrow}{\to} \epsilon$ describes the c-call.
Now consider which information we can gain from G_2. We have that the whole state $\mathtt{e}_1@\mathtt{e}_2 : \rho$ in left hand side for the c-call is strictly larger than $\mathtt{e}_2 : \rho$. If $\mathtt{x} \in fv(\mathtt{e}_0)$ then in $\mathtt{e}_0 : \rho_0[\mathtt{x} \mapsto v_2]$ we have that $\mathtt{x}$ is bound to the whole state in the right hand side for the evaluation of the operand. So in this case, if there is an arc $\mathtt{y} \overset{r}{\to} \epsilon$ in G_2 then the arc $\mathtt{y} \overset{r}{\to} \mathtt{x}$ describes the c-call, and if there is an arc $\epsilon \overset{r}{\to} \epsilon$ in G_2 then the arc $\epsilon \overset{\downarrow}{\to} \mathtt{x}$ describes the c-call. If $\mathtt{x} \notin fv(\mathtt{e}_0)$ then we cannot gain any information from G_2. The restriction built into the definition of $\cup_{e_0}$ ensures that this holds.

$$\text{(ApplyG)}\quad \frac{\mathtt{e_1@e_2} : \rho \underset{c}{\rightarrow} \mathtt{e}' : \rho', G' \qquad \mathtt{e}' : \rho' \Downarrow v, G}{\mathtt{e_1@e_2} : \rho \Downarrow v, (G'; G)}$$

The size-change graph (G';G) is the composition of the two graphs.

In the size-change graphs generated by the rules above, the less-than relations ($x \overset{\downarrow}{\rightarrow} y$) in (VarG)-rule arise from the sub-environment property of $\succ_1$ from Lemma 5.5. The remaining relations $\overset{\downarrow}{\rightarrow}$ arise from the subexpression property of $\succ_2$. The relations based on the sub-environment property capture the case that the state on the right hand side is fetched from the environment in the left hand side. The equality relations $\overset{=}{\rightarrow}$ describe how values are preserved under calls and evaluations.

Lemma 6.7. $s \rightarrow s'$ (by Definition 2.5) *iff* $s \rightarrow s', G$ (by Definition 6.6) for some G. *Further,* $s \Downarrow s'$ *iff* $s \Downarrow s', G$ *for some* G*.*

Theorem 6.8. (The extracted graphs are safe)
$s \rightarrow s', G$ *or* $s \Downarrow s', G$ (by Definition 6.6) *implies* G *is safe for* (s, s') *(with source and target sets extended as necessary).*

Lemma 6.7 is immediate since the new rules extend the old, without any restriction on their applicability. Proof of "safety" Theorem 6.8 is in Appendix.

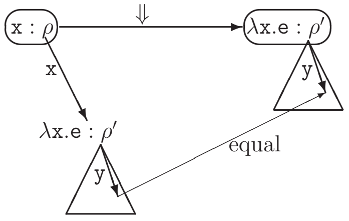


Figure 7: Data-flow in a variable evaluation

The diagram of Figure 7 illustrates the data-flow in a variable evaluation. The diagram of Figure 8 may be of some use in visualising data-flow during evaluation of $\mathtt{e_1@e_2}$. States are in ovals and triangles represent environments. In the application $\mathtt{e_1@e_2} : \rho$ on the left, operator $\mathtt{e_1} : \rho$ evaluates to $\lambda\mathtt{x.e_0} : \rho_0, G_1$ and operand $\mathtt{e_2} : \rho$ evaluates to $\mathtt{e}' : \rho', G_2$. The size-change graphs G_1 and G_2 show relations between variables bound in their environments. There is a call from the application $\mathtt{e_1@e_2} : \rho$ to $\mathtt{e_0} : \rho_0[\mathtt{x} \mapsto \mathtt{e}' : \rho']$ the body of the operator-value with the environment extended with a binding of $\mathtt{x}$ to the operand-value $\mathtt{e}' : \rho'$.

It is possible to approximate the calls and evaluates to relations with different degrees of precision depending on how much information is kept about the bindings in the

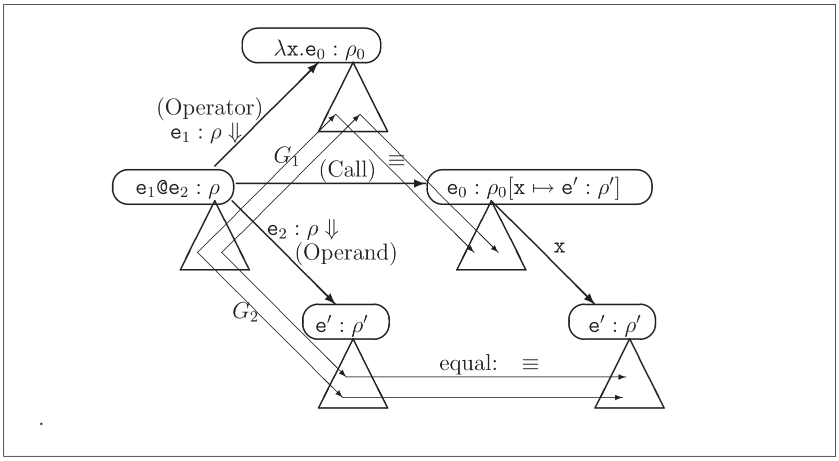


Figure 8: Data-flow in an application

environment. Here we aim at a coarse approximation, where we remove all environment components.[4]

6.3. **Construction of size-change graphs by abstract interpretation.** We now extend the coarse approximation to construct size-change graphs.

Definition 6.9. (Approximate evaluation and call with graph generation)
The judgement forms are now $e \to e', G$ and $e \Downarrow e', G$, where source$(G) = \text{fv}(e) \cup \{\epsilon\}$ and target$(G) = \text{fv}(e') \cup \{\epsilon\}$. The inference rules are:

$$\text{(ValueAG)}\ \frac{}{\lambda x.e \Downarrow \lambda x.e,\ id^{=}_{\lambda x.e}} \qquad \text{(VarAG)}\ \frac{e_1@e_2 \in \mathit{subexp}(P) \quad e_1 \Downarrow \lambda x.e_0, G_1 \qquad e_2 \Downarrow v_2, G_2}{x \Downarrow v_2,\ \{x \overset{\downarrow}{\to} y \mid y \in \mathit{fv}(v_2)\} \cup \{x \overset{=}{\to} \epsilon\}}$$

$$\text{(OperatorAG)}\ \frac{}{e_1@e_2 \underset{r}{\to} e_1, id^{\downarrow}_{e_1}} \qquad \text{(OperandAG)}\ \frac{}{e_1@e_2 \underset{d}{\to} e_2, id^{\downarrow}_{e_2}}$$

$$\text{(CallAG)}\ \frac{e_1 \Downarrow \lambda x.e_0, G_1 \qquad e_2 \Downarrow v_2, G_2}{e_1@e_2 \underset{c}{\to} e_0, G_1^{-\epsilon/\lambda x.e_0} \cup_{e_0} G_2^{\epsilon \mapsto x}} \qquad \text{(ApplyAG)}\ \frac{e_1@e_2 \underset{c}{\to} e', G' \qquad e' \Downarrow v, G}{e_1@e_2 \Downarrow v, G'; G}$$

Lemma 6.10. *Suppose* $P : [] \to^* e : \rho$. *If* $e : \rho \to e' : \rho', G$ *by definition 6.6 then* $e \to e', G$. *Further, if* $e : \rho \Downarrow e' : \rho', G$ *then* $e \Downarrow e', G$.

Proof. Follows from Lemma 3.11; see the Appendix. □

[4]It is possible to keep a little more information in the graphs than we do here even with no knowledge about value-bindings in the environment. We have chosen the given presentation for simplicity.

Definition 6.11.

$absint(\mathtt{P}) = \{\ G_j \mid j > 0 \wedge \exists \mathtt{e}_i, G_i (0 \leq i \leq j) : \mathtt{P} = \mathtt{e}_0 \wedge (\mathtt{e}_0 \rightarrow \mathtt{e}_1, G_1) \wedge \ldots \wedge (\mathtt{e}_{j-1} \rightarrow \mathtt{e}_j, G_j)\ \}$

Theorem 6.12.

(1) *The set* $absint(\mathtt{P})$ *is safe for* P.
(2) *The set* $absint(\mathtt{P})$ *can be effectively computed from* P.

Proof. Part 1: Suppose $\mathtt{P} : [] = s_0 \rightarrow s_1 \rightarrow \ldots \rightarrow s_j$. Theorem 6.8 implies $s_i \rightarrow s_{i+1}, G_i$ where each G_i is safe for the pair (s_i, s_{i+1}). Let $s_i = \mathtt{e}_i : \rho_i$. By Lemma 6.10, $\mathtt{e}_i \rightarrow \mathtt{e}_{i+1}, G_i$. By the definition of $absint(\mathtt{P})$, $G_j \in absint(\mathtt{P})$.

Part 2: There is only a fixed number of subexpressions of P, or of possible size-change graphs with source and target $\subseteq \{\epsilon\} \cup \{\mathtt{x} \mid \mathtt{x}$ is a variable in P $\}$. Thus $absint(\mathtt{P})$ can be computed by applying Definition 6.9 exhaustively, starting with P, until no new graphs or subexpressions are obtained. □

7. Some examples

7.1. A simple example.

Using Church numerals ($\mathtt{n} = \lambda \mathtt{s} \lambda \mathtt{z}.\mathtt{s}^n(\mathtt{z})$), we expect `2 succ 0` to reduce to `succ(succ 0)`. However this contains unreduced redexes because call-by-value does not reduce under a λ, so we force the computation to carry on through by applying `2 succ 0` to the identity (twice). This gives:

```
2 succ 0 id1 id2 where
  succ = λm.λs.λz. m s (s z)
  id1  = λx.x
  id2  = λy.y
```

After writing this out in full as a λ-expression, our analyser yields (syntactically sugared):

```
     [λs2.λz2.(s2 @ (s2 @ z2))]        -- two --
  @  [λm.λs.λz.  15: ((m@s)@(s@z))]    -- succ --
  @  [λs1.λz1.z1]                      -- zero --
  @  [λx.x]                            -- id1 --
  @  [λy.y]                            -- id2 --
```

```
Output of loops from an analysis of this program:

15→* 15: [(m,>,m),(s,=,s),(z,=,z)], []

Size Change Termination: Yes
```

The first number refers to the program point, then comes a list of edges. The loop occurs because application of 2 forces the code for the successor function to be executed twice, with decreasing argument values m. The notation for edges is a little different from previously, here `(m,>,m)` stands for $\mathtt{m} \overset{\downarrow}{\rightarrow} \mathtt{m}$.

7.2. $fnx = x + 2^n$ by Church numerals.

This more interesting program computes $fnx = x + 2^n$ by higher-order primitive recursion. If n is a Church numeral then expression `n g x` reduces to $\mathtt{g}^n(\mathtt{x})$. Let x be the successor function, and g be a "double application" functional. Expressed in a readable named combinator form, we get:

```
f n x    where
f n   =  if n=0 then succ else g(f(n-1))
g r a =  r(ra)
```

As a lambda-expression (applied to values $n = 3, x = 4$) this can be written:

```
[λn.λx.  n                                          -- n --
         @  [λr.λa. 11: (r@ 13: (r@a))]             -- g --
         @  [λ k.λ s.λ z.(s@((k@s)@z))]             - succ-
         @  x ]                                     -- x --

@           [λs2.λz2.  (s2@(s2@(s2@z2))) ]          -- 3 --
@           [λs1.λz1.  (s1@(s1@(s1@(s1@z1))))]      -- 4 --
```

Following is the output from program analysis. The analysis found the following loops from a program point to itself with the associated size-change graph and path. The first number refers to the program point, then comes a list of edges and last a list of numbers, the other program points that the loop passes through.

```
SELF Size Change Graphs, no repetition of graphs:

11 →* 11: [(r,>,r)]             []
11 →* 11: [(a,=,a),(r,>,r)]     [13]
13 →* 13: [(a,=,a),(r,>,r)]     [11]
13 →* 13: [(r,>,r)]             [11,11]

Size Change Termination: Yes
```

7.3. **Ackermann's function, second-order.** This can be written without recursion using Church numerals as: `a m n` where `a = λm. m b succ` and `b = λg. λn. n g (g 1)`. Consequently `a m =` b^m`(succ)` and `b g n =` g^{n+1}`(1)`, which can be seen to agree with the usual first-order definition of Ackermann's function. Following is the same as a lambda-expression applied to argument values `m=2, n=3`, with numeric labels on some subexpressions.

```
(λm.m b succ) 2 3  =  (λm.m@b@succ)@2@3
(λm.m@(λg.λn.n@g@(g@1))@succ)@2@3
(λm.m@(λg.λn. 9: (n@g@ 13: (g@1)))@succ)@2@3
 where
  1    =  λs1.λz1.  17: (s1@z1)
succ =  λk.λs.λz.  23: (s@ 25: (k@s@z))
  2    =  λs2.λz2. s2@(s2@z2)
  3    =  λs3.λz3.  39: (s3@ 41: (s3@ 43: (s3@z3)))
```

Output from an analysis of this program is shown here.

```
SELF Size Change Graphs, no repetition of graphs:
(Because graphs are only taken once it is not always the case that the
same loop is shown for all program points in its path)

 9 →*  9: [(ϵ,>,n),(g,>,g)]            [13]
 9 →*  9: [(g,>,g)]                    [17]
13 →* 13: [(g,>,g)]                    [9]
17 →* 17: [(s1,>,s1)]                  [9]
23 →* 23: [(k,>,k),(s,=,s),(z,=,z)]    [25]
23 →* 23: [(s,>,s)]                    [9]
23 →* 23: [(s,>,s),(z,>,k)]            [25,17,9]
25 →* 25: [(k,>,k),(s,=,s),(z,=,z)]    [23]
25 →* 25: [(s,>,s),(z,>,k)]            [17,9,23]
25 →* 25: [(s,>,s)]                    [23,9,23]
39 →* 39: [(s3,>,s3)]                  [9]
41 →* 41: [(s3,>,s3)]                  [9,39]
43 →* 43: [(s3,>,s3)]                  [9,39,41]

Size Change Termination: Yes
```

7.4. **Arbitrary natural numbers as inputs.** The astute reader may have noticed a limitation in the above examples: each only concerns *a single λ-expression*, e.g., Ackermann's function applied to argument values `m=2, n=3`.

In an implemented version of the λ-termination analysis a program may have an arbitrary natural number as input; this is represented by •. Further, programs can have as constants the predecessor, successor and zero-test functions, and if-then-else expressions. We show, by some examples using •, that the size-change termination approach can handle the Y-combinator.

In Section 8 we show how to do size-change analysis of λ-expressions applied to *sets of argument values* in a more classic context, using Church or other numeral notations instead of •.

7.5. **A minimum function, with general recursion and Y-combinator.** This program computes the minimum of its two inputs using the call-by-value combinator $Y = \lambda$`p. [λq.p@(λs.q@q@s)] @ [λt.p@(λu.t@t@u)]`. The program, first as a first-order recursive definition.

```
m x y = if x=0 then 0 else if y=0 then 0 else succ (m (pred x) (pred y))
```

Now, in λ-expression form for analysis.

```
    {λp.  [λq.p@(λs.q@q@s)] @ [λt.p@(λu.t@t@u)]}   -- the Y combinator --
    @
    [ λm.λx.λy.  27: if(  (ztst @ x),
                          0,
                          32:    if(   (ztst @ y),
                                       0,
                                       37: succ @  39:  m @ (pred@x) @ (pred@y) ]
    @ •
    @ •
```

```
Output of loops from an analysis of this program:

27 →* 27: [(x,>,x),(y,>,y)]    [32,37,39]
32 →* 32: [(x,>,x),(y,>,y)]    [37,39,27]
37 →* 37: [(x,>,x),(y,>,y)]    [39,27,32]
39 →* 39: [(x,>,x),(y,>,y)]    [27,32,37]

Size Change Termination: Yes
```

7.6. **Ackermann's function, second-order with constants and Y-combinator.** Ackermann's function can be written as: `a m n` where `a m = b`^{m}`(suc)` and `b g n = g`$^{n+1}$`(1)`. The following program expresses the computations of both `a` and `b` by loops, using the Y combinator (twice).

```
    [λ y.λ y1.
    (y1 @
    λ a.λ m.  11: if(  (ztst@m),
                       λ v.(suc@v),
                       19: (  (y @
                              λ b.λ f.λ n.
                              25: if(  (ztst@n),
                                       29: (f@1),
                                       32: f@ 34: b @ f @ (pred@n))
                       @  41:  a @ (pred@m)                          ]

    @ {λp.  [λq.p@(λs.  q@q@s)] @ [λt.p@(λu.  t@t@u)]}
    @ {λp1.  [λq1.p1@(λs.  72: q1@1q@s1)] @ [λt1.p1@(λu1.  81: t1@t1@u1)]}
    @ •
    @ •
```

```
Output of loops from an analysis of this program:

SELF Size Change Graphs no repetition of graphs:
```

```
11 →* 11: [(a,>,y),(m,>,m)]     [19,41,72]
11 →* 11: [(m,>,m)]             [19,41,72,11,19,41,72]
19 →* 19: [(a,>,y),(m,>,m)]     [41,72,11]
19 →* 19: [(m,>,m)]             [41,72,11,19,41,72,11]
25 →* 25: [(f,>,b),(f,>,f)]     [29]
25 →* 25: [(f,=,f),(n,>,n)]     [32,34]
25 →* 25: [(f,>,f)]             [29,25,32,34]
29 →* 29: [(f,>,f)]             [25]
32 →* 32: [(f,>,b),(f,>,f)]     [25]
32 →* 32: [(f,=,f),(n,>,n)]     [34,25]
32 →* 32: [(f,>,f)]             [25,32,34,25]
34 →* 34: [(f,=,f),(n,>,n)]     [25,32]
34 →* 34: [(f,>,b),(f,>,f)]     [25,29,25,32]
34 →* 34: [(f,>,f)]             [25,29,25,32,34,25,32]
41 →* 41: [(m,>,m)]             [72,11,19]
72 →* 72: [(s1,>,s1)]           [11,19,41]
81 →* 81: [(u1,>,u1)]           [11,19,41]

Size Change Termination: Yes
```

7.7. **Imprecision of abstract interpretation.** It is natural to wonder whether the gross approximation of Definition 3.10 comes at a cost. The (VarA) rule can in effect "mix up" different function applications, losing the coordination between operator and operand that is present in the exact semantics.

We have observed this in practice: The first time we had programmed Ackermann's using explicit recursion, we used the same instance of Y-combinator for both loops, so the single Y-combinator expression was "shared". The analysis did not discover that the program terminated.

However when this was replaced by the "unshared" version above, with two instances of the Y-combinator (`y` and `y1`) (one for each application), the problem disappeared and termination was correctly recognised.

7.8. **A counterexample to a conjecture.** Sereni disproved in [48, 49] our **conjecture** that the size-change method would recognise as terminating any simply typed λ-expression. The root of the problem is the imprecision of abstract interpretation just noted. A counterexample: the λ-expression

$$E = (\lambda a.a(\lambda b.a(\lambda cd.d)))(\lambda e.e(\lambda f.f))$$

is simply-typable but not size-change terminating. Its types are any instantiation of

$$\begin{array}{lcl} a & : & ((\tau \to \tau) \to \mu \to \mu) \to \mu \to \mu \\ b, c & : & \tau \to \tau \\ d & : & \mu \\ e & : & (\tau \to \tau) \to \mu \to \mu \\ f & : & \tau \end{array}$$

8. Arbitrary λ-regular program inputs (Extended λ-calculus)

Above we have analysed the termination behaviour of a single closed λ-expression. We now analyse the termination behaviour for a program in the λ-calculus for all possible inputs from a given input-set of λ-expressions (e.g., Church numerals). The first step is to define which sets of λ-expressions we consider. A well-defined input set will be the set of closed expressions in the "language" generated by a λ-regular grammar.

We extend the syntax and semantics of the λ-calculus to handle expressions containing nonterminals. An extended lambda term represents all instances of a program with input taken from the input set. If our analysis certifies that the extended term terminates, then this implies that the program will terminate for all possible inputs.

8.1. λ-regular grammars.

We are interested in a λ-regular grammar for the sake of the *language* that it generates: a set of pure λ-expressions (without nonterminals). This is done using the derivation relation $\Rightarrow^*_\Gamma$, soon to be defined.

Definition 8.1.

(1) A *λ-regular grammar* has form $\Gamma = (N, \Pi)$ where N is a finite set of *nonterminal* symbols and Π is a finite set of *productions*.

(2) A *Γ-extended λ-expression* has the following syntax:

$$\begin{array}{lcl} \mathtt{e,\ P} & ::= & \mathtt{x} \mid \mathtt{A} \mid \mathtt{e\ @\ e} \mid \lambda\mathtt{x.e} \\ \mathtt{A} & ::= & \text{Non-terminal name, } \mathtt{A} \in N \\ \mathtt{x} & ::= & \text{Variable name} \end{array}$$

Exp_Γ denotes the set of Γ-extended λ-expressions. Exp denotes the set of pure λ-expressions (without nonterminals). Clearly $Exp_\Gamma \supseteq Exp$.

(3) A production has form $\mathtt{A} ::= \mathtt{e}$ where $\mathtt{e}$ is a Γ-extended λ-expression.

Definition 8.2. Let $\mathtt{nt(e)} = \{\mathtt{X}_1, \ldots, \mathtt{X}_k\}$ denote the *multi-set* of nonterminal occurrences in $\mathtt{e} \in Exp_\Gamma$. The *derivation relation* $\Rightarrow^*_\Gamma \subseteq Exp_\Gamma \times Exp$ is the smallest relation such that

(1) If $\mathtt{nt(e)} = \{\mathtt{X}_1, \ldots, \mathtt{X}_k\}$ and $\mathtt{X}_i \Rightarrow^*_\Gamma \mathtt{t}_i \in Exp$ for $i = 1, \ldots, k$, then $\mathtt{e} \Rightarrow^*_\Gamma \mathtt{e}[\mathtt{t}_1/\mathtt{X}_1, \ldots, \mathtt{t}_k/\mathtt{X}_k]$

(2) If $\mathtt{A} ::= \mathtt{e} \in \Gamma$ and $\mathtt{e} \Rightarrow^*_\Gamma \mathtt{e}'$ then $\mathtt{A} \Rightarrow^*_\Gamma \mathtt{e}'$.

Notice that $\Rightarrow^*_\Gamma$ relates extended λ-terms to pure λ-terms.

In the above definition 8.2 $\mathtt{nt(e)} = \{\mathtt{X}_1, \ldots, \mathtt{X}_k\}$ denotes the multi-set of nonterminals in $\mathtt{e}$ so two different $\mathtt{X}_i, \mathtt{X}_j$ may be instances of the same nonterminal $\mathtt{A}$. In the substitution $\mathtt{e}[\mathtt{t}_1/\mathtt{X}_1, \ldots, \mathtt{t}_k/\mathtt{X}_k]$ such two different instances of a nonterminal may be replaced by different pure λ-terms.

Example 8.3. A grammar for Church Numerals: Consider

$$\Gamma = (\{\mathtt{C}, \mathtt{A}\}, \{\mathtt{C} ::= \lambda\,\mathtt{s}\lambda\,\mathtt{z}\,.\,\mathtt{A},\ \mathtt{A} ::= \mathtt{z},\ \mathtt{A} ::= \mathtt{s}\,@\,\mathtt{A}\}$$

Here $\mathtt{A} \Rightarrow^*_\Gamma \mathtt{v}$ iff $\mathtt{v}$ has form $\mathtt{s}^n(\mathtt{z})$ for some $n \geq 0$. Clearly $\mathtt{C} \Rightarrow^*_\Gamma \mathtt{v}$ iff $\mathtt{v}$ has form $\lambda\,\mathtt{s}\lambda\,\mathtt{z}\,.\,\mathtt{s}^n(\mathtt{z})$ for some $n \geq 0$.

The following assumption makes proofs more convenient; proof is standard and so omitted.

Lemma 8.4. *For any λ-regular grammar Γ_0 there exists an equivalent λ-regular grammar Γ_1 such that no production in Γ_1 has form $\mathtt{A} ::= \mathtt{A}'$ where $\mathtt{A}' \in N$. We henceforth assume that all productions in a λ-regular grammar have form $\mathtt{A} ::= \mathtt{e}$ where $\mathtt{e} \notin N$.*

Definition 8.5. In the following $\mathtt{e}$ is a Γ-extended λ-expression:

(1) Define the *free variables of* $\mathtt{e}$ by $fv(\mathtt{e}) = \{\mathtt{x} \mid \exists \mathtt{t}.\mathtt{e} \Rightarrow^*_\Gamma \mathtt{t} \text{ and } \mathtt{x} \in fv(\mathtt{t})\}$

(2) Define that $\mathtt{e}$ is *closed* iff t is closed for all t such that $e \Rightarrow^*_\Gamma t$. It follows that $\mathtt{e}$ is closed iff $fv(\mathtt{e}) = \{\}$.

(3) Define *subterms*$(\mathtt{e})$ inductively by:
For a variable $\mathtt{x}$: $subterms(\mathtt{x}) = \{\mathtt{x}\}$.
For an abstraction $\lambda\mathtt{x}.\mathtt{e}$: $subterms(\lambda\mathtt{x}.\mathtt{e}) = \{\lambda\mathtt{x}.\mathtt{e}\} \cup subterms(\mathtt{e})$.
For an application $\mathtt{e}_1\mathtt{@}\mathtt{e}_2$: $subterms(\mathtt{e}_1\mathtt{@}\mathtt{e}_2) = \{\mathtt{e}_1\mathtt{@}\mathtt{e}_2\} \cup subterms(\mathtt{e}_1) \cup subterms(\mathtt{e}_2)$.
For a nonterminal $\mathtt{A}$: $subterms(\mathtt{A}) = \{\mathtt{A}\}$.

(4) Define *subexps*$(\mathtt{e})$ as the smallest set satisfying:
For a variable $\mathtt{x}$: $subexps(\mathtt{x}) = \{\mathtt{x}\}$.
For an abstraction $\lambda\mathtt{x}.\mathtt{e}$: $subexps(\lambda\mathtt{x}.\mathtt{e}) = \{\lambda\mathtt{x}.\mathtt{e}\} \cup subexps(\mathtt{e})$.
For an application $\mathtt{e}_1\mathtt{@}\mathtt{e}_2$: $subexps(\mathtt{e}_1\mathtt{@}\mathtt{e}_2) = \{\mathtt{e}_1\mathtt{@}\mathtt{e}_2\} \cup subexps(\mathtt{e}_1) \cup subexps(\mathtt{e}_2)$.
For a nonterminal $\mathtt{A}$: $subexps(\mathtt{A}) = \{\mathtt{A}\} \cup \{\mathtt{t} \mid \exists \mathtt{e}.\mathtt{A} ::= \mathtt{e} \in \Gamma \text{ and } \mathtt{t} \in subexps(\mathtt{e})\}$.

If $\mathtt{e}' \in subterms(\mathtt{e})$ then $\mathtt{e}'$ is syntactically present as part of $\mathtt{e}$.
If $\mathtt{e}' \in subexps(\mathtt{e})$ then $\mathtt{e}'$ is either a subterm of $\mathtt{e}$ or a subexpression of a nonterminal $\mathtt{A} \in subterms(\mathtt{e})$.

Sets $subterms(\mathtt{e}), subexps(\mathtt{e})$ are both finite, and $subterms(\mathtt{e}) = subexps(\mathtt{e})$ for expressions $\mathtt{e}$ in the pure λ-calculus.

Example 8.6. In the grammar for Church Numerals $\mathtt{C}$ is a closed Γ-extended expression, but $\mathtt{A}$ is not a closed Γ-extended expression. Further, $subexps(\mathtt{A}) = \{\mathtt{A},\ \mathtt{z},\ \mathtt{s@A},\ \mathtt{s}\}$, $subexps(\mathtt{C}) = \{\mathtt{C},\ \lambda\,\mathtt{s}\lambda\,\mathtt{z}\,.\,\mathtt{A},\ \lambda\,\mathtt{z}\,.\,\mathtt{A},\ \mathtt{A},\ \mathtt{z},\ \mathtt{s@A},\ \mathtt{s}\}$, $fv(\mathtt{C}) = \{\}$, $fv(\mathtt{A}) = \{\mathtt{s},\mathtt{z}\}$

Lemma 8.7. *Let $\mathtt{x}$ be a variable. If $\mathtt{A} \Rightarrow^*_\Gamma \mathtt{x}$ then $\mathtt{A} ::= \mathtt{x} \in \Gamma$.*
If $\mathtt{A} \Rightarrow^*_\Gamma \lambda\mathtt{x}.\mathtt{e}$ then there exists $\mathtt{e}' \in Exp_\Gamma$ such that $\mathtt{A} ::= \lambda\mathtt{x}.\mathtt{e}' \in \Gamma$.
If $\mathtt{A} \Rightarrow^*_\Gamma \mathtt{e}_1\mathtt{@}\mathtt{e}_2$ then there exist $\mathtt{e}_1', \mathtt{e}_2' \in Exp_\Gamma$ such that $\mathtt{A} ::= \mathtt{e}_1'\mathtt{@}\mathtt{e}_2' \in \Gamma$.

Any production has one of the forms $\mathtt{A} ::= \mathtt{x}$, $\mathtt{A} ::= \lambda\mathtt{x}.\mathtt{e}$, $\mathtt{A} ::= \mathtt{e}_1\mathtt{@}\mathtt{e}_2$. No production performed on a subterm (which must be a nonterminal) can give a new outermost syntactic term-constructor.

The following Lemma follows from the definition of free variables of an extended expression.

Lemma 8.8. *For a variable $\mathtt{x}$: $fv(\mathtt{x}) = \{\mathtt{x}\}$.*
For an abstraction $\lambda\mathtt{x}.\mathtt{e}$: $fv(\lambda\mathtt{x}.\mathtt{e}) = fv(\mathtt{e}) \setminus \{\mathtt{x}\}$.
For an application $\mathtt{e}_1\mathtt{@}\mathtt{e}_2$: $fv(\mathtt{e}_1\mathtt{@}\mathtt{e}_2) = fv(\mathtt{e}_1) \cup fv(\mathtt{e}_2)$.
For a nonterminal $\mathtt{A} \in N$: $fv(\mathtt{A}) = \{\mathtt{x} \mid \exists \mathtt{t}.\mathtt{A} \Rightarrow^*_\Gamma \mathtt{t} \text{ and } \mathtt{x} \in fv(\mathtt{t})\}$.

Lemma 8.9. *For $\mathtt{A} \in N$ the sets $subexps(\mathtt{A})$ and $fv(\mathtt{A})$ are finite and computable.*

Proof is straightforward.

8.2. **Extended environment-based semantics.** A semantics extending Definition 3.3 addresses the problem of substitution in expressions with non-terminals. Environments bind λ-variables (and not non-terminals) to values.

Definition 8.10. (Extended states, values and environments) *State*, *Value*, *Env* are the smallest sets such that

$$\begin{array}{lcllll}
\mathit{State} & = & \{ & \mathtt{e} : \rho & \mid & \mathtt{e} \in \mathit{Exp}_\Gamma, \rho \in \mathit{Env} \text{ and } fv(\mathtt{e}) \subseteq \mathit{dom}(\rho) \ \} \\
\mathit{Value} & = & \{ & \lambda\mathtt{x.e} : \rho & \mid & \lambda\mathtt{x.e} : \rho \in \mathit{State} \ \} \\
\mathit{Env} & = & \{ & \rho : X \to \mathit{Value} & \mid & X \text{ is a finite set of variables} \ \}
\end{array}$$

The empty environment with domain $X = \emptyset$ is written $[]$. The evaluation judgement form is $s \Downarrow v$ where $s \in \mathit{State}, v \in \mathit{Value}$.

The following rules for calls and evaluations in the extended language are simple extensions of the rules for pure λ-calculus to also handle nonterminals.

Definition 8.11. (Extended environment-based evaluation) The judgement forms are $\mathtt{e} : \rho \to \mathtt{e}' : \rho'$ and $\mathtt{e} : \rho \Downarrow \mathtt{e}' : \rho'$, where $\mathtt{e}, \mathtt{e}' \in \mathit{Exp}_\Gamma$, $e : \rho$ and $e' : \rho'$ are states. The evaluation and call relations $\Downarrow, \to$ are defined by the following inference rules, where $\to = \underset{r}{\to} \cup \underset{d}{\to} \cup \underset{c}{\to} \cup \underset{n}{\to}$.

(GramX) $$\frac{}{A : \rho \underset{n}{\to} \mathtt{e} : \rho} \ (A ::= \mathtt{e} \in \Gamma)$$ New rule

(ResultX) $$\frac{\mathtt{e} : \rho \underset{x}{\to} \mathtt{e}' : \rho' \qquad \mathtt{e}' : \rho' \Downarrow v}{\mathtt{e} : \rho \Downarrow v} \ (x \in \{c, n\})$$ Extended **Def.** 3.3 (Apply)

The following rules have not been changed (but now expressions belong to Exp_Γ).

(ValueX) $$\frac{}{\lambda\mathtt{x.e} : \rho \Downarrow \lambda\mathtt{x.e} : \rho}$$ (VarX) $$\frac{}{\mathtt{x} : \rho \Downarrow e' : \rho'} \ (\rho(x) = e' : \rho')$$

(OperatorX) $$\frac{}{\mathtt{e}_1@\mathtt{e}_2 : \rho \underset{r}{\to} \mathtt{e}_1 : \rho}$$ (OperandX) $$\frac{\mathtt{e}_1 : \rho \Downarrow v_1}{\mathtt{e}_1@\mathtt{e}_2 : \rho \underset{d}{\to} \mathtt{e}_2 : \rho}$$

(CallX) $$\frac{\mathtt{e}_1 : \rho \Downarrow \lambda\mathtt{x.e}_0 : \rho_0 \qquad \mathtt{e}_2 : \rho \Downarrow v_2}{\mathtt{e}_1@\mathtt{e}_2 \underset{c}{\to} \mathtt{e}_0 : \rho_0[x \mapsto v_2]}$$

A Γ-extended *program* is a closed expression $\mathtt{P} \in \mathit{Exp}_\Gamma$. While evaluating a program in the extended language ($\mathtt{P} : [] \Downarrow \_$), all calls and subevaluations will be from state to state.

In pure λ-calculus the evaluation relation is deterministic. The extended language is nondeterministic since a nonterminal `A` may have `A` ::= `e` for more than one `e`.

Informally explained, consider closed extended λ-expression `e@B` where nonterminal `B` satisfies $fv(\mathtt{B}) = \{\}$. Then `e@B` represents application of `e` to all possible inputs generated by `B`. The analysis developed below can safely determine that `e` terminates on all inputs by analysing `e@B`.

If a program in the extended language takes more than one input at a time, then we may rename the nonterminals and bound variables similarly as in α-conversion. As an example, if a program takes two Church numerals as input, then they can be given by two grammars identical in structure:

$\mathtt{C}_1 ::= \lambda \mathtt{s}_1.\lambda \mathtt{z}_1.\mathtt{A}_1 \quad \mathtt{A}_1 ::= \mathtt{z}_1 \quad \mathtt{A}_1 ::= \mathtt{s}_1\mathtt{@}\mathtt{A}_1$ and
$\mathtt{C}_2 ::= \lambda \mathtt{s}_2.\lambda \mathtt{z}_2.\mathtt{A}_2 \quad \mathtt{A}_2 ::= \mathtt{z}_2 \quad \mathtt{A}_2 ::= \mathtt{s}_2\mathtt{@}\mathtt{A}_2$

and we can analyse the termination behaviour for $(\mathtt{e@C}_1)\mathtt{@C}_2$. Such renaming can sometimes make the termination analysis more precise.

Definition 8.12. Suppose $\mathtt{e}$ is a closed Γ-extended expression and $nt(\mathtt{e}) = \{\mathtt{A}_1, \ldots, \mathtt{A}_k\}$ where $\Gamma = (N, \Pi)$ is a λ-regular grammar. By definition $\mathtt{e}$ is Γ-*terminating* iff

$$\mathtt{e}[\mathtt{t}_1/\mathtt{A}_1, \ldots, \mathtt{t}_k/\mathtt{A}_k] : [] \Downarrow$$

for all pure λ-expressions $\mathtt{t}_1, \ldots, \mathtt{t}_k$ such that $\mathtt{A}_i \Rightarrow^*_\Gamma \mathtt{t}_i$ for $i = 1, \ldots, k$.

The following rules for calls and evaluations with size-change graphs in the extended language are simple extensions of the rules for pure λ-calculus to also handle nonterminals.

Definition 8.13. (Environment-based evaluation and call semantics with size-change graphs) The judgement forms are $\mathtt{e} : \rho \to \mathtt{e}' : \rho', G$ and $\mathtt{e} : \rho \Downarrow \mathtt{e}' : \rho', G$, where $\mathtt{e}, \mathtt{e}' \in Exp_\Gamma$, $e : \rho$ and $e' : \rho'$ are states, $source(G) = fv(e) \cup \{\epsilon\}$ and $target(G) = fv(e') \cup \{\epsilon\}$. The evaluation and call relations $\Downarrow, \to$ are defined by the following inference rules, where $\to = \underset{r}{\to} \cup \underset{d}{\to} \cup \underset{c}{\to} \cup \underset{n}{\to}$.

(GramG) $$\frac{}{A : \rho \underset{n}{\to} \mathtt{e} : \rho, id^{=}_{e}} \; (A ::= \mathtt{e} \in \Gamma)$$ New rule

(ResultG) $$\frac{\mathtt{e} : \rho \underset{x}{\to} \mathtt{e}' : \rho', G' \qquad \mathtt{e}' : \rho' \Downarrow v, G}{\mathtt{e} : \rho \Downarrow v, G'; G} \; (x \in \{c, n\})$$ Ext. **Def.** 6.6 (ApplyG)

The following rules have not been changed (but now expressions belong to Exp_Γ).

(ValueG) $$\frac{}{\lambda\mathtt{x.e} : \rho \Downarrow \lambda\mathtt{x.e} : \rho, id^{=}_{\lambda\mathtt{x.e}}}$$

(VarG) $$\frac{}{\mathtt{x}\text{:}\ \rho \Downarrow e' : \rho', \{\mathtt{x} \overset{=}{\to} \epsilon\} \cup \{\mathtt{x} \overset{\downarrow}{\to} \mathtt{y} \mid \mathtt{y} \in fv(e')\}} \; (\rho(x) = e' : \rho')$$

(OperatorG) $$\frac{}{\mathtt{e}_1\mathtt{@e}_2 : \rho \underset{r}{\to} \mathtt{e}_1 : \rho, id^{\downarrow}_{\mathtt{e}_1}}$$ (OperandG) $$\frac{\mathtt{e}_1 : \rho \Downarrow v_1}{\mathtt{e}_1\mathtt{@e}_2 : \rho \underset{d}{\to} \mathtt{e}_2 : \rho, id^{\downarrow}_{\mathtt{e}_2}}$$

(CallG) $$\frac{\mathtt{e}_1 : \rho \Downarrow \lambda\mathtt{x.e}_0 : \rho_0, G_1 \qquad \mathtt{e}_2 : \rho \Downarrow v_2, G_2}{\mathtt{e}_1\mathtt{@e}_2 \underset{c}{\to} \mathtt{e}_0 : \rho_0[x \mapsto v_2], G_1^{-\epsilon/\lambda x.e_0} \cup_{e_0} G_2^{\epsilon \mapsto \mathtt{x}}}$$

Theorem 8.14. (The extracted graphs are safe) *$s \to s', G$ or $s \Downarrow s', G$ implies G is safe for (s, s').*

Proof. This is shown by a case analysis as in the pure λ-calculus. For the (GramG) rule it is immediate from the definition of free variables for non-terminals. □

8.3. **Relating extended and pure λ-calculus.**

The aim is now to show that execution of a program `P` in the extended language can simulate execution of any program `Q` in the pure λ-calculus, where `Q` is derived from `P` by replacing each nonterminal occurrence `A` in `P` with a pure λ-expression `A` can produce. The converse does not hold: it is possible that there are simulated executions that do not correspond to any instantiated program `Q`. We have however certified a number of programs to terminate when applied to arbitrary Church numerals. An example is given at the end of this section.

Properties of the relation $\Rightarrow^*_\Gamma$

$\Rightarrow^*_\Gamma$ relates expressions $\mathtt{e}' \in \mathit{Exp}_\Gamma$ in the extended language to expressions $\mathtt{e} \in \mathit{Exp}$ in the pure lambda-calculus. Notice that there are only the following possible forms of $\Rightarrow^*_\Gamma$-related expressions:

$\mathtt{x} \Rightarrow^*_\Gamma \mathtt{x}$	$\lambda\mathtt{x.e}' \Rightarrow^*_\Gamma \lambda\mathtt{x.e}$	$\mathtt{e}'_1\mathtt{@e}'_2 \Rightarrow^*_\Gamma \mathtt{e}_1\mathtt{@e}_2$
$\mathtt{A} \Rightarrow^*_\Gamma \mathtt{x}$	$\mathtt{A} \Rightarrow^*_\Gamma \lambda\mathtt{x.e}$	$\mathtt{A} \Rightarrow^*_\Gamma \mathtt{e}_1\mathtt{@e}_2$

The relation $\Rightarrow^*_\Gamma$ has the following inductive properties:

$\mathtt{A} \Rightarrow^*_\Gamma \mathtt{t}$, for $\mathtt{A} \in N$ is given by definition 8.2.
$\mathtt{x} \Rightarrow^*_\Gamma \mathtt{x}$, – a variable `x` corresponds to the same variable `x` and nothing else.
$\lambda\mathtt{x.e}' \Rightarrow^*_\Gamma \lambda\mathtt{x.e}$, iff $\mathtt{e}' \Rightarrow^*_\Gamma \mathtt{e}$, same `x`.
$\mathtt{e}'_1\mathtt{@e}'_2 \Rightarrow^*_\Gamma \mathtt{e}_1\mathtt{@e}_2$ iff $\mathtt{e}'_1 \Rightarrow^*_\Gamma \mathtt{e}_1$ and $\mathtt{e}'_2 \Rightarrow^*_\Gamma \mathtt{e}_2$.

Lemma 8.15. *If $\mathtt{e}' \Rightarrow^*_\Gamma \mathtt{e}$ then $\mathit{fv}(\mathtt{e}') \supseteq \mathit{fv}(\mathtt{e})$.*

Proof. This is by induction on the structure of $\mathtt{e}'$.
Case $\mathtt{x} \Rightarrow^*_\Gamma \mathtt{x}$, immediate.
Case $\mathtt{A} \Rightarrow^*_\Gamma \mathtt{t}$ where $\mathtt{A} \in N$. By definition $\mathit{fv}(\mathtt{A}) = \{\mathtt{x} | \exists \mathtt{t}.\mathtt{A} \Rightarrow^*_\Gamma \mathtt{t} \text{ and } \mathtt{x} \in \mathit{fv}(\mathtt{t})\}$.
Case $\lambda\mathtt{x.e}' \Rightarrow^*_\Gamma \lambda\mathtt{x.e}$, iff $\mathtt{e}' \Rightarrow^*_\Gamma \mathtt{e}$. By induction the lemma holds for $\mathtt{e}'$ and $\mathtt{e}$. Therefore $\mathit{fv}(\lambda\mathtt{x.e}') = \mathit{fv}(\mathtt{e}') \setminus \{\mathtt{x}\} \supseteq \mathit{fv}(\mathtt{e}) \setminus \{\mathtt{x}\} = \mathit{fv}(\lambda\mathtt{x.e})$.
Case $\mathtt{e}'_1\mathtt{@e}'_2 \Rightarrow^*_\Gamma \mathtt{e}_1\mathtt{@e}_2$, iff $\mathtt{e}'_1 \Rightarrow^*_\Gamma \mathtt{e}_1$ and $\mathtt{e}'_2 \Rightarrow^*_\Gamma \mathtt{e}_2$. By induction the lemma holds for $\mathtt{e}'_1, \mathtt{e}_1$ and $\mathtt{e}'_2, \mathtt{e}_2$. Hence $\mathit{fv}(\mathtt{e}'_1\mathtt{@e}'_2) = \mathit{fv}(\mathtt{e}'_1) \cup \mathit{fv}(\mathtt{e}'_2) \supseteq \mathit{fv}(\mathtt{e}_1) \cup \mathit{fv}(\mathtt{e}_2) = \mathit{fv}(\mathtt{e}_1\mathtt{@e}_2)$. □

If $\mathtt{e} \in \mathit{Exp}$, i.e., no nonterminals occur in `e`, then $\mathtt{e} \Rightarrow^*_\Gamma \mathtt{e}$.
If $\mathtt{A} \Rightarrow^*_\Gamma \mathtt{e}$ then there exist $\mathtt{t} \notin N$ such that $\mathtt{A} ::= \mathtt{t}$ and $\mathtt{t} \Rightarrow^*_\Gamma \mathtt{e}$.

Definition 8.16. The relation S between states

Define relation S between states in the extended language and states in the pure λ-calculus as the smallest relation S such that:

$$S(\mathtt{e}' : \rho', \mathtt{e} : \rho) \text{ if } \mathtt{e}' \Rightarrow^*_\Gamma \mathtt{e} \text{ and for all } \mathtt{x} \in \mathit{fv}(\mathtt{e}) \text{ it holds that } S(\rho'(\mathtt{x}), \rho(\mathtt{x})).$$

If $\mathtt{e} : \rho$ is a state in the pure lambda calculus then it is also a state in the extended language and $S(\mathtt{e} : \rho, \mathtt{e} : \rho)$.

Lemma 8.17. *If $S(\mathtt{A}:\rho', \mathtt{e}:\rho)$ and $\mathtt{A} ::= \mathtt{t},\ \mathtt{t} \Rightarrow^*_\Gamma \mathtt{e}$ then also $S(\mathtt{t}:\rho', \mathtt{e}:\rho)$.*

We now define a relation T between size-change graphs. The intention is that $T(G', G)$ is to hold when the only difference in the generation of the graphs is due to nonterminals that take the place of pure lambda expressions.

Definition 8.18. The relation T between size-change graphs
Define $T(G', G)$ to hold iff

i) $\mathit{source}(G') \supseteq \mathit{source}(G)$ and $\mathit{target}(G') \supseteq \mathit{target}(G)$.
ii) The subgraph of G' restricted to $\mathit{source}(G)$ and $\mathit{target}(G)$ is a subset of G.
iii) Furthermore if $\mathtt{z} \in \mathit{source}(G') \setminus \mathit{source}(G)$ then either there is no edge from $\mathtt{z}$ in G' or the only edge from $\mathtt{z}$ in G' is $(\mathtt{z} \overset{=}{\rightarrow} \mathtt{z})$, and if $(\mathtt{z} \overset{=}{\rightarrow} \mathtt{z}) \in G'$ then $\mathtt{z} \notin \mathit{target}(G)$.

We have that $T(G'_0, G_0)$, $T(G'_1, G_1)$, $\text{target}(G'_0) = \text{source}(G'_1)$ and $\text{target}(G_0) = \text{source}(G_1)$ together imply that $T((G'_0; G'_1), (G_0; G_1))$ holds.

Lemma 8.19. Simulation Property

i) If $S(\mathtt{e}':\rho', \mathtt{e}:\rho)$ and $\mathtt{e}:\rho \Downarrow \mathtt{e}_0:\rho_0, G$ then there exist $\mathtt{e}'_0:\rho'_0, G'$ with $S(\mathtt{e}'_0:\rho'_0, \mathtt{e}_0:\rho_0)$ and $T(G', G)$ such that $\mathtt{e}':\rho' \Downarrow \mathtt{e}'_0:\rho'_0, G'$.
ii) If $S(\mathtt{e}':\rho', \mathtt{e}:\rho)$ and $\mathtt{e}:\rho \underset{x}{\rightarrow} \mathtt{e}_0:\rho_0, G$ with $x \in \{r, d, c\}$ then there exist $\mathtt{e}'_0:\rho'_0, G'$ and possibly s such that either $\mathtt{e}':\rho' \underset{x}{\rightarrow} \mathtt{e}'_0:\rho'_0, G'$ or $\mathtt{e}':\rho' \underset{n}{\rightarrow} s \underset{x}{\rightarrow} \mathtt{e}'_0:\rho'_0, G'$ with $S(\mathtt{e}'_0:\rho'_0, \mathtt{e}_0:\rho_0)$, $T(G', G)$, and in the last case $S(s, \mathtt{e}:\rho)$.
The composite size-change graph for the double-call $\mathtt{e}':\rho' \underset{n}{\rightarrow} s \underset{x}{\rightarrow} \mathtt{e}'_0:\rho'_0$ will have the same edges as G' because the $\underset{n}{\rightarrow}$ call generates an $\mathit{id}^=$ graph.

Corollary 8.20. *For programs $\mathtt{P} \in \mathit{Exp}_\Gamma$ and $\mathtt{Q} \in \mathit{Exp}$ with $\mathtt{P} \Rightarrow^*_\Gamma \mathtt{Q}$ it holds that:
If $\mathtt{Q}:[] \rightarrow^* \mathtt{e}:\rho$ then there exists $\mathtt{e}':\rho'$ such that $\mathtt{P}:[] \rightarrow^* \mathtt{e}':\rho'$ and $S(\mathtt{e}':\rho', \mathtt{e}:\rho)$.
If $\mathtt{Q}:[] \Downarrow \mathtt{e}:\rho$ then there exist $\mathtt{e}':\rho'$ such that $\mathtt{P}:[] \Downarrow \mathtt{e}':\rho'$ and $S(\mathtt{e}':\rho', \mathtt{e}:\rho)$.*

Also notice that if $\mathtt{e}_1:\rho_1 \underset{n}{\rightarrow} \mathtt{e}_2:\rho_2$ then $\mathit{fv}(\mathtt{e}_1) \supseteq \mathit{fv}(\mathtt{e}_2)$ by the definition of free variables for nonterminals. (By definition, $S(\mathtt{e}':\rho', \mathtt{e}:\rho)$ implies $\mathit{fv}(\mathtt{e}') \supseteq \mathit{fv}(\mathtt{e})$.)

Proof. Lemma 8.19 is shown by induction on the tree for the proof of evaluation or call in the pure λ-calculus and uses the observation about free variables. Proof is in the appendix. □

8.4. **The subexpression property.**

Definition 8.21. Given a state s in the extended language, we define its *expression support* $\mathit{exp\_sup}(s)$ by

$$\mathit{exp\_sup}(\mathtt{e}:\rho) = \mathit{subexps}(\mathtt{e}) \cup \bigcup_{\mathtt{x} \in fv(\mathtt{e})} \mathit{exp\_sup}(\rho(\mathtt{x}))$$

Lemma 8.22. (Subexpression property) *If $s \Downarrow s'$ or $s \rightarrow s'$ then $\mathit{exp\_sup}(s) \supseteq \mathit{exp\_sup}(s')$.*

Corollary 8.23. *If $\mathtt{P}:[] \Downarrow \lambda\mathtt{x}.\mathtt{e}:\rho$ then $\lambda\mathtt{x}.\mathtt{e} \in \mathit{subexp}(\mathtt{P})$. If $\mathtt{P}:[] \rightarrow^* \mathtt{e}:\rho$ then $\mathtt{e} \in \mathit{subexps}(\mathtt{P})$.*

The proof of Lemma 8.22 follows the same lines as the proof of Lemma 3.8. The proof for the rule (Gram) is immediate from the definition of subexpressions in the extended language. Proof omitted.

8.5. **Approximate extended semantics with size-change graphs.**

Definition 8.24. (Approximate evaluation and call rules for extended semantics with size-change graphs). The judgement forms are now $\mathtt{e} \to \mathtt{e}', G$ and $\mathtt{e} \Downarrow \mathtt{e}', G$, where $\mathtt{e}, \mathtt{e}' \in Exp_\Gamma$, and $source(G) = fv(e) \cup \{\epsilon\}$ and $target(G) = fv(e') \cup \{\epsilon\}$.

$$\text{(GramAG)} \quad \frac{}{A \underset{n}{\to} \mathtt{e}, id_e^{=}} \; (A ::= \mathtt{e} \in \Gamma)$$

$$\text{(ResultAG)} \quad \frac{\mathtt{e} \underset{x}{\to} \mathtt{e}', G' \qquad \mathtt{e}' \Downarrow v, G}{\mathtt{e} \Downarrow v, G'; G} \; (x \in \{c, n\})$$

$$\text{(ValueAG)} \quad \frac{}{\lambda\mathtt{x.e} \Downarrow \lambda\mathtt{x.e}, id_{\lambda\mathtt{x.e}}^{=}}$$

$$\text{(VarAG)} \quad \frac{\mathtt{e}_1\mathtt{@e}_2 \in subexps(\mathtt{P}) \quad \mathtt{e}_1 \Downarrow \lambda\mathtt{x.e}_0, G_1 \qquad \mathtt{e}_2 \Downarrow v_2, G_2}{\mathtt{x} \Downarrow v_2, \{\mathtt{x} \overset{=}{\to} \epsilon\} \cup \{\mathtt{x} \overset{\downarrow}{\to} \mathtt{y} \mid \mathtt{y} \in fv(v_2)\}}$$

$$\text{(OperatorAG)} \quad \frac{}{\mathtt{e}_1\mathtt{@e}_2 \underset{r}{\to} \mathtt{e}_1, id_{\mathtt{e}_1}^{\downarrow}} \qquad \text{(OperandAG)} \quad \frac{}{\mathtt{e}_1\mathtt{@e}_2 \underset{d}{\to} \mathtt{e}_2, id_{\mathtt{e}_2}^{\downarrow}}$$

$$\text{(CallAG)} \quad \frac{\mathtt{e}_1 \Downarrow \lambda\mathtt{x.e}_0, G_1 \qquad \mathtt{e}_2 \Downarrow v_2, G_2}{\mathtt{e}_1\mathtt{@e}_2 \underset{c}{\to} \mathtt{e}_0, G_1^{-\epsilon/\lambda x.e_0} \cup_{e_0} G_2^{\epsilon \mapsto \mathtt{x}}}$$

Putting the pieces together, we now show how to analyse any program in the regular grammar-extended λ-calculus . Let P be a program in the extended language.

Definition 8.25.

$absintExt(\mathtt{P}) = \{\, G_j \mid j > 0 \wedge \exists \mathtt{e}_i, G_i, (0 \le i \le j) : \mathtt{P} = \mathtt{e}_0 \wedge (\mathtt{e}_0 \to \mathtt{e}_1, G_1) \wedge \ldots \wedge (\mathtt{e}_{j-1} \to \mathtt{e}_j, G_j) \,\}$

Theorem 8.26.

The set absintExt(P) *can be effectively computed from* P.

Proof. In the extended λ-calculus there is only a fixed number of subexpressions of P, and a fixed number of of possible size-change graphs with

$$\text{source, target} \subseteq \{\epsilon\} \cup \{\mathtt{x} \mid \mathtt{x} \text{ is a variable that occurs in a subexpression of } \mathtt{P}\}$$

Thus *absintExt*(P) can be computed in finite time by applying Definition 8.24 exhaustively, starting with P, until no new graphs or subexpressions are obtained. □

8.6. **Simulation properties of approximate extended semantics.**
We will show the following properties of approximate extended semantics:

(1) Calls and evaluations for a program in *extended semantics with environments* can be stepwise simulated by *approximate extended semantics* with identical size-change graphs associated with corresponding calls and evaluations. To a call or evaluation in the extended λ-calculus with environments corresponds the same call or evaluation with environments removed.
(2) Suppose $\mathtt{P} \Rightarrow^*_\Gamma \mathtt{Q}$ for programs $\mathtt{P}$,$\mathtt{Q}$. Then calls and evaluations for $\mathtt{Q}$ in the pure lambda calculus with environments can be simulated by calls and valuations in the approximate extended semantics for $\mathtt{P}$ using the relations $\Rightarrow^*_\Gamma$ and T.
(3) The extra edges in the size-change graphs in extended semantics can never give rise to incorrect termination analysis.

Lemma 8.27. *Let* $\mathtt{P}$ *be a program in the extended language and* $\mathtt{P} : [] \to^* \mathtt{e} : \rho$.
If $\mathtt{e} : \rho \to \mathtt{e}_0 : \rho_0, G$ *then* $\mathtt{e} \to \mathtt{e}_0, G$ *in approximate semantics.*
If $\mathtt{e} : \rho \Downarrow \mathtt{e}_0 : \rho_0, G$ *then* $\mathtt{e} \Downarrow \mathtt{e}_0, G$ *in approximate semantics.*

Proof. The proof is similar to the proof for approximation of the pure lambda-calculus 3.11 and 6.10. For rules (Value), (Operator), (Operand) it is immediate. The (Gram)-rule do not refer to the environment, hence the lemma holds if the (Gram)-rule has been applied. For rules (Call) and (Result) it holds by induction. For the (Var)-rule we need induction on the total size of the derivation, and we can argue as in the case of the pure lambda calculus. □

Lemma 8.28. *Let* $\mathtt{P}$ *be a program in the extended language and* $\mathtt{Q}$ *a program in the pure* λ*-calculus with* $\mathtt{P} \Rightarrow^*_\Gamma \mathtt{Q}$.

If $\mathtt{Q} : [] \to^* \mathtt{e} : \rho$ *and* $\mathtt{e} : \rho \Downarrow \mathtt{e}_0 : \rho_0, G$ *then there exist* $\mathtt{e}', \mathtt{e}'_0, G'$ *with* $\mathtt{e}' \Rightarrow^*_\Gamma \mathtt{e}$, $\mathtt{e}'_0 \Rightarrow^*_\Gamma \mathtt{e}_0$, $T(G', G)$ *such that* $\mathtt{P} \to^* \mathtt{e}'$ *and* $\mathtt{e}' \Downarrow \mathtt{e}'_0, G'$.

If $\mathtt{Q} : [] \to^* \mathtt{e} : \rho$ *and* $\mathtt{e} : \rho \underset{x}{\to} \mathtt{e}_0 : \rho_0, G,\ x \in \{r, d, c\}$ *then there exist* $\mathtt{e}', \mathtt{e}'_0, G'$ *with* $\mathtt{e}' \Rightarrow^*_\Gamma \mathtt{e}$,$\mathtt{e}'_0 \Rightarrow^*_\Gamma \mathtt{e}_0$, $T(G', G)$ *such that* $\mathtt{P} \to^* \mathtt{e}'$ *and either* $\mathtt{e}' \underset{x}{\to} \mathtt{e}'_0, G'$ *or* $\mathtt{e}' \underset{n}{\to} \mathtt{e}'' \underset{x}{\to} \mathtt{e}'_0, G'$ *where in the last case* G' *is the composite size-change graph for the double call.*

Proof. The lemma follows from the simulation property lemma 8.19 together with lemma 8.27. □

Theorem 8.29.

(1) *Let* $\mathtt{P}$ *be a program in the extended language. If there is a program* $\mathtt{Q}$ *in the pure lambda-calculus such that* $\mathtt{P} \Rightarrow^*_\Gamma \mathtt{Q}$ *and there exists an infinite call-sequence in the call-graph for* $\mathtt{Q}$ *in the exact semantics, then there exists an infinite call-sequence with no infinitely descending thread in the call-graph for* $\mathtt{P}$ *in the approximate extended semantics.*
(2) *It follows that if each infinite call-sequence in the call-graphs for* $\mathtt{P}$ *in the approximate extended semantics has an infinitely descending thread, then* $\mathtt{P}$ *is* Γ*-terminating.*

Proof. (1): Assume an infinite call-sequence exists in the call-graph for $\mathtt{Q}$. By the safety of the size-change graphs in the pure λ-calculus, the size-change graphs associated with this call sequence cannot have an infinitely descending thread. By lemma 8.28 there exists a simulating call-sequence in the call-graph for $\mathtt{P}$ such that the corresponding size-change graphs are in the T-relation. Let G_P, G_Q be any such two corresponding T-related size-change graphs from these call-sequences, $T(G_P, G_Q)$. By the definition of the T-relation

it holds that the largest subgraph of G_P, with *source* and *target* the same as $source(G_Q)$ and $target(G_Q)$, is equal to or a subset of G_Q. We need to show that the possible extra variables in the size-change graphs for the simulating sequence in the call-graph for `P` can never take part in an infinitely descending thread. By the definition of the T-relation it holds that an edge leaving from such a variable `x` must have have the form $(\mathtt{x} \overset{=}{\to} \mathtt{x})$ if any exists in the simulating sequence. Also by the definition of the T-relation, if $T(G_P, G_Q)$ and $(\mathtt{x} \overset{=}{\to} \mathtt{x}) \in G_P$ then $\mathtt{x} \notin codomain(G_Q)$. Hence either an extra thread in the size-change graphs going out from `x` will be finite or it will be infinitely equal $\mathtt{x} \overset{=}{\to} \mathtt{x} \overset{=}{\to} \mathtt{x} \overset{=}{\to} \ldots$, i.e. an extra variable can never take part in an infinitely descending thread in the simulating sequence.
(2) is a corollary to (1). □

Example 8.30.
The following is an example of a program certified to terminate by our proof method. The program computes $x + 2^n$ when applied to two arbitrary Church numerals for x and n. In Section 7 we analysed the program applied to Church numerals 3 and 4 (Example 7.2).

Grammar for Church numerals: C ::= λ`s`.λ`z`.A A ::= `z` | `s@`A

The program applied to two Church numerals:

```
[λn1.λn2. n1                                   -- n --
          @  [λr.λa.|11:|(r@|13:|(r@a))]        -- g --
          @  [λ k.λ p.λ q.(p@((k@p)@q))]       - succ-
          @  n2 ]                              -- x --

@            C                                 -- Church numeral --
@            C                                 -- Church numeral --
```

Following is the output from program analysis. The analysis found the following loops from a program point to itself with the associated size-change graph and path. The first number refers to the program point, then comes a list of edges and last a list of numbers, the other program points that the loop passes through. The program points are found automatically by the analysis. The program points 30 and 32 are not written into the presentation of the program because they involve the subexpression A of a Church numeral. The subexpression associated with 30 is A and the subexpression associated with 32 is `s@`A. The loops from 30 to itself and from 32 to itself in the output correspond to the call sequence A→`s@`A→A→`s@`A. . . .

```
SELF SCGS no repetition of graphs:

11 →* 11: [(r,>,r)]                                  []
11 →* 11: [(a,=,a),(r,>,r)]                          [13]
13 →* 13: [(a,=,a),(r,>,r)]                          [11]
13 →* 13: [(r,>,r)]                                  [11,11]
30 →* 30: [(ϵ,>,ϵ),(s,=,s),(z,=,z)]    [32]
32 →* 32: [(ϵ,>,ϵ),(s,=,s),(z,=,z)]    [30]

Size Change Termination: Yes
```

9. Concluding matters

We have developed a method based on The Size Change Principle to show termination of a closed expression in the untyped λ-calculus. This is further developed to analyse if a program in the λ-calculus will terminate when applied to any input from a given input set defined by a tree grammar. The analysis is safe and the method can be completely automated. We have a simple first implementation. The method certifies termination of many interesting recursive programs, including programs with mutual recursion and parameter exchange.

Acknowledgements. The authors gratefully acknowledge detailed and constructive comments by Arne Glenstrup, Chin Soon Lee and Damien Sereni, and insightful comments by Luke Ong, David Wahlstedt and Andreas Abel.

Appendix A. Proof of Lemma 2.6

Proof. $\Rightarrow$: Assume $\mathtt{P}\Downarrow$. To show: CT has no infinite call chain starting with $\mathtt{P}$. The proof is by induction on the height of the proof tree. Each call rule of Definition 2.6 is associated with a use of rule (ApplyS) from Definition 2.2. So if $\mathtt{P}$ is a value, there is no call from $\mathtt{P}$. If $\mathtt{P}\Downarrow$ is concluded by rule (ApplyS), then $\mathtt{P} = \mathtt{e}_1\mathtt{@}\mathtt{e}_2$ and by induction there is no infinite call chain starting with $\mathtt{e}_1$, $\mathtt{e}_2$ and $\mathtt{e}_0[v_2/\mathtt{x}]$. All call chains starting with $\mathtt{P}$ go directly to one of these. So, there are no infinite call chains starting with $\mathtt{P}$.

$\Leftarrow$: Assume CT has no infinite call chain starting with $\mathtt{P}$. To show: $\mathtt{P}\Downarrow$. Since the call tree is finitely branching, by König's lemma the whole call tree is finite, and hence there exists a finite number m bounding the length of all branches.

We prove that $\mathtt{e}\Downarrow$ for any expression in the call tree, by induction on the maximal length n of a call chain from $\mathtt{e}$.

$n = 0$: $\mathtt{e}$ is an abstraction that evaluates to itself.

$n > 0$: $\mathtt{e}$ must be an application $\mathtt{e} = \mathtt{e}_1\mathtt{@}\mathtt{e}_2$. By rule (Operator) there is a call $\mathtt{e}_1\mathtt{@}\mathtt{e}_2 \underset{d}{\rightarrow} \mathtt{e}_1$, and the maximal length of a call chain from $\mathtt{e}_1$ is less than n. By induction there exists v_1 such that $\mathtt{e}_1 \Downarrow v_1$. We now conclude by rule (Operand) that $\mathtt{e}_1\mathtt{@}\mathtt{e}_2 \underset{r}{\rightarrow} \mathtt{e}_2$. By induction there exists v_2 such that $\mathtt{e}_2 \Downarrow v_2$.

All values are abstractions, so we can write $v_1 = \lambda\mathtt{x}.\mathtt{e}_0$. We now conclude by rule (Call) that $\mathtt{e}_1\mathtt{@}\mathtt{e}_2 \underset{c}{\rightarrow} \mathtt{e}_0[v_2/\mathtt{x}]$. By induction again, $\mathtt{e}_0[v_2/\mathtt{x}] \Downarrow v$ for some v. This gives us all premises for the (ApplyS) rule of Definition 2.2, so $\mathtt{e} = \mathtt{e}_1\mathtt{@}\mathtt{e}_2 \Downarrow v$. □

Appendix B. Proof of Lemma 3.11

Proof. To be shown:

$$\begin{array}{l} \text{If } \mathtt{P} : [] \rightarrow^* \mathtt{e} : \rho \quad \text{and} \quad \mathtt{e} : \rho \Downarrow \mathtt{e}' : \rho', \quad \text{then} \quad \mathtt{e} \Downarrow \mathtt{e}'. \\ \text{If } \mathtt{P} : [] \rightarrow^* \mathtt{e} : \rho \quad \text{and} \quad \mathtt{e} : \rho \rightarrow \mathtt{e}' : \rho', \quad \text{then} \quad \mathtt{e} \rightarrow \mathtt{e}'. \end{array}$$

We prove both parts of Lemma 3.11 by course-of-value induction over the size $n = |\mathcal{D}|$ of a deduction $\mathcal{D}$ by Definition 3.3 of the assumption

$$\mathtt{P} : [] \rightarrow^* \mathtt{e} : \rho \wedge \mathtt{e} : \rho \Downarrow \mathtt{e}' : \rho' \text{ or } \mathtt{P} : [] \rightarrow^* \mathtt{e} : \rho \wedge \mathtt{e} : \rho \rightarrow \mathtt{e}' : \rho'$$

The deduction size may be thought of as the number of steps in the computation of $\mathtt{e} : \rho \Downarrow \mathtt{e}' : \rho'$ or $\mathtt{e} : \rho \rightarrow \mathtt{e}' : \rho'$ starting from $\mathtt{P} : []$.

The induction hypothesis $IH(n)$ is that the Lemma holds for all deductions of size not exceeding n. This implies that the Lemma holds for all calls and evaluations performed in the computation before the last conclusion giving ($\mathtt{P} : [] \rightarrow^* \mathtt{e} : \rho$ and $\mathtt{e} : \rho \Downarrow \mathtt{e}' : \rho'$)

or ($\mathtt{P} : [] \rightarrow^* \mathtt{e} : \rho$ and $\mathtt{e} : \rho \rightarrow \mathtt{e}' : \rho'$), i.e., the Lemma holds for premises of the rule last applied, and *for any call and evaluation in the computation until then.*

Proof is by cases on which rule is applied to conclude $\mathtt{e} : \rho \Downarrow \mathtt{e}' : \rho'$ or $\mathtt{e} : \rho \rightarrow \mathtt{e}' : \rho'$. In all cases we show that some corresponding abstract interpretation rules can be applied to give the desired conclusion.

Base cases: Rule (Value), (Operator) and (Operand) in the exact semantics (def. 3.2) are modeled by axioms (ValueA), (OperatorA) and (OperandA) in the abstract semantics (def. 3.10). These are the same as their exact-evaluation counterparts, after removal of environments for (ValueA) and (OperatorA), and a premise as well for (OperandA). Hence the Lemma holds if one of these rules was the last one applied.

The (Var) rule is, however, rather different from the (VarA) rule. If (Var) was applied to a variable $\mathtt{x}$ then the assumption is ($\mathtt{P} : [] \rightarrow^* \mathtt{x} : \rho$ and $\mathtt{x} : \rho \Downarrow \mathtt{e}' : \rho'$). In this case $x \in dom(\rho)$ and $\mathtt{e}' : \rho' = \rho(x)$. The total size of the deduction (of both parts together) is n.

Now $\mathtt{P} : [] \rightarrow^* \mathtt{x} : \rho$ begins from the empty environment, and we know all calls are from state to state. The only possible way $\mathtt{x}$ can have been bound is by a previous use of the (Call) rule, the only rule that extends an environment.[5]

The premises of the (Call) rule require that operator and operand in an application have previously been evaluated. So it must be the case that there exist $\mathtt{e}_1\mathtt{@}\mathtt{e}_2 : \rho''$ and $\lambda\mathtt{x}.\mathtt{e}_0 : \rho_0$ such that ($P : [] \rightarrow^* \mathtt{e}_1\mathtt{@}\mathtt{e}_2 : \rho''$ and $\mathtt{e}_1 : \rho'' \Downarrow \lambda\mathtt{x}.\mathtt{e}_0 : \rho_0$ and $\mathtt{e}_2 : \rho'' \Downarrow \mathtt{e}' : \rho'$) and the size of both deductions are strictly smaller than n. By the Subexpression Lemma, $\mathtt{e}_1\mathtt{@}\mathtt{e}_2 \in \mathit{subexp}(\mathtt{P})$. By induction, Lemma 3.11 holds for both $\mathtt{e}_1 : \rho'' \Downarrow \lambda\mathtt{x}.\mathtt{e}_0 : \rho_0$ and $\mathtt{e}_2 : \rho'' \Downarrow \mathtt{e}' : \rho'$, so $\mathtt{e}_1 \Downarrow \lambda\mathtt{x}.\mathtt{e}_0$ and $\mathtt{e}_2 \Downarrow \mathtt{e}'$ in the abstract semantics. Now we have all premises of rule (VarA), so we can conclude that $\mathtt{x} \Downarrow \mathtt{e}'$ as required.

For remaining rules (Apply) and (Call), when we assume that the Lemma holds for the premises in the rule applied to conclude $\mathtt{e} \Downarrow \mathtt{e}'$ or $\mathtt{e} \rightarrow \mathtt{e}'$, then this gives us the premises for the corresponding rule for abstract interpretation. From this we can conclude the desired result. □

Appendix C. Proof of Lemma 5.4

Proof. Define the length $L(\mathtt{e})$ of an expression e by:

$$L(\mathtt{x}) = 1 \qquad L(\lambda\mathtt{x}.\mathtt{e}) = 1 + L(\mathtt{e}) \qquad L(\mathtt{e}_1@\mathtt{e}_2) = 1 + L(\mathtt{e}_1) + L(\mathtt{e}_2)$$

For any expression $\mathtt{e}$, $L(\mathtt{e})$ is a natural number > 0. For a program, the length of the initial expression bounds all lengths of occurring expressions.

Define for a state s the height $H(s)$ of the state to be the height of the environment:

$$H(\mathtt{e} : \rho) = max\{(1 + H(\rho(\mathtt{x}))) \mid \mathtt{x} \in \mathit{fv}(\mathtt{e})\}$$

So, $H(\mathtt{e} : []) = 0$ the maximum of the empty set, and for any state $\mathtt{e} : \rho, H(\mathtt{e} : \rho)$ is a natural number ≥ 0. Let $>_{lex}$ stand for lexicographic order relation on pairs of natural numbers, hence $>_{lex}$ is well-founded. We prove that the relation $\succ$ on states is well-founded by proving that $\mathtt{e}_1 : \rho_1 \succ \mathtt{e}_2 : \rho_2$ implies that

$$(H(\mathtt{e}_1 : \rho_1), L(\mathtt{e}_1)) >_{lex} (H(\mathtt{e}_2 : \rho_2), L(\mathtt{e}_2))$$

[5] This must have occurred in the part $\mathtt{P} : [] \rightarrow^* \mathtt{x} : \rho$.

First, consider $\succ_1$. Clearly, if $\mathtt{e}_1 : \rho_1 \succ_1 \mathtt{e}_2 : \rho_2$ then $H(\mathtt{e}_1 : \rho_1) > H(\mathtt{e}_2 : \rho_2)$. Hence even though $L(\mathtt{e}_2)$ might be larger than $L(\mathtt{e}_1)$, it holds that in the lexicographic order $(H(\mathtt{e}_1 : \rho_1), L(\mathtt{e}_1)) >_{lex} (H(\mathtt{e}_2 : \rho_2), L(\mathtt{e}_2))$.

Now, consider $\succ_2$. If $\mathtt{e}_1 : \rho_1 \succ_2 \mathtt{e}_2 : \rho_2$ then $H(\mathtt{e}_1 : \rho_1) \geq H(\mathtt{e}_2 : \rho_2)$ and $L(\mathtt{e}_1) > L(\mathtt{e}_2)$, hence in the lexicographic order $(H(\mathtt{e}_1 : \rho_1), L(\mathtt{e}_1)) >_{lex} (H(\mathtt{e}_2 : \rho_2), L(\mathtt{e}_2))$. Trivially, $\mathtt{e}_1 : \rho_1 = \mathtt{e}_2 : \rho_2$ implies $(H(\mathtt{e}_1 : \rho_1), L(\mathtt{e}_1)) =_{lex} (H(\mathtt{e}_2 : \rho_2), L(\mathtt{e}_2))$.

Recall, by definition $\succeq$ is the transitive closure of $\succ_1 \cup \succ_2 \cup =$, and $s_1 \succ s_2$ holds when $s_1 \succeq s_2$ and $s_1 \neq s_2$. So, from the derivations above we can conclude that $\mathtt{e}_1 : \rho_1 \succ \mathtt{e}_2 : \rho_2$ implies $(H(\mathtt{e}_1 : \rho_1), L(\mathtt{e}_1)) >_{lex} (H(\mathtt{e}_2 : \rho_2), L(\mathtt{e}_2))$, hence the relation $\succ$ on states is well-founded.

□

Appendix D. Proof of Theorem 6.8

Proof. For the "safety" theorem we use induction on proofs of $s \Downarrow s', G$ or $s \rightarrow s', G$. Safety of the constructed graphs for rules (ValueG), (OperatorG) and (OperandG) is immediate by Definitions 6.2 and 5.3.

In the following $\mathtt{x}, \mathtt{y}, \mathtt{z}$ are variables and p, q can be variables or ϵ.

The variable lookup rule **(VarG)** yields $\mathtt{x} : \rho \Downarrow \rho(\mathtt{x}), G$ with $G = \{\mathtt{x} \overset{\downarrow}{\rightarrow} \mathtt{y} \mid \mathtt{y} \in \mathit{fv}(\mathtt{e}')\} \cup \{\mathtt{x} \overset{=}{\rightarrow} \epsilon\}$ and $\rho(\mathtt{x}) = \mathtt{e}' : \rho'$. By Definition 5.2, $\overline{\mathtt{x} : \rho}(\mathtt{x}) = \overline{\rho(\mathtt{x})}(\epsilon)$, so arc $\mathtt{x} \overset{=}{\rightarrow} \epsilon$ satisfies Definition 6.2. Further, if $\mathtt{x} \overset{\downarrow}{\rightarrow} \mathtt{y} \in G$ then $\mathtt{y} \in \mathit{fv}(\mathtt{e}')$. Thus $\overline{x : \rho}(x) = \rho(\mathtt{x}) = e' : \rho' \succ \rho'(\mathtt{y}) = \overline{\rho(x)}(y)$ as required.

The rule **(CallG)** concludes $s \underset{c}{\rightarrow} s', G$, where $s = \mathtt{e}_1 @ \mathtt{e}_2 : \rho$ and $s' = \mathtt{e}_0 : \rho_0[\mathtt{x} \mapsto v_2]$ and $G = G_1^{-\epsilon/\lambda x.e_0} \cup_{e_0} G_2^{\epsilon \mapsto \mathtt{x}}$. Its premises are $\mathtt{e}_1 : \rho \Downarrow \lambda\mathtt{x}.\mathtt{e}_0 : \rho_0, G_1$ and $\mathtt{e}_2 : \rho \Downarrow v_2, G_2$. We assume inductively that G_1 is safe for $(\mathtt{e}_1 : \rho, \lambda\mathtt{x}.\mathtt{e}_0 : \rho_0)$ and that G_2 is safe for $(\mathtt{e}_2 : \rho, v_2)$. Let $v_2 = \mathtt{e}' : \rho'$.

We wish to show safety: that $p \overset{=}{\rightarrow} p' \in G$ implies $\overline{s}(p) = \overline{s'}(p')$, and $p \overset{\downarrow}{\rightarrow} p' \in G$ implies $\overline{s}(p) \succ \overline{s'}(p')$. By definition of $G_1^{-\epsilon/\lambda x.e_0}$ and $G_2^{\epsilon \mapsto \mathtt{x}}$, $p \overset{r}{\rightarrow} p' \in G = G_1^{-\epsilon/\lambda x.e_0} \cup_{e_0} G_2^{\epsilon \mapsto \mathtt{x}}$ breaks into 7 cases:

Case 1: $\mathtt{y} \overset{\downarrow}{\rightarrow} \mathtt{z} \in G_1^{-\epsilon/\lambda x.e_0}$ because $\mathtt{y} \overset{\downarrow}{\rightarrow} \mathtt{z} \in G_1$. By safety of G_1, $\overline{\mathtt{e}_1 : \rho}(\mathtt{y}) \succ \overline{\lambda\mathtt{x}.\mathtt{e}_0 : \rho_0}(\mathtt{z})$. Thus, as required,

$$\overline{s}(\mathtt{y}) = \overline{\mathtt{e}_1 @ \mathtt{e}_2 : \rho}(\mathtt{y}) = \overline{\mathtt{e}_1 : \rho}(\mathtt{y}) \succ \overline{\lambda\mathtt{x}.\mathtt{e}_0 : \rho_0}(\mathtt{z}) = \overline{\mathtt{e}_0 : \rho_0[\mathtt{x} \mapsto v_2]}(\mathtt{z}) = \overline{s'}(\mathtt{z})$$

Case 2: $\mathtt{y} \overset{=}{\rightarrow} \mathtt{z} \in G_1^{-\epsilon/\lambda x.e_0}$ because $\mathtt{y} \overset{=}{\rightarrow} \mathtt{z} \in G_1$. Like Case 1.

Case 3: $\mathtt{y} \overset{\downarrow}{\rightarrow} \epsilon \in G_1^{-\epsilon/\lambda x.e_0}$ because $\mathtt{y} \overset{r}{\rightarrow} \epsilon \in G_1$, then $x \notin \mathit{fv}(\mathtt{e}_0)$ by the definition of $G_1^{-\epsilon/\lambda x.e_0}$ and then $\mathtt{e}_0 : \rho_0[\mathtt{x} \mapsto v_2] = \mathtt{e}_0 : \rho_0$. By safety of G_1, $\overline{\mathtt{e}_1 : \rho}(\mathtt{y}) \succeq \overline{\lambda\mathtt{x}.\mathtt{e}_0 : \rho_0}(\epsilon) = \lambda\mathtt{x}.\mathtt{e}_0 : \rho_0$. Thus, as required,

$$\overline{s}(\mathtt{y}) = \overline{\mathtt{e}_1 @ \mathtt{e}_2 : \rho}(\mathtt{y}) = \overline{\mathtt{e}_1 : \rho}(\mathtt{y}) \succeq \lambda\mathtt{x}.\mathtt{e}_0 : \rho_0 \succ \mathtt{e}_0 : \rho_0 = \overline{s'}(\epsilon)$$

Case 4: $\epsilon \overset{\downarrow}{\rightarrow} p \in G_1^{-\epsilon/\lambda x.e_0}$ because $\epsilon \overset{r}{\rightarrow} p \in G_1$. Then it holds that either p is a variable-name or $\mathtt{x} \notin \mathit{fv}(\mathtt{e}_0)$. Now ϵ in G_1 refers to $\mathtt{e}_1 : \rho$, so $\mathtt{e}_1 : \rho \succeq \overline{\lambda\mathtt{x}.\mathtt{e}_0 : \rho_0}(p)$ by safety of G_1.

Thus, as required,

$$\overline{s}(\epsilon) = \mathtt{e}_1\mathtt{@e}_2 : \rho \succ \mathtt{e}_1 : \rho \succeq \overline{\lambda\mathtt{x.e}_0 : \rho_0}(p) \succeq \overline{\mathtt{e}_0 : \rho_0[\mathtt{x} \mapsto v_2]}(p) = \overline{s'}(p)$$

Case 5: $\mathtt{y} \xrightarrow{\downarrow} \mathtt{x} \in G$ because $\mathtt{x} \in fv(\mathtt{e}_0)$ and $\mathtt{y} \xrightarrow{\downarrow} \mathtt{x} \in G_2^{\epsilon \mapsto \mathtt{x}}$ because $\mathtt{y} \xrightarrow{\downarrow} \epsilon \in G_2$. By safety of G_2, $\overline{\mathtt{e}_2 : \rho}(\mathtt{y}) \succ \overline{v_2}(\epsilon)$. Thus, as required,

$$\overline{s}(\mathtt{y}) = \overline{\mathtt{e}_1\mathtt{@e}_2 : \rho}(\mathtt{y}) = \overline{e_2 : \rho}(\mathtt{y}) \succ \overline{v_2}(\epsilon) = \overline{\mathtt{e}_0 : \rho_0[\mathtt{x} \mapsto v_2]}(\mathtt{x}) = \overline{s'}(\mathtt{x})$$

Case 6: $\mathtt{y} \xrightarrow{=} \mathtt{x} \in G$ because $x \in fv(e_0)$ and $\mathtt{y} \xrightarrow{=} \mathtt{x} \in G_2^{\epsilon \mapsto \mathtt{x}}$ because $\mathtt{y} \xrightarrow{=} \epsilon \in G_2$. Like Case 5.

Case 7: $\epsilon \xrightarrow{\downarrow} \mathtt{x} \in G$ because $\mathtt{x} \in fv(\mathtt{e}_0)$ and $\epsilon \xrightarrow{\downarrow} \mathtt{x} \in G_2^{\epsilon \mapsto \mathtt{x}}$ because $\epsilon \xrightarrow{r} \epsilon \in G_2$. By safety of G_2, $\overline{\mathtt{e}_2 : \rho}(\epsilon) = \mathtt{e}_2 : \rho$. Thus, as required,

$$\overline{s}(\epsilon) = \mathtt{e}_1\mathtt{@e}_2 : \rho \succ \mathtt{e}_2 : \rho \succeq \overline{v_2}(\epsilon) = \overline{\rho_0[\mathtt{x} \mapsto v_2]}(\mathtt{x}) = \overline{s'}(\mathtt{x})$$

The rule **(ApplyG)** concludes $s \Downarrow v, G'; G$ from premises $s \underset{c}{\to} s', G'$ and $s' \Downarrow v, G$, where $s = \mathtt{e}_1\mathtt{@e}_2 : \rho$ and $s' = \mathtt{e}' : \rho'$. We assume inductively that G' is safe for (s, s') and G is safe for (s', v). Let $G_0 = G'; G$.

We wish to show that G_0 is safe: that $p \xrightarrow{=} q \in G_0$ implies $\overline{s}(p) = \overline{v}(q)$, and $p \xrightarrow{\downarrow} q \in G_0$ implies $\overline{s}(p) \succ \overline{v}(q)$ (p, q can be variables or ϵ). First, consider the case $p \xrightarrow{=} q \in G_0$. Definition 4.2 implies $p \xrightarrow{=} p' \in G'$ and $p' \xrightarrow{=} q \in G$ for some p'. Thus by the inductive assumptions we have $\overline{s}(p) = \overline{s'}(p') = \overline{v}(q)$, as required.

Second, consider the case $p \xrightarrow{\downarrow} q \in G_0$. Definition 4.2 implies $p \xrightarrow{r_1} p' \in G'$ and $p' \xrightarrow{r_2} q \in G$ for some p', where either one or both of r_1, r_2 are $\downarrow$. By the inductive assumptions we have $\overline{s}(p) \succeq \overline{s'}(p')$ and $\overline{s'}(p') \succeq \overline{v}(q)$, and one or both of $\overline{s}(p) \succ \overline{s'}(p')$ and $\overline{s'}(p') \succ \overline{v}(q)$ hold. By Definition of $\succ$ and $\succeq$ this implies that $\overline{s}(p) \succ \overline{v}(q)$, as required. □

Appendix E. Proof of Lemma 6.10

Proof. The rules are the same as in Section 3.10, only extended with size-change graphs. We need to add to Lemma 3.11 that the size-change graphs generated for calls and evaluations can also be generated by the abstract interpretation. The proof is by cases on which rule is applied to conclude $\mathtt{e} \Downarrow \mathtt{e}', G$ or $\mathtt{e} : \rho \to \mathtt{e}' : \rho', G$.

We build on Lemma 3.11, and we saw in the proof of this that in abstract interpretation we can always use a rule corresponding to the one used in exact computation to prove corresponding steps. The induction hypothesis is that the Lemma holds for the premises of the rule in exact semantics.

Base case (VarAG): By Lemma 3.11 we have $\mathtt{x} : \rho \Downarrow \mathtt{e}' : \rho'$ implies $\mathtt{x} \Downarrow \mathtt{e}'$. The size-change graph built in (VarAG) is derived in the same way from $\mathtt{x}$ and $\mathtt{e}'$ as in rule (VarG), and they will therefore be identical.

For other call- and evaluation rules without premises, the abstract evaluation rule is as the exact-evaluation rule, only with environments removed, and the generated size-change graphs are not influenced by environments. Hence the Lemma will hold if these rules are applied.

For all other rules in a computation: When we know that Lemma 3.11 holds and assume that Lemma 6.10 hold for the premises, then we can conclude that if this rule is applied, then Lemma 6.10 holds by the corresponding rule from abstract interpretation. □

Appendix F. Proof of Lemma 8.19

Proof. By induction on the tree for the proof of evaluation or call in the pure λ-calculus. Possible cases of the structure of $\mathtt{e}' : \rho'$ and $\mathtt{e} : \rho$ in S-related states:

$(\mathtt{x} : \rho', \mathtt{x} : \rho)$	$(\lambda\mathtt{x}.\mathtt{e}' : \rho', \lambda\mathtt{x}.\mathtt{e} : \rho)$	$(\mathtt{e}_1'\mathtt{@}\mathtt{e}_2' : \rho', \mathtt{e}_1\mathtt{@}\mathtt{e}_2 : \rho)$
$(\mathtt{A} : \rho', \mathtt{x} : \rho)$	$(\mathtt{A} : \rho', \lambda\mathtt{x}.\mathtt{e} : \rho)$	$(\mathtt{A} : \rho', \mathtt{e}_1\mathtt{@}\mathtt{e}_2 : \rho)$

Base cases, evaluations and calls in pure λ-calculus by rules without premisses.

Case $S(\mathtt{x} : \rho', \mathtt{x} : \rho)$: No calls from $\mathtt{x} : \rho$.
(Var)-rule, $\mathtt{x} : \rho \Downarrow \rho(\mathtt{x}) = \mathtt{e}_0 : \rho_0$, $\{\mathtt{x} \overset{=}{\to} \epsilon\} \cup \{\mathtt{x} \overset{\downarrow}{\to} \mathtt{y} \mid \mathtt{y} \in fv(\mathtt{e}_0)\}$ and $\mathtt{x} : \rho' \Downarrow \rho'(x) = \mathtt{e}_0' : \rho_0'$, $\{\mathtt{x} \overset{=}{\to} \epsilon\} \cup \{\mathtt{x} \overset{\downarrow}{\to} \mathtt{y} \mid \mathtt{y} \in fv(\mathtt{e}_0')\}$. Beginning from S-related states, by defintion of the relation S we have $S(\rho'(\mathtt{x}), \rho(\mathtt{x}))$ and $fv(\mathtt{e}_0') \supseteq fv(\mathtt{e}_0)$. $source(G') = source(G)$ and the generation of size-change graphs gives that the restriction of G' to $target(G)$ equals G, hence $T(G', G)$.

Case $S(\lambda\mathtt{x}.\mathtt{e}' : \rho', \lambda\mathtt{x}.\mathtt{e} : \rho)$: No calls from $\lambda\mathtt{x}.\mathtt{e} : \rho$.
(Value)-rule, $\lambda\mathtt{x}.\mathtt{e} : \rho \Downarrow \lambda\mathtt{x}.\mathtt{e} : \rho, id^{=}_{\lambda\mathtt{x}.\mathtt{e}}$ and $\lambda\mathtt{x}.\mathtt{e}' : \rho' \Downarrow \lambda\mathtt{x}.\mathtt{e}' : \rho', id^{=}_{\lambda\mathtt{x}.\mathtt{e}'}$. $T(id^{=}_{\lambda\mathtt{x}.\mathtt{e}'}, id^{=}_{\lambda\mathtt{x}.\mathtt{e}})$.

Case $S(\mathtt{e}_1'\mathtt{@}\mathtt{e}_2' : \rho', \mathtt{e}_1\mathtt{@}\mathtt{e}_2 : \rho)$:
(Operator)-rule, $\mathtt{e}_1\mathtt{@}\mathtt{e}_2 : \rho \underset{r}{\to} \mathtt{e}_1 : \rho, id^{\downarrow}_{e_1}$ and $\mathtt{e}_1'\mathtt{@}\mathtt{e}_2' : \rho' \underset{r}{\to} \mathtt{e}_1' : \rho', id^{\downarrow}_{e_1'}$.Beginning from S-related states, by defintion of the relation S we have $S(\mathtt{e}_1' : \rho', \mathtt{e}_1 : \rho)$. Then $T(id^{\downarrow}_{e_1'}, id^{\downarrow}_{e_1})$

Case $S(\mathtt{A} : \rho', \mathtt{x} : \rho)$:
(Var)-rule: $\mathtt{x} : \rho \Downarrow \rho(\mathtt{x}) = \mathtt{e}_0 : \rho_0, G$ where $G = \{\mathtt{x} \overset{=}{\to} \epsilon\} \cup \{\mathtt{x} \overset{\downarrow}{\to} \mathtt{y} \mid \mathtt{y} \in fv(\mathtt{e}_0)\}$. By the definition of S we must have $\mathtt{A} \Rightarrow^*_\Gamma x$. This agaíng by lemma 8.7 gives that we must have $\mathtt{A} ::= \mathtt{x}$. Then $\mathtt{A} : \rho' \underset{n}{\to} \mathtt{x} : \rho'$, $id^{=}_x$ by (Gram)-rule, and we have $S(\mathtt{x} : \rho', \mathtt{x} : \rho)$. Also $\mathtt{x} : \rho' \Downarrow \rho'(\mathtt{x}) = \mathtt{e}_0' : \rho_0', G''$ where $G'' = \{\mathtt{x} \overset{=}{\to} \epsilon\} \cup \{\mathtt{x} \overset{\downarrow}{\to} \mathtt{y} \mid \mathtt{y} \in fv(\mathtt{e}_0')\}$ by (Var)-rule. The edges in G'' are the same as the edges in $G' = id^{=}_x; G''$. Hence by (Result)-rule $A \Downarrow \rho'(\mathtt{x}), G'$. As before $S(\rho'(\mathtt{x}), \rho(\mathtt{x}))$ and $T(G', G)$.

Cases $S(\mathtt{A} : \rho', \lambda\mathtt{x}.\mathtt{e} : \rho)$ with (Value)-rule, and $S(\mathtt{A} : \rho', \mathtt{e}_1\mathtt{@}\mathtt{e}_2 : \rho)$ with (Operator)-rule: similarly by use of lemma 8.7 and reasoning as above. We wil use the rules (Gram)(Value)(Result) and (Gram)(Operator) respectively, where (Value) and (Operator) do not have premises.

Step cases.

Case $S(\mathtt{e}_1'\mathtt{@}\mathtt{e}_2' : \rho', \mathtt{e}_1\mathtt{@}\mathtt{e}_2 : \rho)$. $\mathtt{e}_1\mathtt{@}\mathtt{e}_2 : \rho \underset{d}{\to} \mathtt{e}_2 : \rho, id^{\downarrow}_{\mathtt{e}_2}$ by (Operand)-rule. It follows from the definition of S that also $S(\mathtt{e}_1' : \rho', \mathtt{e}_1 : \rho)$ hence by IH since $\mathtt{e}_1 : \rho \Downarrow$ then also $\mathtt{e}_1' : \rho' \Downarrow$ and then $\mathtt{e}_1'\mathtt{@}\mathtt{e}_2' : \rho' \underset{d}{\to} \mathtt{e}_2' : \rho', id^{\downarrow}_{\mathtt{e}_2'}$ and by the definition of S we have $S(\mathtt{e}_2' : \rho', \mathtt{e}_2 : \rho)$,

$T(id^{\downarrow}_{\mathtt{e}'_2}, id^{\downarrow}_{\mathtt{e}_2})$.

The next case is the one that requires the most consideration to see that we stay within the T-relation. Assume we know for graphs $\tilde{G}', \tilde{G}$, that the restriction of $\tilde{G}'$ to source and target of $\tilde{G}$ is a subset of $\tilde{G}$. Notice, if $x, y \in \mathit{source}(\tilde{G}') \setminus \mathit{source}(\tilde{G})$ and $\mathtt{x}, \mathtt{z} \in \mathit{target}(\tilde{G}') \setminus \mathit{target}(\tilde{G})$, then for testing $T(\tilde{G}', \tilde{G})$ we only need to look at which edges leaves from $\mathtt{x}, \mathtt{y}$, we do not need to care about if other edges goes into $\mathtt{x}, \mathtt{z}$.

Case $S(\mathtt{e}'_1 \mathtt{@} \mathtt{e}'_2 : \rho', \mathtt{e}_1 \mathtt{@} \mathtt{e}_2 : \rho)$. $\mathtt{e}_1 \mathtt{@} \mathtt{e}_2 : \rho \underset{c}{\rightarrow} \mathtt{e}_0 : \rho_0[\mathtt{x} \mapsto v_2], G_1^{-\epsilon/\lambda \mathtt{x}.\mathtt{e}_0} \cup_{\mathtt{e}_0} G_2^{\epsilon \mapsto \mathtt{x}}$ by (Call)-rule, where we have the premises $\mathtt{e}_1 : \rho \Downarrow \lambda \mathtt{x}.\mathtt{e}_0 : \rho_0, G_1$ and $\mathtt{e}_2 : \rho \Downarrow v_2, G_2$.

It follows from the definition of S that also $S(\mathtt{e}'_1 : \rho', \mathtt{e}_1 : \rho)$ and $S(\mathtt{e}'_2 : \rho', \mathtt{e}_2 : \rho)$. Hence by IH since $\mathtt{e}_1 : \rho \Downarrow \lambda \mathtt{x}.\mathtt{e}_0 : \rho_0, G_1$ then also $\mathtt{e}'_1 : \rho' \Downarrow v, G'_1$ where $T(G'_1, G_1)$ and $S(v, \lambda \mathtt{x}.\mathtt{e}_0 : \rho_0)$. Then by definition of values, relations $\Rightarrow^*_\Gamma$ and S we must have $v = \lambda \mathtt{x}.\mathtt{e}'_0 : \rho'_0$. Also by IH since $\mathtt{e}_2 : \rho \Downarrow v_2, G_2$ then also $\mathtt{e}'_2 : \rho' \Downarrow v'_2, G'_2$ where $T(G'_2, G_2)$ and $S(v'_2, v_2)$. Then we have the premises to conclude $\mathtt{e}'_1 \mathtt{@} \mathtt{e}'_2 : \rho' \underset{c}{\rightarrow} \mathtt{e}'_0 : \rho'_0[\mathtt{x} \mapsto v'_2], G_1'^{-\epsilon/\lambda x.e'_0} \cup_{e'_0} G_2'^{\epsilon \mapsto \mathtt{x}}$. By definition of S we have $S(\mathtt{e}'_0 : \rho'_0[\mathtt{x} \mapsto v'_2], \mathtt{e}_0 : \rho_0[\mathtt{x} \mapsto v_2])$. We notice that $\mathtt{x} \notin \mathit{fv}(\lambda \mathtt{x}.\mathtt{e}'_0)$ and therefore $(p \xrightarrow{r} \mathtt{x}) \notin G'_1$.
We consider different possibilities for the generated graphs:

If $\mathtt{x} \in \mathit{fv}(\mathtt{e}'_0)$ but $\mathtt{x} \notin \mathit{fv}(\mathtt{e}_0)$ then we can have some extra edges going to $\mathtt{x}$ in extended semantics where we will have no edges to $\mathtt{x}$ in pure semantics because $\mathtt{x}$ is not in the target, but this is acceptable in the T-relation. There can also be some extra edges going to ϵ in pure semantics where no edges go to ϵ in exact semantics, but as ϵ is within the codomain in pure semantics, this is also acceptable in the T-relation. Since $T(G'_1, G_1)$ it will still hold that $T(G_1'^{-\epsilon/\lambda x.e'_0} \cup_{e'_0} G_2'^{\epsilon \mapsto \mathtt{x}}, G_1^{-\epsilon/\lambda x.e_0} \cup_{e_0} G_2^{\epsilon \mapsto \mathtt{x}})$.

If $\mathtt{x} \in \mathit{fv}(\mathtt{e}_0)$ then also $\mathtt{x} \in \mathit{fv}(\mathtt{e}'_0)$ and if $\mathtt{x} \notin \mathit{fv}(\mathtt{e}'_0)$ then $\mathtt{x} \notin \mathit{fv}(\mathtt{e}_0)$, in these cases since $T(G'_1, G_1)$ and $T(G'_2, G_2)$ also $T(G_1'^{-\epsilon/\lambda x.e'_0} \cup_{e'_0} G_2'^{\epsilon \mapsto \mathtt{x}}, G_1^{-\epsilon/\lambda x.e_0} \cup_{e_0} G_2^{\epsilon \mapsto \mathtt{x}})$.

Case $S(\mathtt{A} : \rho', \mathtt{e}_1 \mathtt{@} \mathtt{e}_2 : \rho)$ with $\mathtt{e}_1 \mathtt{@} \mathtt{e}_2 : \rho \underset{d}{\rightarrow} \mathtt{e}_2 : \rho, id^{\downarrow}_{\mathtt{e}_2}$ by(Operand)-rule. By the definition of S we must have $\mathtt{A} \Rightarrow^*_\Gamma \mathtt{e}_1 \mathtt{@} \mathtt{e}_2$. This againg by lemma 8.7 gives that we must have $\mathtt{A} ::= \mathtt{e}'_1 \mathtt{@} \mathtt{e}'_2$. Then $\mathtt{A} : \rho' \underset{n}{\rightarrow} \mathtt{e}'_1 \mathtt{@} \mathtt{e}'_2 : \rho'$, $id^{=}_{e'_1 @ e'_2}$ by (Gram)-rule, and we have $S(\mathtt{e}'_1 \mathtt{@} \mathtt{e}'_2 : \rho', \mathtt{e}_1 \mathtt{@} \mathtt{e}_2 : \rho)$. Then we have seen that $\mathtt{e}'_1 \mathtt{@} \mathtt{e}'_2 : \rho' \underset{d}{\rightarrow} \mathtt{e}'_2 : \rho', id^{\downarrow}_{\mathtt{e}'_2}$ with $S(\mathtt{e}'_2 : \rho', \mathtt{e}_2 : \rho)$, $T(id^{\downarrow}_{\mathtt{e}'_2}, id^{\downarrow}_{\mathtt{e}_2})$ and we have that the edges of $id^{\downarrow}_{\mathtt{e}'_2}$ are the same as the edges of $(id^{=}_{e'_1 @ e'_2}; id^{\downarrow}_{\mathtt{e}'_2})$ hence $T((id^{=}_{e'_1 @ e'_2}; id^{\downarrow}_{\mathtt{e}'_2}), id^{\downarrow}_{\mathtt{e}_2})$.

Case $S(A : \rho', e_1 @ e_2 : \rho)$ with (Call)-rule $e_1 @ e_2 : \rho \underset{c}{\rightarrow} e_0 : \rho_0[x \mapsto v_2], G$: Similarly as before we have $A : \rho' \underset{n}{\rightarrow} e'_1 @ e'_2 : \rho'$, $id^{=}_{e'_1 @ e'_2}$ by (Gram)-rule, and we have $S(e_1 @ e_2 : \rho, e'_1 @ e'_2 : \rho')$. We can now use the derivation above and with the notation from above we have $e'_1 @ e'_2 : \rho' \underset{c}{\rightarrow} e'_0 : \rho'_0[x \mapsto v'_2], G'$ with $S(e'_0 : \rho'_0[x \mapsto v'_2], e_0 : \rho_0[x \mapsto v_2])$ and $T(G', G)$. Looking into the derivation of G' we find that the edges of G' are the same as the edges of $(id^{=}_{e'_1 @ e'_2}; G')$.

Case $S(\mathtt{e}' : \rho', \mathtt{e} : \rho)$, $\mathtt{e} : \rho \Downarrow v, G$ by (Result)-rule, where we have the premises $\mathtt{e} : \rho \underset{c}{\rightarrow} \mathtt{e}_s : \rho_s, G_s$ and $\mathtt{e}_s : \rho_s \Downarrow v, G_v$, $G = G_s; G_v$: By IH since $e : \rho \underset{c}{\rightarrow} \mathtt{e}_s : \rho_s, G_s$

then $\mathtt{e}' : \rho' \underset{n}{\to}{}^j s : \rho' \underset{c}{\to} \mathtt{e}'_s : \rho'_s, G'_s$ with $S(\mathtt{e}'_s : \rho'_s, \mathtt{e}_s : \rho_s)$, and $T(G'_s, G_s)$, $j \in \{0, 1\}$. Again by IH since $\mathtt{e}_s : \rho_s \Downarrow v, G_v$ then $\mathtt{e}'_s : \rho'_s \Downarrow v', G'_v$ with $S(v, v')$ and $T(G'_v, G_v)$. Let $G' = G'_s; G'_v$ then $T(G', G)$. If $j = 0$ we have the premises to conclude $\mathtt{e}' : \rho' \Downarrow v', G'$. If $j = 1$ by lemma 8.17 we have $S(s : \rho', \mathtt{e} : \rho)$ and we have the premises to conclude $s : \rho' \Downarrow v', G$, and by applications of (Result)-rule once more in the extended semantics we can also conclude $\mathtt{e}' : \rho' \Downarrow v', id_s^{=}; G'$ where the edge set of G' is the same as the edge set of $id_s^{=}; G'$. □

References

[1] A. Abel. Termination checking with types. *RAIRO - Theoretical Informatics and Applications, Special Issue: Fixed Points in Computer Science*, 38(4):277–319, 2004.

[2] A. Abel. *A Polymorphic Lambda-Calculus with Sized Higher-Order Types.* PhD thesis, Ludwig-Maximilians-Universität München, 2006.

[3] A. Ahmed. Step-indexed syntactic logical relations for recursive and quantified types. In P. Sestoft, editor, *Programming Languages and Systems. 15th European Symposium on Programming, ESOP 2006*, volume 3924 of *Lecture Notes in Computer Science*, pages 69–83. Springer, 2006.

[4] T. Arts and J. Giesl. Termination of term rewriting using dependency pairs. *Theoretical Computer Science*, 236:133–178, 2000.

[5] G. Barthe, M. J. Frade, E. Giménez, L. Pinto, and T. Uustalu. Type-based termination of recursive definitions. *Mathematical. Structures in Comp. Sci. 14:97–141*, 2004.

[6] N. Benton, A. Kennedy, M. Hofmann, and L. Beringer. Relational semantics for effect-based program transformations with dynamic allocation. In *Proceedings of the Ninth International ACM SIGPLAN Symposium on Principles and Practice of Declarative Programming (PPDP '07)*, 2007.

[7] N. Benton and B. Leperchey. Relational reasoning in a nominal semantics for storage. In *Proceedings of the Seventh International Conference on Typed Lambda Calculi and Applications (TLCA'05)*, volume 3461 of *Lecture Notes in Computer Science*. Springer, 2005.

[8] G. M. Bierman, A. M. Pitts, and C. V. Russo. Operational properties of Lily, a polymorphic linear lambda calculus with recursion. In *Fourth International Workshop on Higher Order Operational Techniques in Semantics, Montréal*, volume 41 of *Electronic Notes in Theoretical Computer Science*. Elsevier, September 2000.

[9] L. Birkedal and R. Harper. Constructing interpretations of recursive types in an operational setting. *Information and Computation*, 155:3–63, 1999.

[10] L. Birkedal and R. E. Møgelberg. Categorical models of Abadi-Plotkin's logic for parametricity. *Mathematical Structures in Computer Science*, 15(4):709–772, 2005.

[11] L. Birkedal, R. E. Møgelberg, and R. L. Petersen. Linear Abadi & Plotkin logic. *Logical Methods in Computer Science*, 2(5:1):1–33, 2006.

[12] N. Bohr. *Advances in Reasoning Principles for Contextual Equivalence and Termination.* PhD thesis, IT University of Copenhagen, 2007. Submitted.

[13] N. Bohr and L. Birkedal. Relational reasoning for recursive types and references. In *In Proc. of the Fourth ASIAN Symposium on Programming Languages and Systems (APLAS'06)*, 2006.

[14] P. Cousot and R. Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction of approximation of fixpoints. *POPL - 77: 4th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 238–252, 1977.

[15] K. Crary and R.W. Harper. Syntactic logical relations for polymorphic and recursive types. *Gordon Plotkin Festschrift*, 2007. To Appear.

[16] C.C. Frederiksen and N.D. Jones. Running-time analysis and implicit complexity. *Journal of Automated reasoning*, 2006.

[17] J. Giesl, S. Swiderski, P. Schneider-Kamp, and R. Thiemann. Automated termination analysis for haskell: From term rewriting to programming languages. In *RTA 2006: Rewriting Techniques and Applications: (Frank Pfenning, eds), Lecture Notes in Computer Science*, volume 4098, pages 297–312, 2006.

[18] J. Giesl, R. Thiemann, and P. Schneider-Kamp. Proving and disproving termination of higher-order functions. Technical report, RWTH Aachen, 2005.

[19] M. Hasegawa. Relational parametricity and control. *Logical Methods in Computer Science*, 2006.

[20] P. Johann. On proving the correctness of program transformations based on free theorems for higher-order polymorphic calculi. *Mathematical Structures in Computer Science*, 10(2):201–229, 2005.

[21] P. Johann and J. Voigtlaender. The impact of seq on free theorems-based program transformations. *Fundamenta Informaticae*, 69(1–2):63–102, 2006.

[22] N.D. Jones. Flow analysis of lambda expressions. In *ICALP, Lecture Notes in Computer Science*. Springer-Verlag, 1981.

[23] N.D. Jones and N. Bohr. Termination analysis of the untyped λ-calculus. In *RTA: Rewriting Techniques and Applications (V. van Oostrom, eds.), Lecture Notes in Computer Science*, volume 3091, pages 1–23. Springer-Verlag, June 2004.

[24] N.D. Jones and A. Glenstrup. Partial evaluation termination analysis and specialization-point insertion. *ACM Transactions on Programming Languages and Systems*, 2005.

[25] N.D. Jones and F. Nielson. Abstract interpretation:a semantics-based tool for program analysis. In *Handbook of Logic in Computer Science*, volume 4, pages 527–629, 1994.

[26] P. Taylor J.Y. Girard, Y. Lafont. *Proofs and Types*. Cambridge University Press, 1989.

[27] V. Koutavas and M. Wand. Bisimulations for untyped imperative objects. In Peter Sestoft, editor, *Programming Languages and Systems, 15th European Symposium on Programming, ESOP 2005, Vienna, Austria.*, volume 3924 of *Lecture Notes in Computer Science*. Springer, 2006. to appear.

[28] V. Koutavas and M. Wand. Small bisimulations for reasoning about higher-order imperative programs. In *POPL '06: Proceedings of the 33rd ACM SIGPLAN-SIGACT symposium on Principles of Programming Languages*, pages 141–152, New York, NY, USA, 2006. ACM Press.

[29] C.S. Lee. *Program Termination Analysis and the Termination of Offline Partial Evaluation*. PhD thesis, University of Western Australia, March 2001.

[30] C.S. Lee. Finiteness analysis in polynomial time. In *Static Analysis: 9th International Symposium, SAS 2002 (M Hermenegildo and G Puebla, eds.), Lecture Notes in Computer Science*, volume 2477, pages 493–508. Springer Verlag, 2002.

[31] C.S. Lee. Program termination analysis in polynomial time. In *In Generative Programming and Component Engineering: ACM SIGPLAN/SIGSOFT Conference, GPCE 2002 (D Batory, C Consel, and W Taha, eds.), Lecture Notes in Computer Science*, volume 2487, pages 218–235. Springer Verlag, 2002.

[32] C.S. Lee, N.D. Jones, and A.M. Ben-Amram. The size-change principle for program termination. In *POPL 2001: Proceedings* 28^{th} *ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, January 2001.

[33] I. A. Mason, S. Smith, and C. L. Talcott. From operational semantics to domain theory. *Information and Computation*, 128(1):26–47, 1996.

[34] Paul-André Melliès and Jérôme Vouillon. Recursive polymorphic types and parametricity in an operational framework. In *In Proc. of the 21th Conference on Logic in Computer Science (LICS'05)*, 2005.

[35] R.E. Møgelberg. Interpreting polymorphic fpc into domain theoretic models of parametric polymorphism. In *In Proc. of 33rd International Colloquium on Automata, Languages and Programming (ICALP'06)*, 2006.

[36] R.E. Møgelberg and A. Simpson. Relational parametricity for computational effects. In *In Proc. of LICS'07*, 2007.

[37] R.E. Møgelberg and A. Simpson. Relational parametricity for control considered as a computational effect. In M. Fiore, editor, *In Proceedings of Twenty-third Conference on the Mathematical Foundations of Programming Semantics (MFPS'07)*, ENTCS, 2007.

[38] A. M. Pitts. Parametric polymorphism and operational equivalence. *Mathematical Structures in computer Science*, 10:321–359, 2000.

[39] A. M. Pitts and I. D. B. Stark. Operational reasoning for functions with local state. In A. D. Gordon and A. M. Pitts, editors, *Higher Order Operational Techniques in Semantics*, Publications of the Newton Institute, pages 227–273. Cambridge University Press, 1998.

[40] A.M. Pitts. Relational properties of domains. *Information and Computation*, 127:66–90, 1996.

[41] A.M. Pitts. *Advanced Topics in Types and Programming Languages*, chapter Typed Operational Reasoning. MIT Press, 2005.

[42] A.M. Pitts and I.D.B. Stark. Observable properties of higher order functions that dynamically create local names, or: What's new? In *Mathematical Foundations of Computer Science, Proc. 18th Int. Symp., Gdańsk, 1993*, volume 711 of *Lecture Notes in Computer Science*, pages 122–141. Springer-Verlag, Berlin, 1993.

[43] G.D. Plotkin. Call-by-name, call-by-value and the lambda-calculus. *Theoretical Computer Science*, 1, 1975.

[44] G.D. Plotkin. Second order type theory and recursion. Notes for a talk at the Scott Fest, February 1993.

[45] Gordon Plotkin and Martín Abadi. A logic for parametric polymorphism. In *Typed lambda calculi and applications (Utrecht, 1993)*, volume 664 of *Lecture Notes in Comput. Sci.*, pages 361–375. Springer, Berlin, 1993.

[46] U. Reddy and H. Yang. Correctness of data representations involving heap data structures. *Science of Computer Programming*, 50(1-3):129–160, March 2004.

[47] J. C. Reynolds. Separation logic: A logic for shared mutable data structures. In *Proc. of the 17th Annual IEEE Symposium on Logic in Computer Science (LICS'02)*, pages 55–74, Copenhagen, Denmark, July 2002. IEEE Press.

[48] D. Sereni. λ-sct and simple types. E-mail communication, April 2005.

[49] D. Sereni. *Termination Analysis of Higher-Order Functional Programs.* PhD thesis, OUCL (Oxford University Computing Laboratory), 2006.

[50] D. Sereni and N.D. Jones. Termination analysis of higher-order functional programs. In *APLAS 2005: The Third Asian Symposium on Programming Languages and Systems (Kwangkeun Yi, ed.), Lecture Notes in Computer Science*, volume 3780, pages 281–297. Springer Verlag, November 2005.

[51] M. R. Shinwell and A. M. Pitts. On a monadic semantics for freshness. *Theoretical Computer Science*, 342:28–55, 2005.

[52] M.R. Shinwell. *The Fresh Approach: Functional Programming with Names and Binders*. PhD thesis, Computer Laboratory, Cambridge University, December 2004.

[53] D. Olin Shivers. *Control-Flow Analysis of Higher-Order Languages.* PhD thesis, Carnegie Mellon University, 1991.

[54] D. Olin Shivers. Higher-order control-flow analysis in retrospect: Lessons learned, lessons abandoned. In K. S. McKinley, editor, *20 Years of the ACM SIGPLAN Conference on Programming Language Design and Implementation (1979-1999): A Selection*, pages 269–270, 2004.

[55] Eijiro Sumii and Benjamin C. Pierce. A bisimulation for type abstraction and recursion. In *ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages (POPL), Long Beach, California*, 2005.

[56] W.W. Tait. Intensional interpretation of functionals of finite type i. *Journal of Symbolic Logic*, 32:198–212, 1967.

[57] Y. Toyama. Termination of s-expression rewriting systems: Lexicographic path ordering for higher-order terms. In *Proceedings of the 15th International Conference on Rewriting Techniques and Applications (RTA 2004), Lecture Notes in Computer Science*, volume 3091, pages 40–54. Springer Verlag, 2004.

[58] P. Wadler. Theorems for free! In *4'th Symposium on Functional Programming Languages and Computer Architecture, ACM, London*, September 1989.

[59] D. Wahlstedt. *Type Theory with First-Order Data Types and Size-Change Termination.* PhD thesis, Chalmers University of Technology, Gothenburg, Sweden, 2004. Licentiate thesis 2004, No. 36L, ISSN:1651-4963.

[60] D. Wahlstedt. *Dependent Type Theory with Paraetrized First-Order Data Types and Well-Founded Recursion.* PhD thesis, Chalmers University of Technology, Gothenburg, Sweden, 2007. Ph.D. thesis.

[61] Hongwei Xi. Dependent types for program termination verification. *Journal of Higher-Order Symbolic Logic*, 15(1):91–131, 2002.