

# Light Functional Interpretation - an optimization of Gödel's technique towards the extraction of (more) efficient programs from (classical) proofs - Technical Appendix

Mircea-Dan Hernest\*

danher@lix.polytechnique.fr  
Laboratoire d'Informatique (LIX),  
École Polytechnique, F-91128 Palaiseau - FRANCE.

**Abstract.** We give a Natural Deduction formulation of an adaptation of Gödel's functional (*Dialectica*) interpretation to the extraction of (more) efficient programs from (classical) proofs. We adapt Jørgensen's formulation of pure *Dialectica* translation by eliminating his "Contraction Lemma" and allowing free variables in the extracted terms (which is more suitable in a Natural Deduction setting). We also adapt Berger's *uniform* existential and universal quantifiers to the *Dialectica*-extraction context. The use of such quantifiers *without computational meaning* permits the identification and isolation of contraction formulas which would otherwise be redundantly included in the pure-*Dialectica* extracted terms. In the end we sketch the possible combination of our refinement of Gödel's *Dialectica* interpretation with its adaptation to the extraction of bounds due to Kohlenbach into a *light monotone* functional interpretation.

**Keywords:** Program extraction from (classical) proofs, Complexity of extracted programs, Berger's uniform quantifiers, Gödel's Functional interpretation, Proof-Carrying Code, Proof Mining.

## 1 Decidability, Case Distinction and Stability for quantifier-free formulas of WE-Z

**Lemma 11 (Decidability of prime formulas)**  $\vdash \forall q^o. \text{at}(q) \vee \neg \text{at}(q)$

**Proof:** By unfolding the definition of  $\vee$  the conclusion becomes

$$\vdash \forall q \exists p ((\text{at}(p) \rightarrow \text{at}(q)) \wedge (\neg \text{at}(p) \rightarrow \neg \text{at}(q)))$$

which is immediate in minimal logic after we set  $p := q$ .  $\square$

**Lemma 12 (Case Distinction on prime formulas)** For every formula  $A$  of WE-Z the following holds:  $\vdash \forall q^o. (\text{at}(q) \rightarrow A) \wedge (\neg \text{at}(q) \rightarrow A) \rightarrow A$ .

**Proof:** Immediate from AxBIA, AxTRH and the definitions of  $\neg$  and  $\perp$ .  $\square$

**Lemma 13 (Equivalence of boolean and logical constants)**

The following lemmas hold in WE-Z<sup>-</sup>:

---

\* *Project LogiCal* - Pôle Commun de Recherche en Informatique du Plateau de Saclay, CNRS, École Polytechnique, INRIA et Université Paris-Sud - FRANCE and *Graduiertenkolleg Logik in der Informatik* (GKLI) - München, GERMANY. Partly financed by *Deutsche Forschungsgemeinschaft*.

$$\begin{aligned}
\text{LmAND:} \quad & \vdash_{-} \forall p^o, q^o (\text{at}(\text{And } pq) \leftrightarrow \text{at}(p) \wedge \text{at}(q)) \\
\text{LmIMP:} \quad & \vdash_{-} \forall p^o, q^o (\text{at}(\text{Imp } pq) \leftrightarrow \text{at}(p) \rightarrow \text{at}(q)) \\
\text{LmNOT:} \quad & \vdash_{-} \forall p^o (\text{at}(\text{Not } p) \leftrightarrow \neg \text{at}(p)) \\
\text{LmOR:} \quad & \vdash_{-} \forall p^o, q^o (\text{at}(\text{Or } pq) \leftrightarrow \text{at}(p) \vee \text{at}(q)) \\
\text{LmEQ:} \quad & \vdash_{-} \forall p^o, q^o (\text{at}(\text{Eq } pq) \leftrightarrow (\text{at}(p) \leftrightarrow \text{at}(q)) \equiv p =_o q)
\end{aligned}$$

**Proof:** By boolean induction on  $p$ , using definitions,  $\text{AxIf}_o$  and logic - directly or as  $\text{WE-Z}$  consequences of the following  $\text{WE-Z}^-$  lemma (denoted  $\text{LmIF}$ ):

$$\vdash_{-} \forall p^o, q_1^o, q_2^o (\text{at}(\text{If}_o p q_1 q_2) \leftrightarrow (\text{at}(p) \rightarrow \text{at}(q_1)) \wedge (\neg \text{at}(p) \rightarrow \text{at}(q_2))) . \square$$

**Lemma 14 (Equivalence of quantifier-free and prime formulas)**

There exists a unique bijective association of boolean terms to quantifier-free formulas  $A_0 \mapsto \mathfrak{t}_{A_0}$  such that for all  $A_0$ ,  $\vdash_{-} A_0 \leftrightarrow \text{at}(\mathfrak{t}_{A_0})$ .

**Proof:** By induction on the structure of  $A_0$ . Obvious for prime formulas. For composed formulas one uses  $\text{LmAND}$  and  $\text{LmIMP}$ .  $\square$

**Lemma 15 (Quantifier-free Decidability and Case Distinction)**

All quantifier-free formulas  $A_0$  of system  $\text{WE-Z}$  are decidable in the sense that  $\vdash_{-} A_0 \vee \neg A_0$ . Moreover, the following holds for an arbitrary formula  $A$  of  $\text{WE-Z}$ :  $\vdash_{-} (A_0 \rightarrow A) \wedge (\neg A_0 \rightarrow A) \rightarrow A$ .

**Proof:** Immediate consequence of Lemmas 14, 11 and 12.  $\square$

**Lemma 16 (Quantifier-free Stability)** The schema  $\vdash_{-} \neg \neg A_0 \rightarrow A_0$  holds for all quantifier-free formulas  $A_0$ .

**Proof:** While  $\vdash_{-} A_0 \rightarrow (\neg \neg A_0 \rightarrow A_0)$  is obvious,  $\vdash_{-} \neg A_0 \rightarrow (\neg \neg A_0 \rightarrow A_0)$  follows immediately by  $\rightarrow^-$  and  $\text{AxEFQ}$ . The result then follows by case distinction on  $A_0$ , see Lemma 15 above.  $\square$

**Lemma 17 (Disjunction Elimination)** For all  $n \in \mathbb{N}$  and  $B, A_i \in \text{WE-Z}^-$

$$\vdash_{-} \bigwedge_{i=1}^n (A_i \rightarrow B) \rightarrow (\bigvee_{i=1}^n A_i \rightarrow B) \quad (1)$$

**Proof:** It is sufficient to establish (1) for  $n = 2$  which immediately gives the induction step in a proof for general  $n$ . By expanding the definition of  $A_1 \vee A_2$  and using  $\text{Ax}\exists^-$  we only need to prove that

$$A_1 \rightarrow B, A_2 \rightarrow B \vdash_{-} \forall p^o. (\text{at}(p) \rightarrow A_1) \wedge (\neg \text{at}(p) \rightarrow A_2) \rightarrow B$$

which reduces to finding a proof of  $B$  from assumptions  $A_1 \rightarrow B$ ,  $A_2 \rightarrow B$ ,  $\text{at}(p) \rightarrow A_1$  and  $\neg \text{at}(p) \rightarrow A_2$ . It is then enough to find a proof of  $B$  from assumptions  $\text{at}(p) \rightarrow B$  and  $\neg \text{at}(p) \rightarrow B$ . This is just a Case Distinction.  $\square$

**Definition 18 (Dialectica terms)** To every ncm-quantifier free formula  $A(a)$  we associate a boolean term  $\mathfrak{t}_A^D[x; y; a]$ , which we call *the Dialectica term associated to  $A(a)$*  such that

$$\vdash_{-} A_D(x; y; a) \leftrightarrow \text{at}(\mathfrak{t}_A^D[x; y; a]) \quad (2)$$

The association is by induction on the structure of the formula  $A$  :

$$\begin{aligned}
\mathfrak{t}_{\text{at}(t[a])}^{\text{D}}[; ; a] &::= t[a] \\
\mathfrak{t}_{A \wedge B}^{\text{D}}[x, u; y, v; a, b] &::= \text{And } \mathfrak{t}_A^{\text{D}}[x; y; a] \mathfrak{t}_B^{\text{D}}[u; v; b] \\
\mathfrak{t}_{\exists z A(z, a)}^{\text{D}}[z^\dagger, x; y; a] &::= \mathfrak{t}_{A(z, a)}^{\text{D}}[x; y; z^\dagger, a] \\
\mathfrak{t}_{\forall z A(z, a)}^{\text{D}}[X; z^\dagger, y; a] &::= \mathfrak{t}_{A(z, a)}^{\text{D}}[X(z^\dagger); y; z^\dagger, a] \\
\mathfrak{t}_{A \rightarrow B}^{\text{D}}[Y, U; x, v; a, b] &::= \text{Imp } \mathfrak{t}_A^{\text{D}}[x; Y(x, v); a] \mathfrak{t}_B^{\text{D}}[U(x); v; b]
\end{aligned}$$

**Remark 19** Notice how the definition of  $\mathfrak{t}_A^{\text{D}}$  parallels the definition of  $A^{\text{D}}$ . The proof of (2) is immediate by structural induction. Since  $\mathfrak{t}_A^{\text{D}} \equiv \mathfrak{t}_{A_{\text{D}}}$  (literal, syntactic equality with the term directly associated to the quantifier-free formula  $A_{\text{D}}$ ), (2) can also be established as direct consequence of Remark 22 from [1].

Below we present the D-interpretation of some (generic) formulas of interest:

$$\begin{aligned}
(\neg A)^{\text{D}}(a) &\equiv \exists Y \forall x \neg A_{\text{D}}(x; Y(x); a) \\
(\neg \neg A)^{\text{D}}(a) &\equiv \exists X \forall Y \neg \neg A_{\text{D}}(X(Y); Y(X(Y)); a) \\
(\exists^\dagger z A(z))^{\text{D}}(a) &\equiv \exists Z, X \forall Y \neg \neg A_{\text{D}}(X(Y); Y(Z(Y), X(Y)); Z(Y), a) \\
(\neg \neg \exists z A(z))^{\text{D}}(a) &\equiv \exists Z, X \forall Y \neg \neg A_{\text{D}}(X(Y); Y(Z(Y), X(Y)); Z(Y), a)
\end{aligned}$$

**Remark 110**  $\exists^\dagger z A(z)$  has the same D-interpretation as  $\neg \neg \exists z A(z)$ .

## 2 Equivalence between formulations of Induction

**Theorem 21** The induction axiom **IA** and the induction rules **IR** and **IR<sub>0</sub>** are all equivalent over Minimal Logic. Recall from Section 2.2 of [1] that

$$\text{IA} : \quad A(0) \rightarrow \forall z (A(z) \rightarrow A(\mathbf{S}z)) \rightarrow \forall z A(z)$$

$$\text{IR} : \quad \frac{A(0) \quad \forall z (A(z) \rightarrow A(\mathbf{S}z))}{\forall z A(z)}$$

$$\text{IR}_0 : \quad \frac{\begin{array}{c} \emptyset \\ \vdots \\ A(0) \end{array} \quad \begin{array}{c} \emptyset \\ \vdots \\ \forall z (A(z) \rightarrow A(\mathbf{S}z)) \end{array}}{\forall z A(z)}$$

**Proof:** We can produce the following proof of **IA** from **IR** :

$$\frac{\frac{\frac{[A(0)]}{A(0)} \quad \frac{[\forall z (A(z) \rightarrow A(\mathbf{S}z))]}{\forall z (A(z) \rightarrow A(\mathbf{S}z))}}{\forall z A(z)} \quad \text{IR}}{A(0) \rightarrow \forall z (A(z) \rightarrow A(\mathbf{S}z)) \rightarrow \forall z A(z)} \rightarrow^+$$

We now show how to simulate **IR** in terms of **IR<sub>0</sub>**. Let  $C$  be the set of all assumptions in the proof  $\mathcal{P}$  whose last rule is **IR**. Let  $C_b$  be the set of assumptions of  $A(0)$  and  $C_s$  be the set of assumptions of  $\forall z (A(z) \rightarrow A(\mathbf{S}z))$ , hence  $C = C_b \cup C_s$ .

We assume that the elements of  $C_b$  and  $C_s$  are given in an arbitrary but fixed order from left to right such that  $C$  is associated their composed order with the elements of  $C_b$  at left followed by the elements of  $C_s$  at right. We ensure (by, e.g., renaming of the bound variable) that  $z$  is *not* free in  $C$ . We now transform  $\mathcal{P}$  to  $\mathcal{P}'$  such that  $\mathcal{P}'$  uses the same assumptions  $C$  of  $\mathcal{P}$  but uses an  $\text{IR}_0$  to simulate the last  $\text{IR}$  of  $\mathcal{P}$ . We also try to make this simulation as efficient as possible, in the sense of efficiency of the extracted programs, by employing only the strictly necessary contractions. The latter (if any) are only due to a number of  $\rightarrow^+$  with 2-contraction<sup>1</sup> (one for each of the elements of  $C_s$ ). Let  $\mathcal{P}_b$  be the proof of  $A(0)$  and  $\mathcal{P}_s$  be the proof of  $\forall z(A(z) \rightarrow A(\mathbf{S}z))$  (both formulas from  $\text{IR}$ ). From  $\mathcal{P}_b$  we can immediately produce the proof  $\emptyset \vdash C_b \rightarrow (C_s \rightarrow A(0))$  by using a number of “dummy” (i.e., with no assumptions)  $\rightarrow^+$  (i.e., over the elements of  $C_s$ ), followed by a number of  $\rightarrow^+$  without contraction (i.e., over the elements of  $C_b$ ). We now build from  $\mathcal{P}_s$  a proof  $\emptyset \vdash \forall z[(C_b \rightarrow (C_s \rightarrow A(z))) \rightarrow C_b \rightarrow (C_s \rightarrow A(\mathbf{S}z))]$  which will require contractions only over the elements of  $C_s$ .

$$\frac{\frac{C_s \quad \frac{C_b \quad C_b \rightarrow (C_s \rightarrow A(z))}{C_s \rightarrow A(z)} \rightarrow^-}{A(z)} \rightarrow^- \quad \left. \begin{array}{c} C_s \\ \vdots \\ \forall z(A(z) \rightarrow A(\mathbf{S}z)) \end{array} \right\} \mathcal{P}_s}{\frac{\forall z(A(z) \rightarrow A(\mathbf{S}z))}{A(z) \rightarrow A(\mathbf{S}z)} \forall^-}{A(\mathbf{S}z)} \rightarrow^-$$

hence by first cancelling  $C_s$  (with contraction) and subsequently  $C_b$  (without contraction) we obtain a proof  $C_b \rightarrow (C_s \rightarrow A(z)) \vdash C_b \rightarrow (C_s \rightarrow A(\mathbf{S}z))$  to which we apply a last  $\rightarrow^+$  (without contraction) and obtain, with a final  $\forall_z^+$

$$\frac{\frac{\emptyset \quad \vdots}{(C_b \rightarrow (C_s \rightarrow A(z))) \rightarrow C_b \rightarrow (C_s \rightarrow A(\mathbf{S}z))} \forall^+}{\forall z[(C_b \rightarrow (C_s \rightarrow A(z))) \rightarrow C_b \rightarrow (C_s \rightarrow A(\mathbf{S}z))]} \forall^+$$

We can now apply  $\text{IR}_0$  to obtain  $\emptyset \vdash \forall z(C_b \rightarrow C_s \rightarrow A(z))$ , hence  $\emptyset \vdash C \rightarrow A(z)$  by an  $\forall^-$ . Now we assume  $C$  and obtain by a number of  $\rightarrow^-$  that  $C \vdash A(z)$ . Since we ensured that  $z$  is not free in  $C$  we can apply  $\forall^+$  to obtain  $\mathcal{P}' : C \vdash \forall z A(z)$ .

For closing the equivalences we mention that  $\text{IR}$  is immediate from  $\text{IA}$  by means of two  $\rightarrow^-$  and  $\text{IR}_0$  is just a particular case of  $\text{IR}$ .  $\square$

### 3 Soundness theorem for the light functional (*Dialectica*) interpretation

**Theorem 31 (Exact realizer synthesis by the LD-interpretation)** There exists an algorithm which given at input a proof  $\mathcal{P} : \{C^i\}_{i=1}^n \vdash_+ A$  produces at output the tuples of terms  $\{T_i\}_{i=1}^n$  and  $T$ , the tuples of variables  $\{x_i\}_{i=1}^n$  and  $y$  all together with the verifying proof

$$\mathcal{P}_D : \{C_D^i(x_i; T_i(\underline{x}, y))\}_{i=1}^n \vdash_- A_D(T(\underline{x}); y)$$

- where we denoted by  $\underline{x} := x_1, \dots, x_n$ . Moreover,

<sup>1</sup> These contractions are unavoidable in the simulation of  $\text{IR}$  in terms of  $\text{IR}_0$ . On the other hand, contractions over the assumptions in  $C_b$  would be redundant, reason why splitting  $C$  into  $C_b$  and  $C_s$ .

1. the variables  $\underline{x}$  and  $y$  do not occur in  $\mathcal{P}$  (they are all completely new)
2. the free variables of  $T$  and  $\{T_i\}_{i=1}^n$  are among the free variables of  $A$  and  $\{C^i\}_{i=1}^n$  (which is called “the *free variable condition* (FVC) for programs extracted by the D-interpretation”)

hence  $\underline{x}$  and  $y$  also do not occur free in the *extracted* terms  $\{T_i\}_{i=1}^n$  and  $T$ .

**Proof:** The algorithm proceeds by recursion on the structure of the input proof  $\mathcal{P}$ . Realizing terms must be presented for all the axioms and then realizing terms for the conclusion of a rule must be deduced from terms which realize the premise of that rule. Since  $\underline{x}, y$  are produced by the D-interpretation of formulas (see Definition 31 from [1]) it is immediate that they do not occur in  $\mathcal{P}$ . The case of Implication Introduction  $\rightarrow^+$  where at least two copies of the assumption get cancelled has already been treated in the proof of Theorem 34 from Section 3 of [1]. We here present all the remaining cases.

### The rules of Intuitionistic Logic

$\boxed{A \vdash A}$  Just take  $T_1 := \lambda x, y. y$  and  $T := \lambda x. x$ . The FVC is obviously satisfied.

$\boxed{\frac{A \wedge B}{A} \wedge_l^-}$  We are given that (with  $\underline{x} \equiv x_1, \dots, x_n$ )

$$\{C_D^i(x_i; S_i(\underline{x}, y', y''))\}_{i=1}^n \vdash A_D(S'(\underline{x}); y') \wedge B_D(S''(\underline{x}); y'')$$

in which we substitute the variables from  $\mathcal{V}_f(B) \setminus [\cup_{i=1}^n \mathcal{V}_f(C^i) \cup \mathcal{V}_f(A)]$  and those from  $y''$  with terms  $0$  of corresponding type. We denote by  $T'$  and  $\{S_i^*\}_{i=1}^n$  the (newly obtained after substitution) terms corresponding to  $S'$  and  $\{S_i\}_{i=1}^n$  respectively. For all  $i \in \overline{1, n}$  let  $T_i := \lambda \underline{x}, y'. S_i^*(\underline{x}, y', 0)$ . We finally obtain after a  $\wedge_l^-$ , with the FVC obviously satisfied, that

$$\{C_D^i(x_i; T_i(\underline{x}, y'))\}_{i=1}^n \vdash A_D(T'(\underline{x}); y').$$

$\boxed{\frac{A \wedge B}{B} \wedge_r^-}$  Similar to the previous case.

$\boxed{\frac{A, B}{A \wedge B} \wedge^+}$  We are given, with  $\underline{x}' \equiv x'_1, \dots, x'_n$  and  $\underline{x}'' \equiv x''_{n+1}, \dots, x''_m$ , that :

$$\{C_D^i(x'_i; T_i(\underline{x}', y'))\}_{i=1}^n \vdash A_D(T'(\underline{x}'); y')$$

$$\{C_D^i(x''_i; T_i(\underline{x}'', y''))\}_{i=1}^m \vdash B_D(T''(\underline{x}''); y'')$$

It has been assumed that  $n \leq m$ . By one application of  $\wedge^+$  we get

$$\frac{\{C_D^i(x'_i; T_i(\underline{x}', y'))\}_{i=1}^n, \{C_D^i(x''_i; T_i(\underline{x}'', y''))\}_{i=n+1}^m}{A_D(T'(\underline{x}'); y') \wedge B_D(T''(\underline{x}''); y'')}.$$

Let  $\underline{x} := \underline{x}', \underline{x}'', S' := \lambda \underline{x}. T'(\underline{x}')$ ,  $S'' := \lambda \underline{x}. T''(\underline{x}'')$  and for  $i \in \overline{1, m}$ ,

$$S_i := \begin{cases} \lambda \underline{x}, y', y''. T_i(\underline{x}', y') & , \text{ if } 1 \leq i \leq n \\ \lambda \underline{x}, y', y''. T_i(\underline{x}'', y'') & , \text{ if } n < i \leq m \end{cases}$$

Then we obviously have (with the FVC immediately satisfied)

$$\{C_{\mathbb{D}}^i(x_i; S_i(\underline{x}, y', y''))\}_{i=1}^m \vdash (A \wedge B)_{\mathbb{D}}(S'(\underline{x}), S''(\underline{x}); y', y'') \quad .$$

$\frac{A, A \rightarrow B}{B} \rightarrow^-$	We are given, with $\underline{x}' \equiv x'_1, \dots, x'_n$ and $\underline{x}'' \equiv x''_{n+1}, \dots, x''_m$ , that :
--	--

$$\{C_{\mathbb{D}}^i(x'_i; T_i(\underline{x}', y'))\}_{i=1}^n \vdash A_{\mathbb{D}}(T'(\underline{x}'); y') \quad (3)$$

$$\{C_{\mathbb{D}}^i(x''_i; T_i(\underline{x}'', y'', y))\}_{i=n+1}^m \vdash (A \rightarrow B)_{\mathbb{D}}(T''(\underline{x}''), T(\underline{x}''); y'', y) \quad (4)$$

It has been assumed that  $n \leq m$ . It is more convenient to redisplay (4) as

$$\{C_{\mathbb{D}}^i(x''_i; T_i(\underline{x}'', y'', y))\}_{i=n+1}^m \vdash A_{\mathbb{D}}(y''; T''(\underline{x}'', y'', y)) \rightarrow B_{\mathbb{D}}(T(\underline{x}'', y''); y) \quad (5)$$

We substitute  $\{y' \leftarrow T''(\underline{x}'', T'(\underline{x}'), y)\}$  in (3) and  $\{y'' \leftarrow T'(\underline{x}')\}$  in (5). Then

$$\frac{\{C_{\mathbb{D}}^i(x'_i; T_i(\underline{x}', T''(\underline{x}'', T'(\underline{x}'), y))\}_{i=1}^n, \{C_{\mathbb{D}}^i(x''_i; T_i(\underline{x}'', T'(\underline{x}'), y))\}_{i=n+1}^m}{B_{\mathbb{D}}(T(\underline{x}'', T'(\underline{x}')); y)} \quad (6)$$

is obtained by an  $\rightarrow^-$  applied to (5) and substituted (3). In (6) we substitute the variables from  $\mathcal{V}_f(A) \setminus [\cup_{i=1}^m \mathcal{V}_f(C^i) \cup \mathcal{V}_f(B)]$  with terms  $\mathbf{0}$  of corresponding type. We denote by  $T^*, T'_*, T''_*$  and  $\{T_i^*\}_{i=1}^n$  the (newly obtained after substitution) terms corresponding to  $T, T', T''$  and  $\{T_i\}_{i=1}^n$  respectively. Let  $\underline{x} \equiv \underline{x}', \underline{x}''$  and then let  $S \equiv \lambda \underline{x}. T^*(\underline{x}'', T'_*(\underline{x}'))$  and

$$S_i := \begin{cases} \lambda \underline{x}, y. T_i^*(\underline{x}', T''_*(\underline{x}'', T'_*(\underline{x}'), y)) & , \text{ if } 1 \leq i \leq n \\ \lambda \underline{x}, y. T_i^*(\underline{x}'', T'_*(\underline{x}'), y) & , \text{ if } n < i \leq m \end{cases}$$

We then immediately obtain (with the FVC obviously satisfied)

$$\{C_{\mathbb{D}}^i(x_i; S_i(\underline{x}, y))\}_{i=1}^m \vdash B_{\mathbb{D}}(S(\underline{x}); y) \quad .$$

$\frac{[A] \dots / B}{A \rightarrow B} \rightarrow^+$	The case when at least two copies of $A$ get cancelled was treated in the proof of Theorem 34 of [1]. In the case when exactly one copy of $A$ gets cancelled we can use the same extracted terms except that the equation which defines $\tilde{S}$ in [1] is replaced by $\tilde{S} := \lambda \underline{x}, z, y. T_1(\underline{z}, \underline{x}, y)$ [in fact $\underline{z} \equiv z$ ]. We now consider the case when no copy of $A$ gets cancelled. We are given that $\{C_{\mathbb{D}}^i(x_i; T_i(\underline{x}, y))\}_{i=1}^m \vdash B_{\mathbb{D}}(T(\underline{x}); y)$ to which we can apply an $\rightarrow^+$ in which no copy of the assumption gets cancelled to obtain
---	--

$$\{C_{\mathbb{D}}^i(x_i; T_i(\underline{x}, y))\}_{i=1}^m \vdash A_{\mathbb{D}}(z; \mathbf{0}) \rightarrow B_{\mathbb{D}}(T(\underline{x}); y) .$$

We then simply define  $\{S_i := \lambda \underline{x}, z, y. T_i(\underline{x}, y)\}_{i=1}^m$ ,  $\tilde{S} := \lambda \underline{x}, z, y. \mathbf{0} \equiv \mathbf{0}$  and  $S := \lambda \underline{x}, z. T(\underline{x})$  to finally obtain, with the FVC obviously satisfied, that

$$\{C_{\mathbb{D}}^i(x_i; S_i(\underline{x}, z, y))\}_{i=1}^m \vdash A_{\mathbb{D}}(z; \tilde{S}(\underline{x}, z, y)) \rightarrow B_{\mathbb{D}}(S(\underline{x}, z); y) .$$

$\frac{\forall z A(z)}{A(t)} \forall_{z,t}^-$	We are given that $\{C_{\mathbb{D}}^i(x_i; S_i(\underline{x}, z^\dagger, y))\}_{i=1}^m \vdash A_{\mathbb{D}}(S(\underline{x}, z^\dagger); y; z^\dagger)$
---	--

where  $\underline{x} \equiv x_1, \dots, x_n$  and (by its construction)  $z^\dagger$  is not free in  $\{C_D^i\}_{i=1}^n$  and (also due to the FVC)  $z^\dagger$  is not free in  $S$  and/or  $\{S_i\}_{i=1}^n$ . By substituting  $\{z^\dagger \leftarrow t\}$  we therefore obtain that  $\{C_D^i(x_i; S_i(\underline{x}, t, y))\}_{i=1}^n \vdash A_D(S(\underline{x}, t); y; t)$ . Then by defining  $\{T_i := \lambda \underline{x}. S_i(\underline{x}, t)\}_{i=1}^n$  and  $T := \lambda \underline{x}. S(\underline{x}, t)$  we conclude, also using VC<sub>2</sub>, modulo some applications of COMPAT, with FVC obviously satisfied, that:

$$\{C_D^i(x_i; T_i(\underline{x}, y))\}_{i=1}^n \vdash A_D(T(\underline{x}); y; t) .$$

$\frac{A(z)}{\forall z A(z)} \forall_z^+$  We are given that  $\{C_D^i(x_i; T_i(\underline{x}, y))\}_{i=1}^n \vdash A_D(T(\underline{x}); y; z)$  where  $\underline{x} \equiv x_1, \dots, x_n$  and (by VC<sub>1</sub>)  $z$  is not free in the assumptions  $\{C^i\}_{i=1}^n$ . Due to the rules for variable naming in the LD-interpretation of a formula,  $z$  is also not among  $\underline{x}, y$  (see Definition 31 from [1]). On the other hand,  $z$  may be free in  $T$  and/or  $\{T_i\}_{i=1}^n$ . Let  $S := \lambda \underline{x}, z. T(\underline{x})$  and  $\{S_i := \lambda \underline{x}, z. T_i(\underline{x})\}_{i=1}^n$ . We then obtain, after some applications of the COMPAT rule and the substitution  $\{z \leftarrow z^\dagger\}$ , that (with the FVC obviously satisfied):

$$\{C_D^i(x_i; S_i(\underline{x}, z^\dagger, y))\}_{i=1}^n \vdash (\forall z A(z))_D(S(\underline{x}); z^\dagger, y) .$$

$\frac{\forall z A(z)}{A(t)} \forall_{z,t}^-$  We are given that  $\{C_D^i(x_i; T_i(\underline{x}, y))\}_{i=1}^n \vdash \forall z A_D(T(\underline{x}); y; z)$  to which we can immediately apply  $\forall_{z,t}^-$  since VC<sub>2</sub>( $z, t$ ) is obviously respected (due to the policy of variable naming in the LD-interpretation of a formula, see Definition 31 from [1]). We then obtain, with the FVC obviously satisfied, that

$$\{C_D^i(x_i; T_i(\underline{x}, y))\}_{i=1}^n \vdash A_D(T(\underline{x}); y; t) .$$

[ We here also used that  $z$  is not free in any of  $T$ ,  $\{T_i\}_{i=1}^n$  (due to the FVC). ]

$\frac{\mathcal{P}: A(z)}{\forall z A(z)} \forall_z^+$  We are given that  $\{C_D^i(x_i; T_i(\underline{x}, y))\}_{i=1}^n \vdash A_D(T(\underline{x}); y; z)$  where  $z$  is neither in  $\cup_{i=1}^n \mathcal{V}_i(C^i)$  (because of VC<sub>1</sub>( $z$ )) nor among  $\underline{x}, y$  (due to the rules for variable naming in the LD-interpretation of a formula, see Definition 31 from [1]). The fact that  $z$  is *not* free in *any* of  $\{T_i\}_{i=1}^n$  is ensured by VC<sub>3</sub>( $z, \mathcal{P}$ ). Since the pre-condition VC<sub>1</sub>( $z$ ) is established, we can apply a  $\forall_z^+$  in the verifying proof to obtain, with the FVC satisfied also due to VC<sub>3</sub>( $z, \mathcal{P}$ ), that

$$\{C_D^i(x_i; T_i(\underline{x}, y))\}_{i=1}^n \vdash \forall z A_D(T(\underline{x}); y; z) .$$

### The axioms of Intuitionistic Logic

$\text{Ax}\exists^- : \exists z_1 A(z_1) \wedge \forall z_2 [A(z_2) \rightarrow B] \rightarrow B$  The D-interpretation of  $\text{Ax}\exists^-$  is (for legibility we abbreviate below by  $\cdot := z, x, y, u, v$ )

$$\begin{array}{c} \exists Y, Z, X, V, U \forall z, x, y, u, v \\ \left[ \begin{array}{c} A_D(x; Y(x, y, u, v); z) \\ \wedge \\ [A_D(X(\cdot); y(Z(\cdot), X(\cdot), V(\cdot))); Z(\cdot)] \rightarrow B_D(u(Z(\cdot), X(\cdot)); V(\cdot)) \\ \rightarrow \\ B_D(U(z, x, y, u); v) \end{array} \right] \end{array}$$

for which a realizing tuple is  $T_Y, T_Z, T_X, T_V, T_U$  where

$$\begin{aligned} T_Y &:= \lambda z, x, y, u, v. y(z, x, v) \\ T_Z &:= \lambda z, x, y, u, v. z \\ T_X &:= \lambda z, x, y, u, v. x \\ T_V &:= \lambda z, x, y, u, v. v \\ T_U &:= \lambda z, x, y, u. u(z, x) \end{aligned}$$

$\boxed{\mathbf{Ax}\exists^+ : \forall z_1 [A(z_1) \rightarrow \exists z_2 A(z_2)]}$  The D-interpretation of  $\mathbf{Ax}\exists^+$  is

$$\exists Y, Z, X \forall z, x, y [A_D(x; Y(z, x, y); z) \rightarrow A_D(X(z, x); y; Z(z, x))]$$

for which a realizing tuple is  $T_Y, T_Z, T_X$  where

$$\begin{aligned} T_Y &:= \lambda z, x, y. y \\ T_Z &:= \lambda z, x. z \\ T_X &:= \lambda z, x. x \end{aligned}$$

$\boxed{\mathbf{Ax}\exists^- : \exists z_1 A(z_1) \wedge \forall z_2 [A(z_2) \rightarrow B] \rightarrow B}$  The D-interpretation of  $\mathbf{Ax}\exists^-$  is  
(for legibility we abbreviate below by  $\cdot := x, y, u, v$ )

$$\begin{aligned} &\exists Y, X, V, U \forall x, y, u, v \\ &\left[ \begin{array}{c} \exists z_1 A_D(x; Y(x, y, u, v); z_1, a) \\ \wedge \\ \forall z_2 [A_D(X(\cdot); y(X(\cdot), V(\cdot)); z_2, a) \rightarrow B_D(u(X(\cdot)); V(\cdot); b)] \\ \rightarrow \\ B_D(U(x, y, u); v; b) \end{array} \right] \end{aligned}$$

for which a realizing tuple is  $T_Y, T_X, T_V, T_U$  where

$$\begin{aligned} T_Y &:= \lambda x, y, u, v. y(x, v) \\ T_X &:= \lambda x, y, u, v. x \\ T_V &:= \lambda x, y, u, v. v \\ T_U &:= \lambda x, y, u. u(x) \end{aligned}$$

The verification is ensured by an instance of  $\mathbf{Ax}\exists^-$ .

$\boxed{\mathbf{Ax}\exists^+ : \forall z_1 [A(z_1) \rightarrow \exists z_2 A(z_2)]}$  The D-interpretation of  $\mathbf{Ax}\exists^+$  is

$$\exists Y, X \forall x, y \forall z_1 [A_D(x; Y(x, y); z_1, a) \rightarrow \exists z_2 A_D(X(x); y; z_2, a)]$$

for which a realizing tuple is  $T_Y, T_X$  where

$$\begin{aligned} T_Y &:= \lambda x, y. y \\ T_X &:= \lambda x. x \end{aligned}$$

The verification is ensured by an instance of  $\mathbf{Ax}\exists^+$ .

$\boxed{\mathbf{AxEFQ} : \perp \rightarrow A}$  The D-interpretation of  $\mathbf{AxEFQ}$  is  $\exists x \forall y (\perp \rightarrow A_D(x; y))$  for which a realizing tuple is  $T_x := 0$ .

### Arithmetical axioms and rules

The axioms  $\mathbf{AxREF}_l$ ,  $\mathbf{AxSYM}_l$ ,  $\mathbf{AxTRZ}_l$ ,  $\mathbf{AxTRH}$ ,  $\mathbf{AxS}$ ,  $\mathbf{AxR}$ ,  $\mathbf{Ax}\geq$  and  $\mathbf{AxIf}_\tau$  are all quantifier-free and the axioms  $\mathbf{AxREF}_\tau$  are purely universal for higher-order  $\tau$ . Therefore none of them produces any realizing term. The  $\mathbf{qfr}$  axioms verify themselves. For the purely universal axioms the verifying proof is given by simple applications of  $\forall^-$  to themselves. The realizing terms for  $\mathbf{AxSYM}_\tau$  and  $\mathbf{AxTRZ}_\tau$  with higher-order  $\tau$  are just simple projections of shape  $\lambda x_1, \dots, x_n. x_i$  (with  $i \in \overline{1, n}$ ) and the verifying proofs are very simple.

$\mathbf{AxBIA} : A(\mathbf{tt}) \wedge A(\mathbf{ff}) \rightarrow \forall p^\circ A(p)$  The D-interpretation of  $\mathbf{AxBIA}$  is

$$\begin{array}{c} \exists Y_1, Y_2, X \forall p^\dagger, x_1, x_2, y \\ \left( \begin{array}{c} A_D(x_1; Y_1(p^\dagger, x_1, x_2, y); \mathbf{tt}) \wedge A_D(x_2; Y_2(p^\dagger, x_1, x_2, y); \mathbf{ff}) \\ \longrightarrow \\ A_D(X(p^\dagger, x_1, x_2); y; p^\dagger) \end{array} \right) \end{array}$$

for which a realizing tuple is  $T_{Y_1}, T_{Y_2}, T_X$  where

$$\begin{array}{lcl} T_{Y_1} & : \equiv & \lambda p^\dagger, x_1, x_2, y. y \\ T_{Y_2} & : \equiv & \lambda p^\dagger, x_1, x_2, y. y \\ T_Z & : \equiv & \mathbf{If}_\tau \end{array}$$

and the verifying proof uses an instance of  $\mathbf{AxBIA}$  for a  $\mathbf{qfr}$  formula and  $\mathbf{AxIf}_\tau$ .

$\frac{A_0 \dots / x =_\tau y}{B(x) \rightarrow B(y)} \text{COMPAT}$  We have that  $B(z^\tau, a)^D \equiv \exists u \forall v B_D(u; v; z^\tau, a)$ , hence

$$(B(x) \rightarrow B(y))^D \equiv \exists U, V \forall u, v B_D(u; V(u, v); x, a) \rightarrow B_D(U(u); v; y, a).$$

Since  $A_0^D = (A_0)_D = A_0$  we are given that  $A_0 \vdash x(z_1, \dots, z_n) =_i y(z_1, \dots, z_n)$  to which we can (since none of  $z_1, \dots, z_n$  appears in  $A_0$ ) apply  $n$  times  $\forall^+$  to (re)obtain  $A_0 \vdash x =_\tau y$ . To this we apply a  $\text{COMPAT}$  for  $B'(z) \equiv B_D(u; v; z, a)$  and thus obtain  $A_0 \vdash B_D(u; v; x, a) \rightarrow B_D(u; v; y, a)$ . Let  $T_V := \lambda u, v. v$  and  $T_U := \lambda u. u$ . It immediately follows (also using some other  $\text{COMPAT}$ ) that

$$A_0 \vdash B_D(u; T_V(u, v); x, a) \rightarrow B_D(T_U(u); v; y, a)$$

which rewrites as  $(A_0)_D \vdash (B(x) \rightarrow B(y))_D(T_U, T_V; u, v; x, y, a)$ .

$\frac{\begin{array}{c} \emptyset \quad \emptyset \\ \vdots \quad \vdots \\ A(0) \quad \forall z (A(z) \rightarrow A(\mathbf{Sz})) \end{array}}{\forall z A(z)} \text{IR}_0$  We are given that (by abuse of notation we use the same  $z$  in the verifying proof)

$$\emptyset \vdash A_D(T'; y'; 0) \tag{7}$$

$$\emptyset \vdash A_D(y''; T_1''(z, y'', y); z) \rightarrow A_D(T_2''(z, y''); y; \mathbf{Sz}) \tag{8}$$

We define by simultaneous primitive recursion in higher types the terms  $T$  such that  $T(0) = T'$  and  $T(\mathbf{Sz}) = T_2''(z, T(z))$ . By  $\forall^+$ , (7) becomes

$$\emptyset \vdash \forall y A_D(T(0); y; 0) \tag{9}$$

We substitute  $\{y'' \leftarrow T(z)\}$  in (8) and obtain

$$\emptyset \vdash A_D(T(z); T_1''(z, T(z), y); z) \rightarrow A_D(T(\mathbf{Sz}); y; \mathbf{Sz}) \tag{10}$$



hence the D-interpretation of AC is

$$\begin{array}{c} \exists X, V, Y^\dagger, U \forall y^\dagger, u^\dagger, x^\dagger, v^\dagger \\ \left( \begin{array}{c} B_{\mathbb{D}}(u^\dagger(X(y^\dagger, u^\dagger, x^\dagger, v^\dagger))); V(y^\dagger, u^\dagger, x^\dagger, v^\dagger); X(y^\dagger, u^\dagger, x^\dagger, v^\dagger), y^\dagger(X(y^\dagger, u^\dagger, x^\dagger, v^\dagger)) \\ \longrightarrow \\ B_{\mathbb{D}}(U(y^\dagger, u^\dagger, x^\dagger); v^\dagger; x, Y^\dagger(y^\dagger, u^\dagger, x^\dagger)) \end{array} \right) \end{array}$$

for which a realizing tuple is

$$\begin{aligned} T_X &::= \lambda y^\dagger, u^\dagger, x^\dagger, v^\dagger. x^\dagger \\ T_V &::= \lambda y^\dagger, u^\dagger, x^\dagger, v^\dagger. v^\dagger \\ T_{Y^\dagger} &::= \lambda y^\dagger, u^\dagger. y^\dagger \\ T_U &::= \lambda y^\dagger, u^\dagger. u^\dagger \end{aligned}$$

□

## References

1. M.-D. Hernest. *Light Functional Interpretation* - an optimization of Gödel's technique towards the extraction of (more) efficient programs from (classical) proofs. Computer Science Logic 2005 (CSL'05). Springer Verlag, 2005.