Errata for "Client-Server Sessions in Linear Logic"

Zesen Qian, G.A. Kavvos and Lars Birkedal

January 15, 2025

Restricting M-True and QueW

Our work is inspired by both πLL (Montesi and Peressotti, 2021) and HCP (Kokke et al., 2019). Typing rules in πLL are more restricted which allows separation. Our system follows πLL and proved separation (Lemma 3.3 in appendix), and further uses separation to prove progress (Theorem 3.5 in appendix). However, due to an oversight, our typing rules M-True and QueW are in the style of HCP and not restricted. This breaks separation. Our separation proof omitted those two rules for triviality.

To restore separation, we should use the restricted M-True as in πLL, and the similarly restricted QueW:

$$\frac{P \vdash \emptyset}{x[].\, P \vdash x : \mathbf{1}}\ \textsc{M-True} \qquad \frac{P \vdash \emptyset}{¿x[].\, P \vdash x : ¿A}\ \textsc{QueW}$$

The restricted rules require P to be of type $\emptyset$ (the empty hypersequent), effectively making stop the only valid process for such P. Some of our examples contains $x[].\, P$ where P is not stop; they should be replaced by $(x[].\,\mathsf{stop}) \mid P$ of the same type. Similarly, $¿x[].\, P$ should be replaced by $(¿x[].\,\mathsf{stop}) \mid P$ of the same type. The reduction steps in our examples remain unchanged, mostly due to the structural equivalence $\mathsf{stop} \mid P \equiv P$.

$$\frac{\dfrac{\dfrac{}{\mathsf{stop} \vdash \emptyset}\ \textsc{HMix0}}{x[].\,\mathsf{stop} \vdash x : \mathbf{1}}\ \textsc{M-True} \quad P \vdash \mathcal{G}}{(x[].\,\mathsf{stop}) \mid P \vdash \mathcal{G} \mid x : \mathbf{1}}\ \textsc{HMix2} \qquad \frac{\dfrac{\dfrac{}{\mathsf{stop} \vdash \emptyset}\ \textsc{HMix0}}{¿x[].\,\mathsf{stop} \vdash x : ¿A}\ \textsc{QueW} \quad P \vdash \mathcal{G}}{(¿x[].\,\mathsf{stop}) \mid P \vdash \mathcal{G} \mid x : ¿A}\ \textsc{HMix2}$$

With the restricted rules, we are able to restore separation:

Lemma (Separation)**.** *If $T \vdash \Gamma_0 \mid \cdots \mid \Gamma_{n-1}$ where $n \geq 1$, then there exist $T_i \vdash \Gamma_i$ for $0 \leq i < n$ such that $T \equiv T_0 \mid \cdots \mid T_{n-1}$.*

Proof. Prove by induction on $T \vdash \Gamma_0 \mid \cdots \mid \Gamma_{n-1}$.

Case(HMix2, Cut, Tensor). The original proof stands.

Case(Ax, M-True, With, OfCourse, QueW). Trivial because $n = 1$.

Case(Par, PlusL, PlusR, M-False, WhyNotW, WhyNotD, WhyNotC). Trivial because the rules apply to a single sequent in a single hyperenvironment. Take Par as an example. T is $y(x).\, P \vdash \mathcal{G} \mid \Gamma, y : A \parr B$, derived from $P \vdash \mathcal{G} \mid \Gamma, x : A, y : B$. By I.H. on P we have $P_i \vdash \mathcal{G}_i$, and $P' \vdash \Gamma, x : A, y : B$. We take $P_i \vdash \mathcal{G}_i$ and $y(x).\, P' \vdash \Gamma, y : A \parr B$.

CASE(QUEA). Similar to TENSOR.

CASE(CLARO). Then T is $¡y\{z, z', y'. Q\}(i, f). P \vdash \mathcal{G} \mid \Gamma, \Delta, y : ¡A$, derived from

$$P \vdash \mathcal{G} \mid \Gamma, i : B \mid \Delta, f : B^{\perp}$$
$$Q \vdash z : B^{\perp}, z' : B, y' : A$$

By I.H. on P we have $P_i \vdash \mathcal{G}_i$ and $P' \vdash \Gamma, i : B$ and $P'' \vdash \Delta, f : B^{\perp}$. We take $P_i \vdash \mathcal{G}_i$, and also use P' and P'' to construct $¡y\{z, z', y'. Q\}(i, f). (P' \mid P'') \vdash \Gamma, \Delta, y : ¡A$.

□

We are grateful to Daniel Frumin and Sarah Baksteen for pointing out this oversight.

References

Wen Kokke, Fabrizio Montesi, and Marco Peressotti. 2019. Better late than never: a fully-abstract semantics for classical processes. *Proceedings of the ACM on Programming Languages* 3, POPL (2019), 1–29.

Fabrizio Montesi and Marco Peressotti. 2021. Linear Logic, the π-calculus, and their Metatheory: A Recipe for Proofs as Processes. *CoRR* abs/2106.11818 (2021). arXiv:2106.11818 `https://arxiv.org/abs/2106.11818`