Classical Linear Logic of Implications

Masahito Hasegawa

Research Institute for Mathematical Sciences, Kyoto University
hassei@kurims.kyoto-u.ac.jp

Abstract. We give a simple term calculus for the multiplicative exponential fragment of Classical Linear Logic, by extending Barber and Plotkin's system for the intuitionistic case. The calculus has the non-linear and linear implications as the basic constructs, and this design choice allows a technically managable axiomatization without commuting conversions. Despite this simplicity, the calculus is shown to be sound and complete for category-theoretic models given by $*$-autonomous categories with linear exponential comonads.

1 Introduction

We propose a linear lambda calculus called *Dual Classical Linear Logic (DCLL)* for the multiplicative exponential fragment of Classical Linear Logic [10] (often called MELL in the literature). It can be regarded as an extension of the *Dual Intuitionistic Linear Logic (DILL)* of Barber and Plotkin [1, 2].

The main feature of DCLL is its *simplicity*: just three logical connectives (intuitionistic implication $\to$, linear implication $\multimap$ and the bottom type $\bot$) and six axioms for the equational theory on terms (proofs) which are just the familiar $\beta\eta$ axioms of the lambda calculus (each for $\to$ and $\multimap$) plus two axioms saying that the type $(\sigma \multimap \bot) \multimap \bot$ is canonically isomorphic to σ. In particular we can avoid axioms for *commuting conversions*, which have always been troublesome on term calculi for Linear Logic. Other logical connectives and their proof expressions of MELL are easily derived in DCLL; for instance the exponential ! is given by $!\sigma \equiv (\sigma \to \bot) \multimap \bot$. All the desired equalities between terms, including the commuting conversions, are provable from the simple axioms of DCLL.

Thus DCLL can be used as a compact linear syntax for reasoning about MELL, to compliment the drawbacks of conventional proof nets-based presentations which are often tiresome to formulate and deal with. For instance, it is much easier to describe and analyze the translations between type systems if we use term calculi like DCLL instead of graph-based systems. Also techniques of logical relations (e.g. [11, 23]) seem to work more smoothly on term-based systems. As future work, we plan to study the compilations of call-by-value programming languages into linearly typed intermediate languages [6, 13] using DCLL as a target calculus. In fact, our choice of the logical connectives has been motivated by this research direction – see the discussion in Sec. 6.

Despite its simplicity, it is shown that DCLL is sound and complete for categorical models of MELL given by $*$-autonomous categories with symmetric

J. Bradfield (Ed.): CSL 2002, LNCS 2471, pp. 458–472, 2002.

monoidal comonads satisfying some coherence conditions (to be called linear exponential comonads). It turns out that our simple axioms are sufficient for giving such a categorical structure on the term model. Although this may not be of big surprise, there seem not many systems for Linear Logic supported by this sort of semantic completeness at the level of proofs, and we think that this completeness result gives a justification on our design of DCLL.

This paper is organized as follows. We introduce the system DCLL in Sec. 2, with some discussions on its alternative formulations. Sec. 3 gives a comparison of DCLL with its precursor DILL. Sec. 4 then states the completeness result of DCLL with respect to the categorical models of MELL. In Sec. 5 the extension with additives (hence a full propositional Classical Linear Logic) is discussed. We conclude the paper by giving some discussions on future work at Sec. 6. Appendix A gives a summary of DILL, while Appendix B is devoted to a variant of DCLL based on the $\lambda\mu$-calculus, called μDCLL. Appendix C describes an alternative axiomatization of DCLL (and MLL) with no base type.

Acknowledgements I am grateful to Hayo Thieletcke for drawing my attention to the $\{\rightarrow, \multimap\}$-fragment. I thank Martin Hofmann, Yoshihiko Kakutani and Valeria de Paiva for discussions and comments related to this work.

2 DCLL

2.1 The System DCLL

In this "dual-context"[1] formulation of the linear lambda calculus, a typing judgement takes the form $\Gamma\ ;\ \Delta \vdash M : \tau$ in which Γ represents an intuitionistic (or additive) context whereas Δ is a linear (multiplicative) context. While the variables in Γ can be used in the term M as many times as we like, those in Δ must be used exactly once. A typing judgement $x_1 : \sigma_1, \ldots, x_m : \sigma_m\ ;\ y_1 : \tau_1, \ldots, y_n : \tau_n \vdash M : \sigma$ can be considered as the proof of the sequent $!\sigma_1, \ldots, !\sigma_m, \tau_1, \ldots, \tau_n \vdash \sigma$, or the proposition $!\sigma_1 \otimes \ldots \otimes !\sigma_m \otimes \tau_1 \otimes \ldots \otimes \tau_n \multimap \sigma$.

As mentioned in the introduction, the system features both intuitionistic (non-linear) arrow type $\rightarrow$ and linear arrow type $\multimap$. We use $\boldsymbol{\lambda} x^\sigma.M$ and $M \mathbin{@} N$ for the non-linear lambda abstraction and application respectively, while $\lambda x^\sigma.M$ and $M\,N$ for the linear ones. For expressing the duality of Classical Linear Logic, there also is a special combinator C_σ which serves as the isomorphism from $(\sigma \multimap \bot) \multimap \bot$ to σ (which, however, can be eliminated when we have no base type – see the discussion at the end of this section).

Types and Terms

$$\begin{aligned} \sigma &::= b \mid \sigma \rightarrow \sigma \mid \sigma \multimap \sigma \mid \bot \\ M &::= x \mid \boldsymbol{\lambda} x^\sigma.M \mid M \mathbin{@} M \mid \lambda x^\sigma.M \mid M\,M \mid \mathsf{C}_\sigma \end{aligned}$$

where b ranges over a set of base types. We may omit the type subscripts for ease of presentation.

[1] As noted in [2] the word "dual" of DILL (and DCLL) comes from this dual-context typing, and has nothing to do with the duality of Classical Linear Logic.

Typing

$$\frac{}{\Gamma_1, x:\sigma, \Gamma_2 \ ; \ \emptyset \vdash x:\sigma}\ (\text{Int-Ax}) \qquad \frac{}{\Gamma \ ; \ x:\sigma \vdash x:\sigma}\ (\text{Lin-Ax})$$

$$\frac{\Gamma, x:\sigma_1 \ ; \ \Delta \vdash M:\sigma_2}{\Gamma \ ; \ \Delta \vdash \boldsymbol{\lambda} x^{\sigma_1}.M : \sigma_1 \to \sigma_2}\ (\to \text{I}) \quad \frac{\Gamma \ ; \ \Delta \vdash M : \sigma_1 \to \sigma_2 \quad \Gamma \ ; \ \emptyset \vdash N : \sigma_1}{\Gamma \ ; \ \Delta \vdash M \mathbin{@} N : \sigma_2}\ (\to \text{E})$$

$$\frac{\Gamma \ ; \ \Delta, x:\sigma_1 \vdash M:\sigma_2}{\Gamma \ ; \ \Delta \vdash \lambda x^{\sigma_1}.M : \sigma_1 \multimap \sigma_2}\ (\multimap \text{I}) \quad \frac{\Gamma \ ; \ \Delta_1 \vdash M : \sigma_1 \multimap \sigma_2 \quad \Gamma \ ; \ \Delta_2 \vdash N : \sigma_1}{\Gamma \ ; \ \Delta_1 \sharp \Delta_2 \vdash M\, N : \sigma_2}\ (\multimap \text{E})$$

$$\frac{}{\Gamma \ ; \ \emptyset \vdash \mathsf{C}_\sigma : ((\sigma \multimap \bot) \multimap \bot) \multimap \sigma}\ (\mathsf{C})$$

where $\Delta_1 \sharp \Delta_2$ is a merge of Δ_1 and Δ_2 [2]. Thus, $\Delta_1 \sharp \Delta_2$ represents one of possible merges of Δ_1 and Δ_2 as finite lists. We assume that, when we introduce $\Delta_1 \sharp \Delta_2$, there is no variable occurring both in Δ_1 and in Δ_2. We write $\emptyset$ for the empty context. We note that any typing judgement has a unique derivation (hence a typing judgement can be identified with its derivation).

Axioms

$$\begin{array}{lllll}
(\beta_\to) & (\boldsymbol{\lambda} x.M) \mathbin{@} N & = & M[N/x] & \\
(\eta_\to) & \boldsymbol{\lambda} x.M \mathbin{@} x & = & M & (x \notin FV(M)) \\
(\beta_\multimap) & (\lambda x.M)\, N & = & M[N/x] & \\
(\eta_\multimap) & \lambda x.M\, x & = & M & \\
(\mathsf{C}_1) & L\,(\mathsf{C}_\sigma\, M) & = & M\, L & (L : \sigma \multimap \bot) \\
(\mathsf{C}_2) & \mathsf{C}_\sigma\,(\lambda k^{\sigma \multimap \bot}.k\, M) & = & M &
\end{array}$$

where $M[N/x]$ denotes the capture-free substitution. Note that there is no side condition $x \notin FV(M)$ for the axiom $(\eta_\multimap)$ (and similarly for (C_2)), as linearity prevents x from occuring in M. The equality judgement $\Gamma \ ; \ \Delta \vdash M = N : \sigma$ for $\Gamma \ ; \ \Delta \vdash M : \sigma$ and $\Gamma \ ; \ \Delta \vdash N : \sigma$ is defined as usual.
We note that the axiom (C_1) is equivalent to $\lambda k^{\sigma \multimap \bot}.k\,(\mathsf{C}_\sigma\, M) = M$; thus the last two axioms say that C_σ is the inverse of $\lambda x^\sigma.\lambda k^{\sigma \multimap \bot}.k\, x : \sigma \multimap (\sigma \multimap \bot) \multimap \bot$.

Lemma 1. *The "naturality" of* C *is provable in DCLL:*

$$L^{\sigma \multimap \tau}\,(\mathsf{C}_\sigma\, M^{(\sigma \multimap \bot) \multimap \bot}) \ = \ \mathsf{C}_\tau\,(\lambda k^{\tau \multimap \bot}.M\,(\lambda x^\sigma.k\,(L\, x))) \ : \ \tau$$

Proof:

$$L\,(\mathsf{C}\, M) \overset{\mathsf{C}_2}{=} \mathsf{C}\,(\lambda k.k\,(L\,(\mathsf{C}\, M))) \overset{\beta_\multimap}{=} \mathsf{C}\,(\lambda k.(\lambda x.k\,(L\, x))\,(\mathsf{C}\, M)) \overset{\mathsf{C}_1}{=} \mathsf{C}\,(\lambda k.M\,(\lambda x.k\,(L\, x))).$$

□

2.2 Alternative Formulations of DCLL

Formulation Based on the $\lambda\mu$-calculus. Instead of the combinator C for the double-negation elimination, we could use the syntax of the $\lambda\mu$-calculus [21] for expressing the duality, as done in [17] for the multiplicative fragment (MLL).

We do not take this approach here as our presentation using C seems sufficiently simple, while the $\lambda\mu$-calculus style formulation requires to introduce yet another typing context. For completeness, in Appendix B we present such a system (μDCLL) which is routinely seen to be equivalent to DCLL. A potential benefit of the $\lambda\mu$-calculus approach is that it may give a confluent and normalizing reduction system (which cannot be expected for DCLL); also it allows natural treatment of the connective $\parr$ (by introducing the binary μ-bindings). See also [8] for relevant results.

Axiomatization without C. In DCLL, the following equations are provable:

Lemma 2.

1. $\mathsf{C}_\perp = \lambda m^{(\perp\multimap\perp)\multimap\perp}.m\,(\lambda x^\perp.x)$
2. $\mathsf{C}_{\sigma\to\tau} = \lambda m^{((\sigma\to\tau)\multimap\perp)\multimap\perp}.\boldsymbol{\lambda} x^\sigma.\mathsf{C}_\tau\,(\lambda k^{\tau\multimap\perp}.m\,(\lambda f^{\sigma\to\tau}.k\,(f\mathbin{@}x)))$
3. $\mathsf{C}_{\sigma\multimap\tau} = \lambda m^{((\sigma\multimap\tau)\multimap\perp)\multimap\perp}.\lambda x^\sigma.\mathsf{C}_\tau\,(\lambda k^{\tau\multimap\perp}.m\,(\lambda f^{\sigma\multimap\tau}.k\,(f\,x)))$

Proof:

1. $\mathsf{C}_\perp\,m = (\lambda x^\perp.x)\,(\mathsf{C}_\perp\,m) = m\,(\lambda x^\perp.x)$.
2. $\mathsf{C}_{\sigma\to\tau}\,m\mathbin{@}x = \mathsf{C}_\tau\,(\lambda k.k\,(\mathsf{C}_{\sigma\to\tau}\,m\mathbin{@}x)) = \mathsf{C}_\tau\,(\lambda k.(\lambda f.k\,(f\mathbin{@}x))\,(\mathsf{C}_{\sigma\to\tau}\,m)) = \mathsf{C}_\tau\,(\lambda k.m\,(\lambda f.k\,(f\mathbin{@}x)))$.
3. $\mathsf{C}_{\sigma\multimap\tau}\,m\,x = \mathsf{C}_\tau\,(\lambda k.k\,(\mathsf{C}_{\sigma\multimap\tau}\,m\,x)) = \mathsf{C}_\tau\,(\lambda k.(\lambda f.k\,(f\,x))\,(\mathsf{C}_{\sigma\multimap\tau}\,m)) = \mathsf{C}_\tau\,(\lambda k.m\,(\lambda f.k\,(f\,x)))$. □

This implies that, if we do not have base types, all DCLL terms can be expressed as just (non-linear and linear) lambda terms, without using the combinator C. By induction we can show

Proposition 1. *For $\sigma = \sigma_1 \Rightarrow_1 \ldots \sigma_n \Rightarrow_n \perp$ (where $\Rightarrow_i$ is either $\to$ or $\multimap$)*

$$\mathsf{C}_\sigma\,M \star_1 N_1 \ldots \star_n N_n = M\,(\lambda f^\sigma.f \star_1 N_1 \ldots \star_n N_n) : \perp$$

is provable in DCLL, where $M : (\sigma \multimap \perp) \multimap \perp$, $N_i : \sigma_i$, and $\star_i$ is a non-linear application if $\Rightarrow_i$ is $\to$, or a linear application if $\Rightarrow_i$ is $\multimap$. □

If we *define* C's as lambda terms by the equations of Lem. 2 or Prop. 1, then the axiom (C_2) follow just from the $\beta\eta$-axioms for $\to$ and $\multimap$. Therefore it is possible to axiomatize DCLL with no base type as a quotient of the $\{\to,\multimap\}$-calculus on the single base type $\perp$ obtained by adding the axiom (C_1) for these defined C's. In fact all of them are derivable from the following single instance and the $\beta\eta$-axioms for $\to$ and $\multimap$:

$$L\,(\lambda x^\sigma.M\,(\lambda f^{\sigma\multimap\perp}.f\,x)) = M\,L$$

where $L : (\sigma \multimap \perp) \multimap \perp$ and $M : ((\sigma \multimap \perp) \multimap \perp) \multimap \perp$.[2] So it suffices to have the standard $\beta\eta$-axioms and this equation; Appendix C describes the resulting system (as well as its multiplicative fragment MLL).

[2] This in fact amounts to the infamous (in)equality known as "triple unit problem" (which asks if two canonical endomorphisms on $((A \multimap I) \multimap I) \multimap I$ are the same in a symmetric monoidal closed category, see [19, 16]) if one replaces $\perp$ by I.

3 DILL in DCLL

The primitive constructs of DILL (summarized in Appendix A) can be defined in DCLL as follows:

$$
\begin{array}{ll}
I & \equiv \bot \multimap \bot \\
\sigma_1 \otimes \sigma_2 & \equiv (\sigma_1 \multimap \sigma_2 \multimap \bot) \multimap \bot \\
!\,\sigma & \equiv (\sigma \to \bot) \multimap \bot \\
 & \\
* & \equiv \lambda x^{\bot}.x \\
\mathsf{let}\ *\ \mathsf{be}\ M^{I}\ \mathsf{in}\ N^{\tau} & \equiv \mathsf{C}_{\tau}\,(\lambda k^{\tau \multimap \bot}.M\,(k\,N)) \\
M^{\sigma_1} \otimes N^{\sigma_2} & \equiv \lambda k^{\sigma_1 \multimap \sigma_2 \multimap \bot}.k\,M\,N \\
\mathsf{let}\ x^{\sigma_1} \otimes y^{\sigma_2}\ \mathsf{be}\ M^{\sigma_1 \otimes \sigma_2}\ \mathsf{in}\ N^{\tau} & \equiv \mathsf{C}_{\tau}\,(\lambda k^{\tau \multimap \bot}.M\,(\lambda x^{\sigma_1}.\lambda y^{\sigma_2}.k\,N)) \\
!M^{\sigma} & \equiv \lambda h^{\sigma \to \bot}.h\,@\,M \\
\mathsf{let}\ !x^{\sigma}\ \mathsf{be}\ M^{!\sigma}\ \mathsf{in}\ N^{\tau} & \equiv \mathsf{C}_{\tau}\,(\lambda k^{\tau \multimap \bot}.M\,(\boldsymbol{\lambda} x^{\sigma}.k\,N))
\end{array}
$$

(It is also possible to introduce connectives ? and ⅋ by $?\sigma \equiv (\sigma \multimap \bot) \to \bot$ and $\sigma_1 \parr \sigma_2 \equiv (\sigma_1 \multimap \bot) \multimap (\sigma_2 \multimap \bot) \multimap \bot$, though giving the term expressions associated to these connectives seems less obvious.)

Below we shall see that this encoding is sound, for both the typing and equational theory.

Lemma 3. *Derivation rules of typing judgements in DILL are admissible in DCLL.*

Proof: We shall spell out the cases of introduction and elimination rules for !

$$
\frac{\Gamma\ ;\ \emptyset \vdash M : \sigma}{\Gamma\ ;\ \emptyset \vdash\, !M : !\sigma}\ (!\,\mathrm{I}) \qquad \frac{\Gamma\ ;\ \Delta_1 \vdash M : !\sigma \quad \Gamma, x : \sigma\ ;\ \Delta_2 \vdash N : \tau}{\Gamma\ ;\ \Delta_1 \sharp \Delta_2 \vdash \mathsf{let}\ !x^{\sigma}\ \mathsf{be}\ M\ \mathsf{in}\ N : \tau}\ (!\,\mathrm{E})
$$

which are derivable in DCLL as follows.

$$
\dfrac{\dfrac{}{\Gamma\ ;\ h : \sigma \to \bot \vdash h : \sigma \to \bot}\ \text{Lin-Ax} \qquad \Gamma\ ;\ \emptyset \vdash M : \sigma}{\dfrac{\Gamma\ ;\ h : \sigma \to \bot \vdash h\,@\,M : \bot}{\Gamma\ ;\ \emptyset \vdash !M \equiv \lambda h^{\sigma \to \bot}.h\,@\,M : (\sigma \to \bot) \multimap \bot \equiv\, !\sigma}\ \multimap \mathrm{I}}\ \to \mathrm{E}
$$

$$
\dfrac{\Gamma\ ;\ \emptyset \vdash \mathsf{C}_{\tau} : ((\tau \multimap \bot) \multimap \bot) \multimap \tau \qquad \dfrac{\dfrac{\dfrac{}{\Gamma, x : \sigma\ ;\ k : \tau \multimap \bot \vdash k : \tau \multimap \bot}\ \text{Lin-Ax} \qquad \Gamma, x : \sigma\ ;\ \Delta_2 \vdash N : \tau}{\Gamma, x : \sigma\ ;\ \Delta_2, k : \tau \multimap \bot \vdash k\,N : \bot}\ \multimap \mathrm{E}}{\Gamma\ ;\ \Delta_2, k : \tau \multimap \bot \vdash \boldsymbol{\lambda} x^{\sigma}.k\,N : \sigma \to \bot}\ \to \mathrm{I}}{\dfrac{\Gamma\ ;\ \Delta_1 \vdash M : !\sigma \equiv (\sigma \to \bot) \multimap \bot \qquad \Gamma\ ;\ \Delta_1 \sharp \Delta_2, k : \tau \multimap \bot \vdash M\,(\boldsymbol{\lambda} x^{\sigma}.k\,N) : \bot}{\Gamma\ ;\ \Delta_1 \sharp \Delta_2 \vdash \lambda k^{\tau \multimap \bot}.M\,(\boldsymbol{\lambda} x^{\sigma}.k\,N) : (\tau \multimap \bot) \multimap \bot}\ \multimap \mathrm{I}}\ \multimap \mathrm{E}
$$

$$
\Gamma\ ;\ \Delta_1 \sharp \Delta_2 \vdash \mathsf{let}\ !x^{\sigma}\ \mathsf{be}\ M\ \mathsf{in}\ N \equiv \mathsf{C}_{\tau}\,(\lambda k^{\tau \multimap \bot}.M\,(\boldsymbol{\lambda} x^{\sigma}.k\,N)) : \tau
$$

The cases of I and $\otimes$ are derived similarly. □

Theorem 1. *Equality axioms in DILL are admissible in DCLL.*

Proof: The β-axioms are easy:

$$\begin{aligned}
\mathsf{let}\ *\ \mathsf{be}\ *\ \mathsf{in}\ N &\equiv \mathsf{C}\,(\lambda k.(\lambda x.x)\,(k\,N))\\
&= \mathsf{C}\,(\lambda k.k\,N)\\
&= N\\
\mathsf{let}\ x\otimes y\ \mathsf{be}\ M_1\otimes M_2\ \mathsf{in}\ N &\equiv \mathsf{C}\,(\lambda k.(\lambda h.h\,M_1\,M_2)\,(\lambda x.\lambda y.k\,N))\\
&= \mathsf{C}\,(\lambda k.(\lambda x.\lambda y.k\,N)\,M_1\,M_2)\\
&= \mathsf{C}\,(\lambda k.k\,N[M_1/x,M_2/y])\\
&= N[M_1/x,M_2/y]\\
\mathsf{let}\ !x\ \mathsf{be}\ !M\ \mathsf{in}\ N &\equiv \mathsf{C}\,(\lambda k.(\lambda h.h\,@\,M)\,(\boldsymbol{\lambda} x.k\,N))\\
&= \mathsf{C}\,(\lambda k.(\boldsymbol{\lambda} x.k\,N)\,@\,M)\\
&= \mathsf{C}\,(\lambda k.k\,N[M/x])\\
&= N[M/x]
\end{aligned}$$

The η-axioms are slightly more subtle.

$$\begin{aligned}
\mathsf{let}\ *\ \mathsf{be}\ M\ \mathsf{in}\ * &\equiv \mathsf{C}\,(\lambda k.M\,(k\,(\lambda x.x)))\\
&= \lambda y.(\lambda k.M\,(k\,(\lambda x.x)))\,(\lambda f.f\,y) && \text{(Prop.1)}\\
&= \lambda y.M\,((\lambda f.f\,y)\,(\lambda x.x))\\
&= \lambda y.M\,((\lambda x.x)\,y)\\
&= \lambda y.M\,y\\
&= M\\
\mathsf{let}\ x\otimes y\ \mathsf{be}\ M\ \mathsf{in}\ x\otimes y &\equiv \mathsf{C}\,(\lambda k.M\,(\lambda xy.k\,(\lambda n.n\,x\,y)))\\
&= \lambda u.(\lambda k.M\,(\lambda xy.k\,(\lambda n.n\,x\,y)))\,(\lambda f.f\,u) && \text{(Prop.1)}\\
&= \lambda u.M\,(\lambda xy.(\lambda f.f\,u)\,(\lambda n.n\,x\,y))\\
&= \lambda u.M\,(\lambda xy.u\,x\,y)\\
&= \lambda u.M\,u\\
&= M\\
\mathsf{let}\ !x\ \mathsf{be}\ M\ \mathsf{in}\ !x &\equiv \mathsf{C}\,(\lambda k.M\,(\boldsymbol{\lambda} x.k\,(\lambda h.h\,@\,x)))\\
&= \lambda u.(\lambda k.M\,(\boldsymbol{\lambda} x.k\,(\lambda h.h\,@\,x)))\,(\lambda f.f\,u) && \text{(Prop.1)}\\
&= \lambda u.M\,(\boldsymbol{\lambda} x.(\lambda f.f\,u)\,(\lambda h.h\,@\,x))\\
&= \lambda u.M\,(\boldsymbol{\lambda} x.(\lambda h.h\,@\,x)\,u)\\
&= \lambda u.M\,(\boldsymbol{\lambda} x.u\,@\,x)\\
&= \lambda u.M\,u\\
&= M
\end{aligned}$$

There remain (30 instances of) axioms for commuting conversions which, for instance, can be shown as

$$\begin{aligned}
L\,(\mathsf{let}\ !x\ \mathsf{be}\ M\ \mathsf{in}\ N) &\equiv L\,(\mathsf{C}\,(\lambda k.M\,(\boldsymbol{\lambda} x.k\,N)))\\
&= \mathsf{C}\,(\lambda h.(\lambda k.M\,(\boldsymbol{\lambda} x.k\,N))\,(\lambda y.h\,(L\,y))) && \text{(Lem. 1)}\\
&= \mathsf{C}\,(\lambda h.M\,(\boldsymbol{\lambda} x.(\lambda y.h\,(L\,y))\,N))\\
&= \mathsf{C}\,(\lambda h.M\,(\boldsymbol{\lambda} x.h\,(L\,N)))\\
&\equiv \mathsf{let}\ !x\ \mathsf{be}\ M\ \mathsf{in}\ L\,N\\
\mathsf{let}\ !x\ \mathsf{be}\ M\ \mathsf{in}\ \lambda y.N &\equiv \mathsf{C}\,(\lambda k.M\,(\boldsymbol{\lambda} x.k\,(\lambda y.N)))\\
&= \lambda y.\mathsf{C}\,(\lambda h.(\lambda k.M\,(\boldsymbol{\lambda} x.k\,(\lambda y.N)))\,(\lambda f.h\,(f\,y))) && \text{(Lem. 2)}\\
&= \lambda y.\mathsf{C}\,(\lambda h.M\,(\boldsymbol{\lambda} x.(\lambda f.h\,(f\,y))\,(\lambda y.N)))\\
&= \lambda y.\mathsf{C}\,(\lambda h.M\,(\boldsymbol{\lambda} x.h\,N))\\
&\equiv \lambda y.(\mathsf{let}\ !x\ \mathsf{be}\ M\ \mathsf{in}\ N)
\end{aligned}$$

We leave the other cases as exercises for the interested readers. □

4 Completeness for Categorical Models

An important implication of Thm. 1, together with the result in [2] (completeness via the term model construction), is that the term model of DCLL forms a model of DILL, i.e., a symmetric monoidal closed category equipped with a symmetric monoidal comonad satisfying certain coherence conditions (see e.g. [7]) which we shall call a "linear exponential comonad" (following [15]).[3]

Definition 1 (linear exponential comonad). *A symmetric monoidal comonad* $! = (!, \varepsilon, \delta, m_{A,B}, m_I)$ *on a symmetric monoidal category* $\mathcal{C}$ *is called a* linear exponential comonad *when the category of its coalgebras is a category of commutative comonoids – that is:*

- *for each free* $!$*-coalgebra* $(!A, \delta_A)$ *there are specified monoidal natural transformations* $e_A : !A \to I$ *and* $d_A : !A \to !A \otimes !A$ *which form a commutative comonoid* $(!A, e_A, d_A)$ *in* $\mathcal{C}$ *and also are coalgebra morphisms from* $(!A, \delta_A)$ *to* (I, m_I) *and* $(!A \otimes !A, m_{!A,!A} \circ (\delta_A \otimes \delta_A))$ *respectively, and*
- *any coalgebra morphism from* $(!A, \delta_A)$ *to* $(!B, \delta_B)$ *is also a comonoid morphism from* $(!A, e_A, d_A)$ *to* $(!B, e_B, d_B)$.

Moreover, the symmetric monoidal closed category given by the term model of DCLL is a $*$*-autonomous category* [3, 4] if we take $\bot$ as the dualizing object. Recall that a $*$-autonomous category can be characterized as a symmetric monoidal closed category with an object $\bot$ such that the canonical morphism from σ to $(\sigma \multimap \bot) \multimap \bot$ is an isomorphism — in the term model of DCLL, the inverse is given by the combinator C_σ.

On the other hand, all the axioms of DCLL are sound with respect to interpretations in such categorical models, where a typing judgement

$$x_1 : \sigma_1, \ldots, x_m : \sigma_m \ ; \ y_1 : \tau_1, \ldots, y_n : \tau_n \vdash M : \sigma$$

is inductively interpreted as a morphism $[\![x_1 : \sigma_1, \ldots \ ; \ y_1 : \tau_1, \ldots \vdash M : \sigma]\!]$ from $![\![\sigma_1]\!] \otimes \ldots \otimes ![\![\sigma_m]\!] \otimes [\![\tau_1]\!] \otimes \ldots \otimes [\![\tau_n]\!]$ to $[\![\sigma]\!]$ in the $*$-autonomous category with the linear exponential comonad $!$. Thus we have:

Theorem 2 (categorical completeness). *The equational theory of DCLL is sound and complete for categorical models given by* $*$*-autonomous categories with linear exponential comonads:* $\Gamma \ ; \ \Delta \vdash M = N : \sigma$ *is provable if and only if* $[\![\Gamma \ ; \ \Delta \vdash M : \sigma]\!] = [\![\Gamma \ ; \ \Delta \vdash N : \sigma]\!]$ *holds for every such models.* □

[3] In [2] a model of DILL is described as a symmetric monoidal adjunction between a cartesian closed category and a symmetric monoidal closed category (Benton's LNL model [5]). It is known that such an "adjunction model" gives rise to a linear exponential comonad on the symmetric monoidal closed category part. Conversely, a symmetric monoidal closed category with a linear exponential comonad has at least one symmetric monoidal adjunction from a cartesian closed category so that it induces the linear exponential comonad (such an adjunction is not unique in general, though). Therefore, for our purpose (the completeness result as stated here), it does not matter which class of structures we choose as models. However we must be careful when we talk about the morphisms between models, e.g. to use the term model of DILL (or DCLL) as a classifying category of such structures.

5 Additives

It is fairly routine to enrich DCLL with additives. We add the cartesian product $\&$ and its unit $\top$, and terms

$$\frac{}{\Gamma\ ;\ \Delta \vdash \langle\rangle : \top}\ (\top\,\mathrm{I}) \qquad \frac{\Gamma\ ;\ \Delta \vdash M : \sigma \quad \Gamma\ ;\ \Delta \vdash N : \tau}{\Gamma\ ;\ \Delta \vdash \langle M, N\rangle : \sigma \,\&\, \tau}\ (\&\,\mathrm{I})$$

$$\frac{\Gamma\ ;\ \Delta \vdash M : \sigma \,\&\, \tau}{\Gamma\ ;\ \Delta \vdash \mathsf{fst}_{\sigma,\tau}\, M : \sigma}\ (\&\,\mathrm{E}_L) \qquad \frac{\Gamma\ ;\ \Delta \vdash M : \sigma \,\&\, \tau}{\Gamma\ ;\ \Delta \vdash \mathsf{snd}_{\sigma,\tau}\, M : \tau}\ (\&\,\mathrm{E}_R)$$

and the standard axioms

$$\begin{array}{lll} M & = \langle\rangle & (M : \top) \\ \mathsf{fst}\,\langle M, N\rangle & = M & \\ \mathsf{snd}\,\langle M, N\rangle & = N & \\ \langle \mathsf{fst}\, M, \mathsf{snd}\, M\rangle & = M & \end{array}$$

Again we do not need any additional axiom for commuting conversions. Furthermore, it is possible to eliminate the C combinators for additives as we can prove (using Lem. 1 for the latter case)

Lemma 4.

1. $\mathsf{C}_\top = \lambda m^{(\top \multimap \bot) \multimap \bot}.\langle\rangle$
2. $\mathsf{C}_{\sigma\,\&\,\tau} =$
   $\lambda m^{((\sigma\,\&\,\tau)\multimap\bot)\multimap\bot}.$
   $\langle \mathsf{C}_\sigma\,(\lambda k^{\sigma\multimap\bot}.m\,(\lambda z^{\sigma\,\&\,\tau}.k\,(\mathsf{fst}_{\sigma,\tau}\,z))), \mathsf{C}_\tau\,(\lambda h^{\tau\multimap\bot}.m\,(\lambda z^{\sigma\,\&\,\tau}.h\,(\mathsf{snd}_{\sigma,\tau}\,z)))\rangle$

□

In particular, if we do not have base types, it is possible to axiomatize DCLL with additives as a quotient of a typed lambda calculus (with $\to$, $\multimap$, $\top$, $\&$) on a single base type $\bot$, in the same way as described at the end of Sec. 2.

The coproduct $\oplus$ and its unit 0 are given by $\sigma_1 \oplus \sigma_2 \equiv ((\sigma_1 \multimap \bot)\,\&\,(\sigma_2 \multimap \bot)) \multimap \bot$ and $0 \equiv \top \multimap \bot$ as usual. The associated term constructs are

$$\frac{\Gamma\ ;\ \Delta \vdash M : \sigma}{\Gamma\ ;\ \Delta \vdash \mathsf{inl}_{\sigma,\tau}\, M \equiv \lambda k^{(\sigma\multimap\bot)\,\&\,(\tau\multimap\bot)}.\mathsf{fst}_{\sigma\multimap\bot,\tau\multimap\bot}\,k\,M : \sigma \oplus \tau}\ (\oplus\,\mathrm{I}_L)$$

$$\frac{\Gamma\ ;\ \Delta \vdash N : \tau}{\Gamma\ ;\ \Delta \vdash \mathsf{inr}_{\sigma,\tau}\, N \equiv \lambda k^{(\sigma\multimap\bot)\,\&\,(\tau\multimap\bot)}.\mathsf{snd}_{\sigma\multimap\bot,\tau\multimap\bot}\,k\,N : \sigma \oplus \tau}\ (\oplus\,\mathrm{I}_R)$$

$$\frac{\Gamma\ ;\ \Delta \vdash L : \sigma \oplus \tau \quad \Gamma\ ;\ \Delta, x : \sigma \vdash M : \theta \quad \Gamma\ ;\ \Delta, y : \tau \vdash N : \theta}{\begin{array}{l}\Gamma\ ;\ \Delta \vdash \mathsf{case}\ L\ \mathsf{of}\ \mathsf{inl}\,x^\sigma \mapsto M \parallel \mathsf{inr}\,y^\tau \mapsto N \equiv \\ \qquad \mathsf{C}_\theta\,(\lambda k^{\theta\multimap\bot}.L\,\langle\lambda x^\sigma.k\,M, \lambda y^\tau.k\,N\rangle) : \theta\end{array}}\ (\oplus\,\mathrm{E})$$

They do satisfy the standard axioms for coproducts as well as a number of commuting conversion axioms.

A category-theoretic model of DCLL extended with additives can be given as a $*$-autonomous category with a linear exponential comonad and finite products. The soundness and completeness results in the last section easily extend for this setting.

6 Discussions and Future Work

6.1 DCLL as a Typed Intermediate Language

The design of DCLL is heavily inspired from our experience (and still on-going project) on the study of compiling (mostly call-by-value typed) programming languages into linearly typed intermediate languages [13], which has been briefly mentioned in the introduction.

In [6] the $\{\to, \multimap\}$-fragment of DILL (with recursive types) is used as the target language of CPS transformations. In [13] we extend the idea of [6] to general monadic transformations into the $\{!, \multimap\}$-fragment of DILL, and have observed that the $\{\to, \multimap\}$-fragment is full in the $\{!, \multimap\}$-fragment[4] (hence both approaches essentially agree, as long as we talk about CPS transformations). In these studies the "linearly-used continuation monad" $((-) \to \theta) \multimap \theta$ plays the key role[5] $:\to$ for continuations, and $\multimap$ for the linearity of their passing. The choice of connectives of DCLL then comes to us naturally; $\to$ and $\multimap$ come first, and we regard the exponential ! as the special case of the linearly-used continuation monad by letting θ be $\bot$: $!\sigma \simeq (!\sigma \multimap \bot) \multimap \bot \simeq (\sigma \to \bot) \multimap \bot$.

It is also interesting to re-examine the previous work on applying Classical Linear Logic to programming languages with control features [9, 20] using DCLL; in particular Filinski's work [9] seems to share several ideas with the design of DCLL.

6.2 Is "!" better than "→"?

A possible criticism on DCLL is on its indirect treatment of the exponentials, which have been regarded as the central feature of Linear Logic by many people (though there are some exceptions, e.g. [24, 22, 18][6]). We used to consider ! as a primitive and $\to$ as a derived connective as $\sigma \to \tau \equiv !\sigma \multimap \tau$, but not in the other way (i.e. $!\sigma \equiv (\sigma \to \bot) \multimap \bot$ as we do in DCLL).

However, even in the Intuitionistic Linear Logic, the full completeness of the $\{\to, \multimap\}$-fragment in the $\{!, \multimap\}$-fragment tells us that $\to$ is no less delicate than ! at the level of proofs (terms), while $\{\to, \multimap\}$ enjoys much simpler term structures and nice properties like confluence and strong normalization. And, in Classical Linear Logic, $\{\to, \multimap, \bot\}$ is literally isomorphic to $\{!, \multimap, \bot\}$ — then it is not unnatural to use the technically simpler presentation.

[4] This result is shown by mildly extending the proof of full completeness of Girard's translation from the simply typed lambda calculus into the $\{!, \multimap\}$-fragment of DILL [12].

[5] This is *not* a monad on the term model of DILL; it is a monad on a suitable subcategory of the category of !-coalgebras.

[6] In particular Plotkin's system [22] is the *second-order* $\{\to, \multimap\}$-calculus in which other connectives of DILL including ! are definable in the similar way as we do in DCLL, for example $!\sigma$ as $\forall X.(\sigma \to X) \multimap X$. In fact it suffices to add an axiom $L^{\sigma \multimap \tau}\,(M^{\forall X.(\sigma \multimap X) \multimap X}\,\sigma\,(\lambda x^{\sigma}.x)) = M\,\tau\,L$ (which just says $\sigma \simeq \forall X.(\sigma \multimap X) \multimap X$) to give the structure of models of DILL to the term model of this calculus – the story is completely analogous to the case of DCLL.

Moreover, as mentioned above, DCLL do have natural advantages in programming language theory. From such an application-oriented view, we think that the simplicity of DCLL is undeniably attractive. See also [18] for relevant discussions on the $\{\to, \multimap, \otimes, I, \mathbin{\&}, \top\}$-fragment and its fibration-based models (which can be adopted for DCLL without problem).

6.3 Why not $\sigma^{\perp\perp} = \sigma$

Another possible source of criticism would be the way we deal with the duality, which again is the essential feature of Classical Linear Logic. Many systems for Classical Linear Logic, especially those of proof nets, identify the type $\sigma^{\perp\perp}$ $(= (\sigma \multimap \perp) \multimap \perp)$ with σ. On the other hand, in DCLL (and some other term-based systems like [8]) they are just isomorphic, and we explicitly have terms for the isomorphisms. The essential reason of this non-identification in DCLL is that we intend it to have $*$-autonomous categories with linear exponential comonads as models, rather than those with *strict* involution (i.e. $(-)^{\perp\perp}$ is the identity functor and the canonical isomorphism $\sigma \xrightarrow{\simeq} \sigma^{\perp\perp}$ is an identity arrow), as we think that having a strict involution is not a natural assumption on semantic models. (However, it might be the case that any $*$-autonomous category is equivalent to a $*$-autonomous category with strict involution, and if this is true, this design choice would be just a matter of taste.)

6.4 ILL vs. CLL

We believe that the relationship between Intuitionistic Linear Logic and Classical Linear Logic – at the level of proofs rather than that of provability – has not been sufficiently sorted out yet. Let us state the problems in terms of DCLL.

The first question concerns the converse of Thm. 1.

Conjecture 1 (conservativity, or completeness). The equational theory of DCLL is conservative over that of DILL. That is, $\Gamma \,;\, \Delta \vdash M = N : \sigma$ is provable in DILL if and only if it is provable in DCLL (via the encoding given in Sec. 3 – the "only if" part follows from Thm. 1).

The second question is on the *fullness* of Intuitionistic Linear Logic in Classical Linear Logic.

Conjecture 2 (fullness). DILL is full in DCLL. That is, if $\Gamma \,;\, \Delta \vdash N : \sigma$ is derivable in DCLL and all the types in Γ, Δ and σ stay in DILL, then there exists a DILL-term $\Gamma \,;\, \Delta \vdash M : \sigma$ so that $\Gamma \,;\, \Delta \vdash M = N : \sigma$ is provable in DCLL.

Note that the corresponding results for multiplicative fragments are already known: MILL is fully complete in MLL, see for instance [15]. We also know that MILL is fully complete in DILL [11] – but how about DILL and DCLL? In fact, one of our motivations to introduce DCLL has been to provide a manageable foundation for attacking this question. We expect that this will be positively solved by using the model construction techniques (categorical glueing / logical relations) in [23, 15].

6.5 Decidability of the Equational Theory

Another natural question on DCLL is

Conjecture 3 (decidability). The equational theory of DCLL is decidable.

We shall note that the equational theory of DILL is known to be decidable, see [1]. The same is true for MLL (in [17] the corresponding coherence problem for $*$-autonomous categories is solved). We hope that some rewriting techniques are effective for this purpose, especially using some $\lambda\mu$-calculus style variant of DCLL (e.g. μDCLL given in Appendix B). However, even though DCLL avoids to deal with commuting conversions explicitly, we still have to work up to certain equivalence classes of terms, e.g. as in [17] (for instance $\lambda x^{\perp}.\lambda f^{\perp \multimap \perp}.\lambda g^{\perp \multimap \perp}.f\,(g\,x) = \lambda x^{\perp}.\lambda f^{\perp \multimap \perp}.\lambda g^{\perp \multimap \perp}.g\,(f\,x)$ holds in DCLL, but there is no natural way to give an orientation on this equation).

References

[1] Barber, A. (1997) *Linear Type Theories, Semantics and Action Calculi*. PhD Thesis ECS-LFCS-97-371, University of Edinburgh. 458, 468

[2] Barber, A. and Plotkin, G. (1997) Dual intuitionistic linear logic. Submitted. An earlier version available as Technical Report ECS-LFCS-96-347, LFCS, University of Edinburgh. 458, 459, 460, 464

[3] Barr, M. (1979) *$*$-Autonomous Categories*. Springer Lecture Notes in Math. **752**. 464

[4] Barr, M. (1991) $*$-autonomous categories and linear logic. *Math. Struct. Comp. Sci.* **1**, 159–178. 464

[5] Benton, P. N. (1995) A mixed linear and non-linear logic: proofs, terms and models (extended abstract). In *Computer Science Logic (CSL'94)*, Springer Lecture Notes in Comput. Sci. **933**, pp. 121–135. 464

[6] Berdine, J., O'Hearn, P. W., Reddy, U. S. and Thielecke, H. (2001) Linearly used continuations. In *Proc. ACM SIGPLAN Workshop on Continuations (CW'01)*, Technical Report No. 545, Computer Science Department, Indiana University, pp. 47–54. 458, 466

[7] Bierman, G. M. (1995) What is a categorical model of intuitionistic linear logic? In *Proc. Typed Lambda Calculi and Applications (TLCA'95)*, Springer Lecture Notes in Comput. Sci. **902**, pp. 78–93. 464

[8] Bierman, G. M. (1999) A classical linear lambda-calculus. *Theoret. Comp. Sci.* **227**(1-2), 43–78. 461, 467

[9] Filinski, A. (1992) Linear continuations. In *Proc. Principles of Programming Languages (POPL'92)*, pp. 27–38. 466

[10] Girard, J.-Y. (1987) Linear logic. *Theoret. Comp. Sci.* **50**, 1–102. 458

[11] Hasegawa, M. (1999) Logical predicates for intuitionistic linear type theories. In *Proc. Typed Lambda Calculi and Applications (TLCA'99)*, Springer Lecture Notes in Comput. Sci. **1581**, pp. 198–213. 458, 467

[12] Hasegawa, M. (2000) Girard translation and logical predicates. *J. Funct. Programming* **10**(1), 77–89. 466

[13] Hasegawa, M. (2002) Linearly used effects: monadic and CPS transformations into the linear lambda calculus. In *Proc. Functional and Logic Programming (FLOPS2002)*, Springer Lecture Notes in Comput. Sci. 458, 466

[14] Hofmann, M., Pavlović, D. and Rosolini, P. (eds.) (1999) *Proc. 8th Conf. on Category Theory and Computer Science*. Electron. Notes Theor. Comput. Sci. **29**. 469
[15] Hyland, M. and Schalk, A. (200x) Glueing and orthogonality for models of linear logic. To appear in *Theoret. Comp. Sci.* 464, 467
[16] Kelly, G. M. and Mac Lane, S. (1971) Coherence in closed categories. *J. Pure Appl. Algebra* **1**(1):97–140. 461
[17] Koh, T. W. and Ong, C.-H. L. (1999) Explicit substitution internal languages for autonomous and $*$-autonomous categories. In [14]. 460, 468
[18] Maietti, M. E., de Paiva, V. and Ritter, E. (2000) Categorical models for intuitionistic and linear type theory. In *Foundations of Software Science and Computation Structure (FoSSaCS 2000)*, Springer Lecture Notes in Comput. Sci. **1784**, pp. 223–237. 466, 467
[19] Murawski, A. S. and Ong, C.-H. L. (1999) Exhausting strategies, Joker games and IMLL with units. In [14]. 461
[20] Nishizaki, S. (1993) Programs with continuations and linear logic. *Science of Computer Programming* **21**(2), 165–190. 466
[21] Parigot, M. (1992) $\lambda\mu$-calculus: an algorithmic interpretation of classical natural deduction. In *Proc. Logic Programming and Automated Reasoning*, Springer Lecture Notes in Comput. Sci. **624**, pp. 190–201. 460
[22] Plotkin, G. (1993) Type theory and recursion (extended abstract). In *Proc. Logic in Computer Science (LICS'93)*, pp. 374. 466
[23] Streicher, T. (1999) Denotational completeness revisited. In [14]. 458, 467
[24] Wadler, P. (1990) Linear types can change the world! In *Proc. Programming Concepts and Methods*, North-Holland, pp. 561–581. 466

A Dual Intuitionistic Linear Logic

Types and Terms

$$\begin{aligned}\sigma &::= b \mid I \mid \sigma \otimes \sigma \mid \sigma \multimap \sigma \mid !\sigma \\ M &::= x \mid * \mid \mathsf{let}\ * \ \mathsf{be}\ M\ \mathsf{in}\ M \mid M \otimes M \mid \mathsf{let}\ x^\sigma \otimes x^\sigma\ \mathsf{be}\ M\ \mathsf{in}\ M \mid \\ &\quad \lambda x^\sigma.M \mid MM \mid !M \mid \mathsf{let}\ !x^\sigma\ \mathsf{be}\ M\ \mathsf{in}\ M\end{aligned}$$

Typing

$$\frac{}{\Gamma_1, x:\sigma, \Gamma_2\ ;\ \emptyset \vdash x:\sigma}\,(\text{Int-Ax}) \qquad \frac{}{\Gamma\ ;\ x:\sigma \vdash x:\sigma}\,(\text{Lin-Ax})$$

$$\frac{}{\Gamma\ ;\ \emptyset \vdash * : I}\,(I\,\text{I}) \qquad \frac{\Gamma\ ;\ \Delta_1 \vdash M : I \quad \Gamma\ ;\ \Delta_2 \vdash N : \sigma}{\Gamma\ ;\ \Delta_1 \sharp \Delta_2 \vdash \mathsf{let}\ *\ \mathsf{be}\ M\ \mathsf{in}\ N : \sigma}\,(I\,\text{E})$$

$$\frac{\Gamma\ ;\ \Delta_1 \vdash M : \sigma_1 \quad \Gamma\ ;\ \Delta_2 \vdash N : \sigma_2}{\Gamma\ ;\ \Delta_1 \sharp \Delta_2 \vdash M \otimes N : \sigma_1 \otimes \sigma_2}\,(\otimes\,\text{I}) \qquad \frac{\begin{array}{c}\Gamma\ ;\ \Delta_1 \vdash M : \sigma_1 \otimes \sigma_2 \\ \Gamma\ ;\ \Delta_2, x:\sigma_1, y:\sigma_2 \vdash N : \tau\end{array}}{\Gamma\ ;\ \Delta_1 \sharp \Delta_2 \vdash \mathsf{let}\ x^{\sigma_1} \otimes y^{\sigma_2}\ \mathsf{be}\ M\ \mathsf{in}\ N : \tau}\,(\otimes\,\text{E})$$

$$\frac{\Gamma\ ;\ \Delta, x:\sigma_1 \vdash M : \sigma_2}{\Gamma\ ;\ \Delta \vdash \lambda x^{\sigma_1}.M : \sigma_1 \multimap \sigma_2}\,(\multimap\,\text{I}) \qquad \frac{\Gamma\ ;\ \Delta_1 \vdash M : \sigma_1 \multimap \sigma_2 \quad \Gamma\ ;\ \Delta_2 \vdash N : \sigma_1}{\Gamma\ ;\ \Delta_1 \sharp \Delta_2 \vdash M\,N : \sigma_2}\,(\multimap\,\text{E})$$

$$\frac{\Gamma\ ;\ \emptyset \vdash M : \sigma}{\Gamma\ ;\ \emptyset \vdash !M : !\sigma}\,(!\,\text{I}) \qquad \frac{\Gamma\ ;\ \Delta_1 \vdash M : !\sigma \quad \Gamma, x:\sigma\ ;\ \Delta_2 \vdash N : \tau}{\Gamma\ ;\ \Delta_1 \sharp \Delta_2 \vdash \mathsf{let}\ !x\ \mathsf{be}\ M\ \mathsf{in}\ N : \tau}\,(!\,\text{E})$$

Axioms

$$\begin{array}{rr}
\mathsf{let}\ *\ \mathsf{be}\ *\ \mathsf{in}\ M = M & \mathsf{let}\ *\ \mathsf{be}\ M\ \mathsf{in}\ * = M \\
\mathsf{let}\ x \otimes y\ \mathsf{be}\ M \otimes N\ \mathsf{in}\ L = L[M/x, N/y] & \mathsf{let}\ x \otimes y\ \mathsf{be}\ M\ \mathsf{in}\ x \otimes y = M \\
(\lambda x.M)\, N = M[N/x] & \lambda x.M\, x = M \\
\mathsf{let}\ !x\ \mathsf{be}\ !M\ \mathsf{in}\ N = N[M/x] & \mathsf{let}\ !x\ \mathsf{be}\ M\ \mathsf{in}\ !x = M
\end{array}$$

$$\begin{array}{c}
C[\mathsf{let}\ *\ \mathsf{be}\ M\ \mathsf{in}\ N] = \mathsf{let}\ *\ \mathsf{be}\ M\ \mathsf{in}\ C[N] \\
C[\mathsf{let}\ x \otimes y\ \mathsf{be}\ M\ \mathsf{in}\ N] = \mathsf{let}\ x \otimes y\ \mathsf{be}\ M\ \mathsf{in}\ C[N] \\
C[\mathsf{let}\ !x\ \mathsf{be}\ M\ \mathsf{in}\ N] = \mathsf{let}\ !x\ \mathsf{be}\ M\ \mathsf{in}\ C[N]
\end{array}$$

where $C[-]$ is a linear context (no ! binds $[-]$).

B μDCLL

B.1 The System μDCLL

Types and Terms

$$\begin{array}{l}
\sigma ::= b \mid \sigma \to \sigma \mid \sigma \multimap \sigma \mid \bot \\
M ::= x \mid \boldsymbol{\lambda} x^{\sigma}.M \mid M \mathbin{@} M \mid \lambda x^{\sigma}.M \mid M\, M \mid [\alpha]M \mid \mu\alpha^{\sigma}.M
\end{array}$$

Typing

$$\frac{}{\Gamma_1, x : \sigma, \Gamma_2 \ ;\ \emptyset \vdash x : \sigma \mid \Sigma}\ (\text{Int-Ax}) \qquad \frac{}{\Gamma \ ;\ x : \sigma \vdash x : \sigma \mid \emptyset}\ (\text{Lin-Ax})$$

$$\frac{\Gamma, x : \sigma_1 \ ;\ \Delta \vdash M : \sigma_2 \mid \Sigma}{\Gamma \ ;\ \Delta \vdash \boldsymbol{\lambda} x^{\sigma_1}.M : \sigma_1 \to \sigma_2 \mid \Sigma}\ (\to\text{I}) \qquad \frac{\Gamma \ ;\ \Delta \vdash M : \sigma_1 \to \sigma_2 \mid \Sigma \quad \Gamma \ ;\ \emptyset \vdash N : \sigma_1 \mid \emptyset}{\Gamma \ ;\ \Delta \vdash M \mathbin{@} N : \sigma_2 \mid \Sigma}\ (\to\text{E})$$

$$\frac{\Gamma \ ;\ \Delta, x : \sigma_1 \vdash M : \sigma_2 \mid \Sigma}{\Gamma \ ;\ \Delta \vdash \lambda x^{\sigma_1}.M : \sigma_1 \multimap \sigma_2 \mid \Sigma}\ (\multimap\text{I}) \qquad \frac{\Gamma \ ;\ \Delta_1 \vdash M : \sigma_1 \multimap \sigma_2 \mid \Sigma_1 \quad \Gamma \ ;\ \Delta_2 \vdash N : \sigma_1 \mid \Sigma_2}{\Gamma \ ;\ \Delta_1 \sharp \Delta_2 \vdash MN : \sigma_2 \mid \Sigma_1 \sharp \Sigma_2}\ (\multimap\text{E})$$

$$\frac{\Gamma \ ;\ \Delta \vdash M : \sigma \mid \Sigma}{\Gamma \ ;\ \Delta \vdash [\alpha]M : \bot \mid \alpha : \sigma, \Sigma}\ (\bot\text{I}) \qquad \frac{\Gamma \ ;\ \Delta \vdash M : \bot \mid \alpha : \sigma, \Sigma}{\Gamma \ ;\ \Delta \vdash \mu\alpha^{\sigma}.M : \sigma \mid \Sigma}\ (\bot\text{E})$$

Axioms

$$\begin{array}{lll}
(\boldsymbol{\lambda} x.M) \mathbin{@} N & = M[N/x] & \\
\boldsymbol{\lambda} x.M \mathbin{@} x & = M & (x \notin FV(M)) \\
(\lambda x.M)\, N & = M[N/x] & \\
\lambda x.M\, x & = M & \\
L\,(\mu\alpha^{\sigma}.M) & = M\left[{}^{L(-)}/_{[\alpha](-)}\right] & (L : \sigma \multimap \bot) \\
\mu\alpha.[\alpha]M & = M &
\end{array}$$

where $M\left[{}^{L(-)}/_{[\alpha](-)}\right]$ is obtained by replacing the (unique) subterm of the form $[\alpha]N$ by $L\,N$ in the capture-free way.

Lemma 5. *The following equations are provable in μDCLL.*

- $L\,(\mu\alpha^{\sigma}.M) = \mu\beta^{\tau}.M\left[{}^{[\beta]L(-)}/_{[\alpha](-)}\right]$ *where* $L : \sigma \multimap \tau$
- $[\alpha'](\mu\alpha^{\sigma}.M) = M[\alpha'/\alpha]$
- $\mu\alpha^{\perp}.M = M\left[{}^{(-)}/_{[\alpha](-)}\right]$
- $\mu\gamma^{\sigma\to\tau}.M = \boldsymbol{\lambda}x^{\sigma}.\mu\beta^{\tau}.M\left[{}^{[\beta](-)\,@\,x}/_{[\gamma](-)}\right]$
- $\mu\gamma^{\sigma\multimap\tau}.M = \lambda x^{\sigma}.\mu\beta^{\tau}.M\left[{}^{[\beta](-)x}/_{[\gamma](-)}\right]$ □

B.2 DCLL vs. μDCLL

We first note that the combinator C_{σ} is easily represented in μDCLL by

$$\mathsf{C}_{\sigma} \;=\; \lambda m^{(\sigma\multimap\perp)\multimap\perp}.\mu\alpha^{\sigma}.m\,(\lambda x^{\sigma}.[\alpha]x) \;:\; ((\sigma\multimap\perp)\multimap\perp)\multimap\sigma.$$

Let us write M° for the induced translation of a DCLL-term M in μDCLL by this encoding.

Lemma 6. *If* $\Gamma\ ;\ \Delta \vdash M : \sigma$ *is derivable in DCLL,* $\Gamma\ ;\ \Delta \vdash M^{\circ} : \sigma \mid \emptyset$ *is derivable in* μ*DCLL.* □

Proposition 2. *If* $\Gamma\ ;\ \Delta \vdash M = N : \sigma$ *is provable in DCLL,* $\Gamma\ ;\ \Delta \vdash M^{\circ} = N^{\circ} : \sigma \mid \emptyset$ *is provable in* μ*DCLL.* □

Conversely, there is a translation $(-)^{\bullet}$ from μDCLL to DCLL given by

$$\begin{aligned}([\alpha]M)^{\bullet} &= [\alpha]M^{\bullet}\\ (\mu\alpha^{\sigma}.M)^{\bullet} &= \mathsf{C}_{\sigma}\left(\lambda k.M^{\bullet}\left[{}^{k(-)}/_{[\alpha](-)}\right]\right)\end{aligned}$$

and so on; for this $(-)^{\bullet}$ we have

Lemma 7. *If* $\Gamma\ ;\ \Delta \vdash M : \sigma \mid \alpha_1 : \sigma_1, \ldots, \alpha_n : \sigma_n$ *is derivable in* μ*DCLL,* $\Gamma\ ;\ \Delta, k_n : \sigma_n \multimap \perp, \ldots, k_1 : \sigma_1 \multimap \perp \vdash M^{\bullet}\left[{}^{k_1(-)}/_{[\alpha_1](-)}, \ldots, {}^{k_n(-)}/_{[\alpha_n](-)}\right] : \sigma$ *is derivable in DCLL. In particular, if* $\Gamma\ ;\ \Delta \vdash M : \sigma \mid \emptyset$ *is derivable in* μ*DCLL,* $\Gamma\ ;\ \Delta \vdash M^{\bullet} : \sigma \mid \emptyset$ *is derivable in DCLL.* □

Proposition 3. *If* $\Gamma\ ;\ \Delta \vdash M = N : \sigma \mid \emptyset$ *is provable in* μ*DCLL,* $\Gamma\ ;\ \Delta \vdash M^{\bullet} = N^{\bullet} : \sigma$ *is provable in DCLL.* □

Proposition 4. *For* $\Gamma\ ;\ \Delta \vdash M : \sigma$ *we have* $\Gamma\ ;\ \Delta \vdash M = M^{\circ\bullet} : \sigma$ *in DCLL. For* $\Gamma\ ;\ \Delta \vdash M : \sigma \mid \emptyset$ *we have* $\Gamma\ ;\ \Delta \vdash M = M^{\bullet\circ} : \sigma \mid \emptyset$ *in* μ*DCLL.* □

Thus we conclude that DCLL is identical to the single conclusion-fragment of μDCLL as typed equational theories.

B.3 Categorical Semantics

The interpretation of a typing judgement of the form

$$x_1 : \sigma_1, \ldots, x_m : \sigma_m\ ;\ y_1 : \tau_1, \ldots, y_n : \tau_n \vdash M : \sigma \mid \alpha_1 : \theta_1, \ldots, \alpha_k : \theta_k$$

is given as an arrow from $!\llbracket\sigma_1\rrbracket \otimes \ldots \otimes !\llbracket\sigma_m\rrbracket \otimes \llbracket\tau_1\rrbracket \otimes \ldots \otimes \llbracket\tau_n\rrbracket$ to $\llbracket\sigma\rrbracket \parr \llbracket\theta_1\rrbracket \parr \ldots \parr \llbracket\theta_k\rrbracket$, by routinely extending the case of DCLL. The soundness and completeness of μDCLL with respect to the same class of categorical models immediately follow.

C Formulation without C

As noted in Sec. 2, we can formalize DCLL using just lambda terms and five axioms, if there is no base type. The same is true for MLL, for which just three axioms are sufficient.

C.1 DCLL

Types and Terms

$$\sigma ::= \sigma \to \sigma \mid \sigma \multimap \sigma \mid \bot \qquad M ::= x \mid \boldsymbol{\lambda} x^{\sigma}.M \mid M \mathbin{@} M \mid \lambda x^{\sigma}.M \mid M\,M$$

Typing

$$\frac{}{\Gamma_1, x:\sigma, \Gamma_2 \;;\; \emptyset \vdash x:\sigma}\ (\text{Int-Ax}) \qquad \frac{}{\Gamma \;;\; x:\sigma \vdash x:\sigma}\ (\text{Lin-Ax})$$

$$\frac{\Gamma, x:\sigma_1 \;;\; \Delta \vdash M:\sigma_2}{\Gamma \;;\; \Delta \vdash \boldsymbol{\lambda} x^{\sigma_1}.M:\sigma_1 \to \sigma_2}\ (\to \text{I}) \qquad \frac{\Gamma \;;\; \Delta \vdash M:\sigma_1 \to \sigma_2 \quad \Gamma \;;\; \emptyset \vdash N:\sigma_1}{\Gamma \;;\; \Delta \vdash M \mathbin{@} N:\sigma_2}\ (\to \text{E})$$

$$\frac{\Gamma \;;\; \Delta, x:\sigma_1 \vdash M:\sigma_2}{\Gamma \;;\; \Delta \vdash \lambda x^{\sigma_1}.M:\sigma_1 \multimap \sigma_2}\ (\multimap \text{I}) \qquad \frac{\Gamma \;;\; \Delta_1 \vdash M:\sigma_1 \multimap \sigma_2 \quad \Gamma \;;\; \Delta_2 \vdash N:\sigma_1}{\Gamma \;;\; \Delta_1 \sharp \Delta_2 \vdash M\,N:\sigma_2}\ (\multimap \text{E})$$

Axioms

$$\begin{aligned}
&(\boldsymbol{\lambda} x.M) \mathbin{@} N && = M[N/x] \\
&\boldsymbol{\lambda} x.M \mathbin{@} x && = M \qquad (x \notin FV(M)) \\
&(\lambda x.M)\,N && = M[N/x] \\
&\lambda x.M\,x && = M \\
&L\,(\lambda x^{\sigma}.M\,(\lambda f^{\sigma \multimap \bot}.f\,x)) && = M\,L \qquad \begin{pmatrix} L:(\sigma \multimap \bot) \multimap \bot \\ M:((\sigma \multimap \bot) \multimap \bot) \multimap \bot \end{pmatrix}
\end{aligned}$$

C.2 MLL

Types and Terms

$$\sigma ::= \sigma \multimap \sigma \mid \bot \qquad M ::= x \mid \lambda x^{\sigma}.M \mid M\,M$$

Typing

$$\frac{}{x:\sigma \vdash x:\sigma}\ (\text{Ax}) \qquad \frac{\Delta, x:\sigma_1 \vdash M:\sigma_2}{\Delta \vdash \lambda x^{\sigma_1}.M:\sigma_1 \multimap \sigma_2}\ (\multimap \text{I}) \qquad \frac{\Delta_1 \vdash M:\sigma_1 \multimap \sigma_2 \;\; \Delta_2 \vdash N:\sigma_1}{\Delta_1 \sharp \Delta_2 \vdash M\,N:\sigma_2}\ (\multimap \text{E})$$

Axioms

$$\begin{aligned}
&(\lambda x.M)\,N && = M[N/x] \\
&\lambda x.M\,x && = M \\
&L\,(\lambda x^{\sigma}.M\,(\lambda f^{\sigma \multimap \bot}.f\,x)) && = M\,L \qquad \begin{pmatrix} L:(\sigma \multimap \bot) \multimap \bot \\ M:((\sigma \multimap \bot) \multimap \bot) \multimap \bot \end{pmatrix}
\end{aligned}$$