Logical Methods in Computer Science
Vol. 10(3:16)2014, pp. 1–36
www.lmcs-online.org

Submitted Oct. 23, 2012
Published Sep. 23, 2014

CATEGORICAL PROOF THEORY OF CO-INTUITIONISTIC LINEAR LOGIC

GIANLUIGI BELLIN

Dipartimento di Informatica, Università di Verona, Strada Le Grazie, 37134 Verona Italy
e-mail address: gianluigi.bellin@univr.it

Revision Note. This is a revised and corrected version of the article originally published on September 10, 2014.

Abstract. To provide a categorical semantics for co-intuitionistic logic, one has to face the fact, noted by Tristan Crolard, that the definition of co-exponents as adjuncts of co-products does not work in the category **Set**, where co-products are disjoint unions. Following the familiar construction of models of intuitionistic linear logic with exponential "**!**", we build models of co-intuitionistic logic in symmetric monoidal closed categories with additional structure, using a variant of Crolard's term assignment to co-intuitionistic logic in the construction of a free category.

Preface

This paper sketches a categorical semantics for co-intuitionistic logic, advancing a line of proof-theoretic research developed in [1, 2, 3, 5, 7]. Co-intuitionistic logic, also called *dual-intuitionistic* [21, 35, 36], may be superficially regarded as completely determined by the duality, as in its lattice-theoretic semantics. A *co-Heyting algebra* is a (distributive) lattice $\mathcal{C}$ such that its opposite $\mathcal{C}^{op}$ is a Heyting algebra. In a Heyting algebra *implication* $B \to A$ is defined as the right adjoint of *meet*, so in a co-Heyting algebra $\mathcal{C}$ *co-implication* (or *subtraction*) $A \smallsetminus B$ is defined as the left adjoint of *join*:

$$\frac{C \wedge B \ \leq \ A}{C \ \leq \ B \to A} \qquad\qquad \frac{A \ \leq \ B \vee C}{A \smallsetminus B \ \leq \ C}$$

A *bi-Heyting* algebra is a lattice that has both the structure of a Heyting and of a co-Heyting algebra. The logic of bi-Heyting algebras was introduced by Cecylia Rauszer [30, 31] (called *Heyting-Brouwer logic*), who defined also its Kripke semantics; a category-theoretical approach to the topic is due to Makkai, Reyes and Zolfaghari [28, 32]. The suggestion by F. W. Lawvere to use co-Heyting algebras as a logical framework to treat the topological notion of boundary has not been fully explored yet (but see recent work by Pagliani [29]).

2012 ACM CCS: [**Theory of computation**]: Logic—Proof theory; Linear logic.
Key words and phrases: Categorical Proof Theory, Intuitionistic duality, Categorical Semantics of Intuitionistic Linear Logic, Semantics of coroutines and concurrent computations.

Early research showed that the extension of first order intuitionistic logic with subtraction yields an *intermediate* logic of *constant domains* [26]. In a rich and interesting paper [18] T. Crolard showed, essentially by Joyal's argument, that Cartesian closed categories with exponents and co-exponents are degenerate; in fact even the topological models of bi-intuitionistic logic, i.e., *bi-topological spaces*, are degenerate. Crolard's motivations are mainly computational: he studies bi-intuitionistic logic in the framework of the *classical* $\lambda\mu$ calculus, to provide a type-theoretic analysis of the notion of *coroutine*; then he identifies a subclass of *safe coroutines* that can be typed constructively [19]. From our viewpoint, Crolard's work suggests two directions of research. On one hand, it opens the way to a "bottom up" approach to safe coroutines, independent of the $\lambda\mu$ calculus, i.e., co-intuitionistic coroutines [3, 1, 2, 7]. On the other hand, the question arises whether the collapse of algebraic and topological models may be avoided by building the intuitionistic and co-intuitionistic sides separately, starting from distinct sets of elementary formulas, and then by joining the two sides with mixed connectives (mainly, two negations expressing the duality): this is our variant of bi-intuitionistic logic, presented in [5, 1, 6].

Both of these tasks were advocated by this author and pursued within a project of "logic for pragmatics" with motivations from linguistics and natural language representation [1, 5, 9]. In the characterization of the logical properties of "illocutionary acts", such as *asserting*, *making hypotheses* and *conjectures* one finds in natural reasoning forms of duality that can be related to intuitionistic dualities. For co-intuitionistic logic Crolard's term assignment has been adapted to a sequent-style natural deduction setting with single-premise and multiple-conclusions. For (our variant of) bi-intuitionistic logic Kripke semantics has been given (both in **S4** and in *bi-modal* **S4**) and a sequent calculus has been proposed where sequents are of the form

$$\Gamma\ ;\ \Rightarrow\ A\ ;\ \Upsilon \qquad \text{or} \qquad \Gamma\ ;\ C\ \Rightarrow\ ;\ \Upsilon$$

where Γ and A are intuitionistic (assertive) formulas and C and Υ co-intuitionistic (hypothetical).

But from the viewpoint of category theory a crucial remark by Crolard shows that already in *co-intuitionistic* logic there is a problem: namely, only trivial co-exponents exist in the category **Set**. Indeed the categorical semantics of intuitionistic disjunction is given by *coproducts* [24], which in **Set** are represented by *disjoint unions*. On the other hand the categorical semantics of subtraction is given by *co-exponents*. The co-exponent of A and B is an object B_A together with an arrow $\ni_{A,B}$: $B \to B_A \oplus A$ such that for any arrow $f : B \to C \oplus B$ there exists a unique $f_* : B_A \to C$ such that the following diagram commutes:

$$\begin{tikzcd} B \arrow[r, "f"] \arrow[dr, "\ni_{A,B}"'] & C \oplus A \\ & B_A \oplus A \arrow[u, "f_* \oplus id_A"'] \end{tikzcd}$$

It follows that

> *in the category of sets, the co-exponent B_A of two sets A and B is defined if and only if $A = \emptyset$ or $B = \emptyset$* (see [18], Proposition 1.15).

The proof is instructive: in **Set**, the coproduct $\oplus$ is *disjoint union*; thus if $A \neq \emptyset \neq B$ then the functions f and $\ni_{A,B}$ for every $b \in B$ must *choose a side*, left or right, of the coproduct in their target and moreover $f_\star \oplus 1_A$ leaves the side unchanged. Hence, if we take a nonempty set C and f with the property that for some b different sides are chosen by f and $\ni_{A,B}$, then the diagram does not commute.

Thus to have a categorical semantics of co-exponents we need categories where a different notion of disjunction is modelled. The connective *par* of linear logic is a good candidate and a treatment of *par* is available in *full intuitionistic linear logic* (**FILL**) [16, 22], with a proof-theory and a categorical semantics. The *multiple-conclusion* consequence relation of **FILL** and its term assignment have given motivation and inspiration to our work, as a calculus where a distinct term is assigned to each formula in the succedent. The language of **FILL** has tensor ($\otimes$), linear implication ($\multimap$) and *par* ($\wp$) and a main proof-theoretic concern has been the compatibility between *par* and *linear implication*, namely, to find restrictions on the introduction of linear implication that guarantee its *functional* intuitionistic character and at the same time allow to prove cut-elimination (on this point see also [4]).

Linear co-intuitionistic logic appears already in Schellinx[33] and Lambek [23]. Works by R. Blute, J. Cockett, R. Sely and T. Trimble on *wekly distributive categories* [14, 15] provide a sophisticated technology of natural deduction, proof-nets and categorical models for various systems of linear logic without an involutory negation; Cockett and Seely [17] consider also non-commutative systems with implications and subtractions. When *boxes* or other conditions are given for intuitionistic linear implications and co-intuitionistic subtractions, these systems provide a suitable categorical proof theory of linear bi-intuitionism.

To construct categorical models of linear co-intuitionistic logic it suffices to notice that in *monoidal categories par* can be modelled by a monoidal operation and co-exponents as the *left adjoint* of *par*. The main task then is to model Girard's exponential *why not?*: in this way a categorical semantics for co-intuitionistic logic can be recovered by applying the dual of Girard's translation of intuitionistic logic into linear logic, namely:

$$\begin{aligned}
(p)^\circ &= \mathrm{p} \\
(\mathbf{f})^\circ &= \mathbf{0} \\
(C \curlyvee D)^\circ &= ?(C^\circ \oplus D^\circ) = ?(C^\circ)\wp ?(D^\circ) \\
(C \smallsetminus D)^\circ &= C^\circ \smallsetminus (?D^\circ) \\
(E \vdash C_1, \ldots, C_n)^\circ &= ?(E^\circ) \vdash ?(C_1^\circ), \ldots, ?(C_n^\circ))
\end{aligned}$$

where $\mathbf{0}$ is the identity of $\oplus$ and we use "$\smallsetminus$" both in linear and in non-linear co-intuitionistic logic.

The task amounts to dualizing Nick Benton, Gavin Bierman, Valeria de Paiva and Martin Hyland's well-known semantics for intuitionistic linear logic [11]. This may be regarded as a routine exercise, except that one has to provide a term assignment suitable for the purpose. In this task we build on a term assignment to multiplicative co-intuitionistic logic, which has been proposed as an abstract *distributed calculus* dualizing the linear λ calculus [2, 3, 7]: in our view such a dualization underlies the translation of the linear λ-calculus into the π-calculus (see [10]).

As a matter of fact, Nick Benton's *mixed Linear and Non-Linear* logic [12] may give us not only an easier approach to modelling the exponentials but also the key to a categorical semantics of (our version of) bi-intuitionistic logic: indeed, by dualizing the linear part of Benton's system we may obtain both a proof-theoretic and a category theoretic framework for mixed co-intuitionistic linear and intuitionistic logic and thus also for bi-intuitionistic logic - of course, we need to use the exponential *why not?* and dualize Girard's translation.

But then a categorical investigation of *linear cointuitionistic logic* and of the *why not?* exponential is a preliminary step in this direction and has an independent interest.

1. Proof Theory

The language of *co-intuitionistic linear logic*, given an infinite sequence of elementary formulas $\eta_1, \eta_2, \ldots$, is defined by the following grammar:[1]

$$C, D \ := \ \eta \mid \bot \mid C \wp D \mid C \setminus D \mid ?C$$

The rules of sequent-style natural deduction **co-ILL** for co-Intuitionistic Linear Logic are given in Table 1. As *co-intuitionistic linear logic* may be quite unfamiliar, we sketch an

$$\begin{array}{cc}
\textit{assumption} & \textit{substitution} \\
A \vdash A & \dfrac{E \vdash \Gamma, A \qquad A \vdash \Delta}{E \vdash \Gamma, \Delta} \\[2ex]
\bot\text{-I} & \bot\text{-E} \\
\dfrac{E \vdash \Gamma}{E \vdash \Gamma, \bot} & \bot \vdash \\[2ex]
\setminus\text{-I} & \setminus\text{-E} \\
\dfrac{E \vdash \Gamma, C \qquad D \vdash \Delta}{E \vdash \Gamma, C \setminus D, \Delta} & \dfrac{H \vdash \Upsilon, C \setminus D \qquad C \vdash D, \Delta}{H \vdash \Upsilon, \Delta} \\[2ex]
\wp\text{-I} & \wp\text{-E} \\
\dfrac{E \vdash \Gamma, C_0, C_1}{E \vdash \Gamma, C_0 \wp C_1} & \dfrac{H \vdash \Upsilon, C_0 \wp C_1 \qquad C_0 \vdash \Gamma_0 \qquad C_1 \vdash \Gamma_1}{H \vdash \Upsilon, \Gamma_0, \Gamma_1} \\[2ex]
\textit{dereliction} & \textit{storage} \\
\dfrac{E \vdash \Gamma,\ C}{E \vdash \Gamma,\ ?C} & \dfrac{H \vdash \Upsilon,\ ?C \qquad C \vdash ?\Delta}{H \vdash \Upsilon,\ ?\Delta} \\[2ex]
\textit{weakening} & \textit{contraction} \\
\dfrac{E \vdash \Gamma}{E \vdash \Gamma,\ ?C} & \dfrac{E \vdash \Gamma,\ ?C, ?C}{E \vdash \Gamma,\ ?C}
\end{array}$$

Table 1: Natural Deduction for **co-ILL**

intuitive explanation of its proof theory. We think of co-intuitionistic logic as being about *making hypotheses* [1, 5, 6]. It has a consequence relation of the form

$$H \vdash H_1, \ldots, H_n. \tag{1.1}$$

Suppose H is a hypothesis: which (disjunctive sequence of) hypotheses H_1 *or* ... *or* H_n follow from H? Since the logic is *linear*, commas in the meta-theory stand for Girard's *par* and the structural rules Weakening and Contraction are not allowed. A relevant feature, which we shall not discuss here, is that the consequence relation may be seen as *distributed*, i.e., we may think of the alternatives $H_1, \ldots, H_n$ in (1.1) as lying in different *locations* [2, 7].

[1] In accordance with our interpretation of co-intuitionism as a logic of hypotheses, we may write elementary formulas η as $\mathcal{H}\, p$, where "$\mathcal{H}$" is a sign for the illocutionary force of hypothesis and p is an atomic proposition. Such a linguistic analysis plays no explicit role in this paper.

The main connectives are *subtraction* $A \smallsetminus B$ (*possibly* A *and not* B) and Girard's *par* $A \wp B$. Natural Deduction inference rules for *subtraction* (in a sequent form) are as follows.

$$\smallsetminus\text{-}intro\ \frac{H \vdash \Gamma, C \qquad D \vdash \Delta}{H \vdash \Gamma, C \smallsetminus D, \Delta} \qquad \smallsetminus\text{-}elim\ \frac{H \vdash \Delta, C \smallsetminus D \qquad C \vdash D, \Upsilon}{H \vdash \Delta, \Upsilon}$$

Notice that in the $\smallsetminus$-*elimination* rule the evidence that D may be derivable from C given by the *right premise* has become *inconsistent with the hypothesis* $C \smallsetminus D$ in the left premise; in the conclusion we drop D and we *set aside* the evidence for the inconsistent alternative. Namely, such evidence is not destroyed, but rather stored somewhere for future use.

If the *left premise* of $\smallsetminus$-*elimination*, deriving $C \smallsetminus D$ or Δ from H, has been obtained by a $\smallsetminus$-*introduction*, this inference has the form

$$\frac{H \vdash \Delta_1, C \qquad D \vdash \Delta_2}{H \vdash \Delta_1, \Delta_2, C \smallsetminus D}.$$

Then the pair of *introduction/elimination* rules can be eliminated: using the *removed evidence* that D with Υ are derivable from C (*right premise* of the $\smallsetminus$-*elim.*) we can conclude that $\Delta_1, \Delta_2, \Upsilon$ are derivable from H. This is, in a nutshell, the principle of normalization (or *cut-elimination*) for subtraction.

The *storage* operation is made explicit in the rules for the **?** operator of linear logic. Here an entire derivation d of $?\Delta$ from C (where $?\Delta = ?D_1, \dots, ?D_n$) is set aside; what is accessible now is something like a non-logical axiom of the form $?C \vdash ?\Delta$. However in the process of normalization the derivation d may be recovered to be *used*, *discarded* or *copied* in the interaction of a *storage* rule with *dereliction*, *weakening* or *contraction*: all of this is conceptually clear, thanks to J-Y. Girard, and has been mathematically analyzed in the *geometry of interaction.*

1.1. **From Crolard's classical coroutines to co-intuitionistic ones.** Crolard [19] provides a term assignment to the subtraction rules in the framework of Parigot's $\lambda\mu$-calculus, typed in a sequent-style natural deduction system. The $\lambda\mu$-calculus provides a typing system for functional programs with *continuations* and a computational interpretation of classical logic (see, e.g., [20, 34]).

In the type system for the $\lambda\mu$ calculus sequents may be written in the form $\Gamma \vdash t : A \mid \Delta$, with contexts $\Gamma = x_1 : C_1, \dots, x_m : C_m$ and $\Delta = \alpha_1 : D_1, \dots, \alpha_n : D_n$, where the x_i are *variables* and the α_j are μ-*variables* (or *co-names*). In addition to the rules of the simply typed lambda calculus, there are *naming rules*

$$\frac{\Gamma \vdash t : A \mid \alpha : A, \Delta}{\Gamma \vdash [\alpha]t : \bot \mid \alpha : A, \Delta}[\alpha] \qquad\qquad \frac{\Gamma \vdash t : \bot \mid \alpha : A, \Delta}{\Gamma \vdash \mu\alpha.t : A \mid \Delta}\mu$$

whose effect is to "change the goal" of a derivation and which allow us to represent the familiar *double negation rule* in Prawitz Natural Deduction.

Crolard extends the $\lambda\mu$ calculus with introduction and elimination rules for subtraction:[2]

[2]Actually in Crolard [19] the introduction rule is given in the more general form of $\smallsetminus$-*introduction* with two sequent premises (which we use below) and more general *continuation contexts* occur in place of β; the above formulation is logically equivalent and suffices for our purpose.

$$\frac{\Gamma \vdash t : A \mid \Delta}{\Gamma \vdash \texttt{make-coroutine}(t,\beta) : A \setminus B \mid \beta : B, \Delta} \setminus I$$

$$\frac{\Gamma \vdash t : A \setminus B \mid \Delta \qquad \Gamma,\ x : A \vdash u : B \mid \Delta}{\Gamma \vdash \texttt{resume}\ t\ \texttt{with}\ x \mapsto u : C \mid \Delta} \setminus E$$

The reduction of a redex of the form

$$\frac{\dfrac{\Gamma \vdash t : A \mid \Delta}{\Gamma \vdash \texttt{make-coroutine}(t,\beta) : A \setminus B \mid \beta : B, \Delta} \setminus\text{-I} \qquad \Gamma, x : A \vdash u : B \mid \Delta}{\Gamma \vdash \texttt{resume}\ (\texttt{make-coroutine}(t,\beta))\ \texttt{with}\ x \mapsto u\ \ : C \mid \beta : B, \Delta} \setminus\text{-E}$$

is as follows:

$$\frac{\dfrac{\dfrac{\Gamma \vdash t : A \mid \Delta \qquad \Gamma, x : A \vdash u\ B \mid \Delta}{\Gamma \vdash u[t/x] : B \mid \Delta}\ \textit{substitution}}{\Gamma \vdash [\beta]u[t/x] : \bot \mid \beta : B, \gamma : C, \Delta'}\ [\beta]}{\Gamma \vdash \mu\gamma.[\beta]u[t/x] : C \mid \beta : B, \Delta'}\ \mu$$

Notice that Crolard's elimination rule involves an application of the rule *ex falso quodlibet*, which is explicit in the definition of the operator `resume` (see [19], Remark 4.3.). This seems unavoidable working within the $\lambda\mu$ calculus, but it may not be desirable outside that framework.

Working with the full power of classical logic, if a constructive system of bi-intuitionistic logic is required, then the implication right and subtraction left rules must be restricted; this can be done by considering *relevant dependencies*.[3] Crolard is able to show that the term assignment for such a restricted logic is a calculus of *safe coroutines*, described as terms in which no coroutine can access the local environment of another coroutine.

Crolard's work suggests the possibility of defining co-intuitionistic coroutines directly, independently of the typing system of the $\lambda\mu$-calculus. Since μ-variable abstraction and the μ-rule are devices to change the "actual thread" of computation, the effect of removing such rules is that all "threads" of computation are simultaneously represented in a multiple conclusion sequent, but variables y that are temporarily inaccessible in a term N are being replaced by a term $\mathtt{y}(M)$ by the substitution $N[y := \mathtt{y}(M)]$, where M contains a free variable x which is accessible in the current context. This is the approach pursued in [1, 3, 5] leading to the present categorical presentation.

1.2. **A dual linear calculus for MNJ$^{\setminus\wp\perp}$.** We present the grammar and the basic definitions of our dual linear calculus for linear co-intuitionistic logic with subtraction, disjunction and *why not?* (**?**) operator.

Definition 1.1. We are given a countable set of *free variables* (denoted by x, y, z ...), and a countable set of *unary functions* (denoted by $\mathtt{x}, \mathtt{y}, \mathtt{z}, \ldots$). The terms of our calculus, denoted by R, are either *m-terms*, denoted by M, N, or *p-terms*, denoted by P.

(i) *Multiplicative* terms, m-terms and p-terms are defined by the following grammar.

[3] For instance, in the derivation of the right premise $\Gamma, x : A \vdash u : B \mid \Delta$ of a subtraction elimination ($\setminus E$), there should be no relevant dependency between the formula B and the assumptions in Γ, but only between B and A. Similar issues arise in **FILL**, see [22] and [4], section 4.

$$\begin{array}{rcl} M,N & := & x \mid \mathtt{x}(M) \mid \mathtt{connect\ to}(R) \mid M\wp N \mid \mathtt{casel}(M) \mid \mathtt{caser}(M) \mid \\ & & \mid \mathtt{make\text{-}coroutine}(M,\mathtt{x}) \mid [\,]. \\ P & := & \mathtt{postpone}(\mathtt{y} \mapsto N, M) \mid \mathtt{postpone}(M). \\ R & := & M \mid P. \end{array}$$

(ii)*Multiplicative and exponential* terms, m-terms and p-terms are obtained by adding to the above grammar the following clauses:

$$\begin{array}{rcl} M,N & := & \ldots \mid [M] \mid [M,N]. \\ P & := & \ldots \mid \mathtt{store}(P_1,\ldots,P_m,M_1,\ldots,M_n,\mathtt{y}_1,\ldots,\mathtt{y}_n,\mathtt{x},N) \end{array}$$

We usually abbreviate "$\mathtt{make}-\mathtt{coroutine}$" as "$\mathtt{mkc}$" and "$\mathtt{postpone}$" as "$\mathtt{postp}$". We often write $\overline{P}$ for a list $P_1,\ldots,P_m$, and similarly $\overline{M}$ for $M_1,\ldots,M_n$. "[]" is the empty term ($\mathtt{nil}$).

Definition 1.2. The *free variables* $FV(M)$ in a term R are defined thus:

$$\begin{array}{rl} FV(x) = & \{x\} \\ FV(\mathtt{x}(M)) = & FV(M) \\ FV(\mathtt{connect\ to}(R)) = & FV(R) \\ FV(M\wp N) = & FV(M) \cup FV(N) \\ FV(\mathtt{casel}(M)) = FV(\mathtt{caser}(M)) = & FV(M) \\ FV(\mathtt{mkc}\,(M,\mathtt{x})) = & FV(M) \\ FV(\mathtt{store}(\overline{P},M_1,\ldots,M_n,\mathtt{y}_1,\ldots,\mathtt{y}_n,\mathtt{x},N)) = & (FV(\overline{P}) \cup (FV(\overline{M})) \smallsetminus \{x\}) \cup FV(N) \\ & \text{where } \overline{M} = M_1,\ldots,M_n. \\ FV(\,[\,]\,) = \emptyset \quad FV(\,[M]\,) = FV(M) & FV(\,[\,M,N\,]\,) \;=\; FV(M) \cup FV(N) \\ FV(\mathtt{postp}(\mathtt{x} \mapsto N,M)) = & (FV(N) \smallsetminus \{x\}) \cup FV(M). \\ FV(\mathtt{postp}(M)) = & FV(M). \end{array}$$

Definition 1.3. Let $\|$ be a binary operation on terms (*parallel composition*) which is associative, commutative and has the empty term [] as the identity. Terms generated by (zero or more) applications of parallel composition are called *contexts*. Thus *contexts* are generated by the following grammar:

$$C \;:=\; R \mid (C\|R)$$

modulo the structural congruences

(i) $R_0\|(R_1\|R_2) \equiv (R_0\|R_1)\|R_2$,
(ii) $R_0\|R_1 \equiv R_1\|R_0$,
(iii) $(R_0\|[\,]) \equiv R_0$,
(iv) $C_0\|R\|C_1 \equiv C_0\|R'\|C_1$ if $R \equiv R'$.

Let $R_1\|\ldots\|R_k$ be a context, where all R_i are non-null , $i \le k$. Notice that the notation is well-defined by generalized associativity. We write $\mathcal{S}_{\overline{x}} : R_1\|\ldots\|R_k$ if all free variables occurring in $R_1,\ldots,R_k$ are in the list $\overline{x}$.

Computational contexts, the basic expressions of our calculus, are contexts satisfying some correctness conditions, that guarantee the identification of a context and rule out circular structures. Our "calculus of coroutines" is used here in a typed setting, where self referential structures are not needed.

Definition 1.4. An expression $\mathcal{S}_x : R_1\|\ldots\|R_k$ is a (correct) *computational context* if it satisfies the following axioms.

(1) Each term in the set $\{R_1,\ldots,R_k\}$ contains x and no other free variable.

(2) In every term of the form $\mathtt{postp}(\mathtt{y} \mapsto N, M)$ the term N contains a free variable y with $y \notin FV(M)$ and no other free variable.
(3) In every term $\mathtt{store}(\overline{P}, N_1, \ldots, N_n, \mathtt{y}_1, \ldots, \mathtt{y}_n, \mathtt{z}, M)$ the terms N_i are of the forms $[N]$ or $\mathtt{connect\ to}(R)$ for some N or R.
(4) Let $\mathbf{S} = \mathtt{store}(\overline{P}, N_1, \ldots, N_n, \mathtt{y}_1, \ldots, \mathtt{y}_n, \mathtt{z}, M)$ occur within a multiplicative computational context $\mathcal{S}_x$; write $\mathcal{S}_x^-$ for $\mathcal{S}_x$ without $\mathbf{S}$. Then $\{\overline{P}, N_1, \ldots, N_n\}$ is a computational context $\mathcal{S}_z$ for some free variable z with $z \neq x$. We say that $\mathcal{S}_z$ *occurs immediately within* $\mathbf{S}$.
(5) In a computational context $\mathcal{S}_x$ the nesting of p-terms of the form $\mathtt{store}$ within $\mathcal{S}_x$ has the structure of a rooted tree, with root $\mathcal{S}_x^-$ itself.

A computational context is said *multiplicative* if it does not contain $\mathtt{store}$ terms.

Remark 1.5. By axiom 1 the relevant components of a computational context are uniquely identified. Axiom 2 is analogue to the *acyclicity* condition in proof nets for linear logic. Axioms 4 and 5 induce a structure on context that corresponds to that of *boxes* in proof nets. Axiom 3 characterizes *exponential boxes* in our framework.

Definition 1.6 (α-equivalence)**.** Let $\mathcal{S}_x$ and $\mathcal{S}_{x'}$ be computational contexts. To define what it means that $\mathcal{S}_x$ and $\mathcal{S}_{x'}$ are α-equivalent, we need to define this property for sets of terms $\mathcal{S}_{\overline{x}}$ and $\mathcal{S}_{\overline{x}'}$ that may contain more than one free variable from the lists $\overline{x}$ and $\overline{x}'$, respectively; therefore they are not correct computational contexts. The definition is by induction on the number of terms occurring in $\mathcal{S}_{\overline{x}}$.
(1) If $\mathcal{S}_x = \{x\}$, then $\mathcal{S}_x \equiv \mathcal{S}_{x'}$ iff $\mathcal{S}_{x'} = \{x'\}$ and $x = x'$;
(2) If $\mathcal{S}_{\overline{x}} = \{\mathtt{x}(M)\}$, then $\mathcal{S}_{\overline{x}} \equiv \mathcal{S}_{\overline{x}'}$ iff $\mathcal{S}_{\overline{x}'} = \{\mathtt{x}(M')\}$ and $M \equiv M'$. A similar definition applies if $\mathcal{S}_x = \{\mathtt{connect\ to}(M)\}$ or $\{\mathtt{casel}(M)\}$ or $\{\mathtt{caser}(M)\}$ or $\{[M]\}$ or $\{\mathtt{postp}(P)\}$;
(3) If $\mathcal{S}_{\overline{x}} = \{M \wp N\}$, then $\mathcal{S}_{\overline{x}} \equiv \mathcal{S}_{\overline{x}'}$ iff $\mathcal{S}_{\overline{x}'} = \{M' \wp N'\}$ and $M \equiv M'$ and $N \equiv N'$. A similar definition applies if $\mathcal{S}_x = \{[M, N]\}$.
(4) Let $\mathcal{S}_{\overline{x}}$ be partitioned as

$$\mathcal{S}_{\overline{x}}^- \cup \ \{\mathtt{mkc}(M, \mathtt{y})\} \ \cup \ \mathcal{S}_{\overline{x}y}[y := \mathtt{y}(M)];$$

then $\mathcal{S}_{\overline{x}} \equiv \mathcal{S}_{\overline{x}'}$ iff $\mathcal{S}_{\overline{x}'}$ can be partitioned as $\mathcal{S}_{\overline{x}'}^- \cup \{\mathtt{mkc}(M', \mathtt{y})\} \cup \mathcal{S}_{\overline{x}'y'}[y' := \mathtt{y}(M')]$ and $\mathcal{S}_{\overline{x}}^- \cup \{M\} \equiv \mathcal{S}_{\overline{x}'}^- \cup \{M'\}$ and, moreover, for all variables v except for a finite number $\mathcal{S}_{\overline{x}y}[y := v] \equiv \mathcal{S}_{\overline{x}'y'}[y' := v]$.
(5) Let $\mathcal{S}_{\overline{x}}$ can be partitioned as

$$\mathcal{S}_{\overline{x}}^- \cup \{\mathtt{postpone}(\mathtt{y} \mapsto N, M)\} \cup \mathcal{S}_{\overline{x}y}[y := \mathtt{y}(M)];$$

then $\mathcal{S}_{\overline{x}} \equiv \mathcal{S}_{\overline{x}'}$ iff $\mathcal{S}_{\overline{x}'}$ can be partitioned as $\mathcal{S}_{x'}^- \cup (\{\mathtt{postpone}(y' \mapsto N', M')\} \cup \mathcal{S}_{\overline{x}'y'})[y' := \mathtt{y}'(x')]$ and $\mathcal{S}_{\overline{x}}^- \equiv \mathcal{S}_{\overline{x}'}^-$ and, moreover, for all variables v except for a finite number $(\mathcal{S}_{\overline{x}y} \cup \{N\})[y := v] \equiv (\mathcal{S}_{\overline{x}'y'} \cup \{N'\})[y' := v]$.
(6) Let $\mathcal{S}_{\overline{x}}$ be partitioned as $\mathcal{S}_{\overline{x}}^- \cup \{\mathbf{S}_1, \ldots, \mathbf{S}_k\}$ where for $i \leq k$, $\mathbf{S}_i$ is a $\mathtt{store}$ term with a set of terms $\mathcal{S}_{\overline{x}z_i}$ immediately inside it. Then $\mathcal{S}_{\overline{x}} \equiv \mathcal{S}_{\overline{x}'}$ iff $\mathcal{S}_{\overline{x}'}$ can be partitioned in a similar way as $\mathcal{S}_{\overline{x}'}^- \cup \{\mathbf{S}'_1, \ldots, \mathbf{S}'_k\}$ where for $i \leq k$ the set $\mathcal{S}_{\overline{x'}z'_i}$ occurs immediately inside $\mathbf{S}'{}_i$, for $i \leq k$ and, moreover
  (i) $\mathcal{S}_{\overline{x}}^- \equiv \mathcal{S}_{\overline{x}'}^-$
  (ii) for all $i \leq k$ and for all variables v except for a finite number $\mathcal{S}_{\overline{x}z_i}[z_i := v] \equiv \mathcal{S}_{\overline{x}'z'_i}[z'_i := v]$.

Remarks 1.7. (i) Consider the terms $\mathtt{make-coroutine}$, binary `postpone` and `store`. They are binders acting on a whole computational context, rather than on a delimited scope within a single term. We may call their action *remote binding*; it is expressed by a substitution of some m-term $\mathtt{x}(M)$ for the free variable x throughout a computational context $\mathcal{S}_x$. One could express remote binding by a more familiar notation, as in the λ-calculus or in the π-calculus; but then the scope of a binder would partition a context and parallel composition would not appear as a top-level operator only. In the typed case one could not assign a separate m-term to each formula in the succedent: at best, one could assign an "access port" to a unique term assigned to the whole sequent, as in the translations of linear logic into the π-calculus (see [10]). This would be against a main motivation of this calculus, namely, to give a "distributed term assignment" for a "multiple conclusion" co-intuitionistic deductive system. An application of our notation for "remote binding" is found in section 2 on a probabilistic interpretation of *subtraction* and *par*.

(ii) The p-terms binary `postpone` and `store` are also *local binders* of the free variable occurring in their argument. Indeed in a term $\mathtt{postpone}(\mathtt{y} \mapsto N, M)$ a free variable y occurring in N becomes locally bound; then y is replaced by $\mathtt{y}(M)$ in the computational context, to express remote binding. In a term $\mathtt{store}(\overline{P}, N_1, \ldots, N_n, \mathtt{y}_1, \ldots, \mathtt{y}_n, \mathtt{z}, M)$ a free variable z becomes bound; then a term $\mathtt{z}(N_i)$ occurs in the term $\mathtt{y}_i(\mathtt{z}(N_i))$, that replaces the stored N_i in the computational context. A more complete notation of our p-terms would be $\mathtt{postpone}(y \mapsto N, M)$ `with y for` y and $\mathtt{store}(\overline{P}, N_1, \ldots, N_n, \mathtt{y}_1, \ldots, \mathtt{y}_n, M)$ `with z for` z, explicitly establishing the connection between the locally bound variable and its corresponding unary function. For terms of the form `store` we are spared the more verbose notation by the *acyclicity axioms*, by which the locally bound variable is uniquely identified.

(iii) In the notation of p-terms of the form $\mathtt{postpone}(y \mapsto N, M)$ it is sometimes convenient to ignore the distinction between *local binding* and *remote binding* and treat the occurrences in N of the variable y as remotely bound; namely, we can write $\mathtt{postpone}(\mathtt{y} \mapsto N[y := \mathtt{y}(M)], M)$. Indeed the distinction between local and remote binding can be recovered from the structure of the terms. This notation allows us to adopt a tree-like notation for decorated co-intuitionistic Natural Deduction derivations analogue to that of Prawitz's Natural Deduction trees decorated with λ-terms. On the other hand, such a treatment seems unmanageable for p-terms of the form `store`; for them we retain a *box-like* notation.

Definition 1.8. Substitution of a term t for a free variable x in a term R is defined as follows:

$$
\begin{array}{rl}
x[x := M] = M, & y[x := M] = y \text{ if } x \neq y; \\
\mathtt{connect\ to}(R)[x := M] = & \mathtt{connect\ to}(R[x := M]) \\
\mathtt{postp}(N)[x := M] = & \mathtt{postp}(N[x := M]) \\
\mathtt{y}(N)[x := M] = & \mathtt{y}(N[x := M]); \\
(N_0 \wp N_1)[x := M] = & (N_0[x := M]\,)\wp(\,N_1[x := M]) \\
\mathtt{casel}(N)[x := M] = & \mathtt{casel}(N[x := M]), \\
\mathtt{caser}(N)[x := M] = & \mathtt{caser}(N[x := M]); \\
\mathtt{mkc}(N, \mathtt{y})[x := M] = & \mathtt{mkc}(N[x := M], \mathtt{y}), \\
\mathtt{store}(\overline{N}, \overline{\mathtt{y}}, \mathtt{z}, N)[x := M] = & \mathtt{store}(\overline{N}[x := M], \overline{\mathtt{y}}, \mathtt{z}, N[x := M) \\
\mathtt{postp}(\mathtt{y} \mapsto (N_1), N_0)[x := M] = & \mathtt{postp}(\mathtt{y} \mapsto (N_1[x := M]), N_0[x := M]). \\
[\,R\,][x := M] = \big[R[x := M]\big] & [R_0, R_1][x := M] = \big[R_0[x := M], R_1[x := M]\big]
\end{array}
$$

Proposition 1.9.

(i) *$\mathcal{S}_x = x$ and $\mathcal{S}_y = \texttt{postp}(y)$ are computational contexts.*

(ii) *Let $\mathcal{S}_x = R_1\| \dots \|R_m\|M$ and $\mathcal{S}_y = R_{m+1}\| \dots \|R_{m+n}$ be computational contexts where $x \neq y$. We write $\mathcal{S}_y[y := N]$ for $R_{m+1}[y := N]\| \dots \|R_{m+n}[y := N]$. Then*

$$\mathcal{S}'_x = R_1\| \dots \|R_m\|(\mathcal{S}_y[y := M])$$

is a computational context (substitution)*;*

(iii) *Let $\mathcal{S}_x$ and $\mathcal{S}_y$ be as in* (ii)*. Then $\mathcal{S}'_x = R_1\| \dots \|R_m\|\texttt{mkc}(M, \texttt{y})\|\mathcal{S}_y[y := \texttt{y}(M)]$ is a computational context* (make coroutine)*;*

(iv) *Let $\mathcal{S}_x = R_1\| \dots \|R_m$ and $\mathcal{S}_y = R_{m+1}\| \dots \|R_{m+n}$ be computational contexts. Then*

$$\mathcal{S}_z = \mathcal{S}_x[x := \texttt{casel}(z)]\|\mathcal{S}_y[y := \texttt{caser}(z)]$$

is a computational context (cases)*;*

(v) *Let $\mathcal{S}_x = R_1\| \dots \|R_m\|M$ be a computational context and let $x \neq y$. Then*

$$\mathcal{S}_y = \texttt{postp}(x \mapsto M, y)\|R_1\| \dots \|R_m$$

is a computational context (postpone)*;*

(vi) *Let $\mathcal{S}_x = R_1\| \dots \|M_i\|M_{i+1}\| \dots \|R_m$ be a computational context. Then*

$\mathcal{S}'_x = R_1\| \dots \|(M_i \wp M_{i+1})\| \dots \|R_m\}$ *and* $\mathcal{S}'_x = R_1\| \dots \|[M_i, M_{i+1}]\| \dots \|R_m\}$

are computational contexts, (par) *and* (contraction)*;*

(vii) *Let $\mathcal{S}_x = R_1\| \dots \|R_m$ be a computational context. Then*

$$\mathcal{S}'_x = R_1\| \dots \| \dots \|R_m\}\|\texttt{connect to}(R)$$

is a computational context (unit) *and* (weakening)*;*

(viii) *Let $\mathcal{S}_z = P_1\| \dots \|P_m\|N_1\| \dots \|N_n$ be a computational context where all terms N_i are of the form $[N]$ or $\texttt{connect to}(R)$ for some N or R. Then*

$$\mathcal{S}_x = \texttt{store}(P_1, \dots, P_m, N_1, \dots, N_n, \texttt{y}_1, \dots, \texttt{y}_n, \texttt{z}, x)$$

is a computational context (store).

Proof. In all cases the proposition is easily proved by checking that the resulting set of terms satisfies all the axioms in definition 1.4. □

The operation of β-reduction transforms a computational context $\mathcal{S}_x$ into a computational context $\mathcal{S}'_x$. It may be either *local*, affecting only the terms where the *redex* occurs, or a *global* operation with side-effects on parts of $\mathcal{S}_x$, mainly relabelling the terms that express binding by $\texttt{make} - \texttt{coroutine}$, $\texttt{postpone}$ or $\texttt{store}$.

Definition 1.10. β-reduction of a *redex $\mathcal{R}ed$* in a *computational context $\mathcal{S}_x$* is defined as follows.

(i) If $\mathcal{R}ed$ is a m-term N of the following form, then the reduction is local and consists of the rewriting $N \rightsquigarrow_\beta N'$ in $\mathcal{S}_x$ as follows:

$$\texttt{postp}(\texttt{connect to}(R)) \rightsquigarrow_\beta [].$$
$$\texttt{casel}\ (N_0 \wp N_1) \rightsquigarrow_\beta N_0; \qquad \texttt{caser}\ (N_0 \wp N_1) \rightsquigarrow_\beta N_1.$$

If the principal operator of $\mathcal{R}ed$ is a binary `postpone` or `store`, then the reduction is global and consists of the following rewriting. By the axioms in definiton 1.4, $\mathcal{R}ed$ occurs inside a computational context $\mathcal{S}_v$ in the rooted tree of nested p-terms of $\mathcal{S}_x$ and the rewriting takes place within $\mathcal{S}_v$.

(ii) If $\mathcal{R}ed$ has the form $\mathtt{postp}(\mathtt{z} \mapsto N, \mathtt{mkc}(M, \mathtt{y}))$, then $\mathcal{S}_v$ is partitioned as

$$\mathcal{S}_v \;=\; \mathcal{R}ed \cup \mathcal{S}_{vyz}\big[y := \mathtt{y}(M),\ z := \mathtt{z}(\mathtt{mkc}(M, \mathtt{y}))\big]$$

(a simultaneous substitution of $\mathtt{y}(M)$ *for* y *and of* $\mathtt{z}(\mathtt{mkc}(M,\mathtt{y}))$ *for* z *in* $\mathcal{S}_{vyz}$*).* Then a *reduction* of $\mathcal{R}ed$ transforms the computational context as follows:

$$\mathcal{S}_v \;=\; \mathcal{S}_{vyz}\big[y := N[z := M],\ z := M\big].$$

(iii) If $\mathcal{R}ed$ is a term with principal operator `store`, then $\mathcal{S}_v$ is partitioned as $\mathcal{S}_v^- \cup \mathbf{S}$ where

$$\mathbf{S} \;=\; \mathtt{store}(\overline{P}, N_1, \ldots, N_n, \mathtt{y}_1, \ldots, \mathtt{y}_n, \mathtt{z}, N)$$

Here N *is either* $[M]$ *or* $\mathtt{connect\ to}(R)$ *or* $[[M_0][M_1]]$.

By the axioms 1.4 $\overline{P}, N_1, \ldots, N_n$ form a computational context $\mathcal{S}_z$. Moreover $\mathtt{y}_1(\mathtt{z}([M]))$, …, $\mathtt{y}_n(\mathtt{z}([M]))$ may occur in $\mathcal{S}_v^-$ so we write $\mathcal{S}_v^-$ as $\mathcal{S}^-_{y_1\ldots y_n v}\big[y_1 := \mathtt{y}_1(\mathtt{z}(M)), \ldots, y_n := \mathtt{y}_n(\mathtt{z}(M))\big]$.

- If $N = [M]$, then

$$\mathcal{S}_v \;\rightsquigarrow_\beta \mathcal{S}^-_{y_1\ldots y_n v}\big[y_1 := N_1[z := M], \ldots, y_n := N_n[z := M]\big] \cup \{\overline{P}[z := M]$$

- If $N = \mathtt{connect\ to}(R)$, where R belongs to $\mathcal{S}_v^-$, then

$$\mathcal{S}_v \;\rightsquigarrow_\beta\; \mathcal{S}^-_{y_1\ldots y_n v}\big[y_1 := \mathtt{connect\ to}(R), \ldots, y_n := \mathtt{connect\ to}(R)\big]$$

- If $N = [M_0, M_1]$, then

$$\begin{aligned}\mathcal{S}_v \;\rightsquigarrow_\beta\quad & \mathcal{S}^-_{y_1\ldots y_n v}\Big[y_1 := \big[[\mathtt{y}_1(\mathtt{z}(M_0))], [\mathtt{y}_1(\mathtt{z}(M_1))]\big], \ldots \\ & \qquad \ldots, y_n := \big[[\mathtt{y}_n(\mathtt{z}(M_0))], [\mathtt{y}_n(\mathtt{z}(M_1))]\big]\Big] \\ & \cup\{\mathtt{store}(\overline{N}, \overline{\mathtt{y}}, \mathtt{z}, M_0), \mathtt{store}(\overline{N}, \overline{\mathtt{y}}, \mathtt{z}, M_1)\}\end{aligned}$$

Remarks 1.11. Here are some informal explanations about our calculus and notations.

(i) In our "distributed" model of computation a redex arises when an m-term M becomes part of another term; the rewriting of the redex has global effect. On the contrary, a p-term can only be sub-term of a p-term of the form `store`, but such nesting has only a structural significance; no redex is created in this way. A p-term P sits in the "control area", waiting to become active as a redex if a suitable m-term is substituted inside it. A term $\mathtt{y}(M)$ denotes a variable y that has become bound because of an operation in which the term M is active; in some sense $\mathtt{y}(M)$ is an input which is no longer accessible. Later in the computation such an input may become active again in a term R and ready for a substitution by a m-term N in a rewriting of the form $R[y := \mathtt{y}(M)] \rightsquigarrow R[y := N]$.

(ii) When a p-term $\mathtt{store}(\overline{P}, N_1, \ldots, N_n, \mathtt{y}_1, \ldots, \mathtt{y}_n, \mathtt{z}, N)$ is created, the m-terms $N_1, \ldots, N_n$ are set aside, together with the local p-terms $P_1, \ldots P_m$, within the new `store` terms sitting in the control area, but the "guarding terms" $\mathtt{y}_1, \ldots, \mathtt{y}_n$ associated with $N_1, \ldots, N_n$ remain active, since they are part of other terms in the context. Also the free variable z occurring in the terms P_j and N_i becomes inaccessible and is substituted with $\mathtt{z}(N)$. Only the term N is active in the storage operation. If $N = [M]$ then the computation is reactivated in

m-terms $N_i[z := M]$ and the guarding terms $\mathtt{y}_i$ are destroyed. If $N = [M_0, M_1]$ then both of the "guarding terms" and the $\mathtt{store}$ term are copied; if $N = \mathtt{connect\ to}(R)$ then the stored terms and the guarding terms are destroyed and replaced by pointers to the term R.

(iii) A term "*make-coroutine*" $\mathtt{mkc}(M, \mathtt{y})$ jumps from the term M to an input y which becomes inaccessible and thus is substituted by a term $\mathtt{y}(M)$ throughout the computational context. On the other hand a "*postpone*" term $\mathtt{postp}(\mathtt{z} \mapsto N, M')$ stores some *threads* of the computation from z to N (possibly a list of terms). As a consequence the input z becomes inaccessible and is substituted by a term $\mathtt{z}(M')$ throughout the computational context. If M' is $\mathtt{mkc}(M, \mathtt{y})$, then we can reactivate the stored threads N and free the variables z and y in the computational context. The variable z is substituted by M wherever it occurs, i.e., as $\xi[z := M]$. Moreover the threads N are connected to M through the substitution $N[z := M]$ and the variable y is substituted by $N[z := M]$. Here we see a calculus with binding and substitution implemented as "global effects" in a co-intutionistic calculus through terms originally conceived by Tristan Crolard [19] as extensions of the $\lambda\mu$ calculus.

The term assignment to **co-ILL** in sequent-style Natural Deduction notation is given in tables 2 and 3. Sequents are of the form

$$x : E \triangleright \overline{P} \mid \overline{M} : \Gamma$$

where

- the area of the succedent to the left of "|" may be called *"control area"*;
- $\overline{P} = P_1, \ldots, P_m$ is a sequence of p-terms;
- $\overline{M} : \Gamma$ stands for $M_1 : C_1, \ldots, M_n : C_n$, where $\Gamma = C_1, \ldots, C_n$;
- if $\overline{R} = R_1, \ldots, R_n$ then $\overline{R}[x := N]$ stands for $R_1[x := N], \ldots, R_n[x := N]$.
- We shall also use the abbreviation $\kappa : \Gamma$ for $\overline{P} \mid \overline{M} : \Gamma$. If also $\zeta : \Delta$ stands for $\overline{Q} \mid \overline{N} : \Delta$, then $\kappa : \Gamma, \zeta : \Delta$ stands for $\overline{P}, \overline{Q} \mid \overline{M} : \Gamma, \overline{N} : \Delta$.

1.3. Examples of multiplicative contexts.

1. The following computational context $\mathcal{S}_x$

$$\mathcal{S}_x = \mathtt{postp}(x) \parallel \mathtt{connect\ to}(\mathtt{postp}(x))$$

is correct. It is typed as follows:

$$\dfrac{\dfrac{\bot\text{-elim}}{x : \bot \triangleright \mathtt{postp}(x)}}{x : \bot \triangleright \mathtt{postp}(x) \mid \mathtt{connect\ to}(\mathtt{postp}(x)) : \bot} \ \bot \text{ intro}$$

This derivation may be regarded as the η-expansion of the axiom

$$x : \bot \triangleright x : \bot.$$

2. Given the computational contexts $\mathtt{S}_x = x \parallel \mathtt{connect\ to}(x)$ and $\mathcal{S}_y = \mathtt{postp}(y)$, we obtain a correct computational context by substitution of $\mathtt{connect\ to}(x)$ for y in $\mathcal{S}_y$:

$$\mathtt{S}'_x = x \parallel \mathtt{postp}(\mathtt{connect\ to}(x))$$

$\mathtt{S}'_x$ β-reduces to $\mathtt{S}''_x = x \parallel []$.

3. The following context $\mathcal{S}'_z$ is not correct: it violates Axiom 2 in definition 1.4.

$$\mathcal{S}'_z = \mathtt{postp}(y \mapsto [\mathtt{x}(y), \mathtt{x}(z)], \mathtt{mkc}(z, \mathtt{x})) \parallel \mathtt{mkc}(\mathtt{y}(t), \mathtt{x})$$

$$\begin{array}{c}\textit{axiom}\\ x : A \triangleright\ \mid x : A\end{array} \qquad \begin{array}{c}\textit{cut}\\ \dfrac{v : E \triangleright \overline{P} \mid \overline{M} : \Gamma, M : A \qquad x : A \triangleright \overline{Q} \mid \overline{N} : \Delta}{x : E \triangleright \overline{P}, \overline{Q}'[x := M] \mid \overline{M} : \Gamma, \overline{N}[x := M] : \Delta}\end{array}$$

$$\begin{array}{c}\bot\text{-intro}\\ \dfrac{x : E \triangleright \overline{P} \mid \overline{M} : \Gamma \qquad (R \in \overline{P} \cup \overline{M})}{x : E \triangleright \overline{P} \mid \overline{M} : \Gamma, \texttt{connect to}(R) : \bot}\end{array} \qquad \begin{array}{c}\bot\text{-elim}\\ x : \bot \triangleright \texttt{postp}(x) \mid\end{array}$$

We write $\overline{Q}'$ for $\overline{Q}[x := \texttt{x}(M)]$ and $\overline{N}'$ for $\overline{N}[x := \texttt{x}(M)]$

$$\begin{array}{c}\setminus\text{-intro}\\ \dfrac{v : E \triangleright \overline{P} \mid \overline{M} : \Gamma, M : C \qquad x : D \triangleright \overline{Q} \mid \overline{N} : \Delta}{v : E \triangleright \overline{P},\ \overline{Q}' \mid \overline{M} : \Gamma, \overline{N}' : \Delta, \texttt{mkc}(M, \texttt{x}) : C \setminus D}\end{array}$$

$$\begin{array}{c}\setminus\text{-elim}\\ \dfrac{z : E \triangleright \overline{P} \mid \overline{M} : \Gamma, M : C \setminus D \qquad x : C \triangleright \overline{Q} \mid N : D, \overline{N} : \Delta}{z : E \triangleright \overline{P},\ \overline{Q}', \texttt{postp}(x \mapsto M, z) \mid \overline{M} : \Gamma, \overline{N}' : \Delta}\end{array}$$

$$\begin{array}{c}\wp\text{-intro}\\ \dfrac{x : E \triangleright \overline{P} \mid \overline{M} : \Gamma, M_0 : C_0,\ M_1 : C_1}{x : E \triangleright \overline{P} \mid \overline{M} : \Gamma, M_0 \wp M_1 : C_0 \wp C_1}\end{array}$$

$$\begin{array}{c}\wp\text{-elim}\\ \dfrac{\begin{array}{l} z : E \triangleright \overline{Q} \mid \overline{N} : \Delta, N : C_0 \wp C_1 \\ \quad \vdots \qquad\qquad x_0 : C_0 \triangleright \overline{P}_0 \mid \overline{M}_0 : \Gamma_0 \qquad x_1 : C_1 \triangleright \overline{P}_1 \mid \overline{M}_1 : \Gamma_1\end{array}}{z : E \triangleright \overline{Q},\ \overline{P}'_0,\ \overline{P}'_1 \mid \overline{N} : \Delta,\ \overline{M}'_0 : \Gamma_0,\ \overline{M}'_1 : \Gamma_1}\\ \text{where } \overline{P}'_0 = \overline{P}[x_0 := \texttt{casel}(z)], \overline{M}'_0 = \overline{M}_0[x_0 := \texttt{casel}(z)],\\ \overline{P}'_1 = \overline{P}_1[x_0 := \texttt{caser}(z)], \overline{M}'_1 = \overline{M}_1[x_0 := \texttt{caser}(z)]\end{array}$$

Table 2: Decorated Natural Deduction for **multiplicative co-ILL**

Here we write t for $\texttt{mkc}(z, \texttt{x})$. The following context $\mathcal{S}_z$ is correct

$$\mathcal{S}_z = \texttt{postp}(y \mapsto \texttt{x}(y), \texttt{mkc}(z, \texttt{x})) \parallel \texttt{mkc}(\texttt{y}(t), \texttt{x}) \parallel \texttt{x}(z)$$

and is typed as follows:

$$\dfrac{\dfrac{z : C \triangleright z : C \qquad x : C \triangleright x : C}{z : C \triangleright \texttt{mkc}(z, \texttt{x}) : C \setminus C, \texttt{x}(z) : C}\setminus\text{-I} \qquad \dfrac{y : C \triangleright y : C \qquad x : C \triangleright x : C}{y : C \triangleright \texttt{mkc}(y, \texttt{x}) : C \setminus C, \texttt{x}(y) : C}\setminus\text{-I}}{z : C \triangleright \texttt{postp}(y \mapsto \texttt{x}(y), \texttt{mkc}(z, \texttt{x})) \mid \texttt{mkc}(\texttt{y}(t), \texttt{x}) : C \setminus C, \texttt{x}(z) : C}\setminus\text{-E}$$

One could check that the above derivation is dual to the derivation

$$f : A \to A, x : A\ \triangleright\ (\lambda x. f x) x : A$$

dereliction
$$\frac{x : E \rhd \kappa : \Gamma, M : C}{x : E \rhd \kappa : \Gamma, [\,M\,] :?C}$$

weakening
$$\frac{x : E \rhd \kappa : \Gamma}{x : E \rhd \kappa : \Gamma, \mathtt{connect\ to}(R) :?C}$$
where $R \in \kappa$.

contraction
$$\frac{x : E \rhd \kappa : \Gamma, M :?C, N :?C}{x : E \rhd \kappa : \Gamma, [\,M, N\,] :?C}$$

storage
$$\frac{z : E \rhd \overline{P} \mid \overline{M} : \Gamma, M :?C \qquad x : C \rhd \overline{Q} \mid \overline{N} : ?\Delta}{z : E \rhd \overline{P}, \mathtt{store}(\overline{Q}, \overline{N}, \overline{\mathtt{y}}, \mathtt{x}, N) \mid \overline{M} : \Gamma, \overline{[\mathtt{y}(\mathtt{x}(N))]} : ?\Delta}$$
where $\overline{N} = N_1, \ldots, N_m$ and $\overline{\mathtt{y}} = \mathtt{y}_1, \ldots, \mathtt{y}_m$
and $\overline{[\mathtt{y}(\mathtt{x}(N))]} = \mathtt{y}_1(\mathtt{x}(N)), \ldots, \mathtt{y}_m(\mathtt{x}(N))$

Table 3: Decorated Natural Deduction for **co-ILL exponential**

in the simply typed λ-calculus. A more substantial example of computation in our typed dual linear calculus is given in Appendix, Section A.

2. Motivations: a probabilistic interpretation.

In our setting co-intuitionistic logic admits a simple *probabilistic interpretation* which fits well in the view of co-intuitionism as a logic of hypotheses. Indeed if co-intuitionistic logic is about the justification properties of hypotheses, then the co-intuitionistic consequence relation must be about the *preservation of probability assignments* from the premise to the conclusions; a term calculus for such a logic must allow us to compute probabilities and verify the preservation property. We sketch our result only for the *multiplicative linear* fragment, i.e., for typing derivations in the linear system with *subtraction* and *par* only.

We find it easier to state our result for a *decorated sequent calculus* for multiplicative co-intuitionistic linear logic. Such a calculus is equivalent to our system of decorated sequent-style natural deduction; in fact its *right* rules coincide with the *introduction rules* and using *cut* the *left* rules given below are shown to be equivalent to the *elimination rules.*

Definition 2.1. To the judgements of linear co-intuitionistic logic we assign *events in a probabilistic setting*. We write $\overline{\mathbf{C}}$, $\mathbf{C} \cap \mathbf{D}$ and $\mathbf{C} \cup \mathbf{D}$ for complementation, intersection and union between events; there is an impossible event $\emptyset$ and a certain event $\overline{\emptyset}$. A *probabilistic assignment* is a map $(\ )^P : \mathbf{judg} \to \mathbf{events}$ satisfying the following constraints:

$$\begin{array}{rl}
\text{if } (M : C)^P = \mathbf{C} \text{ and } (x : D)^P = \mathbf{D}, \text{ then} & (\mathtt{mkc}(M, x) : C \smallsetminus D)^P = \mathbf{C} \cap \overline{\mathbf{D}}; \\
\text{if } (M_0 : C_0)^P = \mathbf{C}_0 \text{ and } (M_1 : C_1)^P = \mathbf{C}_1, \text{ then} & (M_0 \wp M_1 : C_0 \wp C_1)^P = \mathbf{C}_0 \cup \mathbf{C}_1; \\
\text{if } (M : C_0 \wp C_1)^P = \mathbf{C} \text{ then} & (\mathtt{casel}(M) : C_0)^P \subseteq \mathbf{C} \\
\text{and} & (\mathtt{caser}(M) : C_1)^P \subseteq \mathbf{C}; \\
(\mathtt{postp}(M))^P = \emptyset \text{ and} & (\mathtt{postp}(x \mapsto M, N))^P = \emptyset.
\end{array}$$

Proposition 2.2. (**Decomposition property**) *Let d be a sequent calculus derivation of $x : H \triangleright t_1 : C_1, \ldots, t_n : C_n$ and let $(\ )^P : \mathbf{judg} \to \mathbf{events}$ be an assignment to the judgements in d satisfying the constraints of Definition 2.1, and suppose $\mathbf{H}, \mathbf{C}_1, \ldots, \mathbf{C}_n$ are assigned to $x : H \triangleright t_1 : C_1, \ldots, t_n : C_n$. There are pairwise disjoint events $\mathbf{C}'_1 \subseteq \mathbf{C}_1, \ldots, \mathbf{C}'_n \subseteq \mathbf{C}_n$ such that*

$$\left(\mathbf{C}'_1 \cup \cdots \cup \mathbf{C}'_n\right) \cap \mathbf{H} = \mathbf{H}.$$

The events $\mathbf{C}'_1, \ldots, \mathbf{C}'_n$ can be constructed from the dependencies of the terms $t_1, \ldots, t_n$.

Example 2.3. Consider the following very simple example:

$$\frac{x : C \triangleright x : C \qquad y : D \triangleright y : D}{x : C \triangleright \mathtt{mkc}(x, y) : C \smallsetminus D, \mathtt{y}(x) : D}$$

If the event $\mathbf{C}$ is assigned to $x : C$, and $\mathbf{D}$ is assigned to $y : D$, but also to $\mathtt{y}(x) : D$, then we have the following inclusions

$$\frac{\mathbf{C} \subseteq \mathbf{C} \qquad \mathbf{D} \subseteq \mathbf{D}}{\mathbf{C} \subseteq (\mathbf{C} \cap \overline{\mathbf{D}}) \cup \mathbf{D}}$$

We have equality only by assigning $\mathbf{C} \cap \mathbf{D}$ to $\mathtt{y}(x) : D$, as suggested by the dependency of $\mathtt{y}(x) : D$ on $x : C$.

$$\frac{\mathbf{C} = \mathbf{C} \qquad \mathbf{D} = \mathbf{D}}{\mathbf{C} = (\mathbf{C} \cap \overline{\mathbf{D}}) \cup (\mathbf{C} \cap \mathbf{D})}$$

Proof. By induction on d. The case of assumptions $x : H \triangleright x : H$ is obvious and that of cut is immediate from the inductive hypothesis.

Subtraction right: by inductive hypothesis we may assume that the assignments to the premises $v : E \triangleright \kappa : \Gamma, M : C$ and $x : D \triangleright \zeta : \Delta$ satisfy the conditions of the lemma, i.e., that $((\bigcup \mathbf{\Gamma}) \cup \mathbf{C}) \cap \mathbf{E} = \mathbf{E}$ and $(\bigcup \mathbf{\Delta}) \cap \mathbf{D} = \mathbf{D}$, where the events in $\mathbf{\Gamma}$ are pairwise disjoint and so are those in $\mathbf{\Delta}$. In the term assignment to the conclusion

$$v : E \triangleright \kappa : \Gamma, \mathtt{mkc}(M, \mathtt{x}) : C \smallsetminus D, \zeta[x := \mathtt{x}(M)] : \Delta$$

in all terms $\zeta : \Delta$ the variable $x : D$ has been replaced by $\mathtt{x}(M)$, where $M : C$. We interpret this fact as *the instruction that in the context of the conclusion the disjoint events $\mathbf{D'}_j \subseteq \mathbf{D}_j$ must be* $\mathbf{D}_j \cap (\mathbf{C} \cap \mathbf{D})$ for each $D_j \in \Delta$. Then

$$\mathbf{C} = (\mathbf{C} \cap \overline{\mathbf{D}}) \cup (\mathbf{C} \cap \mathbf{D}) = (\mathbf{C} \cap \overline{\mathbf{D}}) \cup \left(\mathbf{C} \cap \mathbf{D} \cap \bigcup \mathbf{\Delta}\right) = (\mathbf{C} \cap \overline{\mathbf{D}}) \cup (\cup_j \mathbf{D'}_j)$$

hence $\mathbf{C} \cap \mathbf{E} = [(\mathbf{C} \cap \overline{\mathbf{D}}) \cap \mathbf{E}] \cup [(\cup_\mathbf{j} \mathbf{D'_j}) \cap \mathbf{E}]$. Thus

$$((\cup_\mathbf{i} \mathbf{C_i}) \cup (\mathbf{C} \cap \overline{\mathbf{D}}) \cup (\cup_\mathbf{j} \mathbf{D'_j})) \cap \mathbf{E} = \mathbf{E}.$$

$$\boxed{\begin{array}{c} \textit{subtraction}\text{-L} \\ \dfrac{x : C \triangleright \overline{P} \mid M : D, \overline{M} : \Delta}{z : C \smallsetminus D \triangleright\ \overline{P}[x := \mathtt{x}(z)], \mathtt{postp}(x \mapsto M, z) \mid \overline{M}[x := \mathtt{x}(z)] : \Delta} \end{array}}$$

Subtraction left: suppose that by inductive hypothesis we have an assignment to the premise $x : C \triangleright \overline{P} \mid M : D, \overline{M} : \Delta$, thus $\mathbf{D}$ and $\mathbf{D_i} \in \mathbf{\Delta}$ are pairwise disjoint and $(\mathbf{D} \cup (\bigcup \mathbf{\Delta})) \cap \mathbf{C} = \mathbf{C}$, then obviously $(\bigcup \mathbf{\Delta}) \cap \mathbf{C} \cap \overline{\mathbf{D}} = \mathbf{C} \cap \overline{\mathbf{D}}$. Clearly p-terms have empty assignment.

$$\begin{array}{c} \textit{par}\text{-L} \\ \dfrac{x_0 : C_0 \triangleright \overline{P}_0 \mid \overline{M}_0 : \Gamma_0 \qquad x_1 : C_1 \triangleright \overline{P}_1 \mid \overline{M}_1 : \Gamma_1}{z : C_0 \wp C_1 \triangleright \overline{P}'_0,\ \overline{P}'_1 \mid \overline{M}'_0 : \Gamma_0,\ \overline{M}'_1 : \Gamma_1} \\ \text{where } \overline{P}'_0 = \overline{P}[x_0 := \mathtt{casel}(z)], \overline{M}'_0 = \overline{M}_0[x_0 := \mathtt{casel}(z)] \\ \overline{P}'_1 = \overline{P}_1[x_0 := \mathtt{caser}(z)], \overline{M}'_1 = \overline{M}_1[x_0 := \mathtt{caser}(z)] \end{array}$$

In the case of *par right* there is nothing to prove; in the case of *par left* we only need to make sure that the events $\mathbf{C}_0$ and $\mathbf{C}_1$ assigned to C_0 and C_1 are disjoint. If not, assign $\mathbf{C}_0$ to C_0 and $\mathbf{C}_1 \cap \overline{\mathbf{C_0}}$ to C_1, or alternatively $\mathbf{C}_0 \cap \overline{\mathbf{C}}_1$ to C_0 and $\mathbf{C}_1$ to C_1. □

Remarks 2.4. (i) Since events are assigned to expressions $t : X$ rather than to formulas X, if $t : X$ and $u : X$ occur in the same context then $(t : X)^P$ and $(u : X)^P$ are events that may or may not be disjoint of each other.

(ii) The *common sense* reading of the co-intuitionistic consequence relation $H \vdash C_1, \ldots, C_n$ is as follows.

> *If it is justified to make the hypothesis H, then it is justified to make the hypotheses $C_1, \ldots, C_n$.*

The probabilistic interpretation gives a mathematical counterpart of this reading.

> *If the probability of the event $\mathbf{H}$ assigned to $x : H$ is greater than zero, then the conditional probability of the union of the events $\mathbf{C}_1, \ldots, \mathbf{C}_n$ assigned to $t_1 : C_1, \ldots, t_n : C_n$, given $\mathbf{H}$ is equal to one.*

The indexing of the terms $t_1, \ldots, t_n$ can be regarded as computational devices for verifying such an interpretation in the sense of the Decomposition Property.

3. Categorical Semantics

We recall the definition of a symmetric monoidal category.

Definition 3.1. A *symmetric monoidal category (SMC)* $(\mathbb{C}, \bullet, 1, \alpha, \lambda, \rho, \gamma)$, is a category $\mathbb{C}$ equipped with a bifunctor $\bullet : \mathbb{C} \times \mathbb{C} \to \mathbb{C}$ with a neutral element 1 and natural isomorphisms α, λ, ρ and γ:

(1) $\alpha_{A,B,C,} : A \bullet (B \bullet C) \xrightarrow{\sim} (A \bullet B) \bullet C$;
(2) $\lambda_A : 1 \bullet A \xrightarrow{\sim} A$
(3) $\rho_A : A \bullet 1 \xrightarrow{\sim} A$
(4) $\gamma_{A,B} : A \bullet B \xrightarrow{\sim} B \bullet A$.

which satisfy the following coherence diagrams.

$$\begin{tikzcd} A \bullet (B \bullet (C \bullet D)) \arrow[r, "\alpha_{A,B,C\bullet D}"] \arrow[d, "id_A \bullet \alpha_{B,C,D}"'] & (A \bullet B) \bullet (C \bullet D) \arrow[r, "\alpha_{A\bullet B,C,D}"] & (((A \bullet B) \bullet C) \bullet D \\ A \bullet ((B \bullet C) \bullet D) \arrow[rr, "\alpha_{A,B\bullet C,D}"'] & & (A \bullet (B \bullet C)) \bullet D \arrow[u, "\alpha_{A,B,C} \bullet id_D"'] \end{tikzcd}$$

$$\begin{tikzcd}
(A \bullet B) \bullet C \arrow[r, "\alpha^{-1}_{A,B,C}"] \arrow[d, "\gamma_{A,B} \bullet id_C"'] & A \bullet (B \bullet C) \arrow[r, "\gamma_{A,B \bullet C}"] & (B \bullet C) \bullet A \arrow[d, "\alpha^{-1}_{B,C,A}"] \\
(B \bullet A) \bullet C \arrow[r, "\alpha^{-1}_{B,A,C}"'] & B \bullet (A \bullet C) \arrow[r, "id_B \bullet \gamma_{A,C}"'] & B \bullet (C \bullet A)
\end{tikzcd}$$

$$\begin{tikzcd}
A \bullet (1 \bullet B) \arrow[rr, "\alpha_{A,1,B}"] \arrow[dr, "id_A \bullet \lambda_B"'] & & (A \bullet 1) \bullet B \arrow[dl, "\rho_A \bullet id_B"'] \\
& A \bullet B &
\end{tikzcd}
\qquad
\begin{tikzcd}
A \bullet B \arrow[r, "\gamma_{A,B}"] \arrow[dr, "id_{A \bullet B}"'] & B \bullet A \arrow[d, "\gamma_{B,A}"] \\
& A \bullet B
\end{tikzcd}$$

$$\begin{tikzcd}
A \bullet 1 \arrow[rr, "\gamma_{A,1}"] \arrow[dr, "\rho_a"'] & & 1 \bullet A \arrow[dl, "\lambda_A"] \\
& A &
\end{tikzcd}$$

The following equality is also required to hold: $\lambda_1 = \rho_1 : 1 \bullet 1 \to 1$.

Given a *signature* Sg, consisting of a collection of types σ_i and a collection of *sorted function symbols* $f_j : \sigma_1, \ldots, \sigma_n \to \tau$ and given a SMC $(\mathbb{C}, \bullet, 1, \alpha, \lambda, \rho, \gamma)$, a *structure* $\mathcal{M}$ for Sg is an assignment of an object $\llbracket \sigma \rrbracket$ of $\mathbb{C}$ for each type σ and of a morphism $\llbracket f \rrbracket : \llbracket \sigma_1 \rrbracket \bullet \ldots \bullet \llbracket \sigma_n \rrbracket \to \llbracket \tau \rrbracket$ for each function $f : \sigma_1, \ldots, \sigma_n \to \tau$ of Sg.

The types of terms in context $\Delta = [M_1 : \sigma_1, \ldots, M_n : \sigma_n]$ are interpreted as $\llbracket \sigma_1, \sigma_2, \ldots, \sigma_n \rrbracket = (\ldots(\llbracket \sigma_1 \rrbracket \bullet \llbracket \sigma_2 \rrbracket) \ldots) \bullet \llbracket \sigma_n \rrbracket$; left associativity is also intended for concatenations of type sequences Γ, Δ. Thus we need the following functions $\texttt{Split}(\Gamma, \Delta) : \llbracket \Gamma, \Delta \rrbracket \to \llbracket \Gamma \rrbracket \bullet \llbracket \Delta \rrbracket$

$$\texttt{Split}(\Gamma, \Delta) \begin{cases} \lambda^{-1}_{\Delta} & \text{if } \Gamma = \emptyset \\ \rho^{-1}_{\Gamma} & \text{if } \Delta = \emptyset \\ id_{\Gamma \bullet A} & \text{if } \Delta = A \\ \texttt{Split}(\Gamma, \Delta') \bullet id_A; \alpha^{-1}_{\Gamma,\Delta',A} & \text{if } \Delta = \Delta', A. \end{cases}$$

and $\texttt{Join}(\Gamma, \Delta) : \llbracket \Gamma \rrbracket \bullet \llbracket \Delta \rrbracket \to \llbracket \Gamma, \Delta \rrbracket$

$$\texttt{Join}(\Gamma, \Delta) \begin{cases} \lambda_{\Delta} & \text{if } \Gamma = \emptyset \\ \rho_{\Gamma} & \text{if } \Delta = \emptyset \\ id_{\Gamma \bullet A} & \text{if } \Delta = A \\ \alpha_{\Gamma,\Delta',A}; \texttt{Join}(\Gamma, \Delta') \bullet id_A; & \text{if } \Delta = \Delta', A. \end{cases}$$

Similarly we have $\texttt{Split}_n(\Gamma_1, \ldots, \Gamma_n) : \llbracket \Gamma_1, \ldots, \Gamma_n \rrbracket \to \llbracket \Gamma_1 \rrbracket \bullet \ldots \bullet \llbracket \Gamma_n \rrbracket$ and $\texttt{Join}_n$.

The semantics of terms in context is then specified by induction on terms:

$$\llbracket x : \sigma \rhd x : \sigma \rrbracket =_{df} id_{\llbracket \sigma \rrbracket}$$

$$\llbracket x : \sigma \rhd f(M_1, \ldots, M_n) : \tau \rrbracket =_{df} \llbracket x : \sigma \rhd M_1 : \sigma_1 \rrbracket \bullet \ldots \bullet \llbracket x : \sigma \rhd M_n : \sigma_n \rrbracket; \llbracket f \rrbracket$$

The *Exchange right* rule is handled implicitly by symmetry in the model (see [13], Lemma 13):

$$\llbracket x : \sigma \rhd \overline{M} : \Gamma, N : \tau, M : \sigma \rrbracket = \llbracket x : \sigma \rhd \overline{M} : \Gamma, M : \sigma, N : \tau \rrbracket; \alpha^{-1}_{\Gamma,\sigma,\tau}; id_\Gamma \bullet \gamma_{\sigma,\tau}; \alpha_{\Gamma,\tau,\sigma}$$

(Notice that, as a notational convenience, we are sometimes reusing the names of types to denote their interpretation as objects). One then proves by induction on the type derivation that substitution in the term calculus corresponds to composition in the category ([13], Lemma 13):

Lemma 3.2. *Let $x : \sigma \triangleright \overline{M} : \Gamma, M : \tau$ and $y : \tau \triangleright \overline{N} : \Delta$ be derivable terms in context, then*

$$\begin{aligned} &[\![x : \sigma \triangleright \overline{M} : \Gamma, \overline{N}[y := M]\,\Delta]\!] = \\ &\qquad [\![x : \sigma \triangleright \overline{M} : \Gamma, M : \tau]\!]; \quad id_\Gamma \bullet [\![y : \tau \triangleright \overline{N} : \Delta]\!]; \mathtt{Join}(\Gamma, \Delta) \end{aligned}$$

Let M be a structure for a signature Sg in a SMC $\mathbb{C}$. Given an equation in context for Sg

$$x : \sigma \triangleright \overline{M} : \Gamma, M = N : \tau$$

we say that the structure *satisfies* the equation if the morphisms assigned to $x : \sigma \triangleright \overline{M} : \Gamma, M : \tau$ and to $x : \sigma \triangleright \overline{M} : \Gamma, N : \tau$ are equal. Then given an algebraic theory $Th = (Sg, Ax)$, a structure $\mathcal{M}$ for Sg is a *model* for Th if it satisfies all the axioms in Ax.

Lemma 3.3. *Let $\mathbb{C}$ be a SMC, Th an algebraic theory and $\mathcal{M}$ a model of Th in $\mathbb{C}$. Then $\mathcal{M}$ satisfies the equations in context in Table 4.*

$$\frac{z : D \triangleright \overline{M} : \Gamma, M : \sigma}{z : D \triangleright \overline{M} : \Gamma, M = M : \sigma}\ \textit{Refl} \qquad \frac{z : D \triangleright \overline{M} : \Gamma, M = N : \sigma}{z : D \triangleright \overline{M} : \Gamma, N = M : \sigma}\ \textit{Symm}$$

$$\frac{z : D \triangleright \overline{M} : \Gamma, M_0 = M_1 : \sigma \qquad z : D \triangleright \overline{M} : \Gamma, M_1 = M_2 : \sigma}{z : D \triangleright \overline{M} : \Gamma, M_0 = M_2 : \sigma}\ \textit{Trans}$$

$$\frac{z : D \triangleright \overline{M} : \Gamma, M_0 = M_1 : \sigma \qquad x : \sigma \triangleright \overline{N} : \Delta, N_0 = N_1 : \tau}{z : D \triangleright \overline{M} : \Gamma, \overline{N}[x := M_0] = \overline{N}[x := M_1] : \Delta, N_0[x := M_0] = N_1[x := M_1] : \tau}\ \textit{Subst}$$

Table 4:

3.1. **Analysis of the rules of co-intuitionistic linear logic.** We work with symmetric monoidal categories satisfying *the dual condition to closure*, namely, with monoidal categories of the form $(\mathbb{C}, \bullet, \smallsetminus, 1, \alpha, \lambda, \rho, \gamma)$ such that for all objects A in $\mathbb{C}$, the functor $A \bullet -$ has a *left* adjoint $- \smallsetminus A$. We call such monoidal categories *left closed*.

Given a symmetric monoidal category $\mathbb{C}$, its opposite is also symmetric monoidal. If $\mathbb{C}$ is closed, i.e., $A \bullet -$ has a right adjoint, then certainly $\mathbb{C}^{op}$ has a left adjoint. It is well-known that in a symmetric monoidal closed category $\mathbb{C}$ we can construct a model of *multiplicative intuitionistic linear logic*, hence it is certainly not surprising that a model of multiplicative co-intuitionistic linear logic may be constructed in $\mathbb{C}^{op}$. The point of the exercise that follows, however, is to check that the dual linear calculus given above in Section 1.2 is indeed suitable for the construction of such an interpretation. We consider the rules for each connective in turn.

3.2. **Linear disjunction Par.** 3.2.1. *Par introduction.* The introduction rule for Par is of the form

$$\frac{x : D \triangleright \kappa : \Theta, M_0 : A, M_1 : B}{x : D \triangleright \kappa : \Theta, M_0 \wp M_1 : A \wp B} \wp \, \mathrm{I}$$

This suggests an operation on Hom-sets of the form

$$\Phi_{D,\Theta} : \mathbb{C}(D, \Theta \bullet A \bullet B) \to \mathbb{C}(D, \Theta \bullet A \wp B)$$

natural in Θ and D. Given $e : D \to \Theta \bullet A \bullet B$, $d : D' \to D$ and $h : \Theta \to \Theta'$, naturality yields

$$\Phi_{D',\Theta'}(d; e; h \bullet id_A \bullet id_B) = d; \Phi_{D,\Theta}(e); h \bullet id_{A \wp B}$$

In particular, letting $e = id_\Theta \bullet id_A \bullet id_B$, $d : D \to \Theta \bullet A \bullet B$ and $h = id_\Theta$ we have

$$\Phi_{D,\Theta}(d) = d; \Phi_\Theta(id_\Theta \bullet id_A \bullet id_B)$$

By functoriality of $\bullet$ we have $id_A \bullet id_B = id_{A \bullet B}$. Hence, writing **PAR** for $\Phi_\Theta(id_\Theta \bullet id_{A \bullet B})$ we have $\Phi_{D,\Theta}(d) = d; \mathbf{PAR}$. We define

$$[\![x : D \triangleright \kappa : \Theta, M \wp N : A \wp B]\!] =_{df} [\![x : D \triangleright \kappa : \Theta, M : A, N : B]\!]; \mathbf{PAR}.$$

3.2.2. *Par elimination.* The Par elimination rule has the form

$$\frac{z : D \triangleright \kappa : \Upsilon, N : A \wp B \qquad x : A \triangleright \zeta : \Gamma \qquad y : B \triangleright \xi : \Delta}{z : D \triangleright \kappa : \Upsilon, \zeta[x := \mathtt{casel}\, N] : \Gamma, \xi[y := \mathtt{caser}\, N] : \Delta} \wp \, \mathrm{E}$$

This suggests an operation on Hom-sets of the form

$$\Psi_{D,\Upsilon,\Gamma,\Delta} : \mathbb{C}(D, \Upsilon \bullet A \wp B) \times \mathbb{C}(A, \Gamma) \times \mathbb{C}(B, \Delta) \to \mathbb{C}(D, \Upsilon \bullet \Gamma \bullet \Delta)$$

natural in $D, \Upsilon, \Gamma, \Delta$. Given morphisms $g : D \to \Upsilon \bullet A \wp B$, $e : A \to \Gamma$ and $f : B \to \Delta$ and also $a : D' \to D$, $p : \Upsilon \to \Upsilon'$, $c : \Gamma \to \Gamma'$ and $d : \Delta \to \Delta'$ naturality yields

$$\begin{aligned} \Psi_{D',\Upsilon',\Gamma',\Delta'}\big((a; g; p \bullet id_{A \wp B}), (e; c), \quad (f; d)\big) = \\ a; \Psi_{D,\Upsilon,\Gamma,\Delta}(g, e, f); \quad p \bullet c \bullet d; \mathtt{Join}(\Upsilon', \Gamma', \Delta'). \end{aligned}$$

In particular, setting $e = id_A$, $f = id_B$ and also $a = id_D$, $p = id_\Upsilon$, we get

$$\Psi_{D,\Upsilon,\Gamma,\Delta}(g, c, d) = \Psi_{D,\Upsilon,\Gamma,\Delta}(g, id_A, id_B); id_\Upsilon \bullet c \bullet d; \mathtt{Join}(\Upsilon, \Gamma, \Delta)$$

Writing $(g)^*$ for $\Psi_{D,\Upsilon}(g, id_A, id_B)$ we define

$$\begin{aligned} &[\![z : D \triangleright \kappa : \Upsilon, \zeta[x := \mathtt{casel}\ N] : \Gamma, \xi[y := \mathtt{caser}\, N] : \Delta]\!] =_{df} \\ &[\![z : D \triangleright \kappa : \Upsilon, N : A \wp B]\!]^*; id_\Upsilon \bullet [\![x : A \triangleright \zeta : \Gamma]\!] \bullet [\![y : B \triangleright \xi : \Delta]\!]; \mathtt{Join}(\Upsilon, \Gamma, \Delta). \end{aligned}$$

3.2.3. *Equations in context.* We have equations in context of the form

$$\boxed{\begin{array}{c} \wp - \beta \ \textbf{rules:} \\ \dfrac{z : D \triangleright \kappa : \Theta, M_0 : A, M_1 : B \qquad x : A \triangleright \zeta : \Gamma \quad y : B \triangleright \xi : \Delta}{z : D \triangleright \kappa : \Theta, \zeta[x := \mathtt{casel}\, (M_0 \wp M_1)] = \zeta[x := M_0] : \Gamma} \\ \\ \dfrac{z : D \triangleright \kappa : \Theta, M_0 : A, M_1 : B \qquad x : A \triangleright \zeta : \Gamma \quad y : B \triangleright \xi : \Delta}{z : D \triangleright \kappa : \Theta, \xi[y := \mathtt{caser}\, (M_0 \wp M_1)] = \xi[y := M_1] : \Delta} \end{array}} \tag{3.1}$$

Let $q : D \to \Theta \bullet A \bullet B$, $m : A \to \Gamma$ and $n : B \to \Delta$. Then to satisfy the above equations in context we need that the following diagram commutes:

$$\begin{tikzcd} D \arrow[r, "q"] & \Theta \bullet A \bullet B \arrow[r, "id_\Theta \bullet m \bullet n"] \arrow[d, "\wp"'] & \Theta \bullet \Gamma \bullet \Delta \\ & \Theta \bullet A \wp B \arrow[r, "*"'] & \Theta \bullet A \bullet B \arrow[u, "id_\Theta \bullet m \bullet n"'] \end{tikzcd}$$

We make the assumption that the above decomposition is unique. Moreover, supposing Θ empty and $m = id_A$, $n = id_B$, $q = id_A \bullet id_B = id_{A \bullet B}$ we obtain $(id_A \bullet id_B; \mathbf{PAR})^* = id_A \bullet id_B$ and similarly $(id_{A \wp B})*; \mathbf{PAR} = id_{A \wp B}$; hence we may conclude that there is a natural isomorphism

$$\frac{D \to \Gamma \bullet A \bullet B}{\overline{D \to \Gamma \bullet A \wp B}}$$

so we can identify $\bullet$ and $\wp$. Finally we see that the following η equation in context are also satisfied:

$$\boxed{\begin{array}{c} \wp - \eta \textbf{ rule:} \\ \dfrac{z : D \rhd \kappa : \Upsilon, M : A \wp B \qquad x : A \rhd x : A \qquad y : B \rhd y : B}{z : D \rhd \kappa : \Upsilon, \mathtt{casel}(M) \wp \mathtt{caser}(M) = M : A \wp B} \end{array}} \tag{3.2}$$

3.3. **Linear subtraction.** 3.3.1. *Subtraction introduction.* The introduction rule for subtraction has the form

$$\frac{x : D \rhd \kappa : \Gamma, M : A \qquad y : B \rhd \zeta : \Delta}{x : D \rhd \kappa : \Gamma, \zeta[y := \mathtt{y}(M)] : \Delta, \mathtt{mkc}(M, \mathtt{y}) : A \setminus B} \setminus \mathrm{I}$$

This suggests a natural transformation with components

$$\Phi_{D,\Gamma,\Delta} : \mathbb{C}(D, \Gamma \bullet A) \times \mathbb{C}(B, \Delta) \to \mathbb{C}(D, \Gamma \bullet \Delta \bullet A \setminus B)$$

natural in D, Γ, Δ. Taking morphisms $e : D \to \Gamma \bullet A$, $f : B \to \Delta$ and $a : D' \to D$, $c : \Gamma \to \Gamma'$, $d : \Delta \to \Delta'$, by naturality we have

$$\Phi_{D',\Gamma',\Delta'}((a; e; c \bullet id_A), (f; d)) = a; \Phi_{D,\Gamma,\Delta}(e, f); c \bullet d \bullet id_{A \setminus B}; \mathtt{Join}(\Gamma', \Delta', A \setminus B)$$

In particular, taking $a = id_D$, $c = id_\Gamma$, $d : B \to \Delta$ and $f = id_B$ we have:

$$\Phi_{D,\Gamma,\Delta}(e, d) = \Phi_{D,\Gamma}(e, id_B); id_\Gamma \bullet d \bullet id_{A \setminus B}; \mathtt{Join}(\Gamma, \Delta, A \setminus B)$$

Writing $\mathbf{MKC}^B_{D,\Gamma}(e)$ for $\Phi_{D,\Gamma}(e, id_B)$, $\Phi_{D,\Gamma,\Delta}(e, d)$ can be expressed as the composition

$$\mathbf{MKC}^B_{D,\Gamma}(e); id_\Gamma \bullet d \bullet id_{A \setminus B}$$

where $\mathbf{MKC}^B_{D,\Gamma}$ is a natural transformation with components

$$\mathbf{MKC}^B_{D,\Gamma} : \mathbb{C}(D, \Gamma \bullet A) \times \mathbb{C}(B, B) \to \mathbb{C}(D, \Gamma \bullet B \bullet A \setminus B)$$

so we make the definition

$$\begin{array}{l} [\![x : D \rhd \kappa : \Gamma, \zeta[y := \mathtt{y}(M)], \mathtt{mkc}(M, \mathtt{y}) : A \setminus B]\!] =_{df} \\ \quad \mathbf{MKC}^B_{D,\Gamma}[\![x : D \rhd \kappa : \Gamma, M : A]\!]; id_\Gamma \bullet [\![y : B \rhd \zeta : \Delta]\!] \bullet id_{A \setminus B}; \mathtt{Join}(\Gamma, \Delta, A \setminus B) \end{array}$$

3.3.2. *Subtraction elimination.* The subtraction elimination rule has the form

$$\frac{x : D \triangleright \kappa : \Gamma, M : A \setminus B \qquad y : A \triangleright \xi : \Delta, N : B}{x : D \triangleright \mathtt{postp}(y \mapsto N, M), \kappa : \Gamma, \xi[y := \mathtt{y}(M)] : \Delta} \setminus \mathrm{E}$$

This suggests a natural transformation with components

$$\Psi_{D,\Gamma,\Delta} : \mathbb{C}(D, \Gamma \bullet (A \setminus B)) \times \mathbb{C}(A, \Delta \bullet B) \to \mathbb{C}(D, 1 \bullet \Gamma \bullet \Delta)$$

natural in D, Γ, Δ. Given $e : D \to \Gamma \bullet (A \setminus B)$, $f : A \to \Delta \bullet B$ and also $a : D' \to D$, $c : \Gamma \to \Gamma'$, $d : \Delta \to \Delta'$, naturality yields

$$\Psi_{D',\Gamma',\Delta'}((a; e; c \bullet id_{A \setminus B}), (f; d \bullet id_B)) = a; \Psi_{D,\Gamma,\Delta}(e, f); c \bullet d; \mathtt{Join}(\Gamma', \Delta')$$

In particular, taking $a : D \to \Gamma \bullet (A \setminus B)$, $e = id_{\Gamma \bullet (A \setminus B)}$, $c = id_\Gamma$, $d : id_\Delta$, we obtain

$$\Psi_{D,\Gamma,\Delta}(a, f) = a; \Psi_{D,\Gamma,\Delta}(id_{\Gamma \bullet (A \setminus B)}, f); \mathtt{Join}(\Gamma, \Delta)$$

Writing $\mathbf{POSTP}(f)$ for $\Phi_{D,\Gamma,\Delta}(id_{\Gamma \bullet (A \setminus B)}, f)$ we define

$$\begin{aligned} &[\![x : D \triangleright \mathtt{postp}(y \mapsto N, M), \kappa : \Gamma, \xi[y := \mathtt{Y}(M)]\!] =_{df} \\ &\quad [\![x : D \triangleright \kappa : \Gamma, M : A \setminus B]\!]; id_\Gamma \bullet \mathbf{POSTP}[\![y : A \triangleright \xi : \Delta, N : B]\!]; \mathtt{Join}(\Gamma, \Delta) \end{aligned}$$

3.3.3. *Equations in context.* We have equations in context of the form

$$\boxed{\begin{array}{c} \setminus - \beta \ \textbf{rule:} \\ \dfrac{x : D \triangleright \overline{M} : \Gamma, M : A \qquad y : B \triangleright \overline{N} : \Delta \qquad z : A \triangleright \overline{L} : \Lambda, L : B}{x : D \triangleright \overline{M} : \Gamma, \left[\overline{N}' : \Delta, \mathbf{red}\overline{L}' : \Lambda\right] = \left[\overline{N}[y := L[z ::= M]\,], \overline{L}[z := M]\right]} \end{array}} \tag{3.3}$$

where $\overline{N}' = \overline{N}[y := Y(M)]$, $\mathbf{red} = \mathtt{postp}\,(z \mapsto L, \mathtt{mkc}(M, Y))$ and $\overline{L}' = \overline{L}[z := \mathtt{mkc}(M, Y)]$. Given morphisms $n : D \to \Gamma \bullet A$ and $m : A \to \Delta \bullet B$, for these equations to be satisfied we need the following diagram to commute:

$$\begin{tikzcd} D \arrow[r, "n"] \arrow[d, "\mathbf{MKC}^B(n)"'] & \Gamma \bullet A \arrow[d, "id_\Gamma \bullet m"] \\ \Gamma \bullet (A \setminus B) \bullet B \arrow[r, "\mathbf{POSTP}(m) \bullet id_B"'] & \Gamma \bullet \Delta \bullet B \end{tikzcd}$$

in particular, taking $n = id_A$ we have

$$\begin{tikzcd} A \arrow[r, "m"] \arrow[d, "\mathbf{MKC}^B(id_A)"'] & \Delta \bullet B \\ (A \setminus B) \bullet B \arrow[ur, "\mathbf{POSTP}(m) \bullet id_B"'] & \end{tikzcd}$$

Assuming the above decomposition to be unique, we can show that the η equation in context is also satisfied:

$$\begin{array}{c} \fallingdotseq - \eta \textbf{ rule} \\ \dfrac{z : D \triangleright \overline{N} : \Delta, M : A \setminus B \qquad x : A \triangleright x : A \qquad y : B \triangleright y : B}{z : D \triangleright \overline{N} : \Delta, \left[\mathtt{mkc}(X(M), Y) : A \setminus B, \mathtt{postp}(x \mapsto Y(x), M)\right] = M : A \setminus B} \end{array} \tag{3.4}$$

and conclude that there is a natural isomorphism between the maps

$$\frac{A \to \Delta \bullet B}{\overline{A \setminus B \to \Delta}}$$

i.e., that $\setminus$ is the left adjoint to the bifunctor $\bullet$.

3.4. **Unit.** 3.4.1 *Unit rules.* The introduction and elimination rules for the unit $\bot$ are

$$\begin{array}{c} \bot \textit{ introduction} \\ \dfrac{x : D \triangleright \kappa : \Gamma}{x : D \triangleright \kappa : \Gamma, \mathtt{connect\ to}(R) : \bot} \\ \text{where } R \in \kappa. \end{array} \qquad \begin{array}{c} \bot \textit{ elimination} \\ x : \bot \triangleright \mathtt{postp}(x) \end{array}$$

The elimination rule is interpreted by a unique map $\langle\rangle : \bot \to 1$.

The introduction rule requires a natural transformation with components

$$\Phi_{D,\Gamma} : \mathbb{C}(D, \Gamma) \to \mathbb{C}(D, \Gamma \bullet \bot)$$

natural in D and Γ. Given morphisms $e : D \to \Gamma$, $d : D' \to D$ and $c : \Gamma \to \Gamma'$, naturality yields

$$\Phi_{D',\Gamma'}(d; e; c) = d; \Phi_{D,\Gamma}(e); c.$$

Letting $d : D \to \Gamma$ and $e = id_\Gamma$, $c = id_{\Gamma \bullet \bot}$ we have

$$\Phi_{D,\Gamma}(d) = d; \mathbf{Bot}_\Gamma$$

where we write $\mathbf{Bot}_\Gamma$ for $\Phi_\Gamma(id_\Gamma)$. We define

$$[\![x : D \triangleright \kappa : \Gamma, \mathtt{connect\ to}(x) : \bot]\!] =_{df} [\![x : D \triangleright \kappa : \Gamma]\!]; \mathbf{Bot}_\Gamma.$$

3.4.2. *Equations in context.* We may assume the operation $\mathbf{Bot}_\Gamma$ to be compatible with the generalized associativity and commutativity properties of $\bullet$, so that for $\Gamma = C_1, \ldots, C_n$ we have

$$\Phi_{C_1,\ldots C_i,\bot,\ldots,C_n}(id_\Gamma) : C_1 \bullet \cdots \bullet C_i \bullet \bot, \bullet \cdots C_n \quad = \quad \Phi_\Gamma(id_\Gamma) : C_1 \bullet \cdots \bullet C_n \bullet \bot$$

for all $i \leq n$. Together with naturality of $\mathbf{Bot}_\Gamma$ these yield the equations in context

$$\begin{array}{c} \dfrac{x : D \triangleright \kappa : \Gamma}{x : D \triangleright \kappa : \Gamma, \left[\mathtt{connect\ to}(R_i) = \mathtt{connect\ to}(R_j)\right]} \\ R_i, R_j \in \kappa \\ \\ \dfrac{\dfrac{x : D \triangleright \kappa : \Gamma \quad (R_i \in \kappa)}{y : E \triangleright \zeta : \Gamma' \quad (R_j \in \zeta)}}{y : E \triangleright \zeta, \left[\mathtt{connect\ to}(R_i) = \mathtt{connect\ to}(R_j)\right]} \end{array} \tag{3.5}$$

that correspond to the *rewiring properties* of $\perp$-links in the proof-net representation by [14, 15]. Moreover the equation in context

$$\boxed{\begin{array}{c}\perp - \beta\ \textbf{rule}\\ \dfrac{x : D \rhd \kappa : \Gamma \quad y : \perp \rhd \mathtt{postp}(y) \qquad (R \in \kappa)}{x : D \rhd [\kappa : \Gamma, \mathtt{postp}(\mathtt{connect\ to}(R)) = \kappa : \Gamma]}\end{array}} \tag{3.6}$$

requires that for any $m : D \to \Gamma$ the following diagram commutes:

$$\begin{tikzcd} D \arrow[r, "m;\mathbf{Bot}"] \arrow[dr, "m"'] & \Gamma \bullet \perp \arrow[d, "id_\Gamma \bullet \langle\rangle;\lambda_A"] \\ & \Gamma. \end{tikzcd}$$

Assuming that this decomposition is unique and taking $m = id_A$ we have that $\mathbf{Bot}_A; id_A \bullet \langle\rangle; \lambda_A = id_A$. Arguing as before, we see that there is a natural isomorphism

$$\begin{array}{c} D \to \Gamma \bullet 1 \\ \hline\hline D \to \Gamma \bullet \perp \end{array}$$

(so we identify $\perp$ and 1) and that the following equation in context is satisfied:

$$\boxed{\begin{array}{c}\perp - \eta\ \textbf{rule:}\\ \dfrac{z : D \rhd \kappa : \Gamma, M : \perp \qquad\qquad x : \perp \rhd \mathtt{postp}(x)}{z : D \rhd \kappa : \Gamma, \left[\mathtt{connect\ to}(\mathtt{postp}(M)) : \perp = M : \perp\right]}\end{array}} \tag{3.7}$$

Let $\mathcal{L}$ be the signature having

- the types given by the following grammar on a collection of ground types γ:

$$A \ := \ \gamma \mid \perp \mid A \wp A \mid A \setminus A$$

- a collection of sorted function symbols including $\mathtt{connect\ to}(-)$, $\mathtt{postp}(-)$, $\wp(-,-)$, $\mathtt{casel}(-)$, $\mathtt{caser}(-)$, $\mathtt{mkc}(-,-)$, $\mathtt{postp}(-,-)$.

We have proved the following

Theorem 3.4. *Let $\mathcal{T} = (\mathcal{L}, \mathcal{A})$ be a theory with signature $\mathcal{L}$ having as axioms the equations in context in Table 4 and in (3.1) - (3.7). Let $(\mathbb{C}, \bullet, 1, \setminus, \alpha, \lambda, \rho, \gamma)$ be a symmetric monoidal left-closed category and $\mathcal{M}$ a structure for $\mathcal{L}$ in $\mathbb{C}$. Then $\mathcal{M}$ satisfies the equations in $\mathcal{A}$.*

Moreover, define the *syntactic category* as the category $\mathcal{C}$ which has the formulas of multiplicative co-intuitionistic linear logic as objects and typed terms of the form $x : E \lhd \kappa : \Gamma$ (*modulo* renaming of the variable x) as morphisms. Set $x : E \rhd \kappa : \Gamma \ = \ y : E \rhd \zeta : \Gamma$ iff $\kappa = \zeta[y := x]$ is derivable from equations in context in Table 4 and in (3.1) - (3.7). Then we have

Theorem 3.5. *The syntactic category is a symmetric monoidal left-closed category.*

From this fact the *categorical completeness* theorem follows.

4. Extension to co-intuitionistic linear logic with coproducts and exponential

Let $\mathcal{L}^{\oplus}$ be $\mathcal{L}$ extended with additive disjunction $\oplus$ and the familiar functions $\mathtt{inl} : A \to A \oplus B$, $\mathtt{inr} : B \to A \oplus B$ and $\mathtt{case} : A \oplus B \times (A \to C) \times (B \to C) \to C$. Then it is easy to extend the above result to show that if $\mathbb{C}$ has also the structure of coproducts, then a structure for $\mathcal{L}^{\oplus}$ in $\mathbb{C}$ satisfies also the theory $\mathcal{T}^{\oplus}$ where $\mathcal{A}$ is extended with familiar equations in context for $\mathtt{inl}$, $\mathtt{inr}$ and $\mathtt{case}$. We shall not pursue this extension here.

The extension of $\mathcal{T}$ to a theory with the exponential ? (*why not?*) is less simple. On one hand, one can dualize Benton, Bierman, De Paiva, Hyland's definition of a linear category [11, 13] and obtain in this way a sound and complete categorical semantics for co-intuitionistic linear logic. The construction of weakly distributive categories with storage operators based on proof-nets by Blute, Cockett and Seely [14] provides a categorical model for both exponentials **!** and **?**. On the other hand, the semantics for the exponential **!** can recovered in the context of Nick Benton's treatment of *Linear Non Linear* logic [12]. After dualizing the linear part of **LNL** one should be able to recover the semantics for **?** and at the same time obtain a framework where the duality of intuitionistic and co-intuitionistic logic can be studied. We leave the development of this approach to future work and focus on the categorical semantics of the multiplicative and exponential **?** fragment of co-intuitionstic linear logic.

4.1. Co-intuitionistic linear categories.

We begin by dualizing the definition of a linear category [11, 13].

Definition 4.1. A *dual linear category* $\mathbb{C}$ consists of

(1) A symmetric monoidal left-closed category together with
(2) a symmetric co-monoidal monad $(?, \eta, \mu, \mathtt{n}_{-,-}, \mathtt{n}_{\perp})$ *(namely, the functor* **?** *is co-monoidal with respect to* $\wp$ *and the linear transformation* η, μ *are co-monoidal)* such that
   (i) - each free ?-algebra $(?A, \mu_A)$ carries naturally the structure of a commutative $\wp$-monoid *(i.e., for each* $(?A, \mu_A)$ *there are distinguished monoidal natural transformations* $i_A : \perp \to ?A$ *and* $c_A : ?A \wp ?A \to ?A$ *which form a commutative monoid and are algebra morphisms)*;
   (ii) - whenever $f : (?A, \mu_A) \to (?B, \mu_B)$ is a morphism of free algebras, then it is also a monoid morphism.

Remarks 4.2. By Maietti, Maneggia de Paiva and Ritter (see [27], Prop. 25), condition 2(ii) is equivalent to the requirement that μ is a monoidal morphism.

(i) To say that the functor **?** is symmetric co-monoidal means that it comes equipped with a *comparison* natural transformation $\mathtt{n}_{A,B} : ?(A \wp B) \to ?A \wp ?B$ and a morphism $\mathtt{n}_{\perp} : ?\perp \to \perp$, satisfying

$$\begin{tikzcd} ?(\perp \wp A) \arrow[r, "\mathtt{n}_{\perp,A}"] \arrow[d, "?\lambda_A"] & ?\perp \wp ?A \arrow[d, "\mathtt{n}_{\perp} \wp id_{?A}"] \\ ?A & \perp \wp ?A \arrow[l, "\lambda_{?A}"] \end{tikzcd} \quad \text{and} \quad \begin{tikzcd} ?(A \wp \perp) \arrow[r, "\mathtt{n}_{A,\perp}"] \arrow[d, "?\rho_A"] & ?A \wp ?\perp \arrow[d, "id_{?A} \wp \mathtt{n}_{\perp}"] \\ ?A & ?A \wp \perp \arrow[l, "\rho_{?A}"] \end{tikzcd}$$

$$\begin{tikzcd}
?((A \wp B) \wp C) \arrow[r, "\mathtt{n}_{A \wp B, C}"] \arrow[d, "?\alpha^{-1}_{A,B,C}"'] & ?(A \wp B) \wp ?C \arrow[r, "\mathtt{n}_{A,B} \wp id_C"] & (?A \wp ?B) \wp ?C \arrow[d, "\alpha^{-1}_{?A,?B,?C}"] \\
?(A \wp (B \wp C)) \arrow[r, "\mathtt{n}_{A,B \wp C}"'] & ?A \wp ?(B \wp C) \arrow[r, "id_{?A} \wp \mathtt{n}_{B,C}"'] & ?A \wp (?B \wp ?C)
\end{tikzcd}$$

$$\begin{tikzcd}
?(A \wp B) \arrow[r, "\mathtt{n}_{A,B}"] \arrow[d, "?\gamma_{A,B}"] & ?A \wp ?B \arrow[d, "\gamma_{?A,?B}"] \\
?(B \wp A) \arrow[r, "\mathtt{n}_{B,A}"] & ?B \wp ?A
\end{tikzcd}
\quad \text{and naturality:} \quad
\begin{tikzcd}
?(A \wp B) \arrow[r, "\mathtt{n}_{A,B}"] \arrow[d, "?(f \wp g)"] & ?A \wp ?B \arrow[d, "?f \wp ?g"] \\
?(A' \wp B') \arrow[r, "\mathtt{n}_{A',B'}"] & ?A' \wp ?B'
\end{tikzcd}$$

(ii) To say that η and μ are co-monoidal is to say that the following diagrams commute:

$$\begin{tikzcd}
A \wp B \arrow[r, "\eta_A \wp \eta_A"] \arrow[d, "\eta_{A \wp B}"'] & ?A \wp ?B \\
?(A \wp B) \arrow[ur, "\mathtt{n}_{A,B}"'] &
\end{tikzcd}
\quad \text{and} \quad
\begin{tikzcd}
\bot \arrow[r, "\eta_\bot"] \arrow[dr, "id"'] & ?\bot \arrow[d, "\mathtt{n}"] \\
& \bot
\end{tikzcd}$$

$$\begin{tikzcd}
??(A \wp B) \arrow[r, "\mu_{A \wp B}"] \arrow[d, "?\mathtt{n}_{A,B}"'] & ?(A \wp B) \arrow[dd, "\mathtt{n}_{A,B}"] \\
?(?A \wp ?B) \arrow[d, "\mathtt{n}_{?A,?B}"'] & \\
??A \wp ??B \arrow[r, "\mu_A \wp \mu_B"'] & ?A \wp ?B
\end{tikzcd}
\quad \text{and} \quad
\begin{tikzcd}
??\bot \arrow[r, "\mu_\bot"] \arrow[d, "?\mathtt{n}_\bot"'] & ?\bot \arrow[d, "\mathtt{n}_\bot"] \\
?\bot \arrow[r, "\mathtt{n}_\bot"'] & \bot
\end{tikzcd}$$

(iii) To say that the natural transformations $i_A : \bot \to ?A$ and $c_A : ?A \wp ?A \to ?A$ are monoidal means that they are compatible with the comparison maps, i.e., that the following diagrams commute:

$$\begin{tikzcd}
\bot \arrow[r, "id"] \arrow[d, "i_\bot"'] & \bot \\
?\bot \arrow[ur, "\mathtt{n}_\bot"'] &
\end{tikzcd}
\qquad
\begin{tikzcd}
\bot \wp \bot \arrow[r, "\lambda = \rho"] & \bot \\
?\bot \wp ?\bot \arrow[r, "c_\bot"'] \arrow[u, "\mathtt{n}_\bot \wp \mathtt{n}_\bot"] & ?\bot \arrow[u, "\mathtt{n}_\bot"']
\end{tikzcd}
\qquad
\begin{tikzcd}
\bot \wp \bot \arrow[r, "i_A \wp i_B"] \arrow[d, "\lambda = \rho"'] & ?A \wp ?B \\
\bot \arrow[r, "i_{A \wp B}"'] & ?(A \wp B) \arrow[u, "\mathtt{n}_{A,B}"']
\end{tikzcd}$$

$$\begin{tikzcd}
(?A \wp ?A) \wp (?B \wp ?B) \arrow[r, "c_A \wp c_B"] & ?A \wp ?B \\
(?A \wp ?B) \wp (?A \wp ?B) \arrow[u, "iso"] & \\
?(A \wp B) \wp ?(A \wp B) \arrow[u, "\mathtt{n}_{A,B} \wp \mathtt{n}_{A,B}"] \arrow[r, "c_{A \wp B}"'] & ?(A \wp B) \arrow[uu, "\mathtt{n}_{A,B}"']
\end{tikzcd}$$

where *iso* is the canonical isomorphism derived from symmetry and associativity;

$$\begin{tikzcd} \bot \arrow[r, "i_A"] & ?A \\ ?\bot \arrow[u, "\mathtt{n}_\bot"] \arrow[r, "?i_A"'] & ??A \arrow[u, "\mu_A"'] \end{tikzcd} \qquad \text{and} \qquad \begin{tikzcd} ?A \wp ?A \arrow[r, "c_A"] & ?A \\ ??A \wp ??A \arrow[u, "\mu_A \wp \mu_A"] & \\ ?(?A \wp ?A) \arrow[u, "\mathtt{n}_{?A,?A}"] \arrow[r, "?c_A"'] & ??A \arrow[uu, "\mu_A"'] \end{tikzcd}$$

(iv) Finally for the free algebra morphisms to be monoid morphism we require that the following diagrams commute:

$$\begin{tikzcd} ?A \wp ?A \arrow[r, "c_A"] & ?A \\ ??A \wp ??A \arrow[u, "\mu_A \wp \mu_A"] \arrow[r, "c_{?A}"'] & ??A \arrow[u, "\mu_A"'] \end{tikzcd} \qquad \begin{tikzcd} \bot \arrow[r, "i_A"] \arrow[dr, "i_{?A}"'] & ?A \\ & ??A \arrow[u, "\mu_A"'] \end{tikzcd}$$

4.2. **Term and equations in context.** To sketch a proof that a dual linear category is a model of co-intuitionistic linear logic with storage operator **?** we give the term in context and the equation in context relevant to the dereliction, weakening, contraction and storage rules. These conditions are dual to those in Figures 4.1-4.5 in G. M. Bierman's thesis [13], pp. 112-142. Since in our context the exponential rules for dereliction, weakening and contraction do not involve `let` constructions, some of these conditions result immediately from properties of substitution. There are three Equations in Context expressing "β reductions" for the

$$\frac{v : E \rhd \kappa : \Gamma, M : ?C \qquad x : C \rhd \overline{Q} \mid \overline{N} : ?\Delta}{v : E \rhd \kappa : \Gamma, \mathtt{store}(\overline{Q}, \overline{N}, \overline{\mathtt{y}}, \mathtt{x}, M) \mid \overline{\mathtt{y}}(\mathtt{x}(M)) : ?\Delta}$$

dereliction

$$\frac{x : E \rhd \kappa : \Gamma, M : C}{x : E \rhd \kappa : \Gamma, [\, M\,] : ?C}$$

weakening

$$\frac{x : E \rhd \kappa : \Gamma}{x : E \rhd \kappa : \Gamma, \mathtt{connect\ to}(R) : ?C}$$

where $R \in \kappa$.

contraction

$$\frac{x : E \vdash \kappa : \Gamma, M : ?C, N : ?C}{x : E \vdash \kappa : \Gamma, [\, M, N\,] : ?C}$$

Table 5: Term in context judgements for the **?** storage operator

storage operator in Table 6. Finally there are *Categorical Equations in Context* in Table 7.

The key decision, discussed at length in G. M. Bierman's thesis [13] pp. 127-131, arises in the analysis of the *Dereliction-Storage* reduction given by the equation in context in Table 6. By repeating for the rules of *dereliction* and *storage* the kind of analysis done for *par*,

Dereliction - Storage:

$$\frac{v : E \triangleright \kappa : \Gamma, M : C \qquad x : C \triangleright \overline{Q} \mid \overline{N} : ?\Delta}{\begin{array}{c} v : E \triangleright \kappa : \Gamma, \big[\mathtt{store}(\overline{Q}, \overline{N}, \overline{\mathtt{y}}, \mathtt{x}, [M]) \mid \overline{\mathtt{y}}(\mathtt{x}(\,[M]\,)) : ?\Delta = \\ = \overline{Q}[x := M] \mid \overline{N}[x := M] : ?\Delta\big] \end{array}}$$

Contraction - Storage:

$$\frac{v : E \triangleright \kappa : \Gamma, M_0 : ?C, M_1 : ?C \qquad x : C \triangleright \overline{Q} \mid \overline{N} : ?\Delta}{\begin{array}{c} v : E \triangleright \kappa : \Gamma, \big[\mathtt{store}(\overline{Q}, \overline{N}, \overline{\mathtt{y}}, \mathtt{x}, [M_0, M_1]) \mid \overline{\mathtt{y}}(\mathtt{x}(\,[M_0, M_1]\,)) : ?\Delta = \\ = \mathtt{store}(\overline{Q}, \overline{N}, \overline{\mathtt{y}}, \mathtt{x}, M_0), \mathtt{store}(\overline{Q}, \overline{N}, \overline{\mathtt{y}}, \mathtt{x}, M_1]) \mid \\ \langle[\overline{\mathtt{y}}(\mathtt{x}(M_0)), \overline{\mathtt{y}}(\mathtt{x}(M_1))]\rangle : ?\Delta\big] \end{array}}$$

where $?\Delta = ?D_1, \ldots ?D_m$ and $\langle[\overline{\mathtt{y}}(\mathtt{x}(M_0)), \overline{\mathtt{y}}(\mathtt{x}(M_1))]\rangle : ?\Delta$ stands for $[\mathtt{y}_1(\mathtt{x}(M_0)), \mathtt{y}_1(\mathtt{x}(M_1))] : ?D_1, \ldots, [\mathtt{y}_m(\mathtt{x}(M_0)), \mathtt{y}_m(\mathtt{x}(M_1))] : ?D_m$

Weakening - Storage:

$$\frac{v : E \triangleright \kappa : \Gamma \qquad x : C \triangleright \overline{Q} \mid \overline{N} : ?\Delta \qquad R \in \kappa}{\begin{array}{c} v : E \triangleright \kappa : \Gamma, \big[\mathtt{store}(\overline{Q}, \overline{N}, \overline{\mathtt{y}}, \mathtt{x}, \mathtt{connect\ to}(R)) \mid \\ \overline{\mathtt{y}}(\mathtt{x}(\mathtt{connect\ to}(R)) : ?\Delta = \mid \langle[\overline{\mathtt{connect\ to}}(R)]\rangle : ?\Delta\big] \end{array}}$$

where $?\Delta = ?D_1, \ldots ?D_m$ and $\mid \langle[\overline{\mathtt{connect\ to}}(R)]\rangle : ?\Delta$ stands for $\mid \mathtt{connect\ to}(R) : ?D_1, \ldots, \mathtt{connect\ to}(R) : ?D_m$

Table 6: Equations in context for the **?** storage operator

subtraction and *unit*, we see that in order to model the *storage* rule we need a natural transformation $\Phi_{E,\Gamma} : \mathbb{C}(E, \Gamma \bullet ?A) \times \mathbb{C}(A, ?\Delta) \to \mathbb{C}(E, \Gamma \bullet ?\Delta)$. By naturality considerations this is given by its action $\Phi_\Gamma(id_{\Gamma \bullet ?A}, d) =_{df} d^*$ on morphisms $d : A \to ?\Delta$. Similarly, for the *dereliction* rule we need a natural transformation $\Psi : \mathbb{C}(\_, A) \to \mathbb{C}(\_, ?A)$ and by applying Yoneda's Lemma we see that its action is given by a morphism $\eta_A : A \to ?A$.

We can certainly define a functor $? : \mathbb{C}(A, \Gamma) \to \mathbb{C}(?A, ?\Gamma)$ by $f \mapsto (f; \eta_\Gamma)^*$. Now by the equation in context for *dereliction-storage* we have that following the diagram commutes:

$$\begin{tikzcd} ??A & ??A \arrow[l, "(\eta_A)^*"'] \\ & ?A \arrow[ul, "?\eta_A"] \arrow[u, "\eta_{?A}"'] \end{tikzcd}$$

Assuming the above decomposition to be unique, we have $(\eta_A)^* = id_{??A}$ and thus the derivations

$$\eta_{?A} : \; x : ?A \triangleright [x] : ??A \quad \text{and} \quad ?\eta_A : \; z : ?A \triangleright \mathtt{store}([\,[x]\,], \mathtt{y}, \mathtt{x}, z) \mid \mathtt{y}(\mathtt{x}(z)) : ??A$$

must be identified. Now it can be shown that identifying $\eta_{?A}$ and $?\eta_A$ forces the functor **?** to be *idempotent*: $??f = ?f$. In order to avoid such collapse, the functor **?** is only assumed to be a **K** modality, and the properties of **S4** are given by the natural transformations

Monad:

$$z :?A \rhd \big[\mathtt{store}([\,[x]\,], \mathtt{y}, \mathtt{x}, z) \mid \mathtt{y}'(\mathtt{x}'(z)) : ?A \ = x :?A\big]$$

Algebra 1

$$\frac{v : E \rhd \kappa : \Gamma, M :?C \qquad x : C \rhd \overline{P} \mid \overline{N} :?\Delta}{\begin{array}{c} v : E \rhd \kappa : \Gamma, \overline{P}[x := \mathtt{x}(M)], \\ \big[\mathtt{store}(\langle \overline{N}, \mathtt{connect\ to}(R)\rangle, \langle \overline{\mathtt{y}}, \mathtt{y}\rangle, \mathtt{x}, M) \mid \overline{\mathtt{y}}(\mathtt{x}(M)) : ?\Delta, \mathtt{y}(\mathtt{x}(M)) : ?A = \\ = \mathtt{store}(\overline{N}, \overline{\mathtt{y}}, \mathtt{x}, M) \mid \overline{\mathtt{y}}(\mathtt{x}(M)) : ?\Delta, \mathtt{connect\ to}(R') : ?A\big] \\ \text{where } R \in \overline{P} \cup \overline{N} \text{ and } R' \in \overline{P}[x := \mathtt{x}(M)] \cup \overline{\mathtt{y}}(\mathtt{x}(M)) \end{array}}$$

Algebra 2

$$\frac{v : E \rhd \kappa : \Gamma, M :?C \qquad x : C \rhd \overline{P} \mid \overline{N} :?\Delta, N_0 :?A, N_1 :?A}{\begin{array}{c} v : E \rhd \kappa : \Gamma, \overline{P}[x := \mathtt{x}(M)], \\ \big[\mathtt{store}(\langle \overline{N}, [N_0, N_1]\rangle, \langle \overline{\mathtt{y}}, \mathtt{y}\rangle, \mathtt{x}, M) \mid \overline{\mathtt{y}}(\mathtt{x}(M)) : \ ?\Delta, \mathtt{y}(\mathtt{x}(M)) : ?A = \\ = \mathtt{store}(\langle \overline{N}, N_0, N_1\rangle, \langle \overline{\mathtt{y}}, \mathtt{y}_0, \mathtt{y}_1\rangle, \mathtt{x}, M) \mid \overline{\mathtt{y}}(\mathtt{x}(M)) : \ ?\Delta, [\mathtt{y}_0(\mathtt{x}(M)), \mathtt{y}_1(\mathtt{x}(M))] : ?A\big] \end{array}}$$

Monoid 1

$$\frac{v : E \rhd \kappa : \Gamma, M : ?C \qquad R \in \kappa \cup M}{v : E \rhd \kappa : \Gamma, \big[\,[M, \mathtt{connect\ to}(R)] : ?C = M : ?C\,\big]}$$

Monoid 2

$$\frac{v : E \rhd \kappa : \Gamma, M : ?C \qquad R \in \kappa \cup M}{v : E \rhd \kappa : \Gamma, \big[\,[\mathtt{connect\ to}(R), M] : ?C = M : ?C\,\big]}$$

Monoid 3

$$\frac{v : E \rhd \kappa : \Gamma, M_0 : ?C, M_1 : ?C}{v : E \rhd \kappa : \Gamma, \big[\,[M_0, M_1] : ?C = [M_1, M_0] : ?C\,\big]}$$

Monoid 4

$$\frac{v : E \rhd \kappa : \Gamma, M_0 : ?C, M_1 : ?C, M_2 : ?C}{v : E \rhd \kappa : \Gamma, \big[\,[[M_0, M_1], M_2] : ?C = [M_0, [M_1, M_2]] : ?C\,\big]}$$

Table 7: Categorical Equations in Context

$\eta : A \to ?A$ and $\mu : ??A \to ?A$ of the *monad* $(?, \eta, \mu)$. Here μ_A is given by the proof

$$z : ??A \rhd \mathtt{store}(x, \mathtt{y}, \mathtt{x}, z) \mid \mathtt{y}(\mathtt{x}(z)) : ?A$$

and the commutative diagram required by the definition of a monad

$$\begin{tikzcd} ?A & \\ ??A \arrow[u, "\mu_A"] & ?A \arrow[l, "?\eta_A"'] \arrow[ul, "id_{?A}"'] \end{tikzcd}$$

identifies $id_{?A} : x :?A \rhd x :?A$ with the following derivation $?\eta_A; \mu_A$:

$$\dfrac{\begin{array}{c}?\eta_A : \\ \dfrac{\dfrac{x : A \rhd x : A}{x : A \rhd [x] :?A}}{z :?A \rhd z :?A \qquad x : A \rhd [\,[x]\,] :??A} \\ \hline z :?A \rhd \texttt{store}([\,[x]\,], \texttt{y}, \texttt{x}, z) \mid t :??A\end{array} \qquad \begin{array}{c}\mu_A : \\ z' :??A \rhd z' :??A \quad x' :?A \rhd x' :?A \\ \hline z' :??A \rhd \texttt{store}(x', \texttt{y}', \texttt{x}', z') \mid \texttt{y}'(\texttt{x}'(t)) :?A\end{array}}{z :?A \rhd \texttt{store}([\,[x]\,], \texttt{y}, \texttt{x}, z), \texttt{store}(x', \texttt{y}', \texttt{x}', t) \mid \texttt{y}'(\texttt{x}'(t)) : ?A}$$

$$\text{where } t = \texttt{y}(\texttt{x}(z)) :??A$$

The normal form of the derivation $?\eta_A; \mu_A$ is the following one:

$$z :?A \rhd \texttt{store}([\,[x]\,], \texttt{y}, \texttt{x}, z) \mid \texttt{y}'(\texttt{x}'(z)) : ?A$$

as in the Categorical Equation in Context for Monad of Table 7. Further details are left to the reader.

5. Conclusion.

In order to provide a categorical semantics for *co-intuitionistic logic* - given that as remarked by Tristan Crolard [18] co-exponents in the category **Set** are trivial - we have given a categorical semantics for *intuitionistic multiplicative and exponential* co-intuitionistic linear logic, from which our desired results follows by dualizing J-Y. Girard's embedding of intuitionistic logic into intuitionistic linear logic.

In this task we started from a term assignment to multiplicative co-intuitionistic logic, which has been proposed as an abstract *distributed calculus* dualizing the linear λ calculus [2, 3, 7]: in our view such dualization underlies the translation of the linear λ-calculus into the π-calculus (see [10]). Our *dual distributed calculus* is itself a restriction to a co-intuitionistic consequence relation of Crolard's term assignment to *subtraction* in the framework of the $\lambda\mu$-calculus: to *subtraction introduction* and *elimination* rules and to their β reduction *global operations of binding* and *global substitution* are assigned; these operations may appear as notationally awkward at first sight but are forced on us by the removal of the μ-rule and of the μ-variable abstraction used in Crolard's approach. A computational application of our notation is suggested in the proof of the Decomposition Property (Proposition 2.2). Since the dependencies of a variable $y : C$ from n binders $\texttt{make} - \texttt{coroutine}$ and $\texttt{postpone}$ are represented in a term $\texttt{y}(t_1(\dots t_n(M)\dots)) : C$ by the terms $t_i : D_i$, this representation can be used to compute the assignment of a probabilistic event $\mathbf{C}$ to such a term according to the assignments $\mathbf{D}_i$ to the terms $t_i : D_i$ and to prove that *probabilities are preserved* from the premise to the disjunction of the conclusions in a multiplicative co-intuitionistic derivation.

Our work required a lengthy exercise on well-known results by Benton, Bierman, Hyland and de Paiva[11, 13], with the considerable help given by Blute, Cockett, Seely and Trimble's work [14, 15]. To assess the merits and advantages of our work we need to evaluate the syntax for the exponential rules: here again the *storage* rule may appear notationally quite heavy, but it is a straightforward implementation of the act of storing. On the other hand the advantages of working in the dual system are completely evident in the treatment of *dereliction* and *contraction*, where the awkward `let` operations and related naturality conditions are replaced by simple operations on lists. Finally, the treatment of *weakening* is also completely standard, thanks also to Blute, Cockett, Seely and Trimble's work [14, 15] on the notion of *rewiring*.

References

[1] G. Bellin. Assertions, Hypotheses, Conjectures, Expectations: Rough-sets semantics and proof-theory. In: L. C. Pereira, E. H. Haeusler, V. de Paiva (eds) *Advances in Natural Deduction. A Celebration of Dag Prawitz's Work*, Trends in Logic 39, Springer Science+Business Media Dordrecht 2014, pp. 193-241.

[2] G. Bellin. On the π calculus and distributed calculi for co-intuitionistic logic, conference paper presented at the LAM-CONCUR'11 Workshop, Aachen, 2011.

[3] G. Bellin. A Term Assignment for Dual Intuitionistic Logic, conference paper presented at the LICS'05-IMLA'05 Workshop, Chicago, IL, June 30, 2005

[4] G. Bellin. Subnets of Proof-nets in multiplicative linear logic with MIX, *Mathematical Structures in Computer Science* vol.7, pp.663-699 (1997).

[5] G. Bellin and C. Biasi. Towards a logic for pragmatics. Assertions and conjectures. In: *Journal of Logic and Computation*, 14, 4, 2004, pp. 473-506.

[6] G. Bellin, M. Carrara, D. Chiffi and A. Menti. Pragmatic and dialogic interpretations of bi-intuitionism, Parts I and II, forthcoming in *Logic and Logical Philosophy*, http://dx.doi.org/10.12775/LLP.2014.011 and http://dx.doi.org/10.12775/LLP.2014.012, Published online June 23 2014.

[7] G. Bellin and A. Menti. On the π-calculus and Co-intuitionistic Logic. Notes on Logic for Concurrency and λP Systems, *Fundamenta Informaticae* 130, pp. 21-65, 2014.

[8] G. Bellin, M. Hyland, E.Robinson and C. Urban. Categorical Proof Theory of Classical Propositional Calculus *Theoretical Computer Science* Vol. 364, 2, November 2006, pp. 146-165.

[9] G. Bellin and C. Dalla Pozza. A pragmatic interpretation of substructural logics. In *Reflection on the Foundations of Mathematics (Stanford, CA, 1998)*, Essays in honor of Solomon Feferman, W. Sieg, R. Sommer and C. Talcott eds., ASL, Lecture Notes in Logic, Volume 15, 2002, pp. 139-163.

[10] G. Bellin and P. J. Scott. On the Pi-calculus and linear logic, *Theoretical Computer Science* 135, pp. 11-65, 1994.

[11] P. N. Benton, G. M. Bierman, J. M. E. Hyland and V. C. V. dePaiva. A term calculus for Intuitionistic Linear Logic. In: *Typed Lambda Calculi and Applications*, Lecture Notes in Computer Science, Volume 664, 1993, pp.75-90.

[12] P. N. Benton. A mixed linear and non-linear logic: Proofs, terms and models, *Computer Science Logic*, Lecture Notes in Computer Science, 1995, Volume 933, 1995, pp.121-135.

[13] G. M. Bierman. On Intuitionistic Linear Logic, PhD Thesis, University of Cambridge Computer Laboratory, Technical Report No. 346, 1994.

[14] R. F. Blute, J. R. B. Cockett and R. A. G.Seely. **!** and **?** as tensorial strength, *Mathematical Structures in Computer Science*, 6, pp.313-351, 1996.

[15] R. F. Blute, J. R. B. Cockett, R. A. G.Seely and T. H. Trimble. Natural Deduction and Coherence for Weakly Distributive Categories, *Journal of Pure and Applied Algebra*, 113, pp. 229-296, 1996.

[16] T. Brauner and V. de Paiva. Cut-elimination for Full Intuitionistic Logic, University of Cambridge Computer Laboratory, Technical Report 395 and BRICS, 1996.

[17] J. Cockett and R. Seely. Proof theory for full intuitionistic linear logic, bilinear logic, and mix categories. *Theory and Applications of Categories*, 3(5):85131, 1997.

[18] T. Crolard. Subtractive logic, in *Theoretical Computer Science* 254,1-2, 2001, pp. 151-185.

[19] T. Crolard. A Formulae-as-Types Interpretation of Subtractive Logic. In: *Journal of Logic and Computation*, vol.14(4), 2004, pp. 529-570

[20] P-L. Curien. Abstract Machines, Control, and Sequents. G. Barthe, P. Dybjer, L. Pinto, J. Saraiva (Eds.): APPSEM 2000, Springer LNCS 2395, 2002

[21] R. Goré. Dual Intuitionistic Logic Revisited, In TABLEAUX00: Automated Reasoning with Analytic Tableaux and Related Methods, LNAI 1847:252-267, 2000. Springer.

[22] M. Hyland and V. de Paiva. Full intuitionsitic linear logic (extended abstract), *Annals of Pure and Applied Logic*, 64, pp. 273-91, 1993

[23] J. Lambek. Cut elimination for classical bilinear logic. *Fundamenta Informaticae*, 22(1/2):5367, 1995.

[24] J. Lambek and P. J. Scott. *Introduction to higher order categorical logic* Cambridge University Press, 1986.

[25] F. W. Lawvere, Intrinsic co-Heyting boundaries and the Leibniz rule in certain toposes. In A. Carboni, M.C. Pedicchio and G. Rosolini (eds.), *Category Theory (Como 1990)*, Springer Lecture Notes in Math. 1488, 1991, pp. 279 - 297.

[26] E. G. K. López-Escobar. On Intuitionistic Sentential Connectives I, *Revista Colombiana de Matemàticas* XIX, 1985, pp. 117-130
[27] M. Maietti, P. Maneggia, V. de Paiva ad E. Ritter. Relating Categorical Semantics for Intuitionistic Linear Logic, *Applied Categorical Structures*, 13, pp. 1-36. 2005.
[28] M. Makkai and G. E. Reyes. Completeness results for intuitionistic and modal logic in a categorical setting, *Annals of Pure and Applied Logic*, 72, 1995, pp.25-101.
[29] P. Pagliani. Intrinsic co-Heyting boundaries and information incompleteness in Rough Set Analysis. In: Polkowski, L., Skowron, A. (eds.) RSCTC 1998. Springer LNCS, vol. 1424, pp. 123-130, 2009.
[30] C. Rauszer. Semi-Boolean algebras and their applications to intuitionistic logic with dual operations, in *Fundamenta Mathematicae*, 83, 1974, pp. 219-249.
[31] C. Rauszer. Applications of Kripke Models to Heyting-Brouwer Logic, in *Studia Logica* 36, 1977, pp. 61-71.
[32] G. Reyes and H. Zolfaghari, Bi-Heyting algebras, Toposes and Modalities, in *Journal of Philosophical Logic*, 25, 1996, pp. 25-43.
[33] H. Schellinx. Some syntactical observations on linear logic. *Journal of Logic and Computation*, 1(4):537559, 1991.
[34] P. Selinger. Control categories and duality: on the categorical semantics of the lambda-mu calculus. Mathematical Structures in Computer Science, 11:207–260, 2001.
[35] L. Tranchini. Natural Deduction for Dual Intuitionistic Logic, *Studia Logica* (online version: 20 June 2012)
[36] H. Wansing. Constructive negation, implication and co-implication, *Journal of Applied Non-Classical Logics* 18, 2008, pp. 341-364.

Appendix A. Example

Consider the following computation in the simply typed λ-calculus. We write $\mathbf{N}$ for $(A \supset A) \supset (A \supset A)$ and $\mathbf{N}^{\perp}$ for $(C \smallsetminus C) \smallsetminus (C \smallsetminus C)$.

$$
\begin{array}{rll}
\lambda h^{A\supset A}.(\lambda f^{A\supset A}.\lambda x^{A}.ffx)(\lambda g^{A\supset A}\lambda y^{A}.ggy)h : \mathbf{N} & \rightsquigarrow_{\beta} & \text{(i)} \\
\lambda h^{A\supset A}.(\lambda f^{A\supset A}.\lambda x^{A}.ffx)(\lambda y^{A}.hhy) : \mathbf{N} & \rightsquigarrow_{\beta} & \text{(ii)} \\
\lambda h^{A\supset A}.\lambda x^{A}.(\lambda y^{A}.hhy)(\lambda y^{A}.hhy)x : \mathbf{N} & \rightsquigarrow_{\beta} & \text{(iii)} \\
\lambda h^{A\supset A}.\lambda x^{A}.((\lambda y^{A}.hhy)hhx : \mathbf{N} & \rightsquigarrow_{\beta} & \text{(iv)} \\
\lambda h^{A\supset A}.\lambda x^{A}.hhhhx : \mathbf{N} & & (v)
\end{array}
$$

In Table 8 we give a Natural Deduction derivation in "tree form" with the assignment of the term (ii) $\vdash \lambda h.(\lambda f.\lambda x.ffx)(\lambda y.hhy) : \mathbf{N}$. In Table 9 we consider a Natural Deduction derivation of $n : \mathbf{N}^{\perp} \vdash P_1, P_2, P_3, P_4 \mid$ in the *subtraction only fragment of co-intuitionistic logic*; such a derivation is exactly dual of that in Table 8: it is also in "tree form", in fact it yields the same tree as in Table 8 read from bottom up. Its term assignment, with the p-terms P_1, P_2, P_3, P_4 in the conclusion, belong to a *dual calculus* defined in [1] and briefly described here, with the property that to each β reduction in the simply typed λ calculus there corresponds a set of rewritings of the computational context in the dual calculus and a reduction sequence $t_1, t_2, \ldots$ of simply typed λ terms terminates if and only if the dual sequence $t_1^{\perp}, t_2^{\perp}, \ldots$ terminates (see [1], section 6.1). The derivation in Table 10 results from that in Table 9 by one step of normalization.

The grammar of the *dual calculus* for the subtraction-only fragment of co-intuitionism is as follows (see [1], section 6, definition 10):

$$
\begin{array}{rcl}
M & := & x \mid \mathtt{x}(M) \mid \mathtt{mkc}(M, \mathtt{x}) \\
\ell & := & [] \mid [M_1, \ldots, M_n] \text{ for some } n. \\
P & := & \mathtt{postp}(\mathtt{y} \mapsto \ell[y := \mathtt{y}(M)], M).
\end{array}
$$

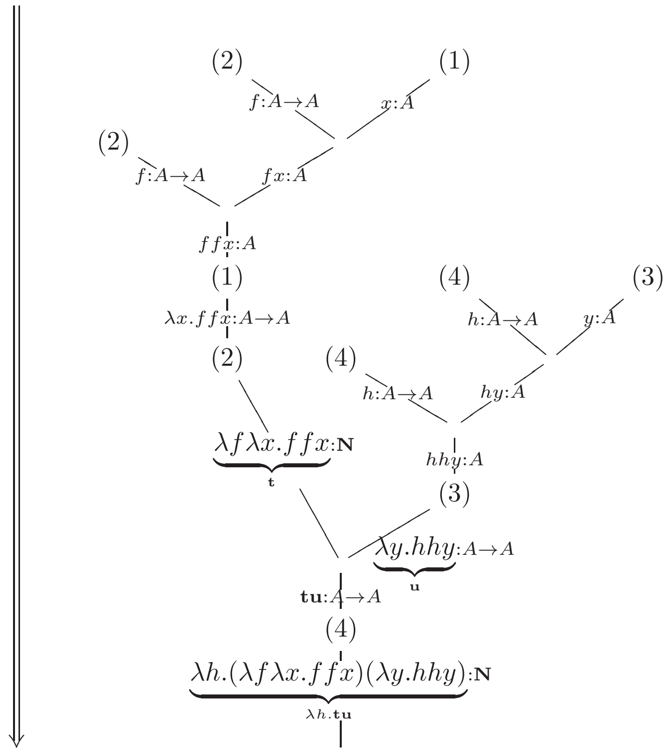

Table 8: Natural Deduction tree for (ii) $\vdash \lambda h.(\lambda f.\lambda x.ffx)(\lambda y.hhy) : \mathbf{N}$

Here non-empty lists ℓ are *flattened* and occur only within p-terms. A notion of *term expansion* allows to define the substitution of a *flat list* for a free variable in (a *flat list* of) terms, yielding a *flat list*. In this way we take care of *contraction of discharged conclusions* resulting from a *subtraction elimination*. A conclusion introduced by *weakening* is assigned an empty list. To decorate natural deduction trees we use a notation which ignores the distinction between local and remote binding, as discussed in Note 1.7 (iii). Notice that the grammar of the *dual calculus* used in this section is actually a fragment of the grammar of the *linear dual calculus* presented in this paper [4].

Next we translate the *co-intuitionistic* natural deduction derivations of Tables 9 and 10 into *co-intuitionistic linear* logic. In Tables 11 and 12 we adapt the graphical notation of Tables 9 and 10 to our linear calculus and notice that that the derivation in Table 12 results from that in Table 11 by applying *two normalization steps*, a *subtraction reduction* followed by a *storage - contraction reduction*. The graphical notation should help to catch a glimpse of the reduction process more vividly. Here we present the Natural Deduction derivation of Table 11 in the *sequent-style typing judgements* of our official calculus.

[4] We have not explored the possibility of assigning the *empty list* to the formulas $?C$ introduced by *weakening* also in the linear calculus; this would distinguish the case of *weakening* from that of the $\bot$-*introduction* rule, where terms of the form `connect to` (R) would still be used.

$$(2) : P_2 = \underbrace{\mathtt{postp}(g \mapsto [m_2, m_1], m_3)}_{\textbf{Redex}} \quad (4) : P_4 = \mathtt{postp}(e \mapsto [m_5, m_4], n)$$

$$(1) : P_1 = \mathtt{postp}(b \mapsto [x], g) \qquad\qquad (3) : P_3 = \mathtt{postp}(d \mapsto [y], j)$$

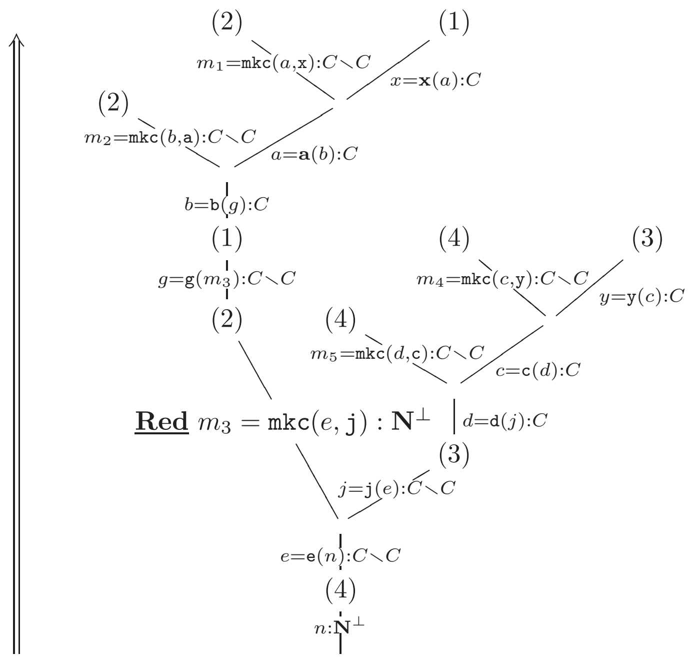

Table 9: **co-IL** tree for the dual of (ii) $\vdash \lambda h.(\lambda f.\lambda x.ffx)(\lambda y.hhy) : \mathbf{N}$

Sequent-style Natural Deduction. (i) The derivation $\mathcal{D}$ corresponding to the graph inside *box* $\mathbf{B}_{k,M}$ of Table 11 is as follows.

$$\dfrac{k : C \setminus C \triangleright \mid k : C \setminus C \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{d : C \triangleright \mid d : C \qquad c : C \triangleright \mid c : C}{d : C \triangleright \mid \underbrace{\mathtt{mkc}(d, \mathtt{c})}_{m_5} : C \setminus C, \underbrace{\mathtt{c}(d)}_{c} : C}\setminus I \qquad y : C \triangleright \mid y : C}{d : C \triangleright \mid m_5 : C \setminus C, \underbrace{\mathtt{mkc}(c, \mathtt{y})}_{m_4} : C \setminus C, \underbrace{\mathtt{y}(c)}_{y} : C}\setminus I}{d : C \triangleright \mid [m_5] :?(C \setminus C), [m_4] :?(C \setminus C), y : C}\,der}{d : C \setminus C \triangleright \mid [[m_5][m_4]] :?(C \setminus C), y : C}\,contr}{k : C \setminus C \triangleright \underbrace{\mathtt{postp}(d \mapsto y, k)}_{P_3} \mid \underbrace{[[m_5][m_4]][d := \mathtt{d}(k)]}_{M} :?(C \setminus C)}\setminus E$$

Applying the ?-E rule with major premise $j :?(C \setminus C) \triangleright j :?(C \setminus C)$ we obtain a derivation of the following sequent:

$$j :?(C \setminus C) \ \triangleright \ \mathtt{store}(P_3, M, \mathtt{y}_0, \mathtt{k}, j) \mid \mathtt{y}_0(\mathtt{k}(j)) :?(C \setminus C).$$

$$(1) : P_1' = \texttt{postp}(b \mapsto [x], e) \qquad (4) : P_4' = \texttt{postp}(e \mapsto [m_5', m_5'', m_4', m_4''], n)$$
$$(3') : P_3' = \underbrace{\texttt{postp}(d' \mapsto [y'], m_2)}_{\textbf{Redex}} \qquad (3'') : P_3'' = \underbrace{\texttt{postp}(d'' \mapsto [y''], m_1)}_{\textbf{Redex}}$$

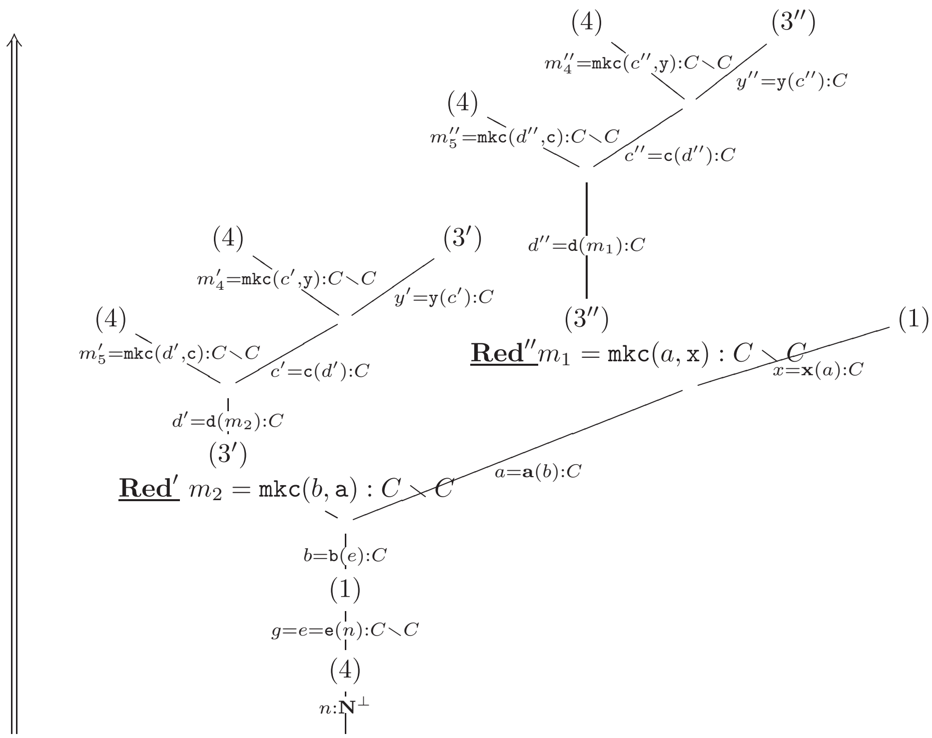

Table 10: **co-IL** tree for the dual of (iii) $\vdash \lambda h.\lambda x.(\lambda y.hhy)(\lambda y.hhy)x : \mathbf{N}$

Finally, by applying *subtraction introduction* to it with the axiom $e : C \smallsetminus C \rhd \mid e : C \smallsetminus C$ we obtain a derivation $\mathcal{D}_1$ of the following sequent:

$$e : C \smallsetminus C \ \rhd\ \underbrace{\texttt{store}(P_3, M, \texttt{y}_0, \texttt{k}, \texttt{j}(e))}_{\textbf{Store}} \mid \underbrace{\texttt{y}_0(\texttt{k}(\texttt{j}(e)))}_{y_0} :?(C \smallsetminus C), \texttt{mkc}(e, \texttt{j}) : \mathbf{N}^\perp$$

where $\mathbf{N}^\perp = (C \smallsetminus C) \smallsetminus ?(C \smallsetminus C)$.

(ii) By applying the same steps as in derivation $\mathcal{D}$, but relabelling of the terms, we obtain a derivation $\mathcal{D}_0$ of the following sequent:

$$g : C \smallsetminus C \rhd \underbrace{\texttt{postp}(b \mapsto x, g)}_{P_1} \mid \underbrace{[[m_1][m_2]][b := \texttt{b}(g)]}_{M_0} :?(C \smallsetminus C)$$

where $m_1 = \texttt{mkc}(b, \texttt{a})$, $m_2 = \texttt{mkc}(a, \texttt{x})$, $b = \texttt{b}(g)$, $a = \texttt{a}(b)$ and $x = \texttt{x}(a)$.
(iii) Now if we apply $\mathcal{D}_1$ to $\mathcal{D}_0$ with the formula $\texttt{mkc}(e, \texttt{j}) : \mathbf{N}^\perp$ as major premise of *subtraction elimination* then we obtain a derivation $\mathcal{D}^+$ ending with following sequent:

$$e : C \smallsetminus C \ \rhd\ \textbf{Store}\ \texttt{postp}(g \mapsto M_0, \texttt{mkc}(e, \texttt{j})) \mid y_0 :?(C \smallsetminus C)$$

$$(2) : P_2 = \underbrace{\mathtt{postp}(g \mapsto [[m_1][m_2]], m_3)}_{\textbf{Redex}}$$

$$(1) : P_1 = \mathtt{postp}(b \mapsto x, g) \qquad\qquad (4') : P_4 = \mathtt{postp}(e \mapsto [y_1], n)$$

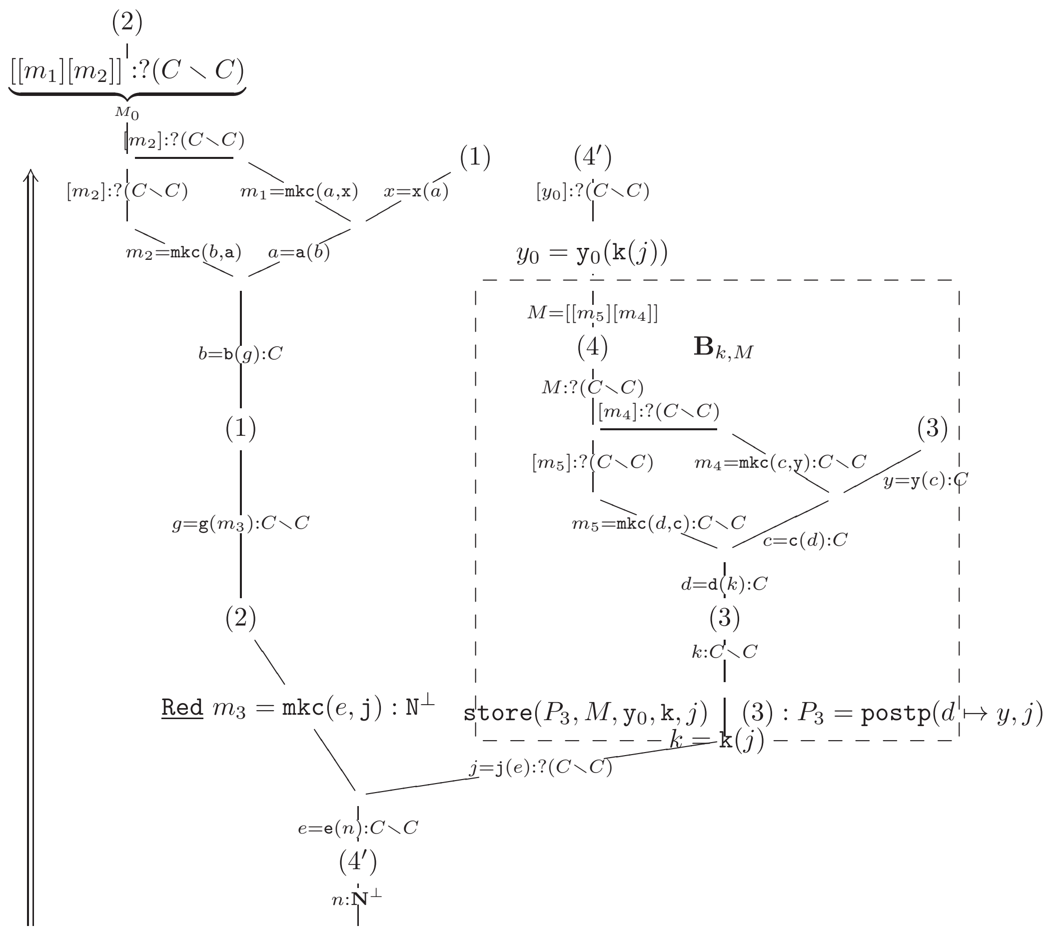

Table 11: **linear co-IL** analysis of the proof in Table 9

The pair *introduction / elimination* inferences determines the only **Redex** in $\mathcal{D}^{+}$. A final *subtraction elimination* with the axiom $n : \mathbf{N}^{\perp} \triangleright n : \mathbf{N}^{\perp}$ concludes the derivation in our example.

$(1) : P_1 = \mathtt{postp}(b \mapsto x, e)$ $\qquad$ $(4') : P_4 = \mathtt{postp}(e \mapsto [[y_2][y_1]], n)$

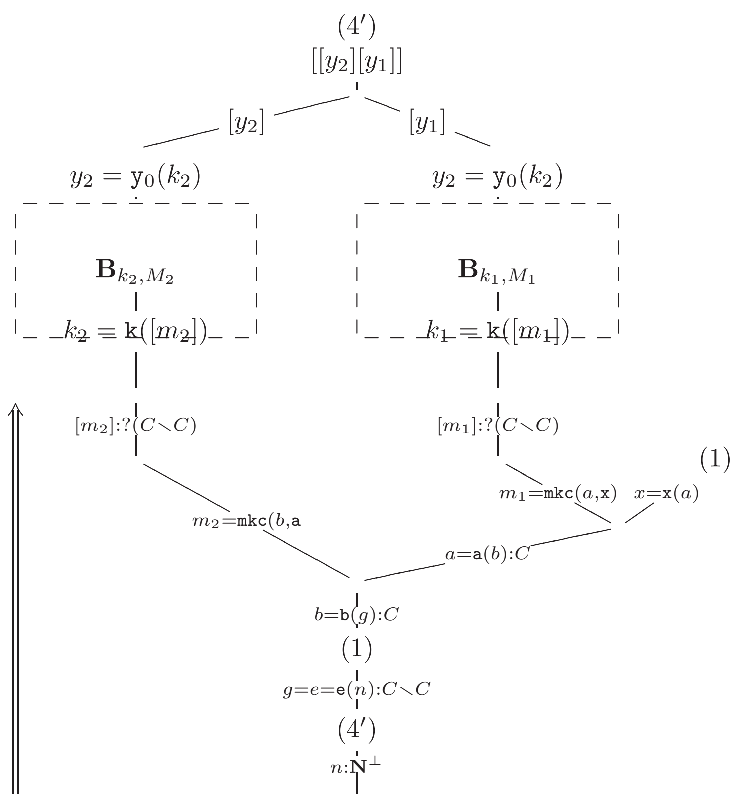

Table 12: **linear co-IL** analysis of the proof in Table 10