A theory of effects and resources: adjunction models and polarised calculi

Pierre-Louis Curien, Marcelo Fiore, Guillaume Munch-Maccagnoni

To cite this version:

Pierre-Louis Curien, Marcelo Fiore, Guillaume Munch-Maccagnoni. A theory of effects and resources: adjunction models and polarised calculi. Principles of Programming Languages, Jan 2016, Saint-Petersburg, Florida, United States. 10.1145/2837614.2837652. hal-01256092

HAL Id: hal-01256092
https://hal.archives-ouvertes.fr/hal-01256092

Submitted on 14 Jan 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
A Theory of Effects and Resources: Adjunction Models and Polarised Calculi

Pierre-Louis Curien
PiR2, INRIA Paris-Rocquencourt
PPS, Univ Paris Diderot, France
Pierre-Louis.Curien@pps.univ-paris-diderot.fr

Marcelo Fiore
Computer Laboratory
University of Cambridge, UK
Marcelo.Fiore@cl.cam.ac.uk

Guillaume Munch-Maccagnoni
Computer Laboratory
University of Cambridge, UK
Guillaume.Munch-Maccagnoni@cl.cam.ac.uk

Abstract
We consider the Curry-Howard-Lambek correspondence for effectful computation and resource management, specifically proposing polarised calculi together with presheaf-enriched adjunction models as the starting point for a comprehensive semantic theory relating logical systems, typed calculi, and categorical models in this context.

Our thesis is that the combination of effects and resources should be considered orthogonally. Model theoretically, this leads to an understanding of our categorical models from two complementary perspectives: (i) as a linearisation of CBPV (Call-by-Push-Value) adjunction models, and (ii) as an extension of linear/non-linear adjunction models with an adjoint resolution of computational effects. When the linear structure is cartesian and the resource structure is trivial we recover Levy’s notion of CBPV adjunction model, while when the effect structure is trivial we have Benton’s linear/non-linear adjunction models. Further instances of our model theory include the dialogue categories with a resource modality of Melliès and Tabareau, and the [E]EC ([Enriched] Effect Calculus) models of Egger, Møgelberg and Simpson. Our development substantiates the approach by providing a lifting theorem of linear models into cartesian ones.

To each of our categorical models we systematically associate a typed term calculus, each of which corresponds to a variant of the sequent calculi LJ (Intuitionistic Logic) or IL (Intuitionistic Linear Logic). The adjoint resolution of effects corresponds to polarisation whereby, syntactically, types locally determine a strict or lazy evaluation order and, semantically, the associativity of cuts is relaxed. In particular, our results show that polarisation provides a computational interpretation of CBPV in direct style. Further, we characterise depolarised models: those where the cut is associative, and where the evaluation order is unimportant. We explain possible advantages of this style of calculi for the operational semantics of effects.

Categories and Subject Descriptors F.3.2 [Logics and meanings of programs]: Semantics of Programming Languages—Denotational semantics, Operational semantics; F.3.3 [Logics and meanings of programs]: Studies of Program Constructs—Type structure; F.4.1 [Mathematical logic and formal languages]: Mathematical Logic—Lambda calculus and related systems, Proof theory

Keywords Categorical semantics, Curry-Howard-Lambek correspondence, Intuitionistic logic, Linear logic, Computational effects, Resource modalities, Polarised calculi, Adjunction models

1. Introduction
The modern study of foundations for programming languages involves looking at the subject from a variety of viewpoints. In particular: (i) syntactic calculi, formalising typing disciplines and computational behaviour, are built; (ii) connections to logical systems, for guiding these developments, are sought; and (iii) models for establishing a mathematical basis, and informing the design space, are investigated. In unison, these activities have the overall aim of discovering deep connections as in the Curry-Howard-Lambek correspondence. It is within this landscape that the work presented here falls, specifically in the context of investigating two prominent aspects of computation: computational effects and resource management.

Computational effects The analysis of computational effects in the above light started with the seminal work of Moggi [34]. There, the notion of monad was put forward as a mathematical abstraction for encapsulating effects and used to extract a metalanguage (in indirect style) in which to give semantic interpretations to call-by-value programming languages (in direct style) with effectful computations.

Prompted by the mathematical analysis of Power and Robinson [42] investigating adjoint resolutions for the monadic theory of effects, Levy [31] developed CBPV (Call-by-Push-Value), a theory based on adjunction models. The mathematical structure of these models is directly reflected in Levy’s CBPV calculus where there is a dichotomy of type and program structure corresponding to each of the categories involved in an adjoint situation. This results in a paradigm with aspects of both call-by-value and call-by-name computation.

Linear logic The analysis of resources in computation started with Girard’s seminal discovery of their importance in denotational semantics [22]. Model theoretically, it corresponds to shifting attention from cartesian (intuitionistic) to symmetric monoidal (linear) structure, recovering the former from the latter by means of a resource (or exponential) modality; which, after much study [4, 5, 8, 43], was revealed to correspond to a comonad arising from a monoidal adjoint resolution.

The study of effects and resources, respectively uncovering the need for adjoint models giving rise to monads and to comonads, offered the tantalising perspective that these two phenomena could somehow be two sides of the same coin. This possibility was first investigated by Benton and Wadler [7], and more recently further pursued by Egger, Møgelberg and Simpson [16]. Our stand in this respect is that there are in fact other phenomena at play, and that the combination of effects and resources should be considered orthogonally.
Polarisation. To understand the computational structures at work, one needs to consider another logical development of Girard: polarisation [23]. In a modern guise, polarisation turns an arbitrary adjunction into an algebraic structure supporting the composition of morphisms with source and target belonging to either of the categories involved; objects are positive or negative depending on their category of origin. We suggest to understand this as a description of direct-style calculi that generalises from monads to adjunctions the direct-style calculus for the Kleisli category in Moggi’s model.

Girard’s work has been related to continuation-passing style (CPS) and with the investigation of the duality between call by value and call by name in this context [10, 12, 13, 29, 30, 41, 44]. However, in continuity with the work of Zeilberger [48, 49] and Melliès [33], we advocate that the perfect symmetry of Girard’s setting is misleading. Polarisation describes a more general model of computation in which strictness and laziness are attributes of the type [36, 39].

Contents of the paper. With the above ingredients at hand, our viewpoint is that the adjoint resolution of a monad describing effects value and call by name in this context [10, 12, 13, 29, 30, 41, 44]. per. (All the logics considered in this paper are propositional.)

The column on the left above consists of enriched adjoint resolutions of intuitionistic models, for multiplicative (top) and for multiplicative-additive (bottom) structure. The column on the right refines this into more general linear models. The latter, when further equipped with an adjoint resolution for resource management, give rise to cartesian models.

Having set up this model-theoretic scenario, our development substantiates the approach by relating it to typed calculi. Rather than reproducing the adjunction structure naively in the syntax, we substantiate the approach by relating it to typed calculi. Rather than reproducing the adjunction structure naively in the syntax, we substantiate the approach by relating it to typed calculi. Rather than reproducing the adjunction structure naively in the syntax, we substantiate the approach by relating it to typed calculi. Rather than reproducing the adjunction structure naively in the syntax, we substantiate the approach by relating it to typed calculi. Rather than reproducing the adjunction structure naively in the syntax, we substantiate the approach by relating it to typed calculi. Rather than reproducing the adjunction structure naively in the syntax, we substantiate the approach by relating it to typed calculi. Rather than reproducing the adjunction structure naively in the syntax, we substantiate the approach by relating it to typed calculi. Rather than reproducing the adjunction structure naively in the syntax, we substantiate the approach by relating it to typed calculi.

These sequent calculi provide the type systems for the calculi that we introduce, and for which we give sound interpretations in their corresponding models. We employ the standard naming scheme whereby $M$ stands for the core multiplicative fragment ($I$, $\otimes$, $\rightarrow$), $E$ for the consideration of a resource (or exponential) modality ($\&$), and $A$ for additives including units ($T$, $\emptyset$, $\&$, $\emptyset$).

Our calculi build on the $L$-calculus $\mu\mu$ relating sequent calculi to abstract machines [10] and their variants with polarities [11, 36, 37, 39]. In particular, building on this earlier work, we demonstrate that a model of computation in which strict and lazy datatypes coexist provides a direct computational interpretation to Levy’s CBPV. Further, we characterise depolarised models in a sense close to that of Melliès and Tabareau [33]: there is a correspondence between proof systems where the cuts associate, calculi where the order of evaluation is unimportant, and models where the enriched adjunction is idempotent.

In summary, our contributions are:

- A comprehensive theory of presheaf-enriched adjunction models, with cartesian, linear, additive, and resource structures as per the diagram, that encompasses CBPV, linear/non-linear adjunction models, dialogue categories, and EEC models.
- A lifting theorem of linear models with resources into cartesian models.
- Polarised L-calculi and corresponding polarised logics, with cartesian, linear, additive, and exponential structure as per the diagram, that are direct-style for the above models, with soundness theorems.
- A characterisation of depolarisation.

Organisation of the paper. Section 2 presents the rudiments of the adjunction models needed here and explains polarisation putting it in the current context. The formal treatment starts in Section 3 where the calculi for multiplicative polarised structure, $MLJ^p$ for cartesian (or intuitionistic) and $IMLL^p$ for linear, are introduced, with $MLJ^p$ models presented. Section 4 follows with $IMLL^p$ models, providing the interpretation of the multiplicative calculi, and the characterisation of depolarisation. Sections 5 and 6 modularly extend the development to respectively include resource structure, establishing the lifting theorem, and additive structure, ending with $ILL^p$ and $LL^p$ calculi and models. Section 7 concludes with perspectives on novel aspects of our calculus in the context of the operational and equational semantics of effects.

2. Adjunction Models and Polarised Calculi

2.1 Notations for Presheaf-Enriched Adjunction Models

We introduce notation for basic presheaf-enriched structure to be used in the paper; for a comprehensive treatment of enriched category theory the reader may consult Kelly [27].

We are interested in this paper in enriched adjoint situations

$$\vdash \frac{\mathcal{V}}{\mathcal{S}} \quad \delta$$

where the categories $\mathcal{V}$ and $\mathcal{S}$ respectively provide mathematical structure for positive and negative worlds to be dwelled upon in the following section. In this context, we will write $P, Q, \ldots$ for the objects of $\mathcal{V}$ and $N, M, \ldots$ for the objects of $\mathcal{S}$. Intuitively, the adjoint functors $F$ and $G$ allow the passage between the two worlds $\mathcal{V}$ and $\mathcal{S}$.

The categories $\mathcal{C}$ at play here are to be specifically enriched over presheaves. Roughly speaking, this means that their homs are parametrised as follows

$$\mathcal{C}(x, y)$$
This is achieved by means of Day’s convolution monoidal structure corresponding to the action of changing environment. For instance, in the cartesian setting, the action \( \mathcal{C}_{\mathcal{C}}(X, Y) \to \mathcal{C}_{\mathcal{C}}(X, Y) \) induced by the duplicator map \( C \to C \times C \) corresponds to the operation of contracting the environment.

In the cartesian setting, the identities and composition of \( \mathcal{C} \) are given pointwise; that is, one has

\[
\text{id}_{C}^{C}(X, X) = \mathcal{C}_{\mathcal{C}}(X, X), \quad \circ_{C}(X, Y, Z) : \mathcal{C}_{\mathcal{C}}(Y, Z) \times \mathcal{C}_{\mathcal{C}}(X, Y) \to \mathcal{C}_{\mathcal{C}}(X, Z).
\]

In the linear setting there is a need for a more refined approach and one instead has

\[
\text{id}_{C}^{C}(X, X) = \mathcal{C}_{\mathcal{C}}(X, X), \quad \circ_{C}(X, Y, Z) : \mathcal{C}_{\mathcal{C}}(Y, Z) \times \mathcal{C}_{\mathcal{C}}(X, Y) \to \mathcal{C}_{\mathcal{C}}(X, Z).
\]

This is achieved by means of Day’s convolution monoidal structure on presheaves [14].

The above theory suffices for discussing multiplicative structure. Incorporating additive structure calls for the further refinement of enriching over distributive presheaves. This amounts to requiring that

\[
\mathcal{C}_{\mathcal{C}}(X, Y) \cong 1.
\]

In this presheaf-enriched context, the adjunction (1) amounts to giving natural bijections

\[
\mathcal{C}_{\mathcal{C}}(F P, N) \cong \mathcal{C}_{\mathcal{C}}(P, G N)
\]

that are invariant under the environment actions; the unit \( \eta \) and the counit \( \varepsilon \) of the adjunction are natural families of morphisms \( \eta_{a} \in \mathcal{T}_{\mathcal{C}}(P, G F P) \) and \( \varepsilon_{a} \in \mathcal{T}_{\mathcal{C}}(F G N, N) \), where \( I \) intuitively stands for the empty environment. Analogous bijective correspondences are used to describe type/logical structure in the vein of traditional category theory.

### 2.2 Polarised L Calculi

**Polarities: two modes of discourse**  
Our first description of polarisation was given by Girard [23] and underpins the interpretation of our calculi in adjunction models. Girard gives a denotational semantics to classical sequent calculus identifying \( P \) and \( \neg \neg A \), via a concrete interpretation based on coherent spaces, and a corresponding abstract construction, given as a negative translation into intuitionistic logic. The negative translation is made by “carefully distinguishing between negative formulas: simply negated, and positive formulas: doubly negated” [24]. However, there is a subtlety in this definition “since it is possible to consider a doubly negated formula as simply negated”. The interpretation then takes advantage of the focusing properties of connectives discovered earlier by Andreoli [1]—a well-studied aspect of polarisation, but which in our sense comes afterwards.

Here is how Girard’s idea is reflected in our interpretation. A cut:

\[
\Gamma_{1}, \Gamma_{2} \vdash \Delta
\]

is interpreted in two different ways depending on the polarity of \( A \). First, sequents are interpreted in the profunctor of oblique morphisms of the adjunction (1):

\[
\mathcal{T}_{\mathcal{C}}(F \neg\neg, \neg\neg) \cong \mathcal{T}_{\mathcal{C}}(\neg\neg, G \neg\neg)
\]

To be more precise, a sequent \( \Gamma, A \vdash B \) is interpreted as \( \mathcal{T}_{\mathcal{C}}(A^{\star}, B^{\star}) \), where \( \cdot^{\star} \) and \( \neg\neg \) suitably suitably add \( G \) and \( F \) wherever necessary.

Then, if \( A \) is positive, the cut is:

\[
\mathcal{T}_{\mathcal{C}}(A^{\star}, \neg\neg I) \times \mathcal{T}_{\mathcal{C}}(\neg\neg I, FA) \to \mathcal{T}_{\mathcal{C}}(\neg\neg I, \neg\neg I^{\neg\neg})
\]

and is therefore interpreted by a composition in \( \mathcal{T}_{\mathcal{C}}^{\star} \):

\[
\mathcal{T}_{\mathcal{C}}^{\star}(F A^{\star}, \neg\neg I) \times \mathcal{T}_{\mathcal{C}}^{\star}(FI, FA) \to \mathcal{T}_{\mathcal{C}}^{\star}(FI, \neg\neg I^{\neg\neg})
\]

which can be seen as generalising the Kleisli composition for the monad \( G F \). On the other hand, if \( A \) is negative, then the cut is:

\[
\mathcal{T}_{\mathcal{C}}^{\star}(A^{\star}, I) \times \mathcal{T}_{\mathcal{C}}^{\star}(I, FA) \to \mathcal{T}_{\mathcal{C}}^{\star}(I, \neg\neg I^{\neg\neg})
\]

and is interpreted by a composition in \( \mathcal{T}_{\mathcal{C}}^{\star} \):

\[
\mathcal{T}_{\mathcal{C}}^{\star}(G A^{\star}, \neg\neg I) \times \mathcal{T}_{\mathcal{C}}^{\star}(I, GA) \to \mathcal{T}_{\mathcal{C}}^{\star}(I, GA^{\star})
\]

which now can be seen as generalising the Kleisli composition for the comonad \( F G \). A new composition is thus obtained; which is
IMLL\textsubscript{p} and MLJ\textsubscript{p} (2/2)

**Judgements**

- \( \Delta \) refers to a pair \( \alpha : A \) of a co-variable and a type.
- \( \Gamma \) refers to a map from a finite set of variables to types provided with a total order \( \leq_T \) on its domain, notation \( \Gamma = (\chi_1 : A_1, \ldots, \chi_n : A_n) \).
- The notation \( \Gamma, \Gamma' \) implies that \( \Gamma \) and \( \Gamma' \) have disjoint domains.
- \( \Sigma(\Gamma; \Gamma') \) is the set of maps \( \sigma : \text{dom} \Gamma \to \text{dom} \Gamma' \) satisfying \( \Gamma'(\sigma(x)) = \Gamma(x) \) for all \( x \in \text{dom} \Gamma \).
- \( \Sigma^*(\Gamma; \Gamma') \) is the subset of \( \Sigma(\Gamma; \Gamma') \) consisting of bijective maps.
- Judgements are:
  \[ \Gamma \vdash t : A \mid \Gamma \vdash V : A \mid \Gamma \vdash e : A \vdash \Delta \mid \Gamma ; S : A \vdash \Delta \mid c : (\Gamma \vdash \Delta) \]

**Typing rules**

- \( \Gamma \vdash V : A; \quad \Gamma \vdash V' : A' \quad (\beta+) \)
  \[ c : (\Gamma \vdash a : N) \]
- \( \Gamma \vdash S : A \vdash \Delta \quad (\beta+) \)
  \[ c : (\Gamma \vdash a : P) \quad \Gamma \vdash \mu a^c : P \quad (\beta+) \]
  \[ x : A \vdash x : A; \quad (\beta a) \]
- \( \Gamma \mid \vdash S : A \vdash \Delta \quad (\beta+) \)
  \[ \Gamma ; \mu x^c : P \vdash \Delta \quad (\beta+) \]
  \[ \Gamma \mid \vdash e : A \vdash \Delta \quad (\beta+) \]
  \[ \alpha : A \vdash \alpha : A \quad (\alpha+) \]
- \( \Gamma \vdash V : N; \quad \Gamma \vdash e : N \vdash \Delta \quad (\text{cut}) \)
  \[ (V \parallel e)^* : (\Gamma, \Gamma' \vdash \Delta) \]

**IMLL\textsubscript{p} \(\forall \sigma \in \Sigma^*(\Gamma; \Gamma')\):**

- \( \Gamma \vdash V : A; \quad \Gamma \vdash V' : A' \quad (\beta+) \)
  \[ \Gamma ; S \vdash a \vdash \Delta \quad (\beta+) \]
  \[ c : (\Gamma \vdash x : P \vdash \Delta) \]
- \( \Gamma \vdash \mu x^c : P \vdash \Delta \quad (\beta+) \)
  \[ \Gamma \mid \vdash S : A \vdash \Delta \quad (\beta+) \]
  \[ \Gamma ; \mu x^c : P \vdash \Delta \quad (\beta+) \]
  \[ \alpha : A \vdash \alpha : A \quad (\alpha+) \]
  
**MLJ\textsubscript{p} \(\forall \sigma \in \Sigma(\Gamma; \Gamma')\):**

- \( \Gamma \vdash V : A; \quad \Gamma \vdash V' : A' \quad (\beta+) \)
  \[ \Gamma ; S \vdash a \vdash \Delta \quad (\beta+) \]
  \[ c : (\Gamma \vdash x : P \vdash \Delta) \]
- \( \Gamma \vdash \mu x^c : P \vdash \Delta \quad (\beta+) \)
  \[ \Gamma ; S : A \vdash \Delta \quad (\beta+) \]
  \[ \Gamma ; \mu x^c : P \vdash \Delta \quad (\beta+) \]
  \[ \alpha : A \vdash \alpha : A \quad (\alpha+) \]

**Coercions**

- \( (\alpha a) \quad \text{Coercions} \)
- \( (\beta a) \quad \text{Weakening and contraction} \)
- \( (\beta+) \quad \text{Identity} \)
- \( (\alpha+) \quad \text{Structures} \)
- \( (\beta+) \quad \text{Identity rule} \)
- \( (\alpha+) \quad \text{Cut rule} \)

**Derivable rules**

- \( \Gamma ; \mu x^c : P \vdash \Delta \quad \Gamma ; S \vdash a \vdash \Delta \quad (\beta+) \)
  \[ c : (\Gamma \vdash x : P \vdash \Delta) \]
- \( \Gamma \vdash V : A; \quad \Gamma \vdash W : B; \quad \Gamma \vdash V \circ W : A \odot B \)
  \[ \Gamma \vdash \mu(x^c \odot y^c) : c : A \odot B \vdash \Delta \quad (\beta+) \]
- \( \Gamma \vdash V : A; \quad \Gamma \vdash W : B; \quad \Gamma \vdash W \circ V : A \odot B \)
  \[ \Gamma \vdash \mu(x^c \odot y^c) : c : A \odot B \vdash \Delta \quad (\beta+) \]

**Figure 2.** Multiplicatives: legal terms and derivable rules

The calculi of polarisation

Our calculi are based on a second description of polarisation. Danos, Joinet and Schellinx [13] reconstruct Girard’s polarised classical logic as a distinguished way of endowing the classical sequent calculus LK with a confluent cut-elimination procedure, the system LK\textsubscript{p}. This is in fact the system in which the order of cut-elimination is preserved by \( \eta \)-expansions. The calculi IMLL\textsubscript{p} and MLJ\textsubscript{p} that we introduce are appropriate variants of LK\textsubscript{p}. The meaning of \( \eta \) is explained in the next section, where the support of \( \lambda \mu \rho \) L-calcus allow a concise presentation and an elegant theoretical development, along the lines of the \( \lambda \) calculus in the standard presentation of Barendregt [3].

On the proof-theoretic side, L-calcus can be understood as term assignments for sequent calculus rather than natural deduction. See [10, 47] for introductions to this aspect. On the side of computation, they can be understood as abstract-machine-like calculi, based on two principles, besides polarisation: the inside-out, defunctionalised representation of contexts is primitive, and language constructs are represented abstractly as solutions to their abstract-machine transitions. See [40] for an introduction based on this aspect.

Danos, Joinet and Schellinx recover other classical systems using annotations on formulae that determine the reduction differently, but do not affect provability. These annotations can be understood as the shifts \( \mathcal{I} \), \( \mathcal{J} \), \( \mathcal{K} \), \( \mathcal{A} \), \( \mathcal{B} \), and \( \mathcal{C} \) are always provably equivalent, albeit not isomorphic. In general, polarised calculi let us describe dif-
f erent reduction strategies using appropriately-shifted types. The fa
miliar decomposition of call-by-value and call-by-name functional
types, as given by Levy [31] and others, can thus be retrieved as
the composition of such annotations with the construction described
previously.

The two qualities we mentioned, compatibility with \( \eta \)-expansions
and derivability of known systems, are two main features of the
range of calculi that we present. This parallels the general and ab-
stract character of their enriched adjunction models.

3. Cartesian Polarised Structure

We start the development by focusing on the multiplicative intu-
itionistic (or cartesian) setting which we refer to as \( \text{MLJ}_p^\circ \). Type-
theoretically, it corresponds to the unit, strict product, and arrow
types. This is model-theoretically simpler than the linear (or sym-
metric monoidal) variant to be considered next.

3.1 \( \text{MLJ}_p^\circ \) and IMLL\(^*_p \) Calculi

The calculus \( \text{MLJ}_p^\circ \) (polarised multiplicative intuitionistic logic)
is introduced in Figures 1 and 2, at the same time with \( \text{IMLL}\(_p^\circ \) \)(po-
larised multiplicative intuitionistic linear logic). On the proof-
theteoric side, \( \text{MLJ}_p^\circ \) corresponds to intuitionistic sequent cal-
gus given with conjunction in multiplicative style (written \( \otimes \)), and
endowed with a polarised cut-elimination procedure. We write
the arrow type as \( \to \), which is to be understood as \( \to \) or \( \to \) depending on
whether the setting is \( \text{IMLL}\(_p^\circ \) \) or \( \text{MLJ}_p^\circ \). (In this paper, the symbol
\( \to \) does not refer to implication but to a compatible closure.)

**Barendregt-style \( \lambda \)-calculus**

The calculi are introduced along the lines of Barendregt’s presentation of the \( \lambda \)-calculus [3]. There is a distinction to make between the latter and the Church-style \( \lambda \)-calculus: in Church style, well-typed terms are directly defined
by induction, whereas in Barendregt style, first a syntax of pseudo-
terms is given, and then the legal terms are defined as those pseudo-
terms that are well-typed. In Church style, type annotations are
located on variables, whereas in Barendregt style, they are located
on binders.

The Barendregt technique consists in proving a **Basis Lemma**
(judgments assign a type to all the free variables, and non-free var-
iables can appear in the context only through weakening), followed by a **Generation Lemma** (the typing relation is determined by induc-
tion on the term). Various properties follow: uniqueness of typing,
decidability of type inference, typability of sub-terms, compatibil-
ity of typing (whereby typed reductions, defined by induction on
judgements, coincide with the legal restriction of reduction). The
**Substitution Lemma** follows (judgements are closed under substitu-
tion), as well as **Subject Reduction** (judgements are closed under
reduction). Finally, one proves that the typed reduction is confluent.
Here, all these properties will be referred to as the **Barendregt-style properties**.

**Description of the calculi \( \text{MLJ}_p^\circ \) and IMLL\(^*_p \)**

In Figure 1, the pseudo-terms of \( \text{MLJ}_p^\circ \) and IMLL\(^*_p \) are given with three main syn-
tactic categories (expressions \( t \), contexts \( e \), and commands \( c \)) as well
as two auxiliary syntactic categories (values \( V \) and stacks \( S \)). There are
explicit coercions \( \tau \) from values to expressions and from stacks
to contexts, which we omit by convention: we display values as
particular expressions, and stacks as particular contexts. Binders \( \mu \) and \( \bar{\mu} \) are annotated with types that are either positive (\( P \)) or nega-
tive (\( N \)).

Head-reductions \( \beta_R \) and \( \beta_E \) are defined between pseudo-terms.
Reductions \( \equiv_R \) and \( \equiv_L \) are obtained by compatible closure as fol-
 lows: one has \( f \to g \) whenever \( g \) is obtained from \( f \) by application
of \( \beta \) to a sub-term. Equivalence is obtained as \( \equiv \equiv (\equiv \cup \equiv) \).

Figure 1 defines the legal terms, *i.e.* well-typed pseudo-terms.
Judgements assign a type to expressions and values on the right,
and a distinguished type to contexts and stacks on the left, while
commands have no type by themselves. Expressions and contexts
determine the **active** (or main) type in the sequence.

The type system is given by rules of **coercion** between judge-
ments, together with the usual **identity**, **structure**, and **logic** groups
from linear sequent calculi. Notice that structural rules (exchange,
weakening, and contraction in the usual terminology) are merged
into a single set of rules; in addition they do not make the term
grow but act via a simultaneous substitution of variables for vari-
ables. For intuitionistic logic, this approach with explicit structural
rules might seem somewhat non-standard, but it is in line with the
viewpoint introduced by linear logic [22], and consists in formulating
\( \text{MLJ}_p^\circ \) as the variant of IMLL\(^*_p \) obtained by lifting the linearity
constraint on the left. The variants of the Basis Lemma, the Gen-
eration Lemma, the uniqueness of typing, and the compatibility of
typing that we obtain are strengthened to take into account such non
type-directed rules.

**Theorem 1.** \( \text{MLJ}_p^\circ \) and IMLL\(^*_p \) have the Barendregt-style properties.

In the theorem above and the rest of the paper, confluence is
restricted to \( \to_{\text{re}} \) where \( \equiv \) is the compatible closure of the four rules
(\( E\mu \)) and (\( E\mu' \)).

**Polarisation**

Danos, Joinet and Schellinx’s \( \rho \)-technique is summarised in L
calculi with two ingredients: the reduction order determined by
the polarity:

\[
\mu c [\mu x^c, e^c] \equiv_{\text{re}} (\mu x, C \parallel \mu x, C^c) \\
(\mu x^c, e^c) \equiv_{\text{re}} (\mu x, C) \equiv_{\text{re}} (\mu x, C) \\
\]

and the \( \eta \)-restric, whereby the notions of values and stacks must
be defined inductively. These two principles ensure that \( \to \) does
not affect the reduction order. Polarisation is also determined by
the presence of shifts. In IMLL\(^*_p \), \( \text{MLJ}_p^\circ \), and the other systems that we
consider, shifts are definable:

\[
\uparrow A \equiv \mathbf{1} \to A \\
\downarrow A \equiv A \otimes A \\
\]

**Theorem 2.** \( \text{LJ} \) without disjunction and IMLL are respectively expressible
in \( \text{MLJ}_p^\circ \) and IMLL\(^*_p \).

Above, **expressible** means that the rules of \( \text{LJ} \) without disjunc-
tion and IMLL are derivable in \( \text{MLJ}_p^\circ \) and IMLL\(^*_p \). In particular,
the latter systems are complete for provability.

3.2 \( \text{MLJ}_p^\circ \) Models

We need consider category-theoretic structure enriched over cate-
geries of presheaves. Background material is provided to keep the
development relatively self-contained.

**Presheaf enrichment**

For a category \( \mathcal{V} \), we write \( \mathcal{V} \) for the large
category of presheaves \( \text{Set}^{\mathcal{V}^{\text{op}}} \) and natural transformations
between them, and let \( \mathcal{P} \mathcal{V} \) be its full subcategory consisting of the small
presheaves. These are equivalently defined as those that are a small
colimit of representables, or the left Kan extension of their restric-
tion to some small full subcategory of their domain, or the left
Kan extension of some presheaf whose domain is small; see [15].
Note that every presheaf on a small category is small, and that
\( \mathcal{P} \mathcal{V} \) is \( \text{Set}^{\mathcal{V}^{\text{op}}} \) for every small category \( \mathcal{V} \).
The **Yoneda embedding** \( \mathcal{V} \hookrightarrow \mathcal{V} \), mapping an object \( X \) to the represen-
table presheaf \( \mathcal{V}(\cdot, X) \), restricts to a Yoneda embedding
\( \mathcal{V} \hookrightarrow \mathcal{P} \mathcal{V} \) that exhibits \( \mathcal{P} \mathcal{V} \) as the free cocompletion of \( \mathcal{V} \).
As for limit structure, \( \mathcal{P} \mathcal{V} \) may not even have a terminal object in general, and we hence-
forth restrict attention to those categories \( \mathcal{V} \) for which \( \mathcal{P} \mathcal{V} \) is carte-
sian closed. This is the case, for instance, for a small or for a carte-
sian closed $\mathcal{V}$; see [15].

A $\mathcal{P}^\mathcal{V}$-category $\mathcal{C}$ is specified by a class of objects $|\mathcal{C}|$ and hom-presheaves $\mathcal{C}(A, B)$ in $\mathcal{P}^\mathcal{V}$ for all $A, B \in |\mathcal{C}|$ together with identities $1 \to \mathcal{C}(A, A)$ and composition operations $\mathcal{C}(B, C) \times \mathcal{C}(A, B) \to \mathcal{C}(A, C)$ for all $A, B, C \in |\mathcal{C}|$ that are subject to the usual monoid laws. In this context, we will conveniently write $\mathcal{C}_x(A, B)$ for the action of the presheaf $\mathcal{C}(A, B) : \mathcal{V}^{op} \to \mathcal{Set}$ on an object $X$ in $\mathcal{V}$.

We shall use the notation $f : A \to B$ for $f \in \mathcal{C}_x(A, B)$. In elementary terms, thus, identities and composition are respectively given by elements of $\mathcal{C}_x(A, A)$ and maps $\mathcal{C}_x(B, C) \times \mathcal{C}_x(A, B) \to \mathcal{C}_x(A, C)$ natural for $X$ in $\mathcal{V}$.

A $\mathcal{P}^\mathcal{V}$-functor $F : \mathcal{A} \to \mathcal{B}$ between $\mathcal{P}^\mathcal{V}$-categories is given by a mapping $F : |\mathcal{A}| \to |\mathcal{B}|$ and functorial mappings $F^\mathcal{V}_Y : \mathcal{C}_x(X, Y) \to \mathcal{D}_x(FX, FY)$ natural for $Z$ in $\mathcal{V}$. Further, a $\mathcal{P}^\mathcal{V}$-natural transformation $\varphi : F \to G : \mathcal{A} \to \mathcal{B}$ between $\mathcal{P}^\mathcal{V}$-functors consists of a family $\varphi^x \in \mathcal{D}_x(FA, GA)$ natural for $X$ in $\mathcal{V}$ and $A \in |\mathcal{A}|$. The category $\mathcal{P}^\mathcal{V}$, being assumed cartesian closed, underlies a $\mathcal{P}^\mathcal{V}$-category $\mathcal{P}^\mathcal{V}$ and the Yoneda embedding $y : \mathcal{V} \hookrightarrow \mathcal{P}^\mathcal{V}$ induces a $\mathcal{P}^\mathcal{V}$-enrichment on $\mathcal{V}$. Indeed, every category $\mathcal{V}$ underlies the $\mathcal{P}^\mathcal{V}$-category $\mathcal{V}$ with objects that of $\mathcal{V}$ and hom-presheaves given by the exponential structure of $\mathcal{P}^\mathcal{V}$; that is, with $\mathcal{V}(X, Y) = \mathcal{P}^\mathcal{V}(yX, yY)$. In particular, for cartesian $\mathcal{V}$, one has that $\mathcal{V}(X, Y) = \mathcal{V}(Z \times X, Y)$. The Yoneda embedding enriches to a $\mathcal{P}^\mathcal{V}$-functor.

A $\mathcal{P}^\mathcal{V}$-category is said to be powered whenever it has powers; represented, for presheaves $P$ in $\mathcal{P}^\mathcal{V}$ and objects $A \in |\mathcal{V}|$, by an object $[P \to A]$ in $|\mathcal{V}|$ together with an isomorphism $\mathcal{V}(C, [P \to A]) \cong \mathcal{P}^\mathcal{V}(\mathcal{P}, \mathcal{C}(C, A))$ $\mathcal{P}^\mathcal{V}$-natural for $C$ in $|\mathcal{V}|$. Powers with respect to representable presheaves are referred to as $\mathcal{V}$-powers.

A $\mathcal{P}^\mathcal{V}$-adjunction $\mathcal{A} \rightleftarrows \mathcal{B}$ consists of $\mathcal{P}^\mathcal{V}$-functors together with a representation $\mathcal{B}_x(FA, B) \equiv \mathcal{A}_x(A, GB)$ natural for $X$ in $\mathcal{V}$, for $A \in |\mathcal{A}|$, and for $B$ in $|\mathcal{B}|$.

**Adjunction models** The notion of model for IMLL\textsuperscript{L} follows; note that this encompasses ML\textsuperscript{L} models.

**Definition 4.** An IMLL\textsuperscript{L} model consists of a symmetric monoidal category $\mathcal{I}$ and a $\mathcal{P}^\mathcal{I}$-category $\mathcal{S}$ with $\mathcal{I}$-powers together with a $\mathcal{P}^\mathcal{I}$-adjunction $\mathcal{I} \rightleftarrows \mathcal{S}$.

To aid the understanding of this structure, we remark that in elementary terms a $\mathcal{P}^\mathcal{I}$-category $\mathcal{S}$ is specified by a class of objects $|\mathcal{S}|$ and hom-presheaves $\mathcal{S}(M, N)$ in $\mathcal{P}^\mathcal{I}$ for all $M, N \in |\mathcal{S}|$ together with identities in $\mathcal{S}_0(N, N)$ and composition operations $\mathcal{S}_0(N, N') \times \mathcal{S}_0(N', N'') \to \mathcal{S}_0(N, N'')$ natural for $P, Q$ in $\mathcal{I}$ and subject to monoid laws.

In addition, $\mathcal{I}$-powers are represented, for objects $P$ in $\mathcal{I}$ and $N \in |\mathcal{S}|$, by an object $[P \to N]$ in $|\mathcal{S}|$ together with an isomorphism $\mathcal{S}_0(M, [P \to N]) \cong \mathcal{S}_0 \otimes \mathcal{S}_0(M, N)$ natural for $Q$ in $\mathcal{I}$ and for $M$ in $|\mathcal{S}|$. Therefore, there are evaluation maps $[P \to N] \to_P N$ in $|\mathcal{S}|$.

**Lemma 5.** For every $P$ in $\mathcal{I}$ and $N$ in $|\mathcal{S}|$, we have an isomorphism $\mathcal{I}(Q \otimes P, GN) \cong \mathcal{S}(Q, G(P \to N))$ natural for $Q$ in $\mathcal{I}$. Therefore, there are evaluation maps $G[\mathcal{P} \to N] \otimes \mathcal{I} \to \mathcal{I}$.

**Example 6.** Every symmetric monoidal closed category $\mathcal{I}$, powered $\mathcal{I}$-category $\mathcal{S}$, and $\mathcal{P}^\mathcal{I}$-adjunction $\mathcal{I} \rightleftarrows \mathcal{S}$ provide a IMLL\textsuperscript{L} model. Such examples include:

1. $\mathcal{S} = \mathcal{I}$ with the monoidal-closure adjunction $(-) \otimes P \to [P \to -]$ for $P$ in $\mathcal{I}$, and $\mathcal{S} = \mathcal{I}^{op}$ with the linear double-dualisation adjunction $[- \to -] P \to [- \to P]$ for $P$ in $\mathcal{I}$.

**Remark 7.** The second example above was considered in the linear case by Melliriës and Tabareau [33]. It has extra structure in the form of an obvious strictly involutive negation $\mathcal{I}^{op} \rightleftarrows \mathcal{S}$ and thus gives models of polarised classical and linear logic. Its proof theory in fact led to the introduction of polarisation [23].

Corresponding systems LK\textsuperscript{L} (essentially Girard’s LC, see [13]) and LL\textsuperscript{L} can be deduced from LJ\textsuperscript{L} and ILL\textsuperscript{L} by adding a (strict or not) involutive negation by following the ideas in [36, 38]. This perspective is novel compared to the original intuitions involving “linear continuations” [17]: it does not presuppose that evaluation order is irrelevant and, as we will see, is more faithful to the focusing properties of linear logic.
Example 8. Let \( C \) be a small symmetric monoidal category with tensor product \( \cdot \). For every presheaf \( S \in \mathcal{P}(C^{op}) \), we have an adjoint situation:
\[
\mathcal{P}C \xrightarrow{\iota_S} \mathcal{P}(C^{op})
\]
where \( F_S(P) = \int_{z\in C} S(z \cdot c) \times P(c) \) and \( G_S(N) = \int_{z\in C} S(z \cdot c) \Rightarrow N(z) \). Each of these adjunctions enriches an \( \text{IMLL}_P^{\text{en}} \) model by taking \( \mathcal{P} \) to be \( \mathcal{P}C \) and \( \iota \) to be the \( \mathcal{P}\mathcal{I}\mathcal{M} \)-enrichment of \( \mathcal{P}(C^{op}) \) with homs \( \mathcal{P}\mathcal{I}\mathcal{M}(M, N) = \mathcal{P}(P, G_M(N)) \). The monads \( G_S F_S \) are therefore given by the formula
\[
G_S F_S(P) = \int_{z\in C} S(z \cdot c) \Rightarrow \int_{z\in C} S(z \cdot c) \times P(c)
\]
and may be computationally interpreted as linear models for local store. There is in fact more mathematical structure in these models, details of which will appear elsewhere.

4.2 \( \text{IMLL}_P^{\text{en}} \) Semantics

We present the categorical interpretation of \( \text{IMLL}_P^{\text{en}} \).

Interpretation of types

to every type \( A \), we associate a positive interpretation \( A^+ \in \mathcal{S} \) and a negative interpretation \( A^- \in \mathcal{S} \). These are defined by mutual induction as follows:
\[
\begin{align*}
  & (X^+) = \text{an assigned object of } \mathcal{S}; I^+ = (\Gamma; A \otimes B)^+ = A^+ \otimes B^+; \\
  & N^+ = GN^+ \text{ for every negative type } N; \\
  & (X^-) = \text{an assigned object of } \mathcal{S}; (A \Rightarrow B)^- = [A^+ \Rightarrow B^+]; \text{ and } \\
  & P^- = FP^- \text{ for every positive type } P.
\end{align*}
\]

The interpretation of types extends pointwise to typing contexts as follows:
\[
\begin{align*}
  & (x_1 : A_1, \ldots, x_n : A_n)^+ = (\ldots (I \otimes A_1^+) \otimes \cdots) \otimes A_n^+; \\
  & (a : A)^- = A^-.
\end{align*}
\]

Interpretation of judgements

denotation of \( V : A ; B \), \( S : A \vdash \Delta \), \( \Gamma \vdash \tau : \Delta \), as explained in Section 3.1. In fact we fix it arbitrarily as follows:
\[
\begin{align*}
  & \beta_1 : A \vdash \Gamma^+ \Rightarrow GA^+ \text{ in } \mathcal{S}; \\
  & \beta_2 : S : A \vdash \Delta^- \Rightarrow A^+ \text{ in } \mathcal{S}; \\
  & \beta_3 : \Gamma \vdash \tau : \Delta^- \Rightarrow \text{the transpose of the map } \Gamma^+ \Rightarrow A^+ \Rightarrow \Delta^- \text{ in } \mathcal{S}.
\end{align*}
\]

The interpretation is defined by mutual induction on derivations as follows.

Coercions:
\[
\begin{align*}
  & \beta_1 : \Gamma \vdash \tau : \Delta^- \Rightarrow \text{the transpose of the map } \Gamma^+ \Rightarrow A^+ \Rightarrow \Delta^- \text{ in } \mathcal{S}; \\
  & \beta_2 : S : A \vdash \Delta^- \Rightarrow A^+ \text{ in } \mathcal{S}; \\
  & \beta_3 : \Gamma \vdash \tau : \Delta^- \Rightarrow \text{the transpose of the map } \Gamma^+ \Rightarrow A^+ \Rightarrow \Delta^- \text{ in } \mathcal{S}.
\end{align*}
\]

Identity rules:
\[
\begin{align*}
  & \beta_1 : \Gamma \vdash x : A ; \beta_2 : I \otimes A^+ \Rightarrow A^+ \text{ is the canonical isomorphism; } \\
  & \beta_3 : [; \alpha : A \vdash \alpha : A] \Rightarrow A^+ \text{ is the identity; } \\
  & \beta_4 : \Gamma \vdash \tau : \Delta^- \Rightarrow \text{the transpose of the map } \Gamma^+ \Rightarrow A^+ \Rightarrow \Delta^- \text{ in } \mathcal{S}.
\end{align*}
\]

Structural rules:
\[
\begin{align*}
  & \beta_1 : \Gamma \vdash \tau : \Delta^- \Rightarrow \text{the transpose of the map } \Gamma^+ \Rightarrow A^+ \Rightarrow \Delta^- \text{ in } \mathcal{S}. \\
  & \beta_2 : S : A \vdash \Delta^- \Rightarrow \text{the transpose of the map } \Gamma^+ \Rightarrow A^+ \Rightarrow \Delta^- \text{ in } \mathcal{S}.
\end{align*}
\]

Logical rules:
\[
\begin{align*}
  & \beta_1 : \Gamma \vdash \tau : \Delta^- \Rightarrow \text{the transpose of the map } \Gamma^+ \Rightarrow A^+ \Rightarrow \Delta^- \text{ in } \mathcal{S}. \\
  & \beta_2 : S : A \vdash \Delta^- \Rightarrow \text{the transpose of the map } \Gamma^+ \Rightarrow A^+ \Rightarrow \Delta^- \text{ in } \mathcal{S}.
\end{align*}
\]

Theorem 9. The \( \text{IMLL}_P^{\text{en}} \) semantics is coherent: all derivations of a judgment have the same interpretation; and sound: if \( \vdash \tau : \Delta^- \Rightarrow \text{the transpose of the map } \Gamma^+ \Rightarrow A^+ \Rightarrow \Delta^- \text{ in } \mathcal{S}.
\]

4.3 \( \text{MLL}_P^{\text{en}} \) Semantics

The categorical interpretation of \( \text{MLL}_P^{\text{en}} \) derivations in its models from Section 3.2 extends that of \( \text{MLL}_P^{\text{en}} \) using the canonical cartesian interpretation \([\sigma] : \Gamma^+ \Rightarrow \Gamma^+ \text{ of maps } \sigma \in \Sigma(\Gamma; \Gamma)\).

Theorem 10. The \( \text{MLL}_P^{\text{en}} \) semantics is coherent and sound.
4.4 Depolarisation

Polarisation, in $\text{MLL}_{\lambda}$, $\text{IMLL}_{\lambda}$, and their extensions, is represented by the fact that the following equations (here given with an orientation) are not always derivable from $\equiv_{\text{RE}}$ when $A$ is positive and $B$ is negative:

\begin{align}
\langle t \parallel \mu x \cdot \langle \mu a \cdot c \parallel e \rangle \rangle_B & \triangleright \langle \mu a \cdot \langle t \parallel \mu x \cdot c \parallel e \rangle \parallel e \rangle \rangle_A \quad (2) \\
\langle t \parallel \mu x \cdot c \rangle & \triangleright \langle e[t/x] \rangle \quad (3) \\
c[e/a] & \triangleright \langle \mu a \cdot \mu c \parallel e \rangle \quad (4)
\end{align}

The equation $\triangleright$ corresponds to an associativity of cuts. The notations $[t/x]$ and $[c/a]$ refer to the substitutions of $t$ for $x$ and of $e$ for $a$ under the condition that every occurrence of $x$ is of the form $(x \parallel S)^*$, and every occurrence of $a$ is of the form $(V \parallel a)^*$. (They can equivalently be defined as standard substitutions, in the sense of higher-order rewriting, by adding the so-called "$\triangleright$-rules".)

These equations are in fact equal modulo $\equiv_{\text{RE}}$ in the following sense:

**Lemma 11.** Let $A$ be a type and $t$ be an expression. We denote with $\triangleright_{\Lambda}$ and $\triangleright_{\Gamma}$ the relations given respectively by (2) and (3) quantified over arbitrary $B$, $c$ and $e$. Symmetrically, let $B$ be a type and $e$ a context. We denote with $\triangleright_{\Lambda_{\nu}}$ and $\triangleright_{\Gamma_{\nu}}$ the relations given respectively by (2) and (4) quantified over arbitrary $A$, $c$ and $t$. One has:

$$\equiv_{\text{RE}} \triangleright_{\Lambda \nu} \equiv_{\text{RE}} \triangleright_{\Lambda_{\nu}} \equiv_{\text{RE}} \triangleright_{\Gamma_{\vdash}} \equiv_{\text{RE}} \triangleright_{\Gamma}$$

With the additional constraint that $c[\mu a \cdot (t \parallel \mu a / c) / x]$ (in the case of $\triangleright_{\vdash}$) and $c[\mu a \cdot e / x]$ (in the case of $\triangleright_{\vdash}$) are typable, these equalities restrict to the corresponding typed relations between legal terms.

In particular, $\triangleright_{\Lambda}$, $\triangleright_{\Gamma}$ and $\triangleright_{\nu}$ are equal modulo $\equiv_{\text{RE}}$. Notice that the final typing constraint is a linearity constraint in the case of $\text{IMLL}_{\lambda}$ and its extensions. A model is depolarised if it validates either of these equivalent equation schemes. The following theorem characterises depolarised models and extends the characterisation in [39].

**Theorem 12.** Let $\mathcal{L}$ be an $\text{IMLL}_{\lambda}$ model. Every interpretation satisfies the typed restriction of (2) if and only if the adjunction is idempotent.

This result generalises the depolarisation criterion for dialogue categories [33]. There, categorial models of linear logic are built from polarised ones by enforcing commutativity of the strong monad $\triangleright$. For the adjunction $\triangleright \dashv \triangleright$, being idempotent is equivalent to $\triangleright$ being commutative (Führmann [21]).

**Example 13.** Erratic choice [31, Section 5.5], based on the powerset monad, is an example of a commutative, but non-idempotent effect.

5. Resource Modalities

We now focus on resource structure relating linear models to cartesian ones. In the game, we give a treatment of the exponential modality $!$ consistent with its focusing properties. Its sound interpretation by means of resource structure extends that of Melliès and Tabareau [33].

5.1 $\text{IMLL}_{\lambda}^p$ Calculus

$\text{IMLL}_{\lambda}^p$ is defined in Figure 3 by adding to $\text{IMLL}_{\lambda}$ the exponential modality $!$ from linear logic [22]. Unlike the other connectives, the pattern-matching form $(\mu a \cdot c)$ is a positive expression; it corresponds to promotion. For this reason, it is difficult to provide syntactic sugar for $\text{IMLL}_{\lambda}^p$ in $\lambda$-calculus style, as was noticed for linear logic in its early days [4, 5, 45, 46]. A main novelty of our syntax is that contraction and weakening are treated by merging and introducing variables in the context, instead of introducing explicit grammatical counterparts to the logical rules. This treatment is similar to [36] but is now substantiated by a semantic interpretation.

Our definition reflects the complex focalisation properties of $\lambda$-calculus on the right (meaning that the rule $(E!)$ is ill-typed unless the context is of the form $\Gamma!$ and not-invertible on the left. A key property to notice is that if $\Gamma \vdash V : !A$, then $\Gamma$ is of the form $\Gamma!$, by analysis on the derivation. Thus, restricting the rule $(E!)$ to values ensures that it is valid. For the same reason, any reduction $\langle V \parallel \mu x \cdot c \rangle$ $\triangleright_R c[V/x]$, which may duplicate or erase $V$ due to contractions or weakening in $x$, is valid because the same contractions or weakening apply to the typing context of $V$, necessarily of the form $\Gamma!$.

**Theorem 14.** $\text{IMLL}_{\lambda}^p$ is expressible in $\text{IMLL}_{\lambda}^p$.

**Theorem 15.** $\text{IMLL}_{\lambda}^p$ has the Barendregt-style properties.

5.2 $\text{IMLL}_{\lambda}^p$ Models

**Adjunction models** We extend $\text{IMLL}_{\lambda}^p$ models with resources in the form of a monoidal adjunction; when this is linear/non-linear [4] it describes $\text{IMLL}_{\lambda}^p$ models. 

**Definition 16.** An $\text{IMLL}_{\lambda}^p$ model with a resource modality consists of an $\text{IMLL}_{\lambda}^p$ model $\mathcal{L} \xrightarrow{\otimes} \mathcal{K}$ together with a symmetric monoidal category $\mathcal{K}$ and a monoidal adjunction $\mathcal{K} \xleftarrow{\otimes} \mathcal{L}$. An $\text{IMLL}_{\lambda}^p$ model is an $\text{IMLL}_{\lambda}^p$ model with a resource modality in which the symmetric monoidal structure of $\mathcal{K}$ is cartesian.

**Example 17.** The dialogue categories with a resource modality of Melliès and Tabareau [33] are $\text{IMLL}_{\lambda}^p$ models with a resource modality.

**Semantics** For an $\text{IMLL}_{\lambda}^p$ model with a resource modality as above, we write $E$ for the monoidal comonad on $\mathcal{L}$ induced by the monoidal adjunction, and define:

$$(!A)^{-} \overset{def}{=} EGA^{-}$$

The terms associated to the exponential are then interpreted as follows:

- $[[\Gamma \vdash \mu a \cdot c : !A ; ]] : EG^{-}\rightarrow EGA^{-}$ is obtained by pre-composing $E[c : (\Gamma \vdash \mu a : A)] : E^G\rightarrow EGA^{-}$ with the comultiplication $E^G\rightarrow E^G\otimes E^G$ of $E$;
- $[[\Gamma ; S : A \vdash \Delta ]] : FEG^{-}\rightarrow_{\mu} A^{-}$ arises from the map $[\Gamma ; S : A \vdash \Delta ] : A^{-}\rightarrow_{\mu} A^{-} \otimes G^{-}$ by pre-composition with the composite $FEG^{-}\rightarrow_{\mu} FEG^{-}\rightarrow_{\gamma} A^{-}$ where the first map is obtained from the application of $F$ to the map $I \otimes EGA^{-}\rightarrow G^{-}A^{-}$ arising from the counit of $E$ and the second map is the counit of $F^{-}\dashv G$.

Assuming that the model is one of $\text{IMLL}_{\lambda}^p$, the interpretation of the structural rules is as before relying on a functorial interpretation $[[\sigma]] : \Gamma^{-}\rightarrow \Gamma^{p}$ of maps $\in \Sigma(\Gamma, \Gamma^{p})$:

- maps $(x_1 : A, \ldots, x_n : A) \rightarrow (x : A)$ are interpreted by

$$(x : A) \equiv A^x \overset{\lambda_{A^x}}{\rightarrow} (A^x)^{\otimes n} \overset{\equiv}{\rightarrow} (x_1 : A, \ldots, x_n : A) \equiv (n \geq 0)$$

\footnote{While the linear/non-linear syntax [4] does offer a similar treatment for structural rules on the non-linear part of the sequent, deriving the actual structural rules of linear logic amounts to encoding contraction and weakening with the explicit syntactic constructs that we manage to avoid.}
We establish a model-theoretic lifting theorem for IMELL_p models as follows:

**Grammar and conversions (pseudo-terms)**

\[ P, Q ::= \ldots \mid A \]

(a) Types

values \( V, W \ ::= \ldots \mid \mu \alpha \delta \epsilon \)

stacks \( S ::= \ldots \mid !S \)

(b) Pseudo-terms

\begin{align*}
(\text{R}) & : \mu \alpha \delta \epsilon \parallel !S \triangleright \epsilon[S/\alpha] \\
(\text{E}) & : \mu \alpha \delta \epsilon (V \parallel !a) \triangleright \epsilon V
\end{align*}

(c) Reduction and expansion

**Typing rules and derivable rules (legal terms)**

- \( \Sigma(\Gamma; \Gamma') \) is the subset of \( \Sigma(\Gamma; \Gamma') \) consisting of the maps that restrict to a bijection on the variables that are not of the type \( !A \).

\[ c : (\Gamma \vdash \alpha : A) \]

\[ \Gamma \vdash \mu \alpha \delta \epsilon : !A \]

and the rules from Figure 2c extended to all \( \sigma \in \Sigma(\Gamma; \Gamma') \).

(d) Structure

\[ \Gamma \vdash \epsilon !A \vdash \Delta \]

\[ \Gamma \vdash !L \epsilon !A \vdash \Delta \]

\[ !Le \overset{\text{def}}{=} \mu \alpha \delta \epsilon \langle \mu \alpha \delta \epsilon (x \parallel !a) \parallel \epsilon \rangle \]

\[ \text{(c) Remaining rule of sequent calculus} \]

Figure 3. Exponentials

where \( \delta_{1}^{(n)} \) is the identity for \( n = 1 \) and otherwise, writing \( L \dashv K \) for the adjoint resolution of \( E \), given by the canonical maps

\[ LKP = \downarrow K P \overline{L}P \overline{L} = (LKP)^{\text{op}} \]

with \( P = GB^ \perp \) for \( A = !B \); arbitrary maps \( \sigma : \Gamma \longrightarrow (x_{1} : A_{1}, \ldots, x_{n} : A_{n}) \) are interpreted by

\[ (x_{1} : A_{1}, \ldots, x_{n} : A_{n})^{*} = (x_{1} : A_{1})^{*} \otimes \cdots \otimes (x : A_{n})^{*} \]

\[ \sigma_{1}^{*} \otimes \cdots \otimes \sigma_{n}^{*} = (\Gamma^{+} \otimes \cdots \otimes \Gamma_{n}^{+}) = \Gamma^{-} \]

where \( \Gamma^{-} \) is the restriction of \( \Gamma \) to \( \sigma^{-1}(x_{i}) \).

**Theorem 18.** The IMELL_p semantics is coherent and sound.

In the particular case of dialogue categories, this interpretation corresponds, when taking into account the presence of an involutive negation, to Melliès and Tabareau’s “focalised” translation of linear logic into tensor logic [33], but for the omission of a shift in their definition that our interpretation corrects.

In fact, this interpretation suggests a decomposition of the exponential \( !A \) in the calculus as a composite \( \Gamma \vdash A \) where \( \Gamma \) is interpreted with \( (\alpha) \Gamma^{*} = E \alpha^{*} \), not investigated here. This new connective \( \Gamma \) has to be understood as a proto-exponential whose promotion rule is restricted to values. As the value restriction of polymorphism, it is the constraint on the typing context that prevents the immediate lifting of this restriction to values.

**5.3 Lifting Theorem**

We establish a model-theoretic lifting theorem for IMELL_p models with a resource modality. In the context of exponential structure it corresponds to a Girard decomposition of cartesian into linear structure.

Every functor \( L : \mathcal{K} \rightarrow \mathcal{L} \) induces the adjoint situation \( L \dashv L^{*} : \mathcal{P}\mathcal{L} \rightarrow \mathcal{P}\mathcal{K} \) where \( L \) is the universal cocontinuous extension of \( L \) and \( L^{*} \) is the symmetric strong monoidal functor, as it is the case when it is a (symmetric) monoidal left adjoint [26], then the adjunction \( L \dashv L^{*} \) is (symmetric) monoidal [25].

In this situation, as it happens with every monoidal functor, one obtains a 2-functor mapping a \( \mathcal{P}\mathcal{L} \)-category \( \mathcal{C} \) to the \( \mathcal{P}\mathcal{K} \)-category \( L^{*}\mathcal{C} \) with objects those of \( \mathcal{C} \) and hom-presheaves \( L^{*}\mathcal{C}(A,B) = L^{*}(\mathcal{C}(A,B)) \). This construction has the following closure property:

- if \( \mathcal{C} \) has \( \mathcal{L} \)-powers \( [\sim \sim] \) then \( L^{*}\mathcal{C} \) has \( \mathcal{K} \)-powers \( [\sim \sim] \).

These are given by

\[ \mathcal{K}(X,Y) \longrightarrow \mathcal{L}^{*}\mathcal{K}(LX,LY) \]

with the property that \( L \) is a left \( \mathcal{P}\mathcal{K} \)-adjoint whenever \( L \) is a monoidal left adjoint. In particular, note that for a monoidal adjunction \( L \dashv K : \mathcal{L} \rightarrow \mathcal{K} \) one has

\[ L^{*}\mathcal{K}(LX,LY) \cong \mathcal{L}(LX \otimes L(-),Y) \]

\[ \cong \mathcal{L}(LX \otimes -,Y) \cong \mathcal{K}(X,Y) \]

To summarise, we have the following lifting theorem.

**Theorem 19.** Every IMELL_p model with a resource modality

\[ \mathcal{L} \rightarrow \mathcal{K} \]

induces an IMELL_p model, obtained by composing the two adjunctions:

\[ \mathcal{K} \rightarrow \mathcal{L}^{*}\mathcal{K} \rightarrow L^{*}\mathcal{K} \]

**6. Additive Structure**

We now consider additive connectives, that is to say positive co-products and negative products. Computationally, they correspond to strict sums and lazy pairs. The semantic subtlety is due to co-products, whereas the model theory of the negative cartesian product is simpler and could have been introduced earlier in Section 4.
\[
\text{IMALL}^n_p / \text{LJ}^0_p = \text{IMLL}^n_p / \text{MLL}^n_p \quad \text{+ additives as follows:}
\]

**Grammar and conversions (pseudo-terms)**

\[
\begin{align*}
N, M & := \ldots \mid A \& B \mid T \\
\text{values} & \quad V, W ::= \ldots \mid t_1(V) \mid t_2(V) \mid \mu < a^\alpha c; \beta^\beta c' > \mid \mu < V > \\
P, Q & := \ldots \mid A \& B \mid \emptyset \\
\text{stacks} & \quad S ::= \ldots \mid \pi_1 S \mid \pi_2 S \mid \mu[x^\lambda c; y^\beta c'] \mid \mu[S] \\
\end{align*}
\]

(a) Types

\[
\begin{align*}
(R\&)(\mu < a^\alpha c_1; c_2 > \parallel \pi_7; S) & \quad \vdash_R c_1[S/\pi_7] \\
(R\oplus)(\pi_7(V)) & \quad \vdash_R c_2[V/x_i] \\
\end{align*}
\]

(no reduction RT)

(no reduction R0)

(c) Reductions

\[
\begin{align*}
(E\&)(\mu < \alpha. (V \parallel \pi_7; \beta)) & \quad \vdash_E V \\
(E\oplus)(\pi_7(V)) & \quad \vdash_E S \\
\end{align*}
\]

(Typing rules (legal terms))

\[
\begin{align*}
\Gamma; S : A_i \vdash \Delta & \quad \frac{\Gamma; \pi_i, S : A_i \& A_j \vdash \Delta}{(\alpha_i \gamma)} \\
\Gamma \vdash V : A_i & \quad \frac{\Gamma; \vdash \pi_i, S : A_i \& A_j \vdash \Delta}{(\beta_i \gamma)} \\
\end{align*}
\]

(βη)

(η)

(Logic)

\[
\begin{align*}
\Gamma ; e : A_i \vdash \Delta & \quad \frac{\Gamma ; \pi_i, e : A_i \& A_j \vdash \Delta}{(\delta_i \gamma)} \\
\Gamma \vdash t : A_i & \quad \frac{\Gamma ; \vdash t : A_i \& A_j \vdash \Delta}{(\beta_i \gamma)} \\
\end{align*}
\]

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)

(η)
Positive additive structure  In the linear setting we consider the following.

**Definition 23.** A linear distributive category is a symmetric monoidal category with finite coproducts in which the distributive law of tensor products over coproducts holds; that is, the canonical map \( \coprod_{i \in I} X_i \otimes Y \to (\coprod_{i \in I} X_i) \otimes Y \) is an isomorphism, for all finite families of objects \( \{X_i\}_{i \in I} \) and object \( Y \).

Distributive presheaves  Incorporating coproducts into IMLL\(^p\) models (Definition 4) requires a further analysis of presheaves crucial to which is the following definition (cf. Fiore, Di Cosmo and Balat [19, Section 3.1]).

**Definition 24.** For a cocartesian category \( \mathcal{L} \), a presheaf \( P : \mathcal{L}^{op} \to \textbf{Set} \) is said to be distributive whenever it maps coproducts in \( \mathcal{L} \) to products in \( \textbf{Set} \), that is, for all finite families of objects \( \{X_i\}_{i \in I} \) of \( \mathcal{L} \), the canonical map \( P(\coprod_{i \in I} X_i) 

We let \( \mathcal{D} \mathcal{V} \) be the full subcategory of \( P \mathcal{V} \) consisting of all the distributive presheaves.

For a linear distributive category \( \mathcal{L} \), the convolution monoidal structure on \( P \mathcal{L} \) restricts to \( \mathcal{D} \mathcal{L} \) yielding a strong monoidal Yoneda embedding \( y : \mathcal{L} \hookrightarrow \mathcal{D} \mathcal{L} \) that induces a \( \mathcal{D} \mathcal{L} \)-category (with respect to the convolution monoidal structure) \( \mathcal{D} \mathcal{L} \). Furthermore, \( \mathcal{D} \mathcal{L} \) is a full reflective linear-exponential ideal of \( P \mathcal{L} \) and thereby underlies a \( \mathcal{D} \mathcal{L} \)-category (with respect to the convolution monoidal structure) \( \mathcal{D} \mathcal{L} \) for which the Yoneda embedding is a \( \mathcal{D} \mathcal{L} \)-functor.

We remark that a \( \mathcal{D} \mathcal{L} \)-category is a \( P \mathcal{L} \)-category in which every hom-presheaf is distributive, while the \( \mathcal{D} \mathcal{L} \)-enriched notions of functor, natural transformation, cartesian structure, and adjunctions coincide with those for \( P \mathcal{L} \)-enrichment. For linear distributive \( \mathcal{L} \), the same holds for powers.

Negative additive structure  A \( \mathcal{D} \mathcal{L} \)-category \( \mathcal{C} \) is cartesian whenever every finite family of objects \( \{A_i\}_{i \in I} \) can be represented by an object \( \prod_{i \in I} A_i \), together with an isomorphism

\[
\mathcal{C}(B, \prod_{i \in I} A_i) \cong \prod_{i \in I} \mathcal{C}(B, A_i)
\]

natural for \( X \) in \( \mathcal{L} \) and \( \mathcal{L} \)-natural for \( B \) in \( \mathcal{C} \).

Adjunction models  The above leads to the following notion of model.

**Definition 25.** An IMALL\(^p\) model consists of a linear distributive category \( \mathcal{L} \) and a cartesian \( \mathcal{D} \mathcal{L} \)-category \( \mathcal{S} \) with \( \mathcal{D} \mathcal{L} \)-powers together with a \( \mathcal{D} \mathcal{L} \)-adjunction \( \mathcal{L} \leftrightarrow \mathcal{S} \).

In view of the remarks above on the relationship between the \( \mathcal{D} \mathcal{L} \) and \( P \mathcal{L} \) enriched notions involved in IMLL\(^p\) and IMALL\(^p\) models, there is only one difference between the two notions; namely, that the latter comes with cartesian structure in \( \mathcal{S} \) and linear distributive structure in \( \mathcal{L} \) crucially satisfying

\[
S_{\prod_{i \in I} (M, N)} \cong \prod_{i \in I} S_{R_i}(M, N) \tag{5}
\]

for all finite families of objects \( \{P_i\}_{i \in I} \) in \( \mathcal{L} \) and objects \( M, N \) in \( \mathcal{S} \). More precisely, we have the following.

**Proposition 26.** An IMALL\(^p\) model \( \mathcal{L} \leftrightarrow \mathcal{S} \) is an IMALL\(^p\) model iff \( \mathcal{L} \) is linear distributive, \( \mathcal{S} \) is cartesian, and every hom-presheaf of \( \mathcal{S} \) is distributive.

Note that the refinement from enriching over \( P \mathcal{L} \) to enriching over \( \mathcal{D} \mathcal{L} \) guarantees that:

\[
\mathcal{S}_{\prod_{i \in I} (M, N)} \cong 1 \quad S_{\coprod_{i \in I} R_i}(M, N) \cong S_{\coprod_{i \in I} R_i}(M, N) \times S_{\coprod_{i \in I} R_i}(M, N)
\]

We mention a few basic examples of models.

**Example 27.** Every bicartesian symmetric monoidal closed category \( \mathcal{L} \), cartesian and powered \( \mathcal{L} \)-category \( \mathcal{C} \), and \( \mathcal{D} \mathcal{L} \)-adjunction \( \mathcal{L} \leftrightarrow \mathcal{S} \) provide a IMALL\(^p\) model. Some examples include:

1. \( \mathcal{S} = \mathcal{L} \) with the monoidal-closure adjunction \((-) \otimes S \vdash \{S \to -\} \) for \( S \) in \( \mathcal{L} \).
2. \( \mathcal{S} = \mathcal{L}^{op} \) with the linear double-dualisation adjunction \([- \to -] \vdash [\to \to R] \) for \( R \) in \( \mathcal{L} \), and
3. \( \mathcal{S} = \mathcal{L}^{op} \) for a cartesian \(*\)-autonomous category \( \mathcal{L} \), with the adjoint equivalence provided by the duality.

**IMALL\(^p\) semantics**  The interpretation of types is given by setting:

\[(A \otimes B)^+ = A^+ \times B^+ \quad T^+ = 1 \quad (A \oplus B)^+ = A^+ + B^+ \quad 0^+ = 0\]

As for terms, the definitions are:

\[\begin{align*}
\Gamma \vdash \mu a : A & \vdash A^+ \\
\Gamma \vdash a : A & \vdash A^+ \quad \Delta \vdash A \to A^+ \\
\Gamma \vdash \mu \langle a, b, \beta \rangle : A & \vdash A^+ \times B^+ \to C \quad \Delta \vdash A \to B
\end{align*}\]

where the second map is the copairing of \( \langle \cdot : (\Gamma, x : A) \vdash \rangle \) and \( \langle \cdot : (\Gamma, y : B) \vdash \rangle \).

**Theorem 28.** The IMALL\(^p\) semantics is coherent and sound.

### 6.3 IMLL\(^p\) Calculus and Models

**Calculus**  The calculus IMLL\(^p\) is given by adding exponentials and additives to IMLL\(^p\).

**Theorem 29.** IMLL\(^p\) is expressible in IMALL\(^p\).

**Theorem 30.** IMALL\(^p\) has the Barendregt-style properties.

**Adjunction models**  As previously, our modular development allows for a straightforward extension with resources.

**Definition 31.** An IMALL\(^p\) model with a resource modality \( \mathcal{L} \leftrightarrow \mathcal{S} \) of an IMALL\(^p\) model \( \mathcal{L} \leftrightarrow \mathcal{S} \) together with a linear distributive category \( \mathcal{K} \) and a monoidal adjunction \( \mathcal{K} \leftrightarrow \mathcal{L} \). An IMALL\(^p\) model is an IMALL\(^p\) model with resources in which the symmetric monoidal structure of \( \mathcal{K} \) is cartesian.

The IMALL\(^p\) semantics is that of IMALL\(^p\) and IMELL\(^p\), and thus also sound.

**Lifting theorem**  For every symmetric monoidal functor \( L : \mathcal{K} \to \mathcal{L} \) between linear distributive categories \( \mathcal{K} \) and \( \mathcal{L} \), the symmetric monoidal adjunction \( L \vdash L^* : P \mathcal{L} \to P \mathcal{K} \) cuts down to a symmetric monoidal adjunction \( L \vdash L^* : \mathcal{D} \mathcal{L} \to \mathcal{D} \mathcal{K} \). As in Section 5.2, the \( \mathcal{D} \mathcal{K} \)-category \( L^* \mathcal{S} \) has \( \mathcal{K} \)-powers and, moreover, it is cartesian whenever \( \mathcal{K} \):

\[L^* \mathcal{S} (N, \coprod_{i \in I} N_i) \cong L^* \prod_{i \in I} \mathcal{S} (N_i, N_i) \cong \prod_{i \in I} L^* \mathcal{S} (N_i, N_i) \]
Hence we have the following lifting theorem.

**Theorem 32.** Every IMALL_p model with a resource modality $(\mathcal{L} \xrightarrow{\phi} \mathcal{S} \xrightarrow{\psi} \mathcal{L})$ induces an IMALL_p model, obtained by composing the following two adjunctions:

$\mathcal{R} \xrightarrow{\phi} L^*\mathcal{L} \xrightarrow{\psi} L^*\mathcal{S}$.

### 6.4 LJ_p, Calculus and Models

**Calculus** The calculus LJ_p is given by adding additives to MLJ_p.

**Theorem 33.** LJ is expressible in LJ_p.

**Theorem 34.** LJ_p has the Barendregt-style properties.

**Adjunction models** Models of LJ_p are the intuitionistic version of IMALL_p models. The LJ_p semantics is that of MLJ_p and IMALL_p and thus sound; it is also coherent.

**Definition 35.** An LJ_p model is an IMALL_p model $(\mathcal{L} \xrightarrow{\phi} \mathcal{S} \xrightarrow{\psi} \mathcal{L})$ in which the symmetric monoidal structure of $\mathcal{L}$ is cartesian.

**Theorem 36.** The notion of LJ_p model and Levy’s notion of CBPV adjunction [32, Definition 5.5] are equivalent.

**Corollary 37.** Every IMALL_p model $(\mathcal{L} \xrightarrow{\phi} \mathcal{S} \xrightarrow{\psi} \mathcal{L})$ induces an LJ_p model $(\mathcal{L} \xrightarrow{\phi} \mathcal{S} \xrightarrow{\psi} \mathcal{L})$.

### 7. Perspectives

**Benefits of L-calculi** L-calculi are an alternative to structured operational semantics. They can be seen as providing a principled reconstruction of defunctionalized CPS in direct style [40], while giving a modern view on the dualities of computation: expression/context, producer/consumer, strict/lazy.

In particular, as a legacy of CPS, there are benefits reminiscent of the advantages of CPS over ANF [20] and monadic meta-languages [35] as reported by Kennedy [28]: simplifications of commuting conversions, of inlining, of sharing of contexts... A natural question is then, whether these simplifications carry over to the operational and equivariant modelling of effects and resources.

**Commuting conversions and effects** Benton and Kennedy [6] show that lessons from the proof theory of λ-calculus with sums, in particular the role of commuting conversions, help in the design and equivariant reasoning of exceptions. It is thus tempting to investigate how far advantages of L calculi on the λ-calculus with sums carry over to the study of effects.

Benton and Kennedy refine the syntax for exceptions using a new form of handler try-unless which is suitable for expressing commuting conversions in the vein of the conversions for sums. Commuting conversions are used to turn an abstract-machine-based operational semantics of exceptions into a definition in terms of the source language, which is simpler since “there is a certain amount of clutter involved in using stacks (extra syntax, type rules, etc.).” They moreover show that it is possible to produce optimised code by combining commuting conversions with effect analysis in the intermediate representation (for instance, compiling a non-tail-recursive source example into a tail-recursive final code). Finally, while commuting conversions duplicate expressions, they mention that it is possible to avoid the explosion in code size by using in the implementation “a special abstraction construct which compiles to a block of code accessed by jumps”.

The calculus LJ_p can be extended so as to give an operational semantics for exceptions, using an auxiliary stack of stacks, as introduced by Ariola, Herbelin and Sabry [2] in the context of delimited control operators.

$\langle \mu c \parallel S \rangle^\rho[\pi] \triangleright_{R} c \{S, \pi\}$

To do so, those stacks are made to correspond to the successive try-unless handlers $(\tau(t, x, u, E(y), e))$ in scope, by solving the following (simplified) equations.

$\langle V \parallel \text{return} \rangle^\rho[\mu(\chi(x), c(E(y), e'), \pi) \rightarrow_{R} c[V/x][\pi]\rangle$

$\langle \text{raise } E(V) \parallel S \rangle^\rho[\mu(\chi(x), c(E(y), e'), \pi) \rightarrow_{R} c[V/y][\pi]\rangle$

$\langle \tau(t, x, u, E(y), e) \parallel S \rangle^\rho[\pi]$ to $\langle \tau(t, x, u, E(y), e) \parallel S \rangle^\rho[\pi]$.

Thus, the abstract machine for exceptions can, as an alternative to adding commuting conversions to the source language, be turned into an L-calculus that reveals a decomposition of try-unless into simpler primitives:

$\tau(t, x, a, E(y), e')$

$$\equiv \mu a^\rho[\mu(\chi(t) \mu x^\rho(\chi(x)) \parallel \mu(\chi(R(x), a) E(y), e']\parallel_{R} c[R(x), (a \parallel S]\parallel E. (\langle e \parallel S \parallel e \rangle)]$$

In particular, it is possible to derive the commuting conversions that Benton and Kennedy use to define the operational semantics.

However, other conversions, characteristic of exceptions¹, are out of reach. Thus, on the one hand, the L-calculus provides a simpler operational semantics than the λ-calculus with try-unless, but on the other hand it is not able to produce by itself exception-specific optimisations as in Benton and Kennedy (as one can expect).

Lastly, the duplications happening during commuting conversions are mediated by µ. Thus, the mismatch between the reduction theory and the implementation in Benton and Kennedy can be explained as the need to determine an appropriate sharing implementation for a single µ binder, as is done with CPS [28].

The challenge is to let L calculi model the operational semantics of a wider class of effects, such as algebraic effects, along the above lines, all the while simplifying the study of their equational theory.

**Completeness** In connection to the model theory, direct algebraic descriptions of our calculi are to be obtained in terms of the duals arising from the effect adjunctions. These direct models are to be put in reflection with the enriched adjunction models, to generalise Fuhrmann’s direct characterisation of LC-calculi-models [21], in the continuity of [39]. In fact, we conjecture that the calculi form initial models, not only for direct duploid models, but also for the adjunction models, provided that the data of values and stacks is appropriately preserved.

In particular, we have observed that the distributivity requirement on presheaves, while needed for the correspondence with CBPV models, and holding in all concrete models we know of, was actually not used for defining the semantic interpretation. However, as syntactic presheaves are naturally distributive, we conjecture that distributivity is required for completeness.

**Biclosed action models** As alluded in Example 8, we are aware of classes of models that display more mathematical structure than the ones presented here and will be studied elsewhere. We are particularly interested in the class given by biclosed symmetric-monoidal skew actions, that seems to be related to delimited control and type-and-effect systems.

**Acknowledgements**

We thank the anonymous reviewers. G. Munch-Maccagnoni was supported by ERC ECSYM; M. Fiore partially so.

¹ try-unless, case-try in the terminology of [6].

² proj-try, app-try, case-try and try-case in [6].