A Term Assignment for Polarized Bi-intuitionistic Logic and its Strong Normalization

Corrado Biasi∗
Department of Computer Science
Queen Mary University of London
London E1 4NS, Great Britain
biasiman@dcs.qmul.ac.uk

Federico Aschieri†
Dipartimento di Informatica
SS.MM.FF.NN. Verona, Italy

Abstract. We propose a term assignment (let calculus) for Intuitionistic Logic for Pragmatics ILPAC, a polarized sequent calculus which includes ordinary positive intuitionistic logic LJ⊃∩, its dual LJ∖⋎ and dual negations ( )⊥ which allow a formula to “communicate” with its dual fragment. We prove the strong normalization property for the term assignment which follows by soundly translating the let calculus into simply typed λ calculus with pairings and projections. A new and simple proof of strong normalization for the latter is also provided.

Keywords: subtractive logic, bi-intuitionistic logic, intuitionistic logic for pragmatics, lambda calculus, strong normalization

1. Introduction

Polarized Bi-intuitionistic Logic ([1],[2]) is a logic which includes the implicative, conjunctive fragment of intuitionistic logic LJ⊃∩ and the subtractive, disjunctive fragment LJ∖⋎ considered as dual to

∗Address for correspondence: Department of Computer Science, Queen Mary University of London, London E1 4NS, Great Britain
†We wish to thank Prof. Gianluigi Bellin and Prof. Stefano Berardi for help and support through all the stages of the article, the two anonymous referees who greatly contributed with precise remarks and detection of errors.
Polarized elementary formulas are regarded as expressing acts of *assertions* or of *conjectures*; non-elementary formulas inherit a polarity: implication and conjunction are applied to and yield *assertive* formulas, subtraction and disjunction are applied to and yield *conjunctural* formulas. Let us suppose that we know what counts as a justification of an elementary assertion and as a refutation of an elementary conjecture; then what counts as a justification of complex assertive formulas and as a refutation of complex conjunctural formulas is explained in terms of Heyting’s interpretation of intuitionistic connectives. For instance, a refutation of \( C \setminus D \) is given by a method that transforms every refutation of \( D \) into a refutation of \( C \). A reason for this choice is the difficulty of characterizing in a purely logical way the condition for a conjecture to be *justified*. Sequent calculi combine a calculus for \( \text{LJ}^{\cap} \) (with multiple assumptions \( \Theta \) and one conclusion \( \epsilon' \)) and for \( \text{LJ}^{\setminus \Vdash} \) (with one assumption \( \epsilon \) and multiple conclusion \( \Upsilon \)) in the same sequent:

\[ \Theta ; \epsilon \Rightarrow \epsilon' ; \Upsilon \]

where \( \Theta \) and \( \epsilon \) are *assertive*, \( \Upsilon \) and \( \epsilon' \) are *conjunctural* and exactly one among \( \epsilon, \epsilon' \) can occur in the sequent.

In a previous work [2] the consideration of Cecylia Rauszer’s bi-intuitionistic logic [18, 19] (also called *Heyting-Brouwer* or *subtractive* logic) from the point of view of the logic for pragmatics has been advocated; the philosophical background and motivations of a logic of assertions and conjectures are discussed and a general outline of such a logic, called \( \text{ILP} \) (*Intuitionistic Logic for Pragmatics*) is presented. The system \( \text{ILP} \) is formalized by a polarized sequent calculus, as presented above, and is complete for Kripke’s semantics over preordered frames. However, the logic \( \text{ILP} \) considerably departs from Rauszer’s tradition, namely, from her definition of the Heyting-Brouwer logic and its Kripke’s semantics [18, 19] in the 1970s, and from the work by several authors, including Lawvere, Makkai, Reyes and Zolfaghari in category theory [16, 20] and most recently by R. Goré and T.Crolard in proof theory [6, 13]. The main difference is that Rauszer’s logic is an intermediate system between intuitionistic and classical logic, while in its polarized version the logic remains intuitionistic. Algebraic models of Heyting-Brouwer logic are *bi-Heyting algebras* which have both the structure of a Heyting algebra and the dual structure of a co-Heyting algebra; topological models are bi-topological spaces, but every bi-topological space consists of the final sections of some preorder; the categorical models are bi-Cartesian closed categories (with co-exponents), but these collapse to partial orderings (see Crolard [5]). Our polarized framework allows us to avoid this collapse: by polarizing formulas in \( \text{LJ}^{\cap} \) as assertive and formulas in \( \text{LJ}^{\setminus \Vdash} \) as conjunctural positive intuitionistic logic and its dual may interact while remaining separated. In this paper we shall not go further on these considerations but focus on the problem to find a term assignment for our Polarized Bi-intuitionistic Logic and prove its strong normalization.

### 1.1. Outline of the Paper

In section 2 we present \( \text{ILP}_{\text{AC}} \) (*Intuitionistic Logic for Pragmatics of Assertions and Conjectures*) the fragment of Polarized Bi-intuitionistic Logic where we will work on. It includes ordinary positive intuitionistic logic \( \text{LJ}^{\cap} \) and its dual \( \text{LJ}^{\setminus \Vdash} \) extended with dual negations \( (\ )^\perp \) which allow a formula to “communicate” with its dual fragment. \( \text{ILP}_{\text{AC}} \) is a restriction of \( \text{ILP} \) which preserves the good property of soundness and completeness over Kripke semantic [2] and cut-elimination [3]. In section 3 we present a possible Curry Howard treatment for \( \text{ILP}_{\text{AC}} \) proposed by Hugo Herbelin\(^1\) short after the second

\(^1\)We thank Hugo Herbelin for kindly letting us work on his unpublished manuscript.
workshop for Intuitionistic Logic for Pragmatics (Paris '04): the simply typed λ-calculus provides a term assignment for the assertive part and also for the dual conjectural part. \( \text{LJ}^\cap \) is formalized with sequents having one conjectural formula on the left and many conjectural formulas on the right: 
\[ m : v \Rightarrow a_1 : v_1, \ldots, a_n : v_n \]
where \( m \) is a (conjectural) term and \( a_1, \ldots, a_n \) are (co)variables. This can be seen as a calculus of refutation i.e. the derivation of the unique refutation of the conjecture taken as assumption from the refutations of the conjectures derived from it.

Each left rule of \( \text{LJ}^\cap \) and right rule of \( \text{LJ}^\\neq \) create a formula in the assertive and conjectural environment respectively and these formulas are associated to a formal context, i.e. a term with a hole in the extension of the λ-calculus with the \texttt{let} construct. The hole can be substituted using a cut-rule or a contraction rule and the resulting term is assigned to the unique conjectural premise \( \epsilon \) or to the unique assertive conclusion \( \epsilon' \). To prove the strong normalization of Herbelin calculus we translate it into λ-calculus with pairings and projections and we prove that this translation is sound with respect to \( \beta \)-reduction and the lambda terms obtained are well typed. Therefore an infinite sequence of reduction in Herbelin calculus would lead to an infinite reduction sequence in a simply typed λ term which is impossible. In chapter 4 we use a simple combinatorial argument to prove strong normalization for simply typed λ-calculus due to René David\(^2\)\cite{9} and independently rediscovered by the second author (Federico Aschieri). The idea is to consider a particular kind of reduction for non strongly normalizable terms, namely reducing the head redex of the leftmost elementary subterm; here an elementary term is a non strongly normalizable term whose (proper) subterms are strongly normalizable. It can be shown that for this particular reduction a term in argument position will eventually appear also as a function and this is incompatible with the ordering of types. In this paper we generalize this argument to the simply typed λ-calculus with pairings and projections.

### 1.2. Related Work

Bi-intuitionistic (subtractive) logic is used by Tristan Crolard \cite{6} as a type system for a variant of the λ\(\mu\) calculus with first-class coroutines. The interpretation of the term assigned to subtraction as coroutines essentially depends on the presence of the \(\mu\) operator while here the terms assigned to subtraction rules are identical to those for implication. Crolard considers also a constructive restriction of subtractive logic, without introducing polarization. Following the approach of \cite{21} and independently rediscovered by de Paiva \cite{17} and Crolard \cite{4}, he introduces restriction on the implication right and subtraction left rules, based on a notion of dependency between occurrences of formulas in the antecedent and in the succedent of a sequent: dependencies are established in axioms and are propagated by inference rules. Crolard then argues that the constructive calculus may be regarded as a type system for a notion of safe coroutines.

Another term assignment related to Crolard’s in our polarized framework has been considered by G. Bellin in \cite{1}. Bellin considers a term assignment for \( \text{LJ}^\\neq \) which is dual to the usual term assignment for intuitionistic logic \( \text{LJ}^\cap \) in a very strong sense: to every step of \( \beta \)-reduction in simply typed λ-calculus there corresponds a step in the dual calculus and viceversa. The most unusual aspect of this calculus is its distributed nature: as sequents in dual intuitionistic logic have a single formula in the antecedent and possibly many formulas in the succedent, so multisets of terms with only one free variable are assigned to the formulas in the succedent. Sequents are formally represented by a distributed term assignment.

\(^2\)We are grateful to Prof René David for having shown to us his new proof of normalization for the simply typed lambda calculus.
\[ \alpha : v \Rightarrow m_1 \Rightarrow 1, \ldots, m_n \Rightarrow 1, \text{ and represent a derivation of conjectures compatible with one assumption. This represent another approach to the understanding of the logic of conjectures.} \]

2. Intuitionistic Logic for Pragmatics \( \text{ILP}_{AC} \)

The logic for pragmatics \( \text{ILP}_{AC} \) deals with acts of assertion \( \vartheta \) and acts of conjecture \( \upsilon \). Its language \( \mathcal{L}_{\vartheta\upsilon} \) has two different atoms: elementary assertions \( \vdash \alpha \), to be read as “certainly” \( \alpha \), and elementary conjectures \( \mathcal{H} \alpha \) to be read as “perhaps” \( \alpha \) where \( \alpha \) is a proposition in classical logic. We have the constant \( \bigvee \), for an assertion which is always justified, and symbols for acts of composite type, conjunctive \( \vartheta_1 \cap \vartheta_2 \) and implicative \( \vartheta_1 \supset \vartheta_2 \) ones (with the usual meaning); similarly, we have a constant \( \bigwedge \), for a conjecture which is always refuted, and symbols for conjectural acts of composite type, disjunctive \( \upsilon_1 \cup \upsilon_2 \), intuitively read as “perhaps \( \upsilon_1 \) or \( \upsilon_2 \)”, and subtractive \( \upsilon_1 \setminus \upsilon_2 \), intuitively read as “perhaps \( \upsilon_1 \) and not \( \upsilon_2 \)”. Moreover we allow assertive formulas to communicate with their dual fragment, and vice versa, through two negations both denoted by \( (\ )^\bot \). The connective of subtraction can be given an intuitive “pragmatic” interpretation by reading this rule

\[
\Theta ; \vartheta_1 \Rightarrow ; \mathcal{Y}_1 \cup \upsilon_2 \Rightarrow ; \mathcal{Y} \setminus \mathcal{Y}_1
\]

as follows: if given the conjecture \( \upsilon_1 \) is admissible the disjunction of the conjectures in \( \mathcal{Y} \) or \( \upsilon_2 \) then given \( \upsilon_1 \) and removing the conjecture \( \upsilon_2 \) only the disjunction of the conjectures in \( \mathcal{Y} \) is admissible.

**Definition 2.1. (The language of \( \mathcal{L}_{\vartheta\upsilon} \))**

The language \( \mathcal{L}_{\vartheta\upsilon} \) is defined as follows

\[
\begin{align*}
\alpha & := p \mid \neg p \\
\vartheta & := \vdash \alpha \mid \bigvee \mid \vartheta_1 \cap \vartheta_1 \mid \vartheta_1 \supset \vartheta_1 \mid \upsilon_1 \bot \\
\upsilon & := \mathcal{H} \alpha \mid \bigwedge \mid \upsilon_1 \cup \upsilon_1 \mid \upsilon_1 \setminus \upsilon_2 \mid \upsilon_1 \bot
\end{align*}
\]

The sequent calculus is given in table 1.

**Definition 2.2. (Informal interpretation)**

Radical formulas \( \alpha \) are interpreted as propositions, which can be true or false. Sentential expressions \( \vartheta \) and \( \upsilon \) are interpreted as impersonal illocutionary acts of assertion and conjecture, respectively. Assertions can be justified or unjustified; dually conjectures can be refuted or unrefuted.

\( \vdash \alpha \) is justified if and only if a proof can be exhibited that \( \alpha \) is true. Dually, \( \mathcal{H} \alpha \) is refuted if and only if a proof that \( \alpha \) is false can be exhibited.

\( \vartheta_1 \supset \vartheta_2 \) is justified if and only if a proof can be exhibited that a justification of \( \vartheta_1 \) can be transformed into a justification of \( \vartheta_2 \); it is unjustified, otherwise. Dually, \( \upsilon_1 \setminus \upsilon_2 \) is refuted if and only if a proof can be exhibited that a refutation of \( \upsilon_1 \) can be transformed into a refutation of \( \upsilon_2 \); it is unrefuted otherwise. See [1] and [2] for further details.

**Definition 2.3. (The duality)**

The duality between assertive formulas in intuitionistic logic \( \text{LJ}^{\bigvee \bigwedge} \) and conjectural formulas in dual intuitionistic logic \( \text{LJ}^{\bigwedge \bigcup} \) can be expressed by maps \( F \) and \( G \) as follows:
\[ F(\top p) = \top \neg p \]
\[ F(\lor) = \lor \]
\[ F(\neg_0 \supset \neg_1) = F(\neg_1) \setminus F(\neg_0) \]
\[ F(\neg_0 \cap \neg_1) = F(\neg_0) \lor F(\neg_1) \]
\[ F(\neg_0) = \top \]
\[ \neg_0 = \bot \]
\[ \neg_1 = \bot \]
\[ \neg_2 = \bot \]
\[ G(\bot) = \bot \]
\[ G(\bot_0) = \bot \]
\[ G(\bot_1) = \bot \]
\[ G(\bot_2) = \bot \]
\[ \neg_0 \lor \neg_1 = \bot \]
\[ \neg_0 \lor \neg_2 = \bot \]
\[ \neg_1 \lor \neg_2 = \bot \]
\[ \neg_0 \lor \neg_1 \lor \neg_2 = \bot \]

**Remark 2.1.** In the framework of Intuitionistic Logic for Pragmatics we have a calculus of assertions for the assertive part which is dual to a calculus of refutations in the conjectural part. The intuition is based on the equality \((\top p) = \top \neg p\) which can be intuitively read as follows: the grounds that justify the assertion \(\top p\) are also necessary and sufficient to refute the conjecture \(\top \neg p\). See [1] and [2] for further details.

### 2.1. Interpretation into Modal Logic S4 and Dualities

**Definition 2.4. (The language \(L_\Box\))**

Let \(p\) range over an enumerable set of propositional variables \(\text{Var} = \{p_1, p_2, \ldots\}\). The language \(L_\Box\) is defined by the following grammar:

\[ \alpha := p \mid \bot \mid T \mid \alpha \land \alpha \mid \alpha \lor \alpha \mid \alpha \rightarrow \alpha \mid \Box \alpha \]

Define \(\Diamond \alpha = \neg_0 \neg \Box \neg \alpha\).

**Definition 2.5. (Interpretation into classical modal logic S4)**

The modal translation \((\ )^M\) of \(L_{\neg_0}\) into \(L_\Box\) is defined inductively:

\[ (\top \alpha)^M = \Box \alpha \]
\[ (\bot)^M = \top \]
\[ (\lor)^M = \lor \]
\[ (\neg_0 \supset \neg_1)^M = \Box (\neg_0^M \rightarrow \neg_1^M) \]
\[ (\neg_0 \cap \neg_1)^M = \Box (\neg_0^M \land \neg_1^M) \]
\[ (\neg_0 \lor \neg_1)^M = \Box (\neg_0^M \lor \neg_1^M) \]

from which one easily shows \(\neg_0^M = \Box \neg_1^M\) and \(\neg_1^M = \Box \neg_0^M\). Note that the orthogonal negation \((\ )\) could have been translated simply as \((\neg_0)^M = \neg_1^M\) (and similar for \(\neg_1\)) since \(\Box \neg \neg_0 = \neg_0 \Diamond \neg_0 = \neg_0\). However we decided to translate the orthogonal negation as the standard intuitionistic negation and maintain, therefore, the modality \(\Box\).

**Theorem 2.1. (The duality)**

The duality between \(L_{\Diamond}^\land\) and \(L_{\Box}^\lor\), as defined in definition 2.3 can be internalized by the connective \((\ )^\perp\) i.e. the following equalities are semantically justified:

\[ (\top p)^\perp = \top \neg p \]
\[ (\bot)^\perp = \bot \]
\[ (\lor)^\perp = \lor \]
\[ (\neg_0 \supset \neg_1)^\perp = \neg_0^\perp \land \neg_1^\perp \]
\[ (\neg_0 \cap \neg_1)^\perp = \neg_0^\perp \lor \neg_1^\perp \]
\[ (\neg_0 \lor \neg_1)^\perp = \neg_0^\perp \land \neg_1^\perp \]
\[ (\top)^\perp = \top \]
\[ (\bot_0)^\perp = \bot_0 \]
\[ (\bot_1)^\perp = \bot_1 \]
\[ (\bot_2)^\perp = \bot_2 \]
\[ (\top_0) = \top_0 \]
\[ (\top_1) = \top_1 \]
\[ (\top_2) = \top_2 \]
\[ (\bot_0) = \bot_0 \]
\[ (\bot_1) = \bot_1 \]
\[ (\bot_2) = \bot_2 \]
Proof:
The proof follows from the modal translation and rely upon the equality $\square = \neg \lozenge \neg$. We give the proof of some interesting cases:

(a) in $\text{ILP}_{\text{AC}}$ we have $(\perp p)^M = \lozenge \neg \lozenge p^M = \lozenge \neg \lozenge \neg p^M = \lozenge p^M = (\neg \lozenge \neg p)^M$. The proof rely on the elimination of classical double negation, the equality $\lozenge = \neg \square \neg$ and idempotence of $\lozenge$. Analogues consideration for $(\neg \lozenge \neg p)^M$.

(c) $(\vartheta_0 \supset \vartheta_1)^M = \lozenge \neg \lozenge (\vartheta_0^M \rightarrow \vartheta_1^M) = \lozenge \lozenge (\neg \vartheta_0^M \land \neg \vartheta_1^M) = \lozenge (\neg \lozenge \neg \vartheta_0^M \land \neg \lozenge \neg \vartheta_1^M) = (\vartheta_1^M \land \vartheta_0^M)^M$

where we use classical equalities, equality $\lozenge = \neg \square \neg$, idempotence of $\lozenge$, $\vartheta = \square \vartheta$ and $\square = \neg \lozenge \neg$.

(e) $(\vartheta^M)^M = \lozenge \neg \lozenge \neg \vartheta^M = \lozenge \lozenge \vartheta^M = \vartheta^M$.

Remark 2.2. All the dualities above are provable in our sequent calculus (table 1). In particular the modal translation of $(\vartheta^\perp)^\perp$ supports the interpretation of these connectives as inverse i.e. it can be easily proved that

$\vartheta^\perp \iff \vartheta$ and $\vartheta^{\perp \perp} \iff \vartheta$

2.2. Sequent Calculus $\text{ILP}_{\text{AC}}$

The intuitionistic fragments of $\text{ILP}_{\text{AC}}$ is formalized by sequent calculi which contains only rules for the pragmatic connectives. The characteristic feature of $\text{G3}$ sequent calculi [22] is that the rules of Weakening and Contraction are implicit. The sequent calculus for our logic $\text{ILP}_{\text{AC}}$ is of this type.

Since all formulas of the language $\mathcal{L}_{\vartheta_\epsilon}$ are polarized as assertive or conjectural, sequents have a restricted form, inspired by Girard’s logic $\text{LU}$ [11].

The sequent is formed in two parts:

1. $\vartheta_1, \ldots, \vartheta_m \Rightarrow \epsilon'$ for usual intuitionistic logic, where $\epsilon'$ is a single assertive formula;
2. $\epsilon \Rightarrow \nu_1, \ldots, \nu_n$ for dual intuitionistic logic, where $\epsilon$ is a single conjectural formula.

These calculus of assertions and conjectures can “communicate” through orthogonal negations, both denoted by $(\cdot)^\perp$, which allow a formula to be moved to its dual part i.e. an assertive formula becomes conjectural and viceversa (see mixed rules in Table 1).

Definition 2.6. (Sequent calculus and inference rules)

The sequent $S$ have the form

$\Theta ; \epsilon \Rightarrow \epsilon' ; \Upsilon$

where

- $\Theta$ is a sequence of assertive formulas $\vartheta_1, \ldots, \vartheta_m$;
- $\Upsilon$ is a sequence of conjectural formulas $\nu_1, \ldots, \nu_n$;
- $\epsilon$ is conjectural and $\epsilon'$ is assertive and exactly one of $\epsilon, \epsilon'$ occurs in $S$.

Notice that polarization here has a different motivation and plays a different role than in linear logic.
The inference rules are given in table 1. \( \bot \text{LAC} \) stands for “left rule for assertive-to-conjectural orthogonal negation” and similarly for \( \bot \text{RAC}, \bot \text{LCA} \) and \( \bot \text{RCA} \). The rule \( \supset R \) and \( \setminus L \) are not semantically invertible as \( \epsilon \) or \( \epsilon' \) are deleted in one of the premises. In order to achieve depth-preserving contraction, feature extensively used in the process of cut-elimination, we must record \( \vartheta_1 \supset \vartheta_2 \) and \( \upsilon_1 \setminus \upsilon_2 \) in the non-invertible premise. More detail in [3].

Remark 2.3.

1. An intuitive interpretation of the logic \( \text{ILP}_{\text{AC}} \) as intuitionistic is obtained by using the duality (as defined in theorem 2.1) to rewrite the sequent as follows:

\[
\Theta, \Upsilon \vdash \Rightarrow \epsilon^\perp, \epsilon'
\]

\( \Upsilon \perp \) means the application of the orthogonal negation to any formula in \( \Upsilon \). Note that as only one between \( \epsilon \), \( \epsilon' \) is non-empty, the form of the sequent is the usual intuitionistic one with \( \supset \) and \( \setminus \).

2. There exist a retraction from the proofs of \( \text{ILP}_{\text{AC}} \) to the proofs of \( \text{LJ}^{\supset \setminus} \), based on the elimination of conjectural connectives and of the corresponding sequent calculus rules suggested in part 1, which is the identity on derivations in the assertive fragment of \( \text{ILP}_{\text{AC}} \). In the next section we consider a less trivial embedding of \( \text{ILP}_{\text{AC}} \), in the style of Herbelin’s \( \lambda \tilde{\lambda} \) calculus [14], into the simply typed \( \lambda \) calculus with pairings and projections.

Theorem 2.2. (Cut elimination for \( \text{ILP}_{\text{AC}} \))

All applications of the cut rule are eliminable in a \( \text{ILP}_{\text{AC}} \) derivation.

Proof:

It can be proved [3] that in \( \text{ILP} \) the rules of weakening and contraction are admissible preserving the depth of the derivation and that the rules cut can be given the context-sharing form, rather than the multiplicative form. In [3] it has also been proved the cut elimination theorem for \( \text{ILP} \); therefore the cut elimination theorem for \( \text{ILP}_{\text{AC}} \) follows easily since it is a subset of \( \text{ILP} \). The procedure for cut elimination is standard and inspired by [10] and [8]: let \( R_i \) be the last rule used in a derivation \( d_i \), let \( d_0 \) be the derivation ending with the cut formula in the stoup and \( d_1 \) the derivation ending with the cut formula outside the stoup

\[
\begin{array}{c}
\vdash d_0 : \Theta \Rightarrow \vartheta ; \Upsilon \\
\vdash d_1 : \vartheta, \Theta \Rightarrow \epsilon \Rightarrow \epsilon' ; \Upsilon \\
\vdash d_2 : \Theta ; \epsilon \Rightarrow \epsilon' ; \Upsilon \\
\vdash d_3 : \vartheta, \Theta \Rightarrow \epsilon \Rightarrow \epsilon' ; \Upsilon \\
\vdash d_4 : \Theta ; \epsilon \Rightarrow \epsilon' ; \Upsilon \\
\vdash d_5 : \Theta ; \epsilon \Rightarrow \epsilon' ; \Upsilon \\
\vdash d_6 : \Theta ; \epsilon \Rightarrow \epsilon' ; \Upsilon \\
\end{array}
\]

1. if the cut formula is not principal in \( R_1 \) permute the cut with \( R_1 \);
2. if the cut formula is principal in \( R_1 \) but not principal in \( R_0 \), permute the cut with \( R_0 \);
3. if the cut formula is principal in \( R_0 \) and \( R_1 \), apply a symmetric reduction

\( \square \)
### Axioms

\[
\begin{align*}
\Theta; \vdash \alpha; & \Rightarrow \vdash \alpha; \top \\
\Theta; \pi \alpha & \Rightarrow; \pi \alpha; \top
\end{align*}
\]

\[
\begin{align*}
\Theta; \wedge & \Rightarrow; \top \\
\Theta; & \Rightarrow; \bigvee; \top
\end{align*}
\]

### Cut rules

\[
\begin{align*}
\Theta; & \Rightarrow \vartheta; \top \Theta; \vartheta; & \Rightarrow \vartheta; \top; \vartheta & \Rightarrow \vartheta; \top \\
\Theta; & \Rightarrow \vartheta; \top
\end{align*}
\]

### Assertive rules

\[
\begin{align*}
\Theta; & \vartheta_1; \Rightarrow \vartheta_2; \top \Theta; & \Rightarrow \vartheta_1; \vartheta_2; \top \vartheta & \Rightarrow \vartheta_1; \vartheta_2; \top \\
\Theta; & \Rightarrow \vartheta_1; \vartheta_2; \top
\end{align*}
\]

### Conjectural rules

\[
\begin{align*}
\Theta; & \vartheta_1; \Rightarrow \vartheta_2; \top \Theta; & \Rightarrow \vartheta_1; \vartheta_2; \top; \vartheta & \Rightarrow \vartheta_1; \vartheta_2; \top \\
\Theta; & \Rightarrow \vartheta_1; \vartheta_2; \top
\end{align*}
\]

### Mixed rules

\[
\begin{align*}
\Theta; & \vartheta; \Rightarrow \vartheta; \top \Theta; & \Rightarrow \vartheta; \top; \vartheta & \Rightarrow \vartheta; \top \\
\Theta; & \Rightarrow \vartheta; \top
\end{align*}
\]

### Table 1. \( \text{ILP}_{\text{AC}} \) : rules.
Theorem 2.3. (Soundness and completeness for $\text{ILP}_{\text{AC}}$)
The intuitionistic sequent calculus $\text{ILP}_{\text{AC}}$ without the rules of cut is sound and complete with respect to the interpretation in $\text{S}_4$ given in definition 2.5.

Proof:
The proof follows from the one given for $\text{ILP}$ [2, 3] as $\text{ILP}_{\text{AC}}$ is a subset of $\text{ILP}$. Sequents are interpreted in the obvious way:

$$(\Theta ; \epsilon \Rightarrow \epsilon' ; \Upsilon)^M = \Theta^M, \epsilon^M \Rightarrow \epsilon'^M, \Upsilon^M$$

\[\square\]

3. Term Assignment for $\text{ILP}_{\text{AC}}$ and Strong Normalization

3.1. The \textit{let} Calculus

The syntax of the calculus, inspired by [14], embodies two syntactic categories: contexts and terms. Both can be either assertive or conjectural.

Definition 3.1. (Terms)

We have two type of terms: assertive and conjectural. We define with $x, y, z$ variables ranging over assertive terms $t, u, s$ and with $a, b, c$ (co)variables ranging over conjectural terms $l, m, n$. They are defined by the following grammar:

$$t, u, s ::= x | \text{let } x = tu \text{ in } s | \text{let } \langle x, y \rangle = t \text{ in } u | \text{let } a^\perp = t \text{ in } u | \text{let } a = lm \text{ in } t | \text{let } x^\perp = t \text{ in } u$$

$$\top | \lambda x. t | \langle t, u \rangle | t^\perp$$

$$l, m, n ::= a | \text{let } x = tu \text{ in } m | \text{let } \langle x, y \rangle = t \text{ in } m | \text{let } a^\perp = t \text{ in } m | \text{let } a = lm \text{ in } n | \text{let } \langle a, b \rangle = m \text{ in } n | \text{let } x^\perp = m \text{ in } n$$

$$\perp | \lambda a. m | \langle m, n \rangle | t^\perp$$

We write $\tau$ for an arbitrary term: $\tau \in \{t, u, v, l, m, n\}$.

Definition 3.2. (Context)

We have two assertive context $E$ and $F$ and two conjectural context $P$ and $Q$ defined over terms and variables by the following grammar:

$$E ::= \text{let } x = [] \text{ in } u | \text{let } \langle x, y \rangle = [] \text{ in } u | \text{let } a^\perp = [] \text{ in } t | \text{let } x = [] \text{ in } x$$

$$F ::= \text{let } x = [] \text{ in } m | \text{let } \langle x, y \rangle = [] \text{ in } m | \text{let } a^\perp = [] \text{ in } m$$

$$P ::= \text{let } a = [] \text{ in } u | \text{let } \langle a, b \rangle = [] \text{ in } u | \text{let } x^\perp = [] \text{ in } u$$

$$Q ::= \text{let } a = [] \text{ in } n | \text{let } \langle a, b \rangle = [] \text{ in } m | \text{let } x^\perp = [] \text{ in } m | \text{let } a = [] \text{ in } a$$
Proposition 3.1. (Terms rephrased)
The syntax of terms as given in definition 3.1 can be rephrased into the following:

\[ t, u, s ::= x | E[t] | P[m] | \top | \lambda x.t \mid t, u | m | \bot \]

\[ l, m, n ::= a | F[t] | Q[m] | \bot | \lambda a.m \mid m, n | t \bot \]

where \( E[t] \) is the capture-avoiding substitution of the hole \([\ ]\) for the term \( t \). The typing system is a sequent calculus with rules given in the appendix and sequents having the form given in the following definition.

Definition 3.3. (Sequents)
The sequents have one of the following forms:

Assertive
\[ \Theta ; \Rightarrow t : \epsilon ; \Upsilon \]

Conjectural
\[ \Theta ; m : \epsilon \Rightarrow ; \Upsilon \]

Assertive assertive-context
\[ \Theta \mid E : \vartheta \Rightarrow \epsilon' ; \Upsilon \]

Conjectural assertive-context
\[ \Theta \mid F : \vartheta \mid \epsilon \Rightarrow ; \Upsilon \]

Conjectural conjectural-context
\[ \Theta ; \epsilon \Rightarrow ; P : \upsilon \mid \Upsilon \]

Assertive conjectural-context
\[ \Theta ; \epsilon ; \Upsilon \Rightarrow ; Q : \upsilon \mid \Upsilon \]

\( \epsilon \) and \( \epsilon' \) represent the active computation; \( \epsilon \) is conjectural and \( \epsilon' \) is assertive and exactly one of \( \epsilon \) and \( \epsilon' \) occurs in the sequent.

On the left of the first semicolon we have an assertive environment and on the right of the second semicolon we have a conjectural environment: the former is a sequence of assertive formulas \( x_1 : \vartheta_1, \ldots, x_m : \vartheta_m \) denoted by \( \Theta \), and possibly an “active” context \( E \) or \( F \) associated to an assertive formula \( \vartheta \); the latter is formed by \( \Upsilon \) which is a sequence of conjectural formulas \( a_1 : \upsilon_1, \ldots, a_n : \upsilon_n \), and possibly an “active” context \( P \) or \( Q \) associated to a conjectural formula \( \upsilon \). Inspired by [7] these contexts occur in a privileged area (stoup) of the environment (denoted by \( [\ ] \) and stands for “a context waiting for a value for the computation to continue”. When the context receives its value the computation continue with \( \epsilon \) or \( \epsilon' \) (see example in remark 3.1).

Definition 3.4. (\( \beta \) reductions)
The redexes are the following:

\[ \text{let } x = y \text{ in } x \Rightarrow y \]
\[ \text{let } a = b \text{ in } a \Rightarrow b \]
\[ \text{let } a \perp = m \perp \text{ in } \tau \Rightarrow \tau[m/a] \]
\[ \text{let } \langle x, y \rangle = (t, u) \text{ in } \tau \Rightarrow \tau[t/x][u/y] \]
\[ \text{let } y = (\lambda x.t)u \text{ in } \tau \Rightarrow \tau[t/x][u/y] \]
\[ \text{let } x \perp = t \perp \text{ in } \tau \Rightarrow \tau[t/x] \]
\[ \text{let } (a, b) \perp = (m, n) \text{ in } \tau \Rightarrow \tau[m/a][n/b] \]
\[ \text{let } b = (\lambda a.m)n \text{ in } \tau \Rightarrow \tau[m/n/b] \]

Remark 3.1. The rules which create a formula in the context (i.e. \( \cap L, \supset L, \wedge L, \vee L, \perp RAC.1, AC.2, \perp LCA.1, \perp LCA.2, |I \) associate to the formula a let term with a hole \([\ ]\) in the stoup of the environment.
(denoted by |). The type of the context is the type of the hole and the hole can be substituted with a term of the same type through a cut rule or a contraction. Only with a cut rule or a contraction a sequent with a let term in the stoup of the context can evolve as shown in the example below:

\[
\frac{\Theta; \Rightarrow \lambda z.s : \vartheta_1 \supset \vartheta_2 ; \Upsilon}{\Theta; \Rightarrow \lambda z.s | \text{let } \Gamma = \varnothing \text{ in } \Gamma : \vartheta_1 \supset \vartheta_2, \Rightarrow \Upsilon : \vartheta; \Upsilon} \quad \supset R_{\text{cut}_1}
\]

\[
\frac{\Theta, y : \vartheta_1 \supset \vartheta_2, \Rightarrow t : \vartheta_1 ; \Upsilon, x : \vartheta_2, \Theta, y : \vartheta_1 \supset \vartheta_2, \Rightarrow u : \vartheta ; \Upsilon}{\Theta, y : \vartheta_1 \supset \vartheta_2 ; \Rightarrow \text{let } \Gamma = \varnothing \text{ in } u : \vartheta_1 \supset \vartheta_2, \Rightarrow \Upsilon : \vartheta; \Upsilon} \quad \supset R_{\text{cut}_2}
\]

Note that there are not two derivations of \( \text{let } x \leftarrow y \text{ in } u \) because, as we allow axioms to be only in atomic form, a cut with an axiom \( \Theta, y : \vartheta_1 \supset \vartheta_2, \Rightarrow y : \vartheta_1 \supset \vartheta_2 ; \Upsilon \) can not occur. This rigidity of the calculus allow us to follow totally the “cut=redex” paradigm.

**Definition 3.5. (Commutative equivalences)**

We say that \( t \) and \( u \) are commutative equivalent, denoted by \( t \simeq_c u \), whenever \( R_1 \,/ R_2 \) is a permutation of the rule \( R_1 \) for the rule \( R_2 \) in a sequent derivation and \( t \) and \( u \) are the terms associated to \( R_1 \) and \( R_2 \) respectively. The list of all the commutative equivalences are generated by all the possible permutations of inferences. Instead of giving a gigantic list of equivalences we show how to generate them by an example. The permutations of \( \supset R \) or \( \setminus L \) with \( \cap L + C \) or \( \cup R + C \) generate four equivalences (which are equal up to renaming of variables):

1) \( \text{let } \langle x_1, x_2 \rangle \leftarrow y \text{ in } \lambda z.t \simeq_c \lambda z. \text{let } \langle x_1, x_2 \rangle \leftarrow y \text{ in } t \quad \supset R \setminus \cap L + C \)
2) \( \text{let } \langle x_1, x_2 \rangle \leftarrow y \text{ in } \lambda a.m \simeq_c \lambda a. \text{let } \langle x_1, x_2 \rangle \leftarrow y \text{ in } m \quad \setminus L \cup \cap L + C \)
3) \( \text{let } \langle a, b \rangle \leftarrow c \text{ in } \lambda x.t \simeq_c \lambda x. \text{let } \langle a, b \rangle \leftarrow c \text{ in } t \quad \supset R \cup \cup R + C \)
4) \( \text{let } \langle a, b \rangle \leftarrow b \text{ in } \lambda c.m \simeq_c \lambda c. \text{let } \langle a, b \rangle \leftarrow b \text{ in } m \quad \setminus L \cup \cup R + C \)

The third equivalence identifies the permutation of

\[
\frac{\Theta, x : \vartheta_1, \Rightarrow \; t : \vartheta_2 ; \; \Upsilon, a : v_1, b : v_2}{\Theta, \Rightarrow \; \lambda x_t : \vartheta_1 \supset \vartheta_2 ; \; \Upsilon, a : v_1, b : v_2 \quad \supset R} \quad \supset R_{\setminus L + C}
\]

\[
\frac{\Theta, \Rightarrow \; \vartheta_1 \supset \vartheta_2 ; \; \text{let } \langle a, b \rangle \leftarrow c \text{ in } \lambda x.t : \vartheta_1 \supset \vartheta_2 ; \; \Upsilon}{\Theta, \Rightarrow \; \text{let } \langle a, b \rangle \leftarrow c \text{ in } t : \vartheta_1 \supset \vartheta_2 ; \; \Upsilon} \quad \cup R_{\setminus L + C}
\]

with

\[
\frac{\Theta, x : \vartheta_1, \Rightarrow \; t : \vartheta_2 ; \; \Upsilon, a : v_1, b : v_2}{\Theta, \Rightarrow \; \lambda x_t : \vartheta_1 \supset \vartheta_2 ; \; \Upsilon, a : v_1, b : v_2 \quad \supset R} \quad \supset R_{\setminus L + C}
\]

\[
\frac{\Theta, \Rightarrow \; \vartheta_1 \supset \vartheta_2 ; \; \text{let } \langle a, b \rangle \leftarrow c \text{ in } t : \vartheta_1 \supset \vartheta_2 ; \; \Upsilon}{\Theta, \Rightarrow \; \lambda x.\text{let } \langle a, b \rangle \leftarrow c \text{ in } t : \vartheta_1 \supset \vartheta_2 ; \; \Upsilon} \quad \setminus L_{\cup R + C}
\]

\[
\frac{\Theta, x : \vartheta_1, \Rightarrow \; t : \vartheta_2 ; \; \Upsilon, a : v_1, b : v_2}{\Theta, \Rightarrow \; \lambda x_t : \vartheta_1 \supset \vartheta_2 ; \; \Upsilon, a : v_1, b : v_2 \quad \supset R} \quad \setminus L_{\cup R + C}
\]

\[
\frac{\Theta, \Rightarrow \; \vartheta_1 \supset \vartheta_2 ; \; \text{let } \langle a, b \rangle \leftarrow c \text{ in } t : \vartheta_1 \supset \vartheta_2 ; \; \Upsilon}{\Theta, \Rightarrow \; \lambda x.\text{let } \langle a, b \rangle \leftarrow c \text{ in } t : \vartheta_1 \supset \vartheta_2 ; \; \Upsilon} \quad \cap L_{\cup R + C}
\]
The “dual” permutations i.e. $\supset R$ with $\cap L + C$, $\setminus L$ with $\cap L + C$ and $\setminus L$ with $\cap R + C$ are similar. Note that as in ordinary intuitionistic logic $\supset R$ is not permutable over $\supset L$, in $\text{ILP}_{\text{AC}}$ also $\supset R$, $\setminus R \mathcal{AC}$, $\setminus L \mathcal{AC}$ and $\setminus L \mathcal{AC}$ are non-permutable over $\supset L$ or $\setminus R$.

3.2. Strong Normalization for the let Calculus

In the sequel we adopt Krivine’s notation [15] for parenthesis where $(u)t$ stands for the application of $u$ to $t$.

**Definition 3.6. (Translation into $\lambda$-calculus)**

The translation $(\cdot)^\lambda$ of the let calculus into $\lambda$-calculus with pairings and projections is defined inductively as follows:

\[
\begin{align*}
(\text{let } a^{-}=\emptyset \text{ in } \tau)^\lambda &= \lambda h(\lambda a.\tau^\lambda)\pi_1 h \quad x^\lambda = x \\
(\text{let } x=y \text{ in } \tau)^\lambda &= \lambda h(\lambda x(\lambda y.\tau^\lambda)\pi_1 h)\pi_0 h \quad a^\lambda = a \\
(\text{let } x^+=u \text{ in } \tau)^\lambda &= \lambda h(\lambda x.\tau^\lambda)(h)u^\lambda \quad (\lambda x.t)^\lambda = \lambda x.t^\lambda \\
(\text{let } a^-=\emptyset \text{ in } \tau)^\lambda &= \lambda h(\lambda a(\lambda b.\tau^\lambda)\pi_1 h)\pi_0 h \quad (t,u)^\lambda = (t^\lambda,u^\lambda) \\
(\text{let } a=m \text{ in } \tau)^\lambda &= \lambda h(\lambda a.\tau^\lambda)(h)m^\lambda \quad (m,n)^\lambda = (m^\lambda,n^\lambda) \\
(t^\lambda)^\lambda &= \langle 0, t^\lambda \rangle \quad \top^\lambda = \top \\
(m^\lambda)^\lambda &= \langle 1, m^\lambda \rangle \quad \perp^\lambda = \perp \\
(E[t])^\lambda &= (E^\lambda t^\lambda) \\
(P[m])^\lambda &= (P^\lambda m^\lambda) \\
(F[t])^\lambda &= (F^\lambda t^\lambda) \\
(Q[m])^\lambda &= (Q^\lambda m^\lambda)
\end{align*}
\]

The interpretation of the let is standard, $h$ stands for the hole $\emptyset$. In order to distinguish between the “assertive to conjectural” negation $t^\perp$ and “conjectural to assertive” negation $m^\perp$ we pair the terms $t$ and $m$ with 0 and 1 respectively. Note how “filling the hole” of a context is translated in $\lambda$-calculus as an application between the context and the term in the hole. The remaining terms are those of usual $\lambda$-calculus and are interpreted in the obvious way.

**Lemma 3.1.** The translation given in definition 3.6 is sound with respect to the $\beta$ reduction.

**Proof:**

we will show that for each redex $t$ given in definition 3.4 if $t \succ u$ then $t^\lambda \succ^* u^\lambda$:

- \[
(\text{let } x=y \text{ in } \tau) \xrightarrow{\omega} \tau[t/x][u/y] \quad \begin{cases} 
(\lambda h(\lambda x(\lambda y.\tau^\lambda)\pi_1 h)\pi_0 h)(t^\lambda,u^\lambda) \xrightarrow{\omega} \tau^\lambda[t^\lambda/x][u^\lambda/y] 
\end{cases}
\]
let $y = (\lambda x.t)u$ in $\tau$ \rightarrow $\tau[t[u/x]/y]$

$(\lambda h(\lambda y.\tau^\lambda)(h)\lambda x.t^\lambda)$ \rightarrow $\tau^\lambda[t^\lambda[u^\lambda/x]/y]$

let $x^\perp = t^\perp$ in $\tau$

$(\lambda h(\lambda x.\tau^\lambda)\pi_1 h)\langle 0, t^\lambda \rangle$ \rightarrow $\tau^\lambda[t^\lambda/x]$

and similar for the remaining redexes.

**Definition 3.7.** Two $\lambda$ terms $t$ and $u$ are $\beta$-equivalent, written $t \simeq^{\beta} u$ if there exist a term $s$ such that $t \triangleright^{*} s$ and $u \triangleright^{*} s$.

**Lemma 3.2.** The commutative equivalences of terms given in definition 3.5 are $\beta$-equivalences in the translation $(\ )^\lambda$:

**Proof:**
we have to show that $t \simeq_c u$ implies $(t)^\lambda \simeq^{\beta} (u)^\lambda$.

Consider the equivalence

let $(x_1, x_2)^y = y$ in $\lambda z. t \simeq_c \lambda z. \text{let } (x_1, x_2)^y = y \text{ in } t$

induced by the permutation $\supset R/ \cap L + C$ we have

let $(x_1, x_2)^y = y$ in $\lambda z. t$

$(\lambda x_1(\lambda x_2.\lambda z. t^\lambda)\pi_1 y)\pi_0 y$

$\simeq^{\beta} \lambda z(\lambda x_1(\lambda x_2. t^\lambda)\pi_1 y)\pi_0 y$

the other cases are similar.

**Theorem 3.1.** The simply typed let calculus for $\text{ILP}_{AC}$ is strong normalizable.

**Proof:**
we show that the lambda terms obtained from the translation of the well typed terms of the let calculus (definition 3.4) are well typed. This is enough to prove the strong normalization property because, by lemma 3.2 and 3.1, an infinite sequence of reduction for the let calculus would be translated in an
infinite sequence of reduction for the simply typed \( \lambda \) calculus but this is impossible as the latter is strong normalizable (proved in the next section). The proof is by induction on the complexity of the 1\texttt{et}-terms. We give the proofs for the most intricate cases of a redex and a context.

- \((\texttt{let}(x,y)\Rightarrow(t,u) \text{ in } \tau)\) = \((\lambda h(\lambda x(\lambda y.\tau)\pi_1 h)\pi_0 h)(t^\lambda, u^\lambda)\)

Given well typed terms \( \Gamma_0 \vdash \tau^\lambda : \varnothing, \Gamma_1 \vdash u^\lambda : \varnothing \) and \( \Gamma \vdash \tau^\lambda : \varnothing \), the term \((\lambda h(\lambda x(\lambda y.\tau)\pi_1 h)\pi_0 h)(t^\lambda, u^\lambda)\) is well typed, and therefore strong normalizable, by the following derivation:

\[
\begin{array}{c}
\frac{\tau^\lambda : \varnothing}{\lambda y.\tau : \varnothing} \\
\frac{\lambda x(\lambda y.\tau): \pi_1 h : \varnothing}{\lambda h(\lambda x(\lambda y.\tau)\pi_1 h)\pi_0 h : \varnothing} \\
\frac{[h : \varnothing \cap \varnothing]_1^3}{\pi_1 h : \varnothing} \\
\frac{\lambda x(\lambda y.\tau) : \pi_1 h : \varnothing}{\lambda h(\lambda x(\lambda y.\tau)\pi_1 h)\pi_0 h : \varnothing} \\
\frac{[h : \varnothing \cap \varnothing]_2^3}{\pi_0 h : \varnothing} \\
\frac{(\lambda x(\lambda y.\tau) : \pi_1 h : \varnothing)}{\lambda h(\lambda x(\lambda y.\tau)\pi_1 h)\pi_0 h : \varnothing} \\
\frac{(t^\lambda : \varnothing)}{\pi_1 h : \varnothing} \\
\frac{[h : \varnothing \cap \varnothing]_1^3}{\pi_1 h : \varnothing} \\
\frac{(t^\lambda, u^\lambda) : \varnothing}{(t^\lambda, u^\lambda) : \varnothing} \\
\frac{(\lambda h(\lambda x(\lambda y.\tau)\pi_1 h)\pi_0 h)(t^\lambda, u^\lambda) : \varnothing}{(\lambda h(\lambda x(\lambda y.\tau)\pi_1 h)\pi_0 h)(t^\lambda, u^\lambda) : \varnothing} \\
\end{array}
\]

- \((\texttt{let} x=[] u \text{ in } \tau)\) = \(\lambda h(\lambda x.\tau^\lambda)(h)u^\lambda\)

Given well typed terms \( \Gamma \vdash \tau^\lambda : \varnothing \) and \( \Gamma_1 \vdash u^\lambda : \varnothing \), the term \(\lambda h(\lambda x.\tau^\lambda)(h)u^\lambda\) is well typed, and therefore strong normalizable, by the following derivation:

\[
\begin{array}{c}
\frac{\tau^\lambda : \varnothing}{\lambda x.\tau^\lambda : \varnothing} \\
\frac{\lambda h(\lambda x.\tau^\lambda)(h)u^\lambda : \varnothing}{\lambda h(\lambda x.\tau^\lambda)(h)u^\lambda : \varnothing} \\
\frac{[h : \varnothing \cap \varnothing]_1^3}{\pi_1 h : \varnothing} \\
\frac{(h)u^\lambda : \varnothing}{(h)u^\lambda : \varnothing} \\
\frac{[h : \varnothing \cap \varnothing]_1^3}{\pi_1 h : \varnothing} \\
\frac{(h)u^\lambda : \varnothing}{(h)u^\lambda : \varnothing} \\
\frac{(\lambda h(\lambda x.\tau^\lambda)(h)u^\lambda) : \varnothing}{(\lambda h(\lambda x.\tau^\lambda)(h)u^\lambda) : \varnothing} \\
\end{array}
\]

\( \square \)

**Remark 3.2.** It should be noticed that the translation of contexts does not respect the typing of the system because in the 1\texttt{et} calculus the type of the context is the type of the hole while in its \( \lambda \) translation is the type of the whole term. However we do not need the translation to be type-preserving to prove the strong normalization.

Each \( \beta \) reduction (definition 3.4) and each commutative equivalence (definition 3.5) encode, respectively, a symmetric reduction and a commutative reduction in the process of cut elimination (theorem 2.2). Thus the process of strong normalization encodes the process of cut elimination for typed \( \text{ILP}_{\text{AC}} \).
4. Strong Normalization for Simply Typed $\lambda$ Calculus with Pairings and Projections

The part regarding $\text{ILP}_{AC}$ finishes with section 3. Here we propose a new, elegant and self-contained proof of simply typed $\lambda$ calculus with pairings and projections. This section is not essential to the paper because to conclude the strong normalization for $\text{ILP}_{AC}$ we could refer to other standard proofs of strong normalization for simply typed $\lambda$ calculus with pairings and projections (for instance in [12]).

**Definition 4.1. (The grammar)**

$$u, t ::= x | \lambda x.t | \langle u, t \rangle | \pi_0 t | \pi_1 t | (u) t$$

for simplicity we denote $t_1 t_2 ... t_n = (...(t_1 t_2 ... t_{n-1}) t_n$.

We remind that every term of the $\lambda$-calculus with pairings and projections has the form

$$\lambda y_1 ... \lambda y_m. (\pi^{*n} ... (\pi^{*2}(\pi^{*1} h) t_1) t_2 ...) t_n$$

where each $\pi^{*i}$ is a sequence, possibly void, of projections $\pi_0$ or $\pi_1$ and in a term $\pi^*(u)t$, projections are applied to the result of the application of $u$ to $t$. The head $h$ has one of the following forms:

$$h ::= x | \lambda x.t | \langle u, t \rangle$$

We define $SN$ to be the set of strong normalizable terms and $\succ^*$ the reflexive and transitive closure of the $\beta$ reduction $\succ$.

**Definition 4.2.** A lambda term $t$ is *elementary* if $t \notin SN$ but all the proper subterm of $t$ are in $SN$.

Note that any elementary term is of the form

- $(\pi^{*n} ... (\pi^{*2}(\lambda x.u) t_2 ...) t_n$ (with $u, t_2, ..., t_n \in SN$);
- $(\pi^{*n} ... (\pi^{*1} \pi_i (u_0, u_1)) t_1 ...) t_n$ (with $i = 0$ or $i = 1$ and $u_0, u_1, t_1, ..., t_n \in SN$).

where $(\lambda x.u) t$ or $\pi_i (u_0, u_1)$ is the head redex. For example $(\lambda x.x)(\lambda x.x)$ or $(\pi_0 (\pi_1 \langle z, \lambda y. (\lambda x.x, u) \rangle)y)\lambda x.x$ are elementary terms. Note that the head of an elementary term must reduce otherwise, as $t_1, ..., t_n \in SN$, it would surely not have an infinite reduction.

**Proposition 4.1.** If $v \notin SN$ then $v$ has an elementary subterm.

**Proof:**

by induction on the complexity of $v$:

- $v = x$, trivial case;
- $v \notin SN$ and all the proper subterms of $v$ are in $SN$ then $v$ is elementary (base case);
- $v = ut$ and $u \notin SN$ or $t \notin SN$, apply inductive hypothesis on $u$ or $t$ respectively;
- $v = \lambda x.u$ then $u \notin SN$, apply inductive hypothesis on $u$;
- $v = \langle u, t \rangle$ then $u \notin SN$ or $t \notin SN$. Apply induction hypothesis on $u$ or $t$;
Definition 4.3. Given \( t \notin SN \) we define:

- the standard subterm of \( t \) as its leftmost elementary subterm;
- the standard redex of \( t \) the head redex of its standard subterm;
- the \( \beta \) standard reduct of \( t \) the term \( t' \) obtained from \( t \) contracting the standard redex. We will write \( t \triangleright_{st} t' \);
- the standard reduction of \( t \) the succession of terms obtained from \( t \) applying only \( \beta \) standard reductions. Thus if \( t \triangleright_{st} t_1 \triangleright_{st} t_n \triangleright_{st} \ldots \) then the standard reduction of \( t \) is \( t, t_1, \ldots, t_n, \ldots \).

Proposition 4.2.

1. if \( (\pi^n \ldots (\pi^2 u[t/x])t_2 \ldots )t_n \in SN \) then \( (\pi^n \ldots (\pi^2 (\lambda x. u)t)t_2 \ldots )t_n \in SN \);
2. if \( (\pi^n \ldots (\pi^1 u_i) t_1 \ldots )t_n \in SN \) and \( u_{1-j} \in SN \) then \( (\pi^n \ldots (\pi^1 \pi_i(u_0, u_1))t_1 \ldots )t_n \in SN \).

Proof:

1. This part is standard. See for instance [15].
2. Look at the possible reductions of \( (\pi^n \ldots (\pi^1 \pi_i(u_0, u_1))t_1 \ldots )t_n \), we have four cases :
   - (a) a term within \( t_1, \ldots, t_n \) is reduced;
   - (b) \( u_0 \) is reduced;
   - (c) \( u_1 \) is reduced;
   - (d) \( \pi_i(u_0, u_1) \) is reduced.

In all the cases the result of the reduction is bounded in length as \( u_0, u_1, t_1, \ldots, t_n \) are strong normalizable by hypothesis. Therefore \( (\pi^n \ldots (\pi^1 \pi_i(u_0, u_1))t_1 \ldots )t_n \in SN \).

Proposition 4.3. \( t \notin SN \) iff the standard reduction of \( t \) is infinite.

Proof:

the implication form right to left is obvious. As regards the implication from left to right if \( t \notin SN \) then, by proposition 4.1, it has an elementary subterm. Let’s call \( s \) the leftmost elementary subterm of \( t \). Then \( t \triangleright_{st} t' \) by a head reduction of \( s \) into \( s' \) which can be in two forms:

1. \( (\pi^n \ldots (\pi^2 (\lambda x. u)t) t_2 \ldots )t_n \triangleright (\pi^n \ldots (\pi^2 u[t/x]) t_2 \ldots )t_n \);
2. \( (\pi^n \ldots (\pi^1 \pi_i(u_0, u_1)) t_1 \ldots )t_n \triangleright (\pi^n \ldots (\pi^1 u_i) t_1 \ldots )t_n \).

Using contraposition from proposition 4.2 and bearing in mind that \( u_{1-j} \notin SN \) is false as we are reducing an elementary term, \( s' \notin SN \). Therefore the standard reduction preserves the non strong normalizability as \( s' \) is a subterm of \( t' \); it is infinite as we can always find the \( \beta \) standard redex to reduce.
Proposition 4.4. Let \( u, t \in SN \) and \( u[t/x] \not\in SN \) then there exists \( v \) such that \( u[t/x] \mathrel{\succ^*_st} v \) and the standard subterm of \( v \) is in the form
\[
(\pi^{s_n}...\pi^{s_1}t_1...)t_n
\]

Proof:
by in induction on \( h(u) \) where, given a term \( s \), \( h(s) \) is the length of the maximal reduction of \( s \). Let the cases considered are the following:

- \( u[t/x] \mathrel{\succ^*_st} u'[t/x] \) apply inductive hypothesis on \( u' \);
- \( u[t/x] \mathrel{\succ^*_st} u' \) because of a redex in \( t \) then, as the standard reduction reduces the head redex of the standard subterm, the standard subterm of \( u[t/x] \) is \( (\pi^{s_n}...\pi^{s_1}t_1...)t_n \) and we end with \( u[t/x] = v \);
- \( u[t/x] \mathrel{\succ^*_st} u' \) and the redex is not in \( t \) nor in \( u \). Therefore \( t \) is a lambda abstraction or a pair in order for \( u[t/x] \) to reduce:
  - \( t \) is a lambda abstraction and the standard subterm of \( u[t/x] \) is in the form \( (\pi^{s_n}...\pi^{s_2}(t)_{t_2...})t_n \).
    We end with \( u[t/x] = v \);
  - \( t \) is a pair and the standard subterm of \( u[t/x] \) is in the form \( (\pi^{s_n}...\pi^{s_1}\pi_1(t)_{t_1...})t_n \). We end with \( u[t/x] = v \).

\( \square \)

Proposition 4.5. Let \( ut \) be elementary. There exists an elementary term \((\lambda x.s)t\) such that \( ut \mathrel{\succ^*_st} (\lambda x.s)t \).

Proof:
by induction on \( h(u) \). We remind that, by definition 4.2, the head of an elementary term must be a redex. Therefore the cases considered are the following:

- \( ut = (\lambda x.s)t \) and we have the result;
- \( (\pi^{s_n}(...\pi^{s_2}(\lambda x.v)_{t_1})_{t_2...})_{t_{n-1}} \mathrel{\succ^*_st} (\pi^{s_n}(...\pi^{s_2}v[t_1/x])_{t_2...})_{t_{n-1}}t \). Apply inductive hypothesis with \( u = \pi^{s_n}(...\pi^{s_2}v[t_1/x])_{t_2...} \);
- \( (\pi^{s_n}(...\pi^{s_2}\pi_i(u_0, u_1)_{t_1...})_{t_{n-1}} \mathrel{\succ^*_st} (\pi^{s_n}(...\pi^{s_1}u_i)_{t_1...})_{t_{n-1}}t \). Apply inductive hypothesis with \( u = \pi^{s_n}(...\pi^{s_1}u_i)_{t_1...} \).

\( \square \)

Proposition 4.6. Let \( ut \) be elementary. There exists \( v \) such that \( ut \mathrel{\succ^*_st} v \) and the standard subterm of \( v \) has the form \( (\pi^{s_n}...\pi^{s_1}t_1...)t_n \) and the head \( t \) is obtained by substitution of \( t \) to some \( x \) in \( u \).

Proof:
by proposition 4.5 and 4.4.

\( \square \)

Theorem 4.1. Every simply typed lambda term \( z \) with pairings and projections is strong normalizable.
Proof: suppose \( z \notin SN \). Let \( E \) be the set of typed elementary terms: \( E \) is not empty. Consider some \( ut \in E \) such that the length (logical complexity) of the type of \( t \) is minimal among all the \( ut \in E \) and all the type assignments given to such \( ut \). Then for no \( u't' \in E \) there is an assignment of a shorter type to \( t' \). By proposition 4.6 \( ut \succst v \) and the standard subterm of \( v \) has the form \((\pi^*n...(\pi^*1)t_1...)t_n\). The type of \( t \) has the same type it has in \( ut \) by subject reduction because \( ut \) reduces to \((\lambda x.t')t \), then to \( t'[t/x] \) and one substitution of \( t \) to \( x \) produces the head of \((\pi^*n...(\pi^*1)t_1...)t_n\). The type of \( t \) strictly includes the type of \( t_n \). Now \((\pi^*n...(\pi^*1)t_1...)t_n \) is a term in \( E \) of the form \( u't_n \) with an assignment to \( t_n \) of a shorter type than \( t \) but this contradicts the choice of the type assignment for \( t \). Therefore \( E = \emptyset \) and \( z \in SN \) because it cannot have any elementary subterm.

4.1. Conclusion and Future Works

In this paper we have provided a term assignment, in the style of Herbelin’s \( \Lambda \lambda \) calculus, to the system \( ILP_{AC} \) of Polarized Bi-intuitionistic Logic. We have proved that this calculus has the strong normalization property through a sound translation into the simply typed \( \lambda \) calculus with pairings and projections. We have been guided by our informal interpretation (definition 2.2) of \( ILP_{AC} \) where assertions are given justification conditions, i.e. the set of their proofs, and conjectures are interpreted in terms of their refutation conditions. If we think of this interpretation as an extension of Heyting’s semantic to Polarized Bi-intuitionistic Logic, then we must acknowledge that it is not very satisfactory. In this paper a refutation of \( C \) is regarded as isomorphic to a proof of \( C \perp \). Another approach, in which a refutation of \( C \) is regarded as dual to a proof of \( C \perp \), is under development. This task presents some conceptual and technical challenges: we may need to consider also non conclusive arguments for a conjecture, and provide a suitable formalization of them. A more sophisticated approach is suggested by the semantics of linear logic: in this setting justification and refutation conditions, for both assertions and conjectures, could be characterized in a purely abstract way. Thus it could be interesting to look at Polarized Bi-intuitionistic Logic through its translation into linear logic.

The combinatorial proof of strong normalization in section 4 comes from an attempt to characterize the class of non strongly normalizable \( \lambda \) terms by describing what happens during a particular infinite reduction. It provides a strikingly simple proof when compared with the existing ones. This particular reduction of a non strongly normalizable term, which we call “standard reduction”, is an infinite chain of terms \( t_1, t_2, t_3, ..., t_n, ... \) such that every \( t_i \) is applied, somewhere in the standard reduction, to \( t_{i+1} \). In other words: \( t_2 \) appears as one of the arguments of \( t_1 \) in some point of the standard reduction; then \( t_3 \) appears as one of the arguments of \( t_2 \) in another point of the standard reduction, and so on, giving rise to an infinite chain. We have not written down the details of this outlined characterization of pure non strong normalizable \( \lambda \) terms, but it is a straightforward corollary.

References


5. Appendix

### Axioms

<table>
<thead>
<tr>
<th>Axioms</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Theta, x : \vdash \alpha \Rightarrow x : \vdash \alpha ; \top )</td>
<td>( \Theta ; \alpha : \top \Rightarrow ; \top )</td>
</tr>
<tr>
<td>( \Theta ; \bot : \land \Rightarrow ; \top )</td>
<td>( \Theta ; \Rightarrow \top : \lor ; \top )</td>
</tr>
</tbody>
</table>

### Cut rules

<table>
<thead>
<tr>
<th>Cut rules</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Theta \Rightarrow t : \emptyset ; \top ) ( \Theta \models E : \emptyset \Rightarrow \epsilon' ; \top )</td>
<td>( \Theta \Rightarrow t : \emptyset ; \top ) ( \Theta \models F : \emptyset \Rightarrow \epsilon ; \top ) cut</td>
</tr>
<tr>
<td>( \Theta \Rightarrow \epsilon' ; P : v \mid \top ) ( \Theta \models l : v \Rightarrow \epsilon' ; \top )</td>
<td>( \Theta \models ; \epsilon \Rightarrow Q : v \mid \top ) ( \Theta \models l : v \Rightarrow \top ) cut</td>
</tr>
</tbody>
</table>

### Contraction rules

<table>
<thead>
<tr>
<th>Contraction rules</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Theta \models E : \emptyset \Rightarrow \epsilon' ; \top ) ( x : \emptyset \in \Theta ) ( \Theta \Rightarrow E[x] : \epsilon' ; \top )</td>
<td>( \Theta \models F : \emptyset \Rightarrow \epsilon ; \top ) ( x : \emptyset \in \Theta ) ( \Theta \Rightarrow F[x] : \epsilon ; \top ) cut</td>
</tr>
<tr>
<td>( \Theta \Rightarrow \epsilon' ; P : v \mid \top ) ( a : v \in \top )</td>
<td>( \Theta \models ; \epsilon \Rightarrow Q : v \mid \top ) ( a : v \in \top ) ( \Theta \Rightarrow Q[a] : \epsilon \Rightarrow ; \top ) cut</td>
</tr>
</tbody>
</table>

### Context rules

<table>
<thead>
<tr>
<th>Context rules</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Theta, x : \vdash \alpha \Rightarrow x : \vdash \alpha ; \top )</td>
<td>( \Theta ; a : \top \Rightarrow ; a : \top ; \top ) IL ( \Theta ; a : \top \Rightarrow ; a : \top ; \top ) IR</td>
</tr>
</tbody>
</table>

Table 2. \( \text{ILP}_{\text{AC}} \): term assignment to structural rules.
<table>
<thead>
<tr>
<th>Assertive rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ \Theta, x : \theta_1 ; t : \theta_2 ; Y \supset R ]</td>
</tr>
<tr>
<td>[ \Theta ; \Rightarrow \lambda x.t : \theta_1 \supset \theta_2 ; Y \supset R ]</td>
</tr>
<tr>
<td>[ \Theta ; \Rightarrow t : \theta_1 ; Y x : \theta_2, \Theta ; \Rightarrow u : \epsilon ; Y \supset L.1 ]</td>
</tr>
<tr>
<td>[ \Theta ; \Rightarrow t : \theta_1 ; Y \theta \mid \text{let } x \mapsto [ t \in u : \theta_1 \supset \theta_2 ; \Rightarrow \epsilon ; Y \supset L.1 ]</td>
</tr>
<tr>
<td>[ \Theta ; \Rightarrow t : \theta_1 ; Y \theta \mid \text{let } x \mapsto [ t \in m : \theta_1 \supset \theta_2 ; \epsilon \Rightarrow ; Y \supset L.1 ]</td>
</tr>
<tr>
<td>[ \Theta, x : \theta_1, y : \theta_1 ; \Rightarrow t : \epsilon ; Y \supset L.1 ]</td>
</tr>
<tr>
<td>[ \Theta ; \Rightarrow (t, u) : \theta_1 \cap \theta_2 ; Y \supset R ]</td>
</tr>
<tr>
<td>[ \Theta, x : \theta_0, y : \theta_1 ; m : \epsilon \Rightarrow ; Y \supset L.2 ]</td>
</tr>
<tr>
<td>[ \Theta ; \Rightarrow (t, u) : \theta_1 \cap \theta_2 ; Y \supset R ]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Conjectural rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ \Theta ; m : v_1 \Rightarrow ; Y a : v_2 \supset L ]</td>
</tr>
<tr>
<td>[ \Theta ; \lambda a.m : v_1 \times v_2 \Rightarrow ; Y \supset L ]</td>
</tr>
<tr>
<td>[ \Theta ; \Rightarrow t : \epsilon ; a : v_1, Y \Theta ; l : v_2 \Rightarrow ; Y \supset R.2 ]</td>
</tr>
<tr>
<td>[ \Theta ; \Rightarrow \epsilon ; \text{let } a = [ l \in m : v_1 \times v_2 ] ; Y \supset R.2 ]</td>
</tr>
<tr>
<td>[ \Theta ; m : v_1 \Rightarrow ; Y \Theta ; n : v_2 \Rightarrow ; Y \supset \gamma L ]</td>
</tr>
<tr>
<td>[ \Theta ; \Rightarrow \epsilon ; \text{let } (a, b) = [ l \in m : v_1 \times v_2 ] ; Y \supset \gamma L ]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mixed rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ \Theta, x : \theta ; \Rightarrow t : \epsilon ; Y \supset RAC.1 ]</td>
</tr>
<tr>
<td>[ \Theta ; \Rightarrow \epsilon ; \text{let } x^{+} = [ t \in \theta ] ; Y \supset RAC.2 ]</td>
</tr>
<tr>
<td>[ \Theta ; \Rightarrow t : \epsilon ; Y a : v \supset LCA.1 ]</td>
</tr>
<tr>
<td>[ \Theta ; m : v \Rightarrow ; Y a : v \supset LCA.2 ]</td>
</tr>
<tr>
<td>[ \Theta ; \Rightarrow t : \theta ; Y \supset LAC ]</td>
</tr>
<tr>
<td>[ \Theta ; m : v \Rightarrow ; Y \supset RCA ]</td>
</tr>
</tbody>
</table>

Table 3. ILPAC : term assignment to inference rules.