Intersection Types for Light Affine Lambda Calculus

Daniel de Carvalho [1]

Institut de Mathématiques de Luminy, UMR 6206
163, avenue de Luminy, case 907
13288 Marseille Cedex 9, France

Abstract

Light Affine Lambda Calculus is a term calculus for polynomial time computation ([12]). Some of the terms of *Light Affine Lambda Calculus* must however be regarded as errors. *Intuitionistic Light Affine Logic* (**ILAL**) types only terms without errors, but not all of them. We introduce two type assignment systems with intersection types : in the first one, typable pseudo-terms are exactly the terms without errors ; in the second one, they are exactly those that reduce to normal terms without errors.

Key words: Implicit Computational Complexity, Intersection Types, Lambda Calculus, Linear Logic

Introduction

One approach to provide languages corresponding to polynomial time computation is that of the proofs-as-programs paradigm and Linear Logic ([7]). In particular, two variants of Linear Logic with a polynomial cut-elimination have been proposed : Light Linear Logic ([8]) and Soft Linear Logic ([10]). They can be seen as refinements of System F allowing to characterize polynomial time functions : by the Curry-Howard correspondence, these systems allow to write programs which can be evaluated in polynomial time.

Two new type free term calculi based on their essential ideas appeared too : *Light Affine Lambda Calculus* ([12]) and *Soft Lambda Calculus* ([3]). K. Terui introduced *Light Affine Lambda Calculus* as a refinement of the type free lambda calculus for which *Intuitionistic Light Affine Logic* (**ILAL**) ([1]), a variant of Light Linear Logic, provides a type assignment system and he proved that it satisfies the polystep strong normalizability : an (untyped) term is normalizable in a polynomial number of steps by any reduction. This seems

[1] Email: carvalho@iml.univ-mrs.fr

to suggest that **ILAL** types are useless. However this is not the case, because if types are not needed to ensure the complexity bound on reduction, they are actually useful to ensure the term does reduce to a sensible result. Indeed light lambda-terms carry more information than ordinary lambda-terms and there is a forgetful map (erasure) from these terms to ordinary lambda- terms. Some light lambda-terms might be in normal form but correspond to ordinary lambda-term with redexes. Indeed some light lambda terms show configurations which can be naturally seen as errors (in particular pattern-matching errors) or deadlocks. So one would like to be able to account for all *usable* light lambda- terms, those for which normalization can be performed without reaching an error. This is all the more natural as light lambda-calculus can be used in other settings than second order **ILAL** : in particular, in [13], K. Terui uses it to extract programs from light affine set theory proofs. Finally, extensions of **ILAL** using recursive types like in [3], can also be considered .

In the present work, we tackle the problem of characterizing light lambda-terms without errors. In section 3, we give a formal definition of the terms which we can reasonably regard as terms without errors : we name them *reasonable terms*. First, we note that, in **ILAL**, even if every typable pseudo-term is a reasonable term (Theorem 3.8) and every normal reasonable term is typable (Theorem 3.9), this system doesn't capture all the reasonable terms (Remark 3.10). Therefore we introduce a new type assignment system, called *Light Intersection Type Assignment System* (**LI**), a system with intersection types. Intersection types were introduced in [5] to overcome the limitations of Curry's type discipline and have been used, for instance, to characterize strongly normalizable terms, solvable terms and normalizable terms (see, e.g., [9]). We overcome the limitations of **ILAL** : we show that typable pseudo-terms in **LI** are exactly the reasonable terms (Theorem 4.17).

Lastly, we introduce a *Relaxed Light Intersection Type Assignment System* (**RLI**) in which typable pseudo-terms are exactly the terms that reduce in terms without errors : the possible errors that the typable pseudo-term contained will be erased during the reduction. So, this sytem, unlike the previous one, has the following property : given two equivalent terms t and t', which amounts to saying that they reduce to the same normal form, t is typable if, and only if, t' is typable.

With another point of view, all this obviously shows that the problems of typability in **LI** and in **RLI**, unlike in the intersection type assignment systems for the Lambda Calculus, are decidable. As far as we know, this problem for **ILAL** is an open question.

In other respects, *Soft Lambda Calculus* has the same particularities as *Light Affine Lambda Calculus* ; so, we conjecture that a similar work could be done for it. With regard to semantics, P. Baillot gave a model for Light Linear Logic in [2] : the present work gives us hope that, inspired by it, we will give a semantics of *Light Affine Lambda Calculus* in the near future.

Notation

If S is a set, then $\mathcal{P}_f(S)$ denotes the set of the finite subsets of S and $\mathcal{P}_f^*(S)$ denotes $\mathcal{P}_f(S) \setminus \{\emptyset\}$.

1 Light Affine Lambda Calculus

For the requisite materials about *Lambda Calculus*, the reader can refer to [4] and [9]. Here we recall only the requisite materials about *Light Affine Lambda Calculus* ; see [12] for a full exposition.
The definition of the terms is done in two steps : first, we define the pseudo-terms ; second, the terms are defined by imposing certain conditions on the pseudo-terms.

Definition 1.1 The set $\mathcal{PT}$ of pseudo-terms is defined by the following grammar :

$$\mathcal{PT} ::= \mathcal{V} \mid (\mathcal{PT})\mathcal{PT} \mid \lambda\mathcal{V}.\mathcal{PT} \mid !\mathcal{PT} \mid \mathsf{let}\ \mathcal{PT}\ \mathsf{be}\ !\mathcal{V}\ \mathsf{in}\ \mathcal{PT} \mid \S\mathcal{PT} \mid \mathsf{let}\ \mathcal{PT}\ \mathsf{be}\ \S\mathcal{V}\ \mathsf{in}\ \mathcal{PT}$$

In the sequel, the symbol † stands for either ! or § ; moreover one identifies α-congruent pseudo-terms (the occurrences of x in v are bound in $\mathsf{let}\ u\ \mathsf{be}\ \dagger\ x\ \mathsf{in}\ v$) : that's why in our proofs, we can always assume that free variables are different from bound variables. For all pseudo-terms t, $FV(t)$ denotes the set of free variables in t, $FO(x,t)$ denotes the number of free occurrences of x in t and $FO(t)$ denotes the number of free occurrences of all variables in t.
The size of a pseudo-term is the number of nodes in its term tree. Given a pseudo-term t and an adress w, the depth of w in t is the number of !-boxes and §-boxes enclosing the subexpression at w. The depth of t is the maximum depth of all adresses in it.

Definition 1.2 Let $X, Y, Z \in \mathcal{P}_f(\mathcal{V})$ mutually disjoint. Then $\mathcal{T}_{X,Y,Z}$ is the set of pseudo-terms defined as follows :

- $x \in \mathcal{T}_{X,Y,Z} \Leftrightarrow x \in X$;
- $\lambda x.t \in \mathcal{T}_{X,Y,Z} \Leftrightarrow t \in \mathcal{T}_{X\cup\{x\},Y,Z}, x \notin X, \mathrm{FO}(x,t) \leq 1$;
- $(t)u \in \mathcal{T}_{X,Y,Z} \Leftrightarrow t, u \in \mathcal{T}_{X,Y,Z}$;
- $!t \in \mathcal{T}_{X,Y,Z} \Leftrightarrow t \in \mathcal{T}_{Y,\emptyset,\emptyset}, \mathrm{FO}(t) \leq 1$;
- $\S t \in \mathcal{T}_{X,Y,Z} \Leftrightarrow t \in \mathcal{T}_{Y\cup Z,\emptyset,\emptyset}$;
- $\mathsf{let}\ t\ \mathsf{be}\ !x\ \mathsf{in}\ u \in \mathcal{T}_{X,Y,Z} \Leftrightarrow t \in \mathcal{T}_{X,Y,Z}, u \in \mathcal{T}_{X,Y\cup\{x\},Z}, x \notin Y$;
- $\mathsf{let}\ t\ \mathsf{be}\ \S x\ \mathsf{in}\ u \in \mathcal{T}_{X,Y,Z} \Leftrightarrow t \in \mathcal{T}_{X,Y,Z}, u \in \mathcal{T}_{X,Y,Z\cup\{x\}}, x \notin Z, \mathrm{FO}(x,u) \leq 1.$

Finally, t is a term ($t \in \mathcal{T}$) if $t \in \mathcal{T}_{X,Y,Z}$ for some X, Y and Z.

Lemma 1.3 *Let $t \in \mathcal{T}_{X,Y,Z}$. If $x \notin FV(t)$, then $t \in \mathcal{T}_{X\setminus\{x\},Y\setminus\{x\},Z\setminus\{x\}}$.*

Examples :

- $\lambda x.\mathsf{let}\ x\ \mathsf{be}\ !y\ \mathsf{in}\ y$, $\lambda x.\mathsf{let}\ x\ \mathsf{be}\ !y\ \mathsf{in}\ !!y \notin \mathcal{T}$;

Name	*Redex*	*Contractum*
(β)	$(\lambda x.t)u$	t[u/x]
(!)	let $!u$ be $!x$ in t	t[u/x]
(§)	let $\S u$ be $\S x$ in t	t[u/x]
(com)	(let u be $\dagger\, x$ in $t)v$	let u be $\dagger\, x$ in $(t)v$
	let let u be $\dagger_1\, x$ in t be $\dagger_2\, y$ in v	let u be $\dagger_1\, x$ in let t be $\dagger_2\, y$ in v

Fig. 1. Reduction rules

- $\lambda x.$let x be $!z$ in $\S\lambda y.(z)\ldots(z)y \in \mathcal{T}$.

K. Terui provided a quadratic time algorithm checking whether a given pseudo-term is a term.
The reduction rules are those given in Figure 1 with the following restriction : in the rule (com), $x \notin FV(v)$. $\longrightarrow_0$ denotes the one step reduction and $\longrightarrow$ denotes the transitive reflexive closure of $\longrightarrow_0$.

K. Terui proved the following proposition and theorem :

Proposition 1.4 *If $t \in \mathcal{T}_{X,Y,Z}$ and $t \longrightarrow u$, then $u \in \mathcal{T}_{X,Y,Z}$.*

Theorem 1.5 *For every term t_0 of size s and depth d, the following hold :*

(i) every reduction sequence from t_0 has a length bounded by $O(s^{2^{d+1}})$;

(ii) every term to which t_0 reduces has a size bounded by $O(s^{2^d})$.

So, by König's Lemma, for all terms t, we can define $N(t)$ as the sum of the lengths of all possible reduction sequences. Moreover, applying Newman's Lemma, we obtain, as a corollary, that $\longrightarrow$ satisfies the Church-Rosser property.
Another corollary is the polytime strong normalization (see [12]).

The *erasure* of a term t is defined, by induction on t, to be a lambda-term :

- if $t = \dagger u$, then erasure(t) = erasure(u) ;
- if $t =$ let u be $\dagger\, x$ in v, then erasure(t) = erasure(v)[erasure$(u)/x$] ;
- erasure commutes to other constructions.

For all terms t and t', if $t \longrightarrow t'$, then erasure(t) β erasure(t'). But here is an example of a normal term, of which the erasure is a non normalizable lambda-term : let $\lambda x.$let x be $!y$ in $\S(y)y$ be $!y$ in $\S(y)y$. This is an example of a term that can be seen as an error.

2 Type Assignment System $\mathbf{ILAL_N}$

We present *Intuitionistic Light Affine Logic* as a type assignment system for *Light Affine Lambda Calculus*. $\mathbf{ILAL_N}$ is a second order type assignment

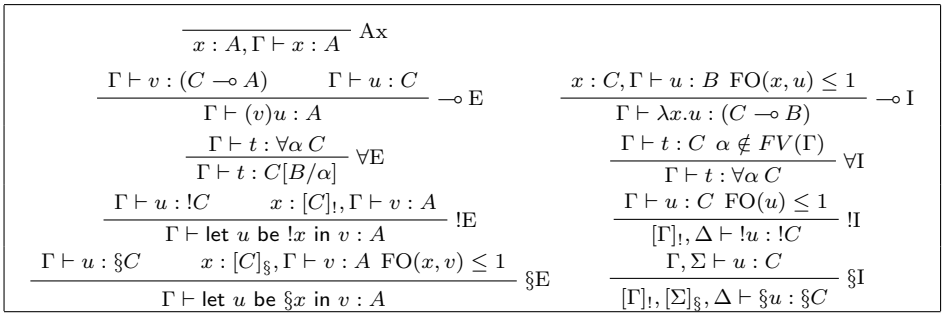

Fig. 2. Type Assignment System $\mathbf{ILAL_N}$

system in natural deduction style.

Definition 2.1 The types of $\mathbf{ILAL_N}$ are given by the following grammar : $\mathcal{F} ::= \mathcal{P} \mid (\mathcal{F} \multimap \mathcal{F}) \mid \forall\mathcal{P}\ \mathcal{F} \mid !\mathcal{F} \mid \S\mathcal{F}$.

$\exists, \otimes, \mathbf{1}, \&, \otimes$ and $\mathbf{0}$ are definable from $\multimap$ and $\forall$. In particular, $\mathbf{0} \equiv \forall\alpha\ \alpha$.

A †-discharged type is an expression of the form $[A]_\dagger$, where A is an (undischarged) type. A declaration is an expression of the form $x : A$ or $x : [A]_\dagger$. A context is a finite set of declarations.

If Γ is the context $x_1 : A_1, \ldots, x_n : A_n$ where all the types in it are undischarged, then $[\Gamma]_\dagger$ denotes the context $x_1 : [A_1]_\dagger, \ldots, x_n : [A_n]_\dagger$. If Γ contains a declaration with a discharged type, then $[\Gamma]_\dagger$ is undefined.

Definition 2.2 The type assignment rules of $\mathbf{ILAL_N}$ are those given in Figure 2.

The following theorems hold :

Theorem 2.3 *Every typable pseudo-term is a term.*

Theorem 2.4 ([8], [11]) *Every function $f : \{0,1\}^* \longrightarrow \{0,1\}^*$ which is computable in time $O(n^d)$ is represented by a term of type* $\mathbf{bint} \multimap \S^{d+6}\mathbf{bint}$.

3 True terms, safe terms and reasonable terms

We already encountered an example of term that can be seen as an error. In this section, we clarify this notion.
We begin by giving a formal definition of the normal terms without errors : we name them *true terms*.

Definition 3.1 We define the set $\mathcal{TT}$ of *true terms* and the set $\mathcal{WT}$ of *wise terms* as follows :
$\mathcal{TT} = \mathcal{TPT} \cap \mathcal{T}$ and $\mathcal{WT} = \mathcal{WPT} \cap \mathcal{T}$, where $\mathcal{RPT}$ and $\mathcal{WPT}$ are the sets defined by the following grammar :

$$\mathcal{WPT} ::= \mathcal{V} \mid (\mathcal{WPT})\mathcal{TPT}$$
$$\mathcal{TPT} ::= \mathcal{WPT} \mid \lambda\mathcal{V}.\mathcal{TPT} \mid !\mathcal{TPT} \mid \mathsf{let}\ \mathcal{WPT}\ \mathsf{be}\ !\mathcal{V}\ \mathsf{in}\ \mathcal{TPT}$$
$$\mid \S\mathcal{TPT} \mid \mathsf{let}\ \mathcal{WPT}\ \mathsf{be}\ \S\mathcal{V}\ \mathsf{in}\ \mathcal{TPT}$$

Name	*Redex*	*Contractum*
$(Ap\dagger)$	$(\dagger v)u$	*error*
$(\lambda\dagger)$	$\mathsf{let}\ \lambda x.u\ \mathsf{be}\ \dagger\, y\ \mathsf{in}\ v$	*error*
$(\S!)$	$\mathsf{let}\ \S u\ \mathsf{be}\ !x\ \mathsf{in}\ v$	*error*
$(!\S)$	$\mathsf{let}\ !u\ \mathsf{be}\ \S x\ \mathsf{in}\ v$	*error*

Fig. 3. New reduction rules

Note that we have : {wise terms} $\subsetneqq$ {true terms} $\subsetneqq$ {normal terms}.

In order to justify this definition, it can be showed that if we add a new term *error* and the reduction rules given in Figure 3, then the new reduction relation has the Church-Rosser property (using Hindley-Rosen's Lemma) and every term reduces either to a true term, or to a term of which *error* is a subterm.

Definition 3.2 A term is said to be *safe* whenever it reduces to a true term ; it is said to be *reasonable* whenever every normal subterm of every term to which it reduces is true.

Note that we have : {true terms} $\subsetneqq$ {reasonable terms} $\subsetneqq$ {safe terms}.

Moreover, the erasure of any true term is normal and the erasure of any safe term is normalizable ; we will prove that the erasure of any reasonable term is strongly normalizable in Section 4.

Definition 3.3 One says that a formula is *open* if it doesn't begin by $\forall$.

Fact 3.4 *For all contexts* Γ*, for all open formulae* A*, for all variables* x*, for all terms* u*, if* $\Gamma \vdash \lambda x.u : A$*, then* A *is written* $(C \multimap B)$*.*

Fact 3.5 *For all contexts* Γ*, for all open formulae* A*, for all terms* u*, if* $\Gamma \vdash !u : A$*, then* A *is written* $!C$*.*

Fact 3.6 *For all contexts* Γ*, for all open formulae* A*, for all terms* u*, if* $\Gamma \vdash \S u : A$*, then* A *is written* $\S C$*.*

Lemma 3.7 *Every typable normal term is true.*

Proof. We prove, by induction on π, that for all derivations π, for all formulae A, for all contexts Γ, for all normal terms t, if π is a derivation of $\Gamma \vdash t : A$, then t is true :

- π is $\dfrac{}{x : A, \Gamma' \vdash x : A}\,\mathrm{Ax}$: t is a variable ;
- π ends in $\dfrac{\Gamma \vdash v : (C \multimap A) \qquad \Gamma \vdash u : C}{\Gamma \vdash (v)u : A} \multimap \mathrm{E}$: v is wise, because :
  - as $(v)u$ is normal, v is normal and isn't written $\lambda x.v_1$, nor $\mathsf{let}\ v_1\ \mathsf{be}\ \dagger\ x\ \mathsf{in}\ v_2$;
  - moreover, by the Facts 3.5 and 3.6, v isn't written $\dagger v_1$;

· finally, by the induction hypothesis, v is true ;
moreover, as $(v)u$ is normal, u is normal, therefore, by the induction hypothesis, u is true ;

- π ends in $\dfrac{x : C, \Gamma \vdash u : B \;\; \mathrm{FO}(x,u) \leq 1}{\Gamma \vdash \lambda x.u : (C \multimap B)} \multimap \mathrm{I}$: as $\lambda x.u$ is normal, u is normal, therefore, by the induction hypothesis, u is true ;
- π ends in $\dfrac{\Gamma \vdash t : \forall \alpha \, C}{\Gamma \vdash t : C[B/\alpha]} \forall \mathrm{E}$: apply the induction hypothesis ;
- π ends in $\dfrac{\Gamma \vdash t : C \;\; \alpha \notin FV(\Gamma)}{\Gamma \vdash t : \forall \alpha \, C} \forall \mathrm{I}$: apply the induction hypothesis ;
- π ends in $\dfrac{\Gamma \vdash u : !C \qquad x : [C]_!, \Gamma \vdash v : A}{\Gamma \vdash \mathsf{let}\ u\ \mathsf{be}\ !x\ \mathsf{in}\ v : A} !\mathrm{E}$: u is wise, because :
  · as $\mathsf{let}\ u\ \mathsf{be}\ !x\ \mathsf{in}\ v$ is normal, u is normal and isn't written $!u_1$, nor $\mathsf{let}\ u_1\ \mathsf{be}\ \dagger\ x\ \mathsf{in}\ u_2$;
  · moreover, by the Facts 3.4 and 3.6, u isn't written $\lambda y.u_1$, nor $\S u_1$;
  · finally, by the induction hypothesis, u is true ;
  moreover, as $\mathsf{let}\ u\ \mathsf{be}\ !x\ \mathsf{in}\ v$ is normal, v is normal, therefore, by the induction hypothesis, v is true ;
- π ends in $\dfrac{\Gamma' \vdash u : C}{[\Gamma']_!, \Delta \vdash !u : !C} !\mathrm{I}$: as $!u$ is normal, u is normal, therefore, by the induction hypothesis, u is true ;
- π ends in $\dfrac{\Gamma \vdash u : \S C \qquad x : [C]_\S, \Gamma \vdash v : A \;\; \mathrm{FO}(x,v) \leq 1}{\Gamma \vdash \mathsf{let}\ u\ \mathsf{be}\ \S x\ \mathsf{in}\ v : A} \S\mathrm{E}$: u is wise, because :
  · as $\mathsf{let}\ u\ \mathsf{be}\ !x\ \mathsf{in}\ v$ is normal, u is normal and isn't written $\S u_1$, nor $\mathsf{let}\ u_1\ \mathsf{be}\ \dagger\ x\ \mathsf{in}\ u_2$;
  · moreover, by the Facts 3.4 and 3.5, u isn't written $\lambda y.u_1$, nor $!u_1$;
  · finally, by the induction hypothesis, u is true ;
  moreover, as $\mathsf{let}\ u\ \mathsf{be}\ !x\ \mathsf{in}\ v$ is normal, v is normal, therefore, by the induction hypothesis, v is true ;
- π ends in $\dfrac{\Gamma, \Sigma \vdash u : C}{[\Gamma]_!, [\Sigma]_\S, \Delta \vdash \S u : \S C} \S\mathrm{I}$: as $\S u$ is normal, u is normal, therefore, by the induction hypothesis, u is true.

□

Theorem 3.8 *Every typable pseudo-term is a reasonable term.*

Proof. The theorem follows from the Theorem 2.3, the Subject Reduction Theorem for $\mathbf{ILAL_N}$, the subterm typability and the Lemma 3.7. □

For all $x_1, \ldots, x_{k_1}, y_1, \ldots, y_{k_2}, z_1, \ldots, z_{k_3} \in \mathcal{V}$, $\Gamma_{\{x_1,\ldots,x_{k_1}\},\{y_1,\ldots,y_{k_2}\},\{z_1,\ldots,z_{k_3}\}}$ denotes the following context :

$$x_1 : \mathbf{0}, \ldots, x_{k_1} : \mathbf{0}, y_1 : [\mathbf{0}]_!, \ldots, y_{k_2} : [\mathbf{0}]_!, z_1 : [\mathbf{0}]_\S, \ldots, z_{k_3} : [\mathbf{0}]_\S.$$

Theorem 3.9 *Every true term is typable.*

Proof. We prove, by induction on t, that for all true (respectively wise) terms t, for all $X, Y, Z \subseteq \mathcal{V}$, if $t \in \mathcal{T}_{X,Y,Z}$, then there exists a formula T such that (respectively for all formulae T, we have) $\Gamma_{X,Y,Z} \vdash t : T$:

- $t \in \mathcal{WT}$: let T be a formula :
  - Case 1 : t is a variable : $t \in X$, therefore we have $\dfrac{\dfrac{}{\Gamma_{X,Y,Z} \vdash t : \mathbf{0}}\,\mathrm{Ax}}{\Gamma_{X,Y,Z} \vdash t : T}\,\forall\mathrm{E}$;
  - Case 2 : $t = (w)r$, $w \in \mathcal{WT}$ and $r \in \mathcal{TT}$: $w, r \in \mathcal{T}_{X,Y,Z}$, therefore, by the hypothesis induction, $\Gamma_{X,Y,Z} \vdash r : R$ and $\Gamma_{X,Y,Z} \vdash w : (R \multimap T)$; so $\Gamma_{X,Y,Z} \vdash t : T$;
- $t = !u$ and $u \in \mathcal{TT}$: $u \in \mathcal{T}_{Y,\emptyset,\emptyset}$; therefore, by the induction hypothesis, $\Gamma_{Y,\emptyset,\emptyset} \vdash u : U$; so $\Gamma_{X,Y,Z} \vdash t : !U$;
- $t = \mathsf{let}\ u\ \mathsf{be}\ !x\ \mathsf{in}\ v$, $u \in \mathcal{WT}$ and $v \in \mathcal{TT}$: $u \in \mathcal{T}_{X,Y,Z}$ and $v \in \mathcal{T}_{X,Y,Z}$; therefore, by the induction hypothesis, $\Gamma_{X,Y,Z} \vdash u : !\mathbf{0}$ and $\Gamma_{X,Y\cup\{x\},Z} \vdash v : V$; so $\Gamma_{X,Y,Z} \vdash t : V$;
- $t = \S u$ and $u \in \mathcal{TT}$: $u \in \mathcal{T}_{Y\cup Z,\emptyset,\emptyset}$; therefore, by the induction hypothesis, $\Gamma_{Y\cup Z,\emptyset,\emptyset} \vdash u : U$; so $\Gamma_{X,Y,Z} \vdash t : \S U$;
- $t = \mathsf{let}\ u\ \mathsf{be}\ \S x\ \mathsf{in}\ v$, $u \in \mathcal{WT}$ and $v \in \mathcal{TT}$: $u \in \mathcal{T}_{X,Y,Z}$ and $v \in \mathcal{T}_{X,Y,Z\cup\{x\}}$; therefore, by the induction hypothesis, $\Gamma_{X,Y,Z} \vdash u : \S\mathbf{0}$ and $\Gamma_{X,Y,Z\cup\{x\}} \vdash v : V$; so $\Gamma_{X,Y,Z} \vdash t : V$.

□

Remark 3.10 *Nevertheless, there exist untypable reasonable terms. Example :*

$$G = \mathit{let}\ !\lambda x.\mathit{let}\ x\ \mathit{be}\ !x'\ \mathit{in}\ \S(x')x'\ \mathit{be}\ !x'\ \mathit{in}\ \S\lambda y.((y)(x')!\lambda x.x)(x')!\lambda x.\lambda y.x.$$

Although its erasure is typable in System F, G is a reasonable term, that isn't typable in $\mathbf{ILAL_N}$. *Indeed, we can define a new translation into the lambda-calculus : the crossing out ; $cross(t)$ denotes the crossing out of a term t, that is defined by induction on t :*

- *if $t = \dagger u$, then $cross(t) = cross(u)$;*
- *if $t = \mathit{let}\ u\ \mathit{be}\ \dagger\ x\ \mathit{in}\ v$, then $cross(t) = (\lambda x.cross(v))cross(u)$;*
- *cross commutes to other constructions ;*

and we can show that if a term is typable in $\mathbf{ILAL_N}$, *then its crossing out is typable in System F. And the crossing out of G, i.e.*

$$(\lambda x'.\lambda y.((y)(x')\lambda x.x)(x')\lambda x.\lambda y.x)\lambda x.(\lambda x'.(x')x')x,$$

isn't typable in System F (see [6]).

4 Light Intersection Type Assignment System

$\mathbf{ILAL_N}$ allowed us to give a sufficient condition for a term to be reasonable. Now, we want to define a system that types more reasonable terms. For that, we use the approach of intersection types.

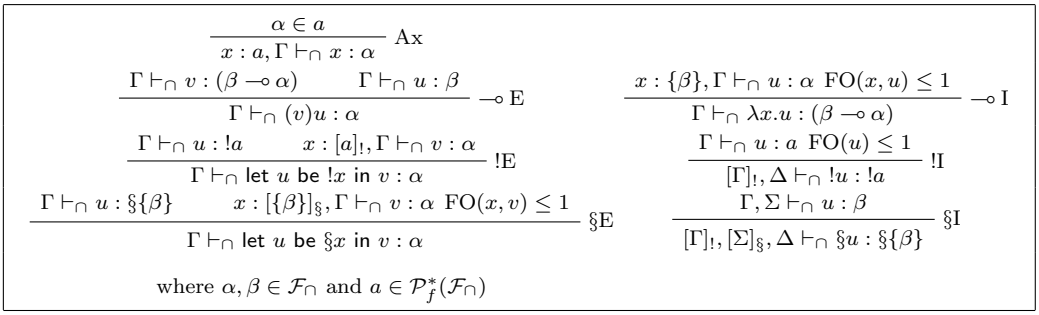

Fig. 4. Light Intersection Type Assignment System

Definition 4.1 The set $\mathcal{F}_\cap$ of types is defined by the following grammar : $\mathcal{F}_\cap ::= \mathcal{P} \mid (\mathcal{F}_\cap \multimap \mathcal{F}_\cap) \mid !\mathcal{P}^*_f(\mathcal{F}_\cap) \mid \S\{\mathcal{F}_\cap\}$.
The set $!\text{-}\mathcal{DF}_\cap$ is $\{[\alpha]_! \ ; \ \alpha \in \mathcal{F}_\cap\}$. The set $\S\text{-}\mathcal{DF}_\cap$ is $\{[\alpha]_\S \ ; \ \alpha \in \mathcal{F}_\cap\}$.
A context is a map from a finite subset of $\mathcal{V}$ to $\mathcal{P}^*_f(\mathcal{F}_\cap) \cup \mathcal{P}^*_f(!\text{-}\mathcal{DF}_\cap) \cup \mathcal{P}^*_f(\S\text{-}\mathcal{DF}_\cap)$.
For all $X, Y, Z \subseteq \mathcal{V}$, $\mathcal{C}_{X,Y,Z}$ is the set of the contexts Γ such that $\forall x \in X \cap \text{dom}(\Gamma), \Gamma(x) \in \mathcal{P}^*_f(\mathcal{F}_\cap)$, $\forall y \in Y \cap \text{dom}(\Gamma), \Gamma(y) \in \mathcal{P}^*_f(!\text{-}\mathcal{DF}_\cap)$ and $\forall z \in Z \cap \text{dom}(\Gamma), \Gamma(z) \in \mathcal{P}^*_f(\S\text{-}\mathcal{DF}_\cap)$.

From now on, we use Greek letters to denote types and Latin ones to denote their finite sets.

An element of $\mathcal{P}^*_f(\mathcal{F}_\cap)$ can be thought of as the intersection of its elements.

For all $a \in \mathcal{P}^*_f(\mathcal{F}_\cap)$, $[a]_\dagger$ denotes $\{[\alpha]_\dagger \ ; \ \alpha \in a\}$ and $\Gamma \vdash_\cap t : a$ denotes $\forall \alpha \in a \ \Gamma \vdash_\cap t : \alpha$.

Definition 4.2 The type assignment rules are those given in Figure 4.

Note that the rule !I has Card(a) premises.

Remark 4.3 *This system satisfies the subterm typability.*

Definition 4.4 We define a binary relation $\leq$ on the set of the contexts as follows : $\Gamma \leq \Gamma'$ if, and only if, $\text{dom}(\Gamma) \subseteq \text{dom}(\Gamma')$ and $\forall x \in \text{dom}(\Gamma) \ \Gamma(x) \subseteq \Gamma'(x)$.

Fact 4.5 *For all $t \in \mathcal{T}_{X,Y,Z}$, for all types α, for all contexts Γ and Γ' such that $\Gamma \leq \Gamma'$, if $\Gamma \vdash_\cap t : \alpha$, then $\Gamma' \vdash_\cap t : \alpha$.*

For all contexts Γ, for all terms t, Γ_t denotes $\{(x, \mathfrak{a}) \in \Gamma \ ; x \text{ is free in } t\}$.

Fact 4.6 *For all contexts Γ, for all terms t, for all types α, $\Gamma \vdash_\cap t : \alpha$ if, and only if, $\Gamma_t \vdash_\cap t : \alpha$.*

4.1 Subject Reduction

Lemma 4.7 *Let Γ be a context and let $x_1, \ldots, x_{k_1}, y_1, \ldots, y_{k_2}, z_1, \ldots, z_{k_3}$ be variables such that $x_1, \ldots, x_{k_1}, \ y_1, \ldots, y_{k_2}, \ z_1, \ldots, z_{k_3} \notin dom(\Gamma)$. If $\Gamma, x_1 : \{\alpha_1\}, \ldots, x_{k_1} : \{\alpha_{k_1}\}, y_1 : [b_1]_!, \ldots, y_{k_2} : [b_{k_2}]_!, z_1 : [\{\gamma_1\}]_\S, \ldots, z_{k_3} : [\{\gamma_{k_3}\}]_\S \vdash_\cap$*

$t : \alpha$, for all $i \in \{1, \ldots, k_1\}, \Gamma \vdash_\cap u_i : \alpha_i$, for all $i \in \{1, \ldots, k_2\}, \Gamma \vdash_\cap !v_i : !b_i$, and for all $i \in \{1, \ldots, k_3\}, \Gamma \vdash_\cap \S w_i : \S\{\gamma_i\}$, then

$$\Gamma \vdash_\cap t[u_1/x_1, \ldots, u_{k_1}/x_{k_1}, v_1/y_1, \ldots, v_{k_2}/y_{k_2}, w_1/z_1, \ldots, w_{k_3}/z_{k_3}] : \alpha.$$

Proof. By induction on t. □

Lemma 4.8 *For all term t, if $\Gamma \vdash_\cap t : \alpha$ and $t \longrightarrow_0 t'$, then $\Gamma \vdash_\cap t' : \alpha$.*

Proof. First, note that, by the Proposition 1.4, t' is a term. Now, we prove the lemma by induction on t : the critical cases are the following :

- $t = (\lambda x.v_1)u$ et $t' = v_1[u/x]$;
- $t = \mathsf{let}\ !u_1\ \mathsf{be}\ !x\ \mathsf{in}\ v$ and $t' = v[u_1/x]$;
- $t = \mathsf{let}\ \S u_1\ \mathsf{be}\ \S x\ \mathsf{in}\ v$ and $t' = v[u_1/x]$.

In all this cases, assume that $x \notin \mathrm{dom}(\Gamma)$ and apply the Lemma 4.7. □

Proposition 4.9 *For all terms t, if $\Gamma \vdash_\cap t : \alpha$ and $t \longrightarrow t'$, then $\Gamma \vdash_\cap t' : \alpha$.*

Proof. Follows from the Lemma 4.8. □

4.2 Typable pseudo-terms are reasonable terms

Proposition 4.10 *For all pseudo-terms t, if*

$$x_1 : a_1, \ldots, x_{k_1} : a_{k_1}, y_1 : [b_1]_!, \ldots, y_{k_2} : [b_{k_2}]_!, z_1 : [\{\gamma_1\}]_\S, \ldots, z_{k_3} : [\{\gamma_{k_3}\}]_\S \vdash_\cap t : \alpha$$

is derivable, then $t \in \mathcal{T}_{\{x_1,\ldots,x_{k_1}\},\{y_1,\ldots,y_{k_2}\},\{z_1,\ldots,z_{k_3}\}}$.

Proof. By induction on the pseudo-term. □

Lemma 4.11 *Every typable normal term is true.*

Proof. By induction on the term. □

Proposition 4.12 *Every typable pseudo-term is a reasonable term.*

Proof. Let t be a typable pseudo-term. First, by the Proposition 4.10, t is a term. Now, let t' such that $t \longrightarrow t'$. By the Proposition 4.9, t' is typable ; so, every normal subterm of t' is typable and, by the Lemma 4.11, is true. □

4.3 Reasonable terms are typable

For all contexts Γ_1 and Γ_2, $\Gamma_1 + \Gamma_2$ denotes $\{(x, \Gamma_1(x) \cup \Gamma_2(x)) ; x \in \mathrm{dom}(\Gamma_1) \cap \mathrm{dom}(\Gamma_2)\} \cup \{(x, \Gamma_1(x)) ; x \in \mathrm{dom}(\Gamma_1) \setminus \mathrm{dom}(\Gamma_2)\} \cup \{(x, \Gamma_2(x)) ; x \in \mathrm{dom}(\Gamma_2) \setminus \mathrm{dom}(\Gamma_1)\}$.

Lemma 4.13 *For all true (respectively wise) terms t, for all $X, Y, Z \subseteq \mathcal{V}$, if $t \in \mathcal{T}_{X,Y,Z}$ and for all $z \in Z$, $FO(z,t) \leq 1$, then there exists $\alpha \in \mathcal{F}_\cap$ (respectively for all $\alpha \in \mathcal{F}_\cap$), there exists $\Gamma \in \mathcal{C}_{X,Y,Z}$ such that :*

(i) $\Gamma \vdash_\cap t : \alpha$;

(ii) and for all $w \in dom(\Gamma)$, if $FO(w,t) \leq 1$, then $Card(\Gamma(w)) = 1$.

Proof. By the Lemma 1.3 and the Fact 4.6, we can assume that $FV(t) = X \cup Y \cup Z$. Now, we prove the lemma by induction on t :

- t is a wise term : let $\alpha \in \mathcal{F}_\cap$:
  - Case 1 : t is a variable : we have $\dfrac{}{t : \{\alpha\} \vdash_\cap t : \alpha}\,\mathrm{Ax}$;
  - Case 2 : $t = (v)r$, v is a wise term and r is a true term : by the induction hypothesis, there exists $\beta \in \mathcal{F}_\cap$ and $\Gamma^r \in \mathcal{C}_{X,Y,Z}$ such that :

  (i) $\Gamma^r \vdash_\cap r : \beta$;

  (ii) and for all $w \in \mathrm{dom}(\Gamma^r)$, if $\mathrm{FO}(w, r) \leq 1$, then $\mathrm{Card}(\Gamma^r(w)) = 1$;

  again by the induction hypothesis, there exists $\Gamma^v \in \mathcal{C}_{X,Y,Z}$ such that :

  (i) $\Gamma^v \vdash_\cap v : (\beta \multimap \alpha)$;

  (ii) and for all $w \in \mathrm{dom}(\Gamma^v)$, if $\mathrm{FO}(w, v) \leq 1$, then $\mathrm{Card}(\Gamma^v(w)) = 1$;

  by the Facts 4.6 and 4.5 and by $\multimap$ E, we have $\Gamma^v_v + \Gamma^r_r \vdash t : \alpha$; hence we can let $\Gamma^t = \Gamma^v_v + \Gamma^r_r$;
- $t = \lambda x.u$, $t = !u$, or $t = \S u$: it is straightforward ;
- $t = \mathsf{let}\ u\ \mathsf{be}\ !x\ \mathsf{in}\ v$, $u \in \mathcal{WT}$ and $v \in \mathcal{TT}$: by the induction hypothesis, there exists $\alpha \in \mathcal{F}_\cap$ and $\Gamma^v, x : [b]_! \in \mathcal{C}_{X,Y\cup\{x\},Z}$ such that :

  (i) $\Gamma^v, x : [b]_! \vdash_\cap v : \alpha$;

  (ii) and for all $w \in \mathrm{dom}(\Gamma^v) \cup \{x\}$, if $\mathrm{FO}(w, v) \leq 1$, then $\mathrm{Card}(\Gamma^v(w)) = 1$;

  again by the induction hypothesis, there exists $\Gamma^u \in \mathcal{C}_{X,Y,Z}$ such that :

  (i) $\Gamma^u \vdash_\cap u : !b$;

  (ii) and for all $w \in \mathrm{dom}(\Gamma^u)$, if $\mathrm{FO}(w, u) \leq 1$, then $\mathrm{Card}(\Gamma^u(w)) = 1$;

  by the Facts 4.6 and 4.5 and by !E, we have $\Gamma^u_u + \Gamma^v_v \vdash t : \alpha$; hence we can let $\Gamma^t = \Gamma^u_u + \Gamma^v_v$;
- $t = \mathsf{let}\ u\ \mathsf{be}\ \S x\ \mathsf{in}\ v$, $u \in \mathcal{WT}$ and $v \in \mathcal{TT}$: by the induction hypothesis, there exists $\alpha \in \mathcal{F}_\cap$ and $\Gamma^v, x : [\{\beta\}]_\S \in \mathcal{C}_{X,Y,Z\cup\{x\}}$ such that :

  (i) $\Gamma^v, x : [\{\beta\}]_\S \vdash_\cap v : \alpha$;

  (ii) and for all $w \in \mathrm{dom}(\Gamma^v)$, if $\mathrm{FO}(w, v) \leq 1$, then $\mathrm{Card}(\Gamma^v(w)) = 1$;

  again by the induction hypothesis, there exists $\Gamma^u \in \mathcal{C}_{X,Y,Z}$ such that :

  (i) $\Gamma^u \vdash_\cap u : \S\{\beta\}$;

  (ii) and for all $w \in \mathrm{dom}(\Gamma^u)$, if $\mathrm{FO}(w, u) \leq 1$, then $\mathrm{Card}(\Gamma^u(w)) = 1$;

  by the Facts 4.6 and 4.5 and by §E, we have $\Gamma^u_u + \Gamma^v_v \vdash t : \alpha$; hence we can let $\Gamma^t = \Gamma^u_u + \Gamma^v_v$.

□

Lemma 4.14 *For all $X, Y, Z \subseteq \mathcal{V}$, for all terms u and v such that $v \in \mathcal{T}_{X,Y,Z}$, for all contexts Γ such that $x \notin dom(\Gamma)$, if $\Gamma \vdash_\cap v[u/x] : \alpha$, then :*

(i) if $x \in X$ and u is typable in the context Γ, then there exists $a \in \mathcal{P}^*_f(\mathcal{F}_\cap)$ such that $\Gamma, x : a \vdash_\cap v : \alpha$ and $\Gamma \vdash_\cap u : a$;

(ii) if $x \in Y$ and $!u$ is typable in the context Γ, then there exists $a \in \mathcal{P}^*_f(\mathcal{F}_\cap)$ such that $\Gamma, x : [a]_! \vdash_\cap v : \alpha$ and $\Gamma \vdash_\cap !u : !a$;

(iii) if $x \in Z$, $FO(x, v) \leq 1$ and $\S u$ is typable in the context Γ, then there exists $\gamma \in \mathcal{F}_\cap$ such that $\Gamma, x : [\{\gamma\}]_\S \vdash_\cap v : \alpha$ and $\Gamma \vdash_\cap \S u : \S\{\gamma\}$.

Proof. By induction on v :

- if v is a variable, then we have the following cases :
  - $v = x : x \in X$ and we can let $\gamma = \alpha$;
  - $v \neq x$: in all the cases, just apply the Fact 4.6 ;
- if $v = (v_2)v_1$, then we have $\Gamma \vdash_\cap v_2[u/x] : (\beta \multimap \alpha)$ and $\Gamma \vdash_\cap v_1[u/x] : \beta$:

(i) by the hypothesis induction, we have $\Gamma, x : a_2 \vdash_\cap v_2 : (\beta \multimap \alpha)$ and $\Gamma \vdash_\cap u : a_2$; again by the hypothesis induction, we have $\Gamma, x : a_1 \vdash_\cap v_1 : \beta$ and $\Gamma \vdash_\cap u : a_1$; by the Fact 4.5, we have $\Gamma, x : a_1 \cup a_2 \vdash_\cap v_2 : (\beta \multimap \alpha)$ and $\Gamma, x : a_1 \cup a_2 \vdash_\cap v_1 : \beta$; by $\multimap$ E, we have $\Gamma, x : a_1 \cup a_2 \vdash_\cap v : \alpha$;

(ii) by the hypothesis induction, we have $\Gamma, x : [a_2]_! \vdash_\cap v_2 : (\beta \multimap \alpha)$ and $\Gamma \vdash_\cap !u : !a_2$; again by the hypothesis induction, we have $\Gamma, x : [a_1]_! \vdash_\cap v_1 : \beta$ and $\Gamma \vdash_\cap !u : !a_1$; by the Fact 4.5, we have $\Gamma, x : [a_1 \cup a_2]_! \vdash_\cap v_2 : (\beta \multimap \alpha)$ and $\Gamma, x : [a_1 \cup a_2]_! \vdash_\cap v_1 : \beta$; by $\multimap$ E, we have $\Gamma, x : [a_1 \cup a_2]_! \vdash_\cap v : \alpha$;

(iii) we have the following cases :
  - $v_2[u/x] = v_2$: by the hypothesis induction, we have $\Gamma, x : [\{\gamma\}]_\S \vdash_\cap v_1 : \beta$ and $\Gamma \vdash_\cap \S u : \S\{\gamma\}$; by the Fact 4.5, we have $\Gamma, x : [\{\gamma\}]_\S \vdash_\cap v_2 : (\beta \multimap \alpha)$; by $\multimap$ E, we have $\Gamma, x : [\{\gamma\}]_\S \vdash_\cap v : \alpha$;
  - $v_2[u/x] \neq v_2$: $FO(x, v) \leq 1$, therefore we have $v_1[u/x] = v_1$; by the hypothesis induction, we have $\Gamma, x : [\{\gamma\}]_\S \vdash v_2 : (\beta \multimap \alpha)$ and $\Gamma \vdash_\cap \S u : \S\{\gamma\}$; by the Fact 4.5, we have $\Gamma, x : [\{\gamma\}]_\S \vdash_\cap v_1 : \beta$; by $\multimap$ E, we have $\Gamma, x : [\{\gamma\}]_\S \vdash_\cap \alpha$;

- if $v = !v_1$, then :
  - in the cases *(i)* and *(iii)*, just apply the Fact 4.5 ;
  - in the case *(ii)*, apply the case *(i)* of the hypothesis induction ;
- the other cases are similar.

□

Proposition 4.15 *For all X, Y, $Z \subseteq \mathcal{V}$, for all reasonable terms $t \in \mathcal{T}_{X,Y,Z}$, for all $\Gamma \in \mathcal{C}_{X,Y,Z}$, for all terms t' such that $t \longrightarrow t'$, if for all $z \in Z$, $FO(z, t) \leq 1$, $\Gamma \vdash_\cap t' : \alpha$ and for all $w \in dom(\Gamma)$ such that $FO(w, t') \leq 1$, $Card(\Gamma(w)) = 1$, then there exists $\Gamma' \in \mathcal{C}_{X,Y,Z}$ such that :*

(i) $\Gamma \leq \Gamma'$;

(ii) $\Gamma' \vdash_\cap t : \alpha$;

(iii) and for all $w \in dom(\Gamma')$ such that $FO(w, t) \leq 1$, $Card(\Gamma'(w)) = 1$.

Proof. By well-founded induction on $(N(t), \text{size}(t))$.
If t is normal, then there is nothing to do. Else, if $t' \neq t$, then there exists a term t_1 such that $t \longrightarrow_0 t_1$ and $t_1 \longrightarrow t'$. t_1 is a reasonable term, so, by the hypothesis induction, there exists $\Gamma_1 \in \mathcal{C}_{X,Y,Z}$ such that $\Gamma \leq \Gamma_1$, $\Gamma_1 \vdash_\cap t_1 : \alpha$ and for all $w \in \text{dom}(\Gamma_1)$ such that $\text{FO}(w, t) \leq 1$, $\text{Card}(\Gamma_1(w)) = 1$; we have the following cases :

a) $t = (v)u$, $t_1 = (v_1)u$ and $v \longrightarrow_0 v_1$: apply the hypothesis induction ;

b) $t = (v)u$, $t_1 = (v)u_1$ and $u \longrightarrow_0 u_1$: apply the hypothesis induction ;

c) $t = (\lambda x.v)u$ and $t_1 = v[u/x]$: apply the Lemma 4.13, the hypothesis induction and the Lemma 4.14 (i) ;

d) $t = (\mathsf{let}\ u_1\ \mathsf{be}\ \dagger\, x_1\ \mathsf{in}\ v_1)u$, $t_1 = \mathsf{let}\ u_1\ \mathsf{be}\ \dagger\, x_1\ \mathsf{in}\ (v_1)u$ and $x_1 \notin FV(u)$: we have $\Gamma_1 \vdash_\cap t : \alpha$;

e) $t = \lambda x.u$, $t_1 = \lambda x.u_1$ and $u \longrightarrow_0 u_1$: apply the hypothesis induction ;

f) $t = \dagger u$, $t_1 = \dagger u_1$ and $u \longrightarrow_0 u_1$: apply the hypothesis induction ;

g) $t = \mathsf{let}\ u\ \mathsf{be}\ \dagger\, x\ \mathsf{in}\ v$, $t_1 = \mathsf{let}\ u_1\ \mathsf{be}\ \dagger\, x\ \mathsf{in}\ v$ and $u \longrightarrow_0 u_1$: apply the hypothesis induction ;

h) $t = \mathsf{let}\ !u\ \mathsf{be}\ !x\ \mathsf{in}\ v$, $t_1 = v[u/x]$: apply the hypothesis induction and the Lemma 4.14 (ii) ;

i) $t = \mathsf{let}\ \S u\ \mathsf{be}\ \S x\ \mathsf{in}\ v$, $t_1 = v[u/x]$: apply the hypothesis induction and the Lemma 4.14 (iii) ;

j) $t = \mathsf{let}\ !u\ \mathsf{be}\ !x\ \mathsf{in}\ v$, $t_1 = \mathsf{let}\ !u\ \mathsf{be}\ !x\ \mathsf{in}\ v_1$, $v \longrightarrow_0 v_1$ and $!u$ is normal : it reduces to the case h) : apply Church-Rosser, the Proposition 4.9 and the hypothesis induction ;

k) $t = \mathsf{let}\ \S u\ \mathsf{be}\ \S x\ \mathsf{in}\ v$, $t_1 = \mathsf{let}\ \S u\ \mathsf{be}\ \S x\ \mathsf{in}\ v_1$, $v \longrightarrow_0 v_1$ and $\S u$ is normal : it reduces to the case i) : apply Church-Rosser, the Proposition 4.9 and the hypothesis induction ;

l) $t = \mathsf{let}\ u\ \mathsf{be}\ \dagger\, x\ \mathsf{in}\ v$, $t_1 = \mathsf{let}\ u\ \mathsf{be}\ \dagger\, x\ \mathsf{in}\ v_1$, $v \longrightarrow_0 v_1$ and $u \in \mathcal{WT}$: apply the hypothesis induction and the Lemma 4.13 ;

m) $t = \mathsf{let}\ u\ \mathsf{be}\ \dagger x\ \mathsf{in}\ v$, $t_1 = \mathsf{let}\ u\ \mathsf{be}\ \dagger x\ \mathsf{in}\ v_1$, $v \longrightarrow_0 v_1$ and u isn't normal : it reduces to the case g) : apply Church-Rosser, the Proposition 4.9 and the hypothesis induction ;

n) $t = \mathsf{let}\ \mathsf{let}\ u_1\ \mathsf{be}\ \dagger_1 x_1\ \mathsf{in}\ v_1\ \mathsf{be}\ \dagger\, x\ \mathsf{in}\ v$, $t_1 = \mathsf{let}\ u_1\ \mathsf{be}\ \dagger_1 x_1\ \mathsf{in}\ \mathsf{let}\ v_1\ \mathsf{be}\ \dagger\, x\ \mathsf{in}\ v$ and $x_1 \notin FV(v)$: we have $\Gamma_1 \vdash_\cap t : \alpha$.

□

Corollary 4.16 *Every reasonable term is typable.*

Proof. Let $t \in \mathcal{T}_{X,Y,Z}$ a reasonable term. First, note that $t \in \mathcal{T}_{X,Y\cup Z,\emptyset}$. Now, there exists a true term $t' \in \mathcal{T}_{X,Y\cup Z,\emptyset}$ such that $t \longrightarrow t'$. Apply the Propositions 4.13 and 4.15. □

Theorem 4.17 *A pseudo-term is typable if, and only if, it is a reasonable term.*

Proof. Follows from the Proposition 4.12 and the Corollary 4.16. □

4.4 The erasure of any reasonable term is strongly normalizable

In order to prove that the erasure of any reasonable term is strongly normalizable, we recall what is the System $\mathcal{D}$ (see [9] for a full exposition) :

Definition 4.18 The set $\mathcal{F}_\mathcal{D}$ of the types of the System $\mathcal{D}$ is defined by the

$$\frac{}{x : A, \Gamma \vdash_{\mathcal{D}} x : A}\ \mathrm{Ax}$$

$$\frac{\Gamma \vdash_{\mathcal{D}} v : (B \multimap A) \qquad \Gamma \vdash_{\mathcal{D}} u : B}{\Gamma \vdash_{\mathcal{D}} (v)u : A} \to \mathrm{E} \qquad \frac{x : B, \Gamma \vdash_{\mathcal{D}} u : A}{\Gamma \vdash_{\mathcal{D}} \lambda x.u : (B \to A)} \to \mathrm{I}$$

$$\frac{\Gamma \vdash_{\mathcal{D}} t : (A \wedge B)}{\Gamma \vdash_{\mathcal{D}} t : A} \wedge_1 \mathrm{E} \qquad \frac{\Gamma \vdash_{\mathcal{D}} t : (A \wedge B)}{\Gamma \vdash_{\mathcal{D}} t : B} \wedge_2 \mathrm{E} \qquad \frac{\Gamma \vdash_{\mathcal{D}} t : A \qquad \Gamma \vdash_{\mathcal{D}} t : B}{\Gamma \vdash_{\mathcal{D}} t : (A \wedge B)} \wedge I$$

Fig. 5. System $\mathcal{D}$

following grammar :
$\mathcal{F}_{\mathcal{D}} ::= \mathcal{P} \mid (\mathcal{F}_{\mathcal{D}} \to \mathcal{F}_{\mathcal{D}}) \mid (\mathcal{F}_{\mathcal{D}} \wedge \mathcal{F}_{\mathcal{D}})$.

Definition 4.19 The type assignment rules are those given in Figure 5.

The following fact, theorem and proposition hold :

Fact 4.20 *If* $\Gamma \vdash_{\mathcal{D}} t : \alpha$, *then* $\Gamma, \Delta \vdash_{\mathcal{D}} t : \alpha$.

Theorem 4.21 *Every typable lambda-term in the System* $\mathcal{D}$ *is strongly normalizable.*

Proposition 4.22 *Let* Γ *be a context and* $x_1, \ldots, x_k$ *variables non declared in* Γ. *If* $\Gamma, x_1 : A_1, \ldots, x_k : A_k \vdash_{\mathcal{D}} u : B$ *and for all* i *such that* $1 \leq i \leq k$ *and* x_i *is free in* u, $\Gamma \vdash_{\mathcal{D}} t_i : A_i$, *then* $\Gamma \vdash_{\mathcal{D}} u[t_1/x_1, \ldots, t_k/x_k] : B$.

Now, we need some definitions :

Definition 4.23 For all $n \geq 1$, for all $\mathfrak{s} \in \mathcal{F}_{\mathcal{D}}^n$, $\bigwedge \mathfrak{s}$ is defined by induction on n :

- if $\mathfrak{s} = (\alpha)$, then $\bigwedge \mathfrak{s} = \alpha$;
- $\bigwedge(\alpha_1, \ldots, \alpha_{n+1}) = (\bigwedge(\alpha_1, \ldots, \alpha_n) \wedge \alpha_{n+1})$.

Fact 4.24 *If* $\Gamma \vdash_{\mathcal{D}} \bigwedge(\alpha_1, \ldots, \alpha_n)$, *then* $\Gamma \vdash_{\mathcal{D}} \alpha_1$, $\ldots$, $\Gamma \vdash_{\mathcal{D}} \alpha_n$.

Let $\Box_{\mathcal{D}}$ be any function from $\mathcal{P}_f^*(\mathcal{F}_{\mathcal{D}})$ to $\mathcal{F}_{\mathcal{D}}^{(\mathbb{N})}$ such that

$$\Box_{\mathcal{D}}(\{A_1, \ldots, A_n\}) = (A_1, \ldots, A_n).$$

The erasure$_{\mathcal{F}_\cap}$ of a type α of **LI** is defined, by induction on α, to be a type of the System $\mathcal{D}$:

- if $\alpha \in \mathcal{P}$, then erasure$_{\mathcal{F}_\cap}(\alpha) = \alpha$;
- if $\alpha = (\beta \multimap \gamma)$, then erasure$_{\mathcal{F}_\cap}(\alpha) = (\text{erasure}_{\mathcal{F}_\cap}(\beta) \to \text{erasure}_{\mathcal{F}_\cap}(\gamma))$;
- if $\alpha = !a$, then erasure$_{\mathcal{F}_\cap}(\alpha) = \bigwedge \Box_{\mathcal{D}}\{\text{erasure}_{\mathcal{F}_\cap}(\beta)\ ;\ \beta \in a\}$;
- if $\alpha = \S\{\gamma\}$, then erasure$_{\mathcal{F}_\cap}(\alpha) = \text{erasure}_{\mathcal{F}_\cap}(\gamma)$.

For any $[\alpha]_\dagger \in \dagger - \mathcal{DF}_\cap$, erasure$_{\mathcal{DF}_\cap}([\alpha]_\dagger) = \text{erasure}_{\mathcal{F}_\cap}(\alpha)$.
For any context Γ, erasure$_{\mathrm{C}}(\Gamma)$ is a map from a finite subset of $\mathcal{V}$ to $\mathcal{F}_{\mathcal{D}}$ defined as follows : the domain is the same as those of Γ and for all $x \in \text{dom}(\Gamma)$, we have the following cases :

- if $\Gamma(x) \in \mathcal{P}_f^*(\mathcal{F}_\cap)$, then erasure$_{\mathrm{C}}(\Gamma)(x) = \bigwedge \Box_{\mathcal{D}}\{\text{erasure}_{\mathcal{F}_\cap}(\alpha)\ ;\ \alpha \in \Gamma(x)\}$;
- if $\Gamma(x) \in \mathcal{P}_f^*(\dagger - \mathcal{DF}_\cap)$, then erasure$_{\mathrm{C}}(\Gamma)(x) = \bigwedge \Box_{\mathcal{D}}\{\text{erasure}_{\mathcal{DF}_\cap}(\alpha)\ ;\ \alpha \in$

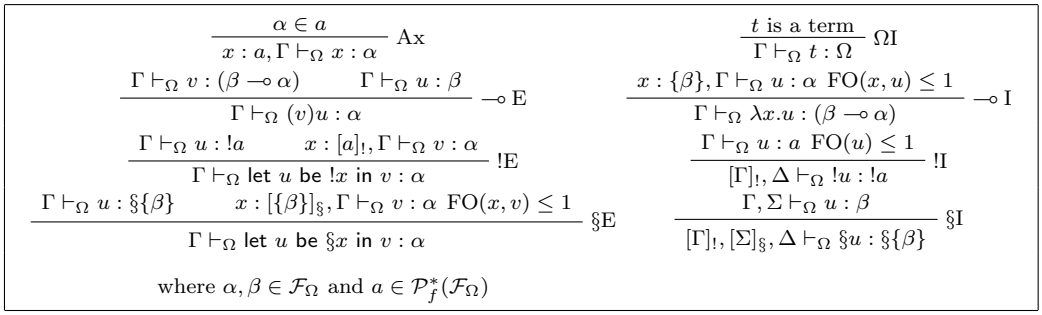

Fig. 6. Relaxed Light Intersection Type Assignment System

$\Gamma(x)\}$.

Proposition 4.25 *If* $\Gamma \vdash_{\mathbf{LI}} t : \alpha$ *is derivable, then* $erasure_{\mathcal{C}}(\Gamma) \vdash_{\mathcal{D}} erasure(t) : erasure_{\mathcal{F}_\cap}(\alpha)$ *is derivable.*

Proof. By induction on t :

- if t is a variable, then apply the Fact 4.24 ;
- if $t = (v)u$ or $t = \lambda x.u$, then it is straightforward ;
- if $t = \mathsf{let}\ u\ \mathsf{be}\ !x\ \mathsf{in}\ v$, then apply the Fact 4.24 and the Proposition 4.22 ;
- if $t = \mathsf{let}\ u\ \mathsf{be}\ \S x\ \mathsf{in}\ v$, then apply the Proposition 4.22 ;
- if $t = \dagger u$, then apply the Fact 4.20.

□

Theorem 4.26 *The erasure of any reasonable term is strongly normalizable.*

Proof. Follows from the Corollary 4.16 and 4.25 and the Theorem 4.21. □

5 Relaxed Light Intersection Type Assignment System

LI allowed us to give a sufficient and necessary condition for a term to be reasonable. Now, we slightly modify this system to obtain a new system that types exactly the safe terms.

Definition 5.1 The set $\mathcal{F}_\Omega$ of types is defined by the following grammar : $\mathcal{F}_\Omega ::= \mathcal{P} \mid (\mathcal{F}_\Omega \multimap \mathcal{F}_\Omega) \mid !\mathcal{P}^*_f(\mathcal{F}_\Omega) \mid \S\{\mathcal{F}_\Omega\} \mid \Omega$.
The set $!\text{-}\mathcal{DF}_\Omega$ is $\{[\alpha]_! \ ; \ \alpha \in \mathcal{F}_\Omega\}$. The set $\S\text{-}\mathcal{DF}_\Omega$ is $\{[\alpha]_\S \ ; \ \alpha \in \mathcal{F}_\Omega\}$.
A context is a map from a finite subset of $\mathcal{V}$ to $\mathcal{P}^*_f(\mathcal{F}_\Omega) \cup \mathcal{P}^*_f(!\text{-}\mathcal{DF}_\Omega) \cup \mathcal{P}^*_f(\S\text{-}\mathcal{DF}_\Omega)$.

For all $a \in \mathcal{P}^*_f(\mathcal{F}_\Omega)$, $[a]_\dagger$ denotes $\{[\alpha]_\dagger \ ; \ \alpha \in a\}$ and $\Gamma \vdash_\Omega t : a$ denotes $\forall \alpha \in a\ \Gamma \vdash_\Omega t : \alpha$.

Definition 5.2 The type assignment rules are those given in Figure 6.

The unique difference between the type assignment rules of **LI** and those of **RLI** is the rule ΩI : every term has type Ω. Note that in order to apply this rule with a pseudo-term you must check that it is a term ; it is not a

problem, because, as noted in Section 1, there is a quadratic algorithm to do it. But be careful with the definition of *typable* :

Definition 5.3 For all sequents $x_1 : a_1, \ldots, x_{k_1} : a_{k_1}, y_1 : [b_1]_{\dagger_1}, \ldots, y_{k_2} : [b_{k_2}]_{\dagger_{k_2}} \vdash t : \alpha$, types$(x_1 : a_1, \ldots, x_{k_1} : a_{k_1}, y_1 : [b_1]_{\dagger_1}, \ldots, y_{k_2} : [b_{k_2}]_{\dagger_{k_2}} \vdash t : \alpha)$ denotes $(\bigcup\{a_i \, ; \, 1 \leq i \leq k_1\}) \cup (\bigcup\{b_i \, ; \, 1 \leq i \leq k_2\}) \cup \{\alpha\}$.

Definition 5.4 A term t is said to be typable whenever there exists a sequent $\Gamma \vdash_\Omega t : \alpha$ such that there exists a derivation of $\Gamma \vdash_\Omega t : \alpha$ and types$(\Gamma \vdash_\Omega t : \alpha) \subseteq \mathcal{F}_\cap$.

5.1 Subject Reduction

Proposition 5.5 *For all terms t, if $\Gamma \vdash_\Omega t : \alpha$ and $t \longrightarrow t'$, then $\Gamma \vdash_\Omega t' : \alpha$.*

Proof. Similar to the proof of the Proposition 4.9. □

5.2 Subformula Property

Definition 5.6 For all $\alpha \in \mathcal{F}_\Omega$, we define, by induction on α, the set $\mathcal{S}(\alpha)$ of the subformulas of α :

- if $\alpha \in \mathcal{P}$, then $\mathcal{S}(\alpha) = \{\alpha\}$;
- if $\alpha = (\beta \multimap \gamma)$, then $\mathcal{S}(\alpha) = \mathcal{S}(\beta) \cup \mathcal{S}(\gamma) \cup \{\alpha\}$;
- if $\alpha = \dagger a$, then $\mathcal{S}(\alpha) = (\bigcup\{\mathcal{S}(\mu) \, ; \, \mu \in a\}) \cup \{\alpha\}$;
- if $\alpha = \Omega$, then $\mathcal{S}(\alpha) = \{\Omega\}$.

Remark 5.7 *If $\alpha \in \mathcal{F}_\cap$, then $\mathcal{S}(\alpha) \subseteq \mathcal{F}_\cap$.*

Proposition 5.8 *Let t be a normal term and let π be a derivation of $\Gamma \vdash_\Omega t : \alpha$. Then for all sequents Θ of π, for all $\mu \in$ types(Θ), there exists $\beta \in$ types$(\Gamma \vdash_\Omega t : \alpha)$ such that $\mu \in \mathcal{S}(\beta)$; moreover if t is written $(v)u$, then $\alpha \in \mathcal{S}(\Gamma)$.*

Proof. By induction on t. □

Corollary 5.9 *Every typable normal term in* **RLI** *is typable in* **LI**.

5.3 Typable pseudo-terms are safe terms

Proposition 5.10 *If $x_1 : a_1, \ldots, x_{k_1} : a_{k_1}$, $y_1 : [b_1]_!, \ldots, y_{k_2} : [b_{k_2}]_!$, $z_1 : [c_1]_\S, \ldots, z_{k_3} : [c_{k_3}]_\S \vdash t : \alpha$ is derivable, then $t \in \mathcal{T}_{\{x_1,\ldots,x_{k_1}\},\{y_1,\ldots,y_{k_2}\},\{z_1,\ldots,z_{k_3}\}}$.*

Proof. Similar to the proof of the Proposition 4.10. □

Proposition 5.11 *Every typable pseudo-term is a safe term.*

Proof. Let t be a typable pseudo-term. By the Proposition 5.10, t is a term. By the Theorem 1.5, t reduces to a normal term t'. By the Proposition 5.5, t' is a typable term. Now, by the Corollary 5.9, t' is typable in **LI**. So, by the Lemma 4.11, t' is a true term. □

5.4 Safe terms are typable terms

Lemma 5.12 *For all $X, Y, Z \subseteq \mathcal{V}$, for all terms u and v such that $v \in \mathcal{T}_{X,Y,Z}$, for all contexts Γ such that $x \notin dom(\Gamma)$, if $\Gamma \vdash_\Omega v[u/x] : \alpha$, then :*

(i) if $x \in X$, then there exists $a \in \mathcal{P}^*_f(\mathcal{F}_\Omega)$ such that $\Gamma, x : a \vdash_\Omega v : \alpha$ and $\Gamma \vdash_\Omega u : a$;

(ii) if $x \in Y$, then there exists $a \in \mathcal{P}^*_f(\mathcal{F}_\Omega)$ such that $\Gamma, x : [a]_! \vdash_\Omega v : \alpha$ and $\Gamma \vdash_\Omega !u : !a$;

(iii) if $x \in Z$ and $FO(x, v) \leq 1$, then there exists $\gamma \in \mathcal{F}_\Omega$ such that $\Gamma, x : [\{\gamma\}]_\S \vdash_\Omega v : \alpha$ and $\Gamma \vdash_\Omega \S u : \S\{\gamma\}$.

Proof. We can assume that $\alpha \neq \Omega$. Now, we prove the lemma by induction on v :

- if v is a variable, then we have the following cases :
  · $v = x : x \in X$ and we can let $\gamma = \alpha$;
  · $v \neq x$:
  (i) since $\Gamma \vdash_\Omega v : \alpha$, by the Fact 4.6, $\Gamma, x : \{\Omega\} \vdash_\Omega v : \alpha$; and we have $\Gamma \vdash_\Omega u : \Omega$;
  (ii) since $\Gamma \vdash_\Omega v : \alpha$, by the Fact 4.6, $\Gamma, x : [\{\Omega\}]_! \vdash_\Omega v : \alpha$; and we have $\Gamma \vdash_\Omega !u : !\{\Omega\}$;
  (iii) since $\Gamma \vdash_\Omega v : \alpha$, by the Fact 4.6, $\Gamma, x : [\{\Omega\}]_\S \vdash_\Omega v : \alpha$; and we have $\Gamma \vdash_\Omega \S u : \S\{\Omega\}$;
- the other cases are similar to the proof of the Lemma 4.14.

□

Proposition 5.13 *For all terms t and t' such that $t \longrightarrow_0 t'$, if $\Gamma \vdash_\Omega t' : \alpha$, then $\Gamma \vdash_\Omega t : \alpha$.*

Proof. We can assume that $\alpha \neq \Omega$. Now, the proposition is proved by induction on t : the critical cases are the following :

- $t = (\lambda x.v_1)u$ et $t' = v_1[u/x]$: apply the Lemma 5.12 (i) ;
- $t = \mathsf{let}\ !u_1\ \mathsf{be}\ !x\ \mathsf{in}\ v$ and $t' = v[u_1/x]$: apply the Lemma 5.12 (ii) ;
- $t = \mathsf{let}\ \S u_1\ \mathsf{be}\ \S x\ \mathsf{in}\ v$ and $t' = v[u_1/x]$: apply the Lemma 5.12 (iii).

□

Theorem 5.14 *For all terms t and t' such that $t \longrightarrow t'$, if $\Gamma \vdash_\Omega t' : \alpha$, then $\Gamma \vdash_\Omega t : \alpha$.*

Proof. Follows from the Proposition 5.13. □

Theorem 5.15 *Every pseudo-term is typable if, and only if, it is a safe term.*

Proof. Let t be a safe term : it reduces to a true term, which, by the Proposition 5.5, is typable ; so, by the Theorem 5.14, t is typable. The converse is the Proposition 5.11. □

Acknowledgement

We have to thank Patrick Baillot for all his comments, remarks and suggestions ; we do it here warmly.

References

[1] A. Asperti. Light affine logic. In *Proceedings of LICS'98*, pages 300–308, 1998.

[2] Patrick Baillot. Stratified coherent spaces : a denotational semantics for light linear logic. *Theoretical Computer Science*, 318(1-2):29–55, June 2004.

[3] Patrick Baillot and Virgile Mogbil. Soft lambda-calculus : a language for polynomial time computation. In *Proceedings of FOSSACS*, volume 2987 of *LNCS*, pages 27–41. Springer, 2004.

[4] H. P. Barendregt. *The Lambda Calculus*. Noth-Holland, 1984.

[5] M. Coppo and M. Dezani-Ciancaglini. An extension of the basic functionality theory for the lambda-calculus. *Notre Dame J. Formal Logic*, 21(4):685–693, 1980.

[6] P. Giannini and S. Ronchi della Rocca. Characterization of typings in polymorphic type discipline. In *Proceedings of LICS'88*, pages 61–70, 1988.

[7] J.-Y. Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.

[8] J.-Y. Girard. Light linear logic. *Information and Computation*, 143(2):175–204, 1998.

[9] J.L. Krivine. *Lambda-calcul types et modèles*. Masson, 1990.

[10] Y. Lafont. Soft linear logic and polynomial time. *Theoretical Computer Science*, 318(1-2):163–180, 2004.

[11] L. Roversi. A P-Time Completeness Proof for Light Logics. In *Ninth Annual Conference of the EACSL (CSL'99)*, volume 1683 of *Lecture Notes in Computer Science*, pages 469 – 483, Madrid (Spain), September 1999. Springer-Verlag.

[12] K. Terui. Light affine lambda calculus and polytime strong normalization. In *Proceedings of LICS'01*, pages 209–220, 2001.

[13] K. Terui. Light affine set theory : a naive set theory of polynomial time. *Studia Logica*, 77:9–40, 2004.