Linear Additives
(Extended Abstract)

Gianluca Curzi
University of Turin
Torino, Italy
curzi@di.unito.it

We introduce LAM, a type assignment system with a weaker version of the additive rules, called *linear additives*. Typable terms of LAM enjoy the strong linear normalization property, and a mildly weakened cut-elimination is established for derivations. Also, we present a sound translation from LAM into IMLL_2 and we study its complexity.

1 Introduction

The additive rules of *Linear Logic* [8] are somehow related to non-determinism. This connection has been investigated in the field of ICC (*Implicit Computational Complexity*), where several approaches to capture NP with non-deterministic variants of the additive rules have been proposed [13, 6, 14].

However, the additive rules and their variants have a serious drawback when applications to ICC are considered: these rules affect the complexity of cut-elimination, that may require an exponential amount of time and space. To circumvent this problem, special reduction strategies are needed. Concerning MALL (*Multiplicative Additive Linear Logic*) and its second-order formulation, Girard has introduced the *lazy cut-elimination* [7]. Another example is in [6], where Marion et al. proved an implicit characterization of NP in STA_+, a system that extends STA (*Soft Type Assignment* [5]) with a *sum logical rule* for the choice operator $M+N$ and the related non-deterministic reductions $M \leftarrow M+N \rightarrow N$. The sum rule is very close to an additive rule, and suffers the same disadvantages. So, a specific reduction strategy has been defined to establish the Non-deterministic Polytime Soundness for STA_+.

In this paper we present a different solution to the complexity-theoretical issues caused by the additive rules. We focus on the second-order intuitionistic formulation of MALL, i.e. IMALL_2. We shall look at IMALL_2 as a type system by considering formulas as types and by decorating logical derivations with λ-terms endowed with pairs and projections. The analysis of the non-linear features of the standard additive rule &R of Linear Logic leads to LAM (*Linearly Additive Multiplicative Type Assignment*). LAM is a subsystem of IMALL_2 with *linear additive rules*, weaker than the standard ones, and gives types to a calculus with a `copy` construct able to manage substitutions linearly.

The cut-elimination rules for LAM are constrained to copy-cat the reduction rules on terms. We then identify the class of $\forall$*-lazy* types, whose derivations can always be turned into cut-free ones in cubic time. Moreover, we show that typable terms in LAM enjoy the Subject reduction property and a *Strong* linear normalization.

Last, we translate LAM into IMLL_2, following [3, 11, 12]. The translation is computationally sound and produces a derivation of IMLL_2 exponentially bigger than its source in LAM.

Submitted to:
TLLA2020

2 The system IMALL_2 and the exponential blow up

We briefly recall the $(\multimap, \&, \forall)$ fragment of IMALL_2. We present it as a type assignment for the calculus $\Lambda_{\pi,\langle\rangle}$, whose terms are defined by the following grammar:

$$M := x \mid \lambda x.M \mid MM \mid \langle M, M\rangle \mid \pi_1(M) \mid \pi_2(M) \tag{1}$$

where x is taken from a denumerable set of variables. The set of free variables of a term M is written $FV(M)$, and the meta-level substitution for terms is denoted $M[N/x]$. The *size* $|M|$ of a term M is the number of nodes in its syntax-tree. The *one-step relation* $\to_\beta$ is the binary relation over $\Lambda_{\pi,\langle\rangle}$ defined by:

$$(\lambda x.M)N \to_\beta M[N/x] \qquad \pi_i\langle M_1, M_2\rangle \to_\beta M_i \qquad i \in \{1,2\} \tag{2}$$

its reflexive and transitive closure is $\to_\beta^*$. The set $\Theta_\&$ of types of IMALL_2 is generated by the following grammar:

$$A := \alpha \mid A \multimap A \mid A \,\&\, A \mid \forall \alpha.A \tag{3}$$

where α belongs to a denumerable set of type variables. The set of free type variables of a type A is written $FV(A)$, and the standard meta-level substitution for types is denoted $A\langle B/\alpha\rangle$. A type A is *closed* if $FV(A) = \emptyset$. The *size* $|A|$ of a type A is the number of nodes in its syntax-tree.

The system IMALL_2 (*Intuitionistic Second-Order Multiplicative Additive Linear Logic*) is displayed in Figure 1, where &R and &Li are the *additives rules*. It derives *judgements* with form $\Gamma \vdash M : A$, where Γ is a finite multiset of *assumptions* $x : A$, $M \in \Lambda_{\pi,\langle\rangle}$, and $A \in \Theta_\&$. The system requires the *linear constraint* $FV(\Gamma) \cap FV(\Delta) = \emptyset$ in both *cut* and $\multimap$L, where $FV(\Gamma)$ denotes the set of all free type variables in Γ. With $\mathscr{D} \triangleleft \Gamma \vdash M : A$ we denote a derivation $\mathscr{D}$ of $\Gamma \vdash M : A$, and in this case we say that M is an *inhabitant* of A. The *size* $|\Gamma|$ of a context $\Gamma = x_1 : A_1, \ldots, x_n : A_n$ is $\sum_{i=1}^n |A_i|$, and the *size* $|\mathscr{D}|$ of a derivation $\mathscr{D}$ is the number of its rules applications.

We recall that IMLL_2 (*Intuitionistic Second-Order Multiplicative Linear Logic*) is obtained from IMALL_2 by excluding the additive rules from Figure 1. It gives a type *exactly* to the class of linear λ-terms (see [9, 12]), i.e. those terms M from the standard λ-calculus such that: (i) each free variables of M occurs in it exactly once and (ii) for each subterm $\lambda x.N$ of M, x occurs in N exactly once. Tensors ($\otimes$) and units ($\mathbf{1}$) can be introduced in IMLL_2 (and hence in IMALL_2) by means of second-order definitions (see for example [12]). Hence, the inference rules for $\otimes$ and $\mathbf{1}$, and the reduction rules $\mathtt{let}\ \mathbf{I}\ \mathtt{be}\ \mathbf{I}\ \mathtt{in}\ N \to_\beta N$ and $\mathtt{let}\ M_1 \otimes M_2\ \mathtt{be}\ x_1 \otimes x_2\ \mathtt{in}\ N \to_\beta N[M_1/x_1, M_2/x_2]$ are derivable in IMLL_2.

The rule &R affects the complexity of normalization in IMALL_2 letting the size of typable terms and the number of their redexes grow exponentially during reduction, as the following example shows:

Example 1. Consider the term $\mathtt{add}_n^x$ defined by $\mathtt{add}_0^x \triangleq x$ and $\mathtt{add}_n^x \triangleq (\lambda y.\mathtt{add}_{n-1}^y)\langle x, x\rangle$ $(n > 0)$. If M is typable in IMALL_2, then so is $(\lambda x.\mathtt{add}_n^x)M$. Moreover $(\lambda x.\mathtt{add}_n^x)M \to_\beta^* M[n]$, where $M[n]$ is defined by $M[0] \triangleq M$ and $M[n] \triangleq \langle M[n-1], M[n-1]\rangle$ $(n > 0)$.

A normalization with linear cost is recovered by considering the *lazy reduction*, originally defined by Girard for proof nets [8]. Intuitively, the lazy reduction "freezes" substitutions inside a pair. Whenever a term M has *lazy type* in IMALL_2, i.e. a type not containing negative occurrences of $\forall$ and positive occurrences of &, M evaluates to a normal form by lazy reduction. This happens because every pair $\langle N_1, N_2\rangle$ in M eventually turns into a redex $\pi_i\langle N_1, N_2\rangle$, so that all the suspended substitutions are performed.

Similarly, we can define a *lazy cut-elimination* for IMALL_2, which does not eliminate all those instances of *cut* whose right premise is the conclusion of &R, because their elimination involves duplications of derivations. In this case, every derivation of a lazy type can be turned into a cut-free one in cubic time (commuting steps require quadratic time).

$$\frac{}{x:A\vdash x:A}\,ax \qquad \frac{\Gamma\vdash N:A \qquad \Delta,x:A\vdash M:C}{\Gamma,\Delta\vdash M[N/x]:C}\,cut \qquad \frac{\Gamma,x:A\vdash M:B}{\Gamma\vdash \lambda x.M:A\multimap B}\,\multimap\mathrm{R}$$

$$\frac{\Gamma\vdash N:A \qquad \Delta,x:B\vdash M:C}{\Gamma,y:A\multimap B,\Delta\vdash M[yN/x]:C}\,\multimap\mathrm{L} \qquad \frac{\Gamma\vdash M_1:A_1 \qquad \Gamma\vdash M_2:A_2}{\Gamma\vdash\langle M_1,M_2\rangle:A_1\,\&\,A_2}\,\&\mathrm{R}$$

$$\frac{\Gamma,x_i:A_i\vdash M:C \quad i\in\{1,2\}}{\Gamma,y:A_1\,\&\,A_2\vdash M[\pi_i(y)/x_i]:C}\,\&\mathrm{L}i \qquad \frac{\Gamma\vdash M:A\langle\gamma/\alpha\rangle \quad \gamma\notin \mathrm{FV}(\Gamma)}{\Gamma\vdash M:\forall\alpha.A}\,\forall\mathrm{R} \qquad \frac{\Gamma,x:A\langle B/\alpha\rangle\vdash M:C}{\Gamma,x:\forall\alpha.A\vdash M:C}\,\forall\mathrm{L}$$

Figure 1: The system IMALL_2.

3 Linear additives

To motivate the introduction of LAM, let us focus on the term $(\lambda x.\mathtt{add}_1^x)M$ in Example 1. Its reduction produces $\langle M,M\rangle$, so that it duplicates both the number of redexes and the size of M. To avoid the increase of redexes, we introduce the new construct `copy` that forbids the substitution of M for some variable in a pair *before* M has been fully normalized. Its typing rule can be seen as a weaker version of &R:

$$\frac{x_1:A\vdash M_1:A_1 \qquad x_2:A\vdash M_2:A_2}{x:A\vdash \mathtt{copy}\ x\ \mathtt{as}\ x_1,x_2\ \mathtt{in}\ \langle M_1,M_2\rangle:A_1\,\&\,A_2} \tag{4}$$

and its reduction rule is $\mathtt{copy}\ M\ \mathtt{as}\ x_1,x_2\ \mathtt{in}\ \langle M_1,M_2\rangle\to\langle M_1[M/x_1],M_2[M/x_2]\rangle$, that can be performed whenever M is a "value":

Definition 1 (Values)**.** A *value* V is any closed normal term of $\Lambda_{\pi,\langle\rangle}$. The set of all values is denoted $\mathscr{V}$.

However, this is not enough, as the new reduction rule doubles the size of M, thus possibly causing a size explosion. To avoid the increase of size during reduction, we add a third premise $\vdash V:A$ to (4), where $V\in\mathscr{V}$, and we require A,A_1,A_2 to be in the following class of types:

Definition 2 ($\forall$-lazy types)**.** A type $A\in\Theta_{\&}$ is $\forall$-*lazy* if it contains no negative occurrences of $\forall$.

We obtain the following refinement of (4):

$$\frac{x_1:A\vdash M_1:A_1 \qquad x_2:A\vdash M_2:A_2 \qquad \vdash V:A}{x:A\vdash \mathtt{copy}^V\ x\ \mathtt{as}\ x_1,x_2\ \mathtt{in}\ \langle M_1,M_2\rangle:A_1\,\&\,A_2} \tag{5}$$

where the value V decorates the `copy` construct. Since every $\forall$-lazy type A has always finitely many values (Remark 1), when V is chosen with largest size among those inhabitants, the construct $\mathtt{copy}^V$ gives a bound to the size of the new copy of V' produced by the reduction $\mathtt{copy}^V V'\ \mathtt{as}\ x_1,x_2\ \mathtt{in}\ \langle M_1,M_2\rangle\to\langle M_1[V'/x_1],M_2[V'/x_2]\rangle$, for every value V' of type A. Notice that the term $\mathtt{copy}^V M\ \mathtt{as}\ x_1,x_2\ \mathtt{in}\ \langle M_1,M_2\rangle$ behaves as a suspended substitution of M in the pair $\langle M_1,M_2\rangle$, like in the case of the lazy reduction discussed in Section 2. The crucial difference is that the latter *always* forbids the substitution of M in $\langle M_1,M_2\rangle$, while the former allows the replacement whenever M is a value, increasing the expressiveness.

Beside the restriction of the additive rules to $\forall$-lazy types ($\forall$-*laziness condition*), we need a further requirement, called *closure condition*, that forbids derivations of judgements $\Gamma\vdash M:A$ such that $FV(A)=\emptyset$ and $FV(\Gamma)\neq\emptyset$. The latter plays an essential role to assure a weak form of cut-elimination.

The above intuitions can be expressed formally by the next two definitions.

Definition 3 (The calculus Λ_{LAM})**.** The set Λ_{LAM} of *terms* extends the grammar (1) with the clause $\mathtt{copy}^V M \ \mathtt{as}\ x,y\ \mathtt{in}\ \langle M,M\rangle$, where $V \in \mathscr{V}$. If $P = \mathtt{copy}^V M \ \mathtt{as}\ x_1,x_2\ \mathtt{in}\ \langle N_1,N_2\rangle$, we define $FV(P) = \mathrm{FV}(M) \cup (\mathrm{FV}(\langle N_1,N_2\rangle) \setminus \{x_1,x_2\})$ and $|P| = |V| + |M| + |\langle N_1,N_2\rangle| + 1$. The *one-step reduction* $\to$ on Λ_{LAM} extends $\to_\beta$ in (2) with the following rule:

$$\mathtt{copy}^{V'} V \ \mathtt{as}\ x_1,x_2\ \mathtt{in}\ \langle M_1,M_2\rangle \to \langle M_1[V/x_1], M_2[V/x_2]\rangle \qquad V,V' \in \mathscr{V} \tag{6}$$

Its reflexive and transitive closure is $\to^*$. A term is a *normal form* if no reduction applies to it.

Definition 4 (The system LAM)**.** The system LAM (*Linearly Additive Multiplicative Type Assignment*) is the type assignment for Λ_{LAM} obtained from IMALL_2 by replacing the rule &R with the following:

$$\frac{\vdash M_1 : A_1 \qquad \vdash M_2 : A_2}{\vdash \langle M_1,M_2\rangle : A_1 \,\&\, A_2}\ \&\mathrm{R0} \qquad \frac{x_1 : A \vdash M_1 : A_1 \qquad x_2 : A \vdash M_2 : A_2 \qquad \vdash V : A}{x : A \vdash \mathtt{copy}^V x \ \mathtt{as}\ x_1,x_2\ \mathtt{in}\ \langle M_1,M_2\rangle : A_1 \,\&\, A_2}\ \&\mathrm{R1}$$

and by imposing the conditions below:

(i) $\forall$-*laziness*: the types A, A_1 and A_2 in &R0, &R1, and &Li (Figure 1) must be *closed and* $\forall$-*lazy*;

(ii) *Closure*: no instance of $\multimap$L has conclusion $\Delta, y : A \multimap B, \Gamma \vdash M : C$ with $FV(B) = \emptyset$ and $FV(A) \neq \emptyset$, and no instance of $\forall$R has conclusion $\Gamma \vdash M : \forall\alpha.A$ with $FV(\forall\alpha.A) = \emptyset$ and $FV(\Gamma) \neq \emptyset$.

The resulting rules &R0, &R1 and &Li in LAM are called *linear additive rules*. We say that $x_1 : A_1, \ldots, x_n : A_n \vdash M : B$ is a $\forall$-*lazy judgement* if $A_1 \multimap \ldots \multimap A_n \multimap B$ is a $\forall$-lazy type. Moreover, $\mathscr{D} \triangleleft \Gamma \vdash M : A$ is called a $\forall$-*lazy derivation* if $\Gamma \vdash M : A$ is a $\forall$-lazy judgement. Finally, given $\mathscr{D} \triangleleft \Gamma \vdash M : A$ a $\forall$-lazy and cut-free derivation, $\mathscr{D}$ is η-*expanded* if all its axioms are *atomic*, i.e. of the form $x : \alpha \vdash x : \alpha$ for some type variable α. In this case, M is called a η-*long normal form* (note that M is a normal form).

The following are basic properties about $\forall$-laziness and η-long normal forms.

Proposition 1 (Properties of $\forall$-laziness)**.**

1. *Given an instance of* $\multimap R$, $\multimap L$, &R0, $\forall R$, *if one of its premises is not* $\forall$*-lazy then its conclusion is not* $\forall$*-lazy. Moreover, given an instance of* &R1, &L, $\forall L$, *its conclusion is not* $\forall$*-lazy;*
2. *If* $\mathscr{D} \triangleleft \Gamma \vdash M : A$ *is a* $\forall$*-lazy and cut-free derivation, then it has no instances of* &R1, &L *and* $\forall L$. *Moreover, M is normal and contains no occurrences of* copy *(in particular, if* $\Gamma = \emptyset$ *then* $M \in \mathscr{V}$*).*

Proposition 2 (Properties of η-long nfs)**.** *Let* $\mathscr{D} \triangleleft \Gamma \vdash M : A$ *be* $\forall$*-lazy and* η*-expanded. Then* $|M| \leq |\Gamma| + |A| \leq 2 \cdot |\mathscr{D}|$. *Moreover, if* $\mathscr{D}' \triangleleft \Gamma \vdash N : A$ *is a cut-free derivation, then both* $|N| \leq |M|$ *and* $|\mathscr{D}'| \leq |\mathscr{D}|$.

Remark 1. W.l.o.g. we shall assume that any derivation of the rightmost premise $\vdash V : A$ of an instance of &R1 is η-expanded. This prevents the increase of size during normalization or cut-elimination, since by Proposition 2 the size of a closed η-long normal form of a $\forall$-lazy type A is larger than the size of any other closed normal inhabitant of A. This also implies that $\forall$-lazy types have always *finitely* many values.

4 Cubic $\forall$-lazy cut-elimination and strong linear normalization

LAM is essentially a subsystem of IMALL_2. Hence, from a purely proof-theoretical point of view, the former inherits the standard cut-elimination rules of the latter (see for example [2]). However, the reduction rule (6) duplicates only values to avoid any exponential blow up, and this causes a mismatch between term reductions and cut-elimination steps, as the following example shows.

$$(\&\mathrm{R}0,\&\mathrm{L}i)\quad \dfrac{\dfrac{\vdash N_1 : A_1 \qquad \vdash N_2 : A_2}{\vdash \langle N_1, N_2\rangle : A_1 \,\&\, A_2} \qquad \dfrac{\Gamma, x_i : A_i \vdash M : B \qquad i \in \{1,2\}}{\Gamma, x : A_1 \,\&\, A_2 \vdash M[\pi_i(x)/x_i] : B}}{\Gamma \vdash M[\pi_i\langle N_1, N_2\rangle / x_i] : B}\, cut \rightsquigarrow \dfrac{\vdash N_i : A_i \qquad \Gamma, x_i : A_i \vdash M : B}{\Gamma \vdash M[N_i/x_i] : B}\, cut$$

$$(X,\&\mathrm{R}1)\quad \dfrac{\dfrac{\mathscr{D}^\dagger}{\vdash V : A} \qquad \dfrac{x_1 : A \vdash N_1 : B_1 \qquad x_2 : A \vdash N_2 : B_2 \qquad \vdash V' : A}{x : A \vdash \mathtt{copy}^{V'} x \;\mathtt{as}\; x_1, x_2 \;\mathtt{in}\; \langle N_1, N_2\rangle : B_1 \,\&\, B_2}}{\vdash \mathtt{copy}^{V'} V \;\mathtt{as}\; x_1, x_2 \;\mathtt{in}\; \langle N_1, N_2\rangle : B_1 \,\&\, B_2}\, cut \rightsquigarrow \dfrac{\dfrac{\dfrac{\mathscr{D}^\dagger}{\vdash V : A} \qquad \dfrac{x_1 : A \vdash N_1 : B_1}{\vdash N_1[V/x_1] : B_1}\, cut}{\vdash \langle N_1[V/x_1], N_2[V/x_2]\rangle : B_1 \,\&\, B_2}\,\&\mathrm{R}0 \qquad \dfrac{\dfrac{\mathscr{D}^\dagger}{\vdash V : A} \qquad \dfrac{x_2 : A \vdash N_2 : B_1}{\vdash N_2[V/x_2] : B_2}\, cut}{}}{}$$

Figure 2: $\forall$-lazy cut-elimination rules for $(\&\mathrm{R}0,\&\mathrm{L}i)$ and for a ready cut, where $\mathscr{D}^\dagger$ is cut-free.

Example 2. Le us consider the following derivation of LAM:

$$\dfrac{y : \mathbf{1} \vdash y\mathbf{I} : \mathbf{1} \qquad \dfrac{x_1 : \mathbf{1} \vdash x_1 : \mathbf{1} \qquad x_2 : \mathbf{1} \vdash x_2 : \mathbf{1} \qquad \vdash \mathbf{I} : \mathbf{1}}{x : \mathbf{1} \vdash \mathtt{copy}^{\mathbf{I}} x \;\mathtt{as}\; x_1, x_2 \;\mathtt{in}\; \langle x_1, x_2\rangle : \mathbf{1} \,\&\, \mathbf{1}}\,\&\mathrm{R}1}{y : \mathbf{1} \vdash \mathtt{copy}^{\mathbf{I}} y\mathbf{I} \;\mathtt{as}\; x_1, x_2 \;\mathtt{in}\; \langle x_1, x_2\rangle : \mathbf{1} \,\&\, \mathbf{1}}\, cut \quad . \tag{7}$$

and let us apply the cut-elimination rule of IMALL_2 moving the *cut* upward. We get a derivation of $y : \mathbf{1} \vdash M : \mathbf{1} \,\&\, \mathbf{1}$, where M is $\mathtt{copy}^{\mathbf{I}} y \;\mathtt{as}\; y_1, y_2 \;\mathtt{in}\; \langle y_1\mathbf{I}, y_2\mathbf{I}\rangle$. But $\mathtt{copy}^{\mathbf{I}} y\mathbf{I} \;\mathtt{as}\; x_1, x_2 \;\mathtt{in}\; \langle x_1, x_2\rangle \not\to^* M$.

Definition 5 here below introduces $\forall$-*lazy cut-elimination rules*. They circumvent the above mismatch by never eliminating instances of *cut* like (7). The drawback is that now cuts exist we cannot eliminate. We shall prove that a $\forall$-*lazy cut-elimination strategy* exists which rewrites *every* derivation of a $\forall$-lazy judgement into a cut-free derivation in cubic time (Theorem 6). This result is analogous to the lazy cut-elimination for lazy types discussed in Section 2. The crucial difference is that, while Girard's laziness rules out both negative occurrences of $\forall$ and positive occurrences of &, the $\forall$-laziness only requires the absence of negative $\forall$. The reward is that $\forall$-lazy derivations with nested applications of &R1 *become legal* and have no exponential blow up.

Definition 5 ($\forall$-lazy cut-elimination rules)**.**

- Cuts are divided into three classes: the *symmetric cuts* are $(\multimap\mathrm{R}, \multimap\mathrm{L})$, $(\&\mathrm{R}0,\&\mathrm{L}1)$, $(\&\mathrm{R}0,\&\mathrm{L}2)$, $(\forall\mathrm{R},\forall\mathrm{L})$ and those involving *ax*; the *critical cuts* have form $(X,\&\mathrm{R}1)$, for some rule X; finally, the *commuting cuts* are all the remaining instances of *cut*.
- We say that a critical cut with premises $\mathscr{D} \triangleleft \Gamma \vdash N : A$ and $\mathscr{D}' \triangleleft x : A \vdash M : B$ is *safe* if $\Gamma = \emptyset$, and *deadlock* otherwise. It is called *ready* if it is safe and $\mathscr{D}$ is cut-free (in this case $N \in \mathscr{V}$, by Proposition 1.2).
- The $\forall$-*lazy cut-elimination rules* are the standard cut-elimination rules for IMALL_2 (see [2]) with the proviso that only *ready* critical cuts can be eliminated (see Figure 2). If $\mathscr{D}$ rewrites to $\mathscr{D}'$ by a $\forall$-lazy cut-elimination rule, we write $\mathscr{D} \rightsquigarrow \mathscr{D}'$. The reflexive and transitive closure of $\rightsquigarrow$ is $\rightsquigarrow^*$.

The following is an easy induction on derivations using the $\forall$-*laziness* and the *closure* conditions.

Proposition 3. *If $\mathscr{D} \triangleleft \Gamma \vdash M : A$ and $FV(A) = \emptyset$ then $FV(\Gamma) = \emptyset$.*

The next two lemmas are essential to ensure a $\forall$-lazy cut-elimination result:

Lemma 4 (Existence of a safe cut)**.** *Let $\mathscr{D} \triangleleft \Gamma \vdash M : A$ be a $\forall$-lazy derivation with only critical cuts in it. At least one of those cuts (if any) is* safe*.*

Proof. First, we prove that the conclusion of a deadlock is never a $\forall$-lazy judgement. So, let $\Delta \vdash N : B$ and $x : B \vdash M' : C$ be the premises of a deadlock. By definition, we have $\Delta \neq \emptyset$, so let $y : D$ be an assumption in Δ. Since $x : B \vdash M' : C$ is a conclusion of &R1, the $\forall$-*laziness* condition implies $FV(B) = \emptyset$. Hence $FV(D) = \emptyset$, by Proposition 3. A closed type must contain at least a $\forall$ in positive position, so that $\Delta \vdash M'[N/x] : C$ cannot be a $\forall$-lazy judgement. Now, suppose towards a contradiction that all critical cuts in $\mathscr{D}$ are deadlocks. Thus, if $\mathscr{D}$ contains at least one critical cut, it also contains a judgement that is not $\forall$-lazy. Let $R_1, \dots R_n$ be the sequence of rule instances from $\Gamma \vdash M : A$ up to this judgement. We prove by induction on n that $\Gamma \vdash M : A$ cannot be $\forall$-lazy, contradicting the assumption. The case $n = 0$ is trivial. If $n > 0$, then we have two cases depending on R_n. If R_n is a critical cut then it is a deadlock and its conclusion cannot be $\forall$-lazy, so that we can apply the induction hypothesis. If R_n is not a critical cut, we apply Proposition 1.1 and the induction hypothesis. Hence, at least one critical cut in $\mathscr{D}$ is safe. □

Definition 6 (Height and weight)**.** Let $\mathscr{D} \triangleleft \Gamma \vdash M : A$ be a derivation of LAM. The *weight of* $\mathscr{D}$, written $\&(\mathscr{D})$, is the number of instances of the rule &R1 in $\mathscr{D}$. Given a rule instance R in $\mathscr{D}$, the *height of* R, written $h(R)$, is the number of rule instances from the conclusion $\Gamma \vdash M : A$ of $\mathscr{D}$ upward to the conclusion of R. The *height of* $\mathscr{D}$, written $h(\mathscr{D})$, is the largest $h(R)$ among its rule instances.

Lemma 5 (Eliminating a ready cut)**.** *Let $\mathscr{D} \triangleleft \Gamma \vdash M : A$ be a $\forall$-lazy derivation with only critical cuts in it. Then, there exists $\mathscr{D}^*$ such that $\mathscr{D} \rightsquigarrow \mathscr{D}^*$ and $|\mathscr{D}^*| + 2 \cdot \&(\mathscr{D}^*) < |\mathscr{D}| + 2 \cdot \&(\mathscr{D})$.*

Proof. By Lemma 4, $\mathscr{D}$ contains at least one safe cut. Let R be the following safe cut:

$$\dfrac{\begin{array}{c}\mathscr{D}' \\ \vdash N : A\end{array} \qquad \dfrac{\begin{array}{ccc}\mathscr{D}_1 & \mathscr{D}_2 & \mathscr{D}'' \\ x_1 : A \vdash M_1 : B_1 & x_2 : A \vdash M_2 : B_2 & \vdash V' : A\end{array}}{x : A \vdash \mathtt{copy}^{V'} x \ \mathtt{as}\ x_1, x_2 \ \mathtt{in}\ \langle M_1, M_2 \rangle : B_1 \,\&\, B_2}\ \&\mathrm{R1}}{\vdash \mathtt{copy}^{V'} N \ \mathtt{as}\ x_1, x_2 \ \mathtt{in}\ \langle M_1, M_2 \rangle : B_1 \,\&\, B_2}\ \mathit{cut} \tag{8}$$

with maximal height. Since A is a $\forall$-lazy type by the $\forall$-*laziness* condition, $\mathscr{D}'$ is a $\forall$-lazy derivation. By Lemma 4 and maximality of $h(R)$, $\mathscr{D}'$ is cut-free. Therefore, R is a ready cut. Let $\mathscr{D}^*$ be the derivation obtained by eliminating R (see Figure 2). By Remark 1, $\mathscr{D}''$ is η-expanded, hence $\forall$-lazy and cut-free. By Proposition 1.2, $\mathscr{D}'$ and $\mathscr{D}''$ have no instances of &R1, so that $\&(\mathscr{D}^*) = \&(\mathscr{D}) - 1$. Moreover, $|\mathscr{D}'| \leq |\mathscr{D}''|$ by Proposition 2. We have: $2 \cdot |\mathscr{D}'| + |\mathscr{D}_1| + |\mathscr{D}_2| + 3 + 2 \cdot \&(\mathscr{D}^*) < |\mathscr{D}| + 2 \cdot \&(\mathscr{D}^*) + 2 = |\mathscr{D}| + 2 \cdot (\&(\mathscr{D}^*) + 1)$. Therefore, $|\mathscr{D}^*| + 2 \cdot \&(\mathscr{D}^*) < |\mathscr{D}| + 2 \cdot \&(\mathscr{D})$. □

Theorem 6 ($\forall$-lazy cut-elimination has a cubic bound)**.** *Let $\mathscr{D} \triangleleft \Gamma \vdash M : A$ be a $\forall$-lazy derivation. Then, the $\forall$-lazy cut-elimination reduces $\mathscr{D}$ to a cut-free $\mathscr{D}^* \triangleleft \Gamma \vdash N : A$ in $\mathscr{O}(|\mathscr{D}|^3)$ steps.*

Proof. We define a $\forall$-*lazy cut-elimination strategy* divided into *rounds*. At each round:

{1} we eliminate all the commuting instances of *cut*;

{2} if a symmetric instance of *cut* exists, we eliminate it; otherwise, all instances of *cut* are critical, and we eliminate a *ready* one, if any.

We proceed by induction on the lexicographical order of the pairs $(|\mathscr{D}| + 2 \cdot \&(\mathscr{D}), H(\mathscr{D}))$, where $H(\mathscr{D})$ is the sum of the heights $h(\mathscr{D}')$ of all subderivations $\mathscr{D}'$ of $\mathscr{D}$ whose conclusion is an instance of *cut*. During round {1}, every commuting $\forall$-lazy cut-elimination step just moves an instance of *cut* upward, strictly decreasing $H(\mathscr{D})$ and leaving $|\mathscr{D}| + 2 \cdot \&(\mathscr{D})$ unaltered. During round {2}, every symmetric $\forall$-lazy cut-elimination step shrinks $|\mathscr{D}|$. If only critical cuts are in $\mathscr{D}$, by Lemma 5, there exists $\mathscr{D}'$ such that $\mathscr{D} \rightsquigarrow \mathscr{D}'$ and $|\mathscr{D}'| + 2 \cdot \&(\mathscr{D}') < |\mathscr{D}| + 2 \cdot \&(\mathscr{D})$. There can be at most $|\mathscr{D}| + 2 \cdot \&(\mathscr{D}) \leq 3 \cdot |\mathscr{D}|$ $\forall$-lazy cut-elimination steps applied to symmetric and critical cuts, and at most $\mathscr{O}(|\mathscr{D}|^2)$ commuting steps. Therefore, the total number of $\forall$-cut elimination steps is $\mathscr{O}(|\mathscr{D}| \cdot |\mathscr{D}|^2)$. □

The strong linear normalization for LAM is a corollary of the following strengthened version of the Subject reduction:

Theorem 7 (Subject reduction)**.** *If $\mathscr{D} \triangleleft \Gamma \vdash M_1 : A$ and $M_1 \to M_2$, then $|M_2| < |M_1|$ and there exists $\mathscr{D}^*$ such that $\mathscr{D}^* \triangleleft \Gamma \vdash M_2 : A$.*

Sketch. Suppose $M_1 \to M_2$ is applied to a redex $N = \mathtt{copy}^{V'}\, V \ \mathtt{as}\ x_1, x_2 \ \mathtt{in}\ \langle N_1, N_2 \rangle$. By Remark 1, V' is a η-long normal form and, by Proposition 2, $|V| \leq |V'|$. Therefore, $|\langle N_1[V/x_1], N_2[V/x_2]\rangle| < |N|$. □

Corollary 8 (Strong linear normalization)**.** *If $\mathscr{D} \triangleleft \Gamma \vdash M : A$ then M reduces to a normal form in at most $|M|$ reduction steps.*

5 A translation of LAM into IMLL_2 and exponential compression

In [12], Mairson and Terui show the existence of linear λ-terms able to duplicate or erase all closed normal inhabitants of a given closed Π_1 type, i.e. a closed type of IMLL_2 containing no negative occurrences of $\forall$.

Theorem 9 (Erasure and duplication [12])**.**

1. *For any closed Π_1 type A there is a (closed) linear λ-term $\mathtt{E}_A$ of type $A \multimap \mathbf{1}$ such that, for all closed and normal inhabitant M of A, $\mathtt{E}_A\, M \to^*_\beta \mathbf{I}$.*
2. *For any closed and* inhabited *Π_1 type A there is a (closed) linear λ-term $\mathtt{D}_A$ of type $A \multimap A \otimes A$ such that, for all closed and normal inhabitant M of A, $\mathtt{D}_A\, M \to^*_{\beta\eta} M \otimes M$.*

We call $\mathtt{E}_A$ *eraser* and $\mathtt{D}_A$ *duplicator* of A. Point 2 of Theorem 9 was only sketched in [12]. A detailed proof of the construction of $\mathtt{D}_A$ is in [3], which also estimates the complexity of duplicators and erasers:

Proposition 10 (Size of $\mathtt{E}_A$ and $\mathtt{D}_A$ [3])**.** *If A is a closed Π_1 type, then $|\mathtt{E}_A| \in \mathscr{O}(|A|)$. Moreover, if A is inhabited, then $|\mathtt{D}_A| \in \mathscr{O}(2^{|A|^2})$.*

Following [3], we define a translation $(\_)^\bullet$ from derivations of LAM into linear λ-terms with type in IMLL_2 which maps closed $\forall$-lazy types into closed Π_1 types, and instances of the inference rules &R1 and &L*i* into, respectively, duplicators and erasers of closed Π_1 types. We prove that the translation is sound and that the linear λ-term $\mathscr{D}^\bullet$ associated with a derivation $\mathscr{D}$ of LAM has a size which is exponential with respect to the one of $\mathscr{D}$.

Definition 7 (From LAM to IMLL_2)**.** We define a map $(\_)^\bullet$ translating a derivation $\mathscr{D} \triangleleft \Gamma \vdash_{\mathsf{LAM}} M : A$ into a linear λ-term $\mathscr{D}^\bullet$ such that $\Gamma^\bullet \vdash_{\mathsf{IMLL}_2} \mathscr{D}^\bullet : A^\bullet$.

1. The map $(\_)^\bullet$ is defined on types of $\Theta_\&$ by induction on their structure:

$$\alpha^\bullet \triangleq \alpha \qquad (A \,\&\, B)^\bullet \triangleq A^\bullet \otimes B^\bullet$$
$$(A \multimap B)^\bullet \triangleq A^\bullet \multimap B^\bullet \qquad (\forall \alpha.A)^\bullet \triangleq \forall \alpha.A^\bullet \ .$$

   For all contexts $\Gamma = x_1 : A_1, \ldots, x_n : A_n$, we set $\Gamma^\bullet \triangleq x_1 : A_1^\bullet, \ldots, x_n : A_n^\bullet$.

2. The map $(\_)^\bullet$ is defined on derivations $\mathscr{D} \triangleleft \Gamma \vdash M : A$ by induction on the last rule. Principal cases are:
   (a) if $\mathscr{D}$ is *ax* with conclusion $x : A \vdash x : A$, then $\mathscr{D}^\bullet \triangleq x$;
   (b) if $\mathscr{D}$ has last rule *cut* with premises $\mathscr{D}_1 \triangleleft \Delta \vdash N : B$ and $\mathscr{D}_2 \triangleleft \Sigma, x : B \vdash P : A$, where $M = P[N/x]$, then $\mathscr{D}^\bullet \triangleq \mathscr{D}_2^\bullet[\mathscr{D}_1^\bullet/x]$;

(c) if $\mathscr{D}$ has last rule &R0 with premises $\mathscr{D}_1 \triangleleft \vdash N_1 : B_1$ and $\mathscr{D}_2 \triangleleft \vdash N_2 : B_2$ then $\mathscr{D}^\bullet \triangleq \mathscr{D}_1^\bullet \otimes \mathscr{D}_2^\bullet$;

(d) if $\mathscr{D}$ has last rule &Li with premise $\mathscr{D}_1 \triangleleft \Delta, x_i : B_i \vdash N : A$, where $\Gamma = \Delta, x : B_1 \,\&\, B_2$, then $\mathscr{D}^\bullet \triangleq \texttt{let}\ x\ \texttt{be}\ x_1 \otimes x_2\ \texttt{in}\ (\texttt{let}\ \mathtt{E}_{B^\bullet_{3-i}} x_{3-i}\ \texttt{be}\ \mathbf{I}\ \texttt{in}\ \mathscr{D}_1^\bullet)$, where $\mathtt{E}_{B^\bullet_{3-i}}$ is the eraser of $B^\bullet_{3-i}$;

(e) if $\mathscr{D}$ ends with &R1 with premises $\mathscr{D}_1 \triangleleft x_1 : B \vdash N_1 : B_1$, $\mathscr{D}_2 \triangleleft x_2 : B \vdash N_2 : B_2$, and $\mathscr{D}_3 \triangleleft \vdash V^\bullet : B^\bullet$ then $\mathscr{D}^\bullet \triangleq \texttt{let}\ \mathtt{D}_{B^\bullet} x\ \texttt{be}\ x_1 \otimes x_2\ \texttt{in}\ \mathscr{D}_1^\bullet \otimes \mathscr{D}_2^\bullet$, where $\mathtt{D}_{B^\bullet}$ is the duplicator of $B^\bullet$;

Remark 2. Points 2(d) and 2(e) are well-defined. Indeed, by the $\forall$-laziness condition of LAM (Definition 4), the types $B^\bullet, B_1^\bullet, B_2^\bullet$ in point 2(d) and point 2(e) are closed Π_1. Moreover, since by Remark 1 the derivation $\mathscr{D}_3$ in point 2(e) is η-expanded (hence cut-free), the type $B^\bullet$ is inhabited by the closed and normal linear λ-term $\mathscr{D}_3^\bullet$. So, Theorem 9 ensures that both $\mathtt{E}_{B^\bullet_{3-i}}$ and $\mathtt{D}_{B^\bullet}$ exist.

Theorem 11 (Soundness of $(\_)^\bullet$)**.** *Let $\mathscr{D}$ be a derivation of* LAM*. If $\mathscr{D} \rightsquigarrow \mathscr{D}'$ then $\mathscr{D}^\bullet \to_{\beta\eta}^* \mathscr{D}'^\bullet$.*

Proof. W.l.o.g. it suffices to consider the case where the last rule of $\mathscr{D}$ is the instance of *cut* the $\forall$-lazy cut-elimination rule $\mathscr{D} \rightsquigarrow \mathscr{D}'$ is applied to. We consider the case where $\mathscr{D}$ ends with a *ready* cut $(X, \&\mathrm{R}1)$, for some X, where the left premises of the *cut* is $\mathscr{D}_1 \triangleleft \vdash V : A$ and the premises of &R1 are $\mathscr{D}_2 \triangleleft x_1 : A \vdash N_1 : B_1$, $\mathscr{D}_3 \triangleleft x_2 : A \vdash N_2 : B_2$ and $\mathscr{D}_4 \triangleleft \vdash V' : A$. Since the cut is ready, $\mathscr{D}_1$ must be cut-free, and hence $\mathscr{D}_1^\bullet$ is a closed and normal linear λ-term of closed Π_1 type $A^\bullet$. Therefore, by applying Theorem 9.2, we have: $\texttt{let}\ \mathtt{D}_{A^\bullet} \mathscr{D}_1^\bullet\ \texttt{be}\ x_1 \otimes x_2\ \texttt{in}\ \mathscr{D}_2^\bullet \otimes \mathscr{D}_3^\bullet \to_{\beta\eta}^* \texttt{let}\ \mathscr{D}_1^\bullet \otimes \mathscr{D}_1^\bullet\ \texttt{be}\ x_1 \otimes x_2\ \texttt{in}\ \mathscr{D}_2^\bullet \otimes \mathscr{D}_3^\bullet$, which reduces in a single step to $\mathscr{D}_2^\bullet[\mathscr{D}_1^\bullet/x_1] \otimes \mathscr{D}_3^\bullet[\mathscr{D}_1^\bullet/x_2]$. Therefore, $\mathscr{D}^\bullet \to_{\beta\eta}^* \mathscr{D}'^\bullet$. □

Theorem 12 (Exponential compression for LAM)**.** *Let $\mathscr{D}$ be a derivation in* LAM*. Then, $|\mathscr{D}^\bullet| = \mathscr{O}(2^{|\mathscr{D}|^k})$, for some $k \geq 1$.*

Proof. By structural induction on $\mathscr{D}$. The only interesting case is when $\mathscr{D}$ ends with &R1 with premises $\mathscr{D}_1 \triangleleft x_1 : A \vdash N_1 : B_1$, $\mathscr{D}_2 \triangleleft x_2 : A \vdash N_2 : B_2$ and $\mathscr{D}_3 \triangleleft \vdash V' : A$. By Definition 7, $\mathscr{D}^\bullet = \texttt{let}\ \mathtt{D}_{A^\bullet} x\ \texttt{be}\ x_1 \otimes x_2\ \texttt{in}\ \mathscr{D}_1^\bullet \otimes \mathscr{D}_2^\bullet$. By Remark 1, $\mathscr{D}_3$ is η-expanded, so that $|A| \leq 2 \cdot |\mathscr{D}_3|$ by Proposition 2. Hence, $|\mathtt{D}_A| \in \mathscr{O}(2^{2 \cdot |\mathscr{D}_3|^2})$ by Proposition 10. We apply the induction hypothesis on $\mathscr{D}_1$ and $\mathscr{D}_2$ and conclude. □

6 Conclusion

We introduce LAM, a type assignment system with a weaker version of the additive rules, called *linear additives*. We prove both a strong linear normalization and a restricted cut-elimination result. Also, we present a sound translation from LAM into IMLL_2 and we study its complexity.

A future direction is to find weaker versions of $\oplus$Ri and $\oplus$L that do not affect the complexity of normalization or cut-elimination. This task happens to be much harder than the case of the rules &R and &Li, due to the "classical flavour" of $\oplus$L. Concerning the rule &R, the hidden contractions of assumptions have been managed in LAM by means of the `copy` construct (and some further conditions) to avoid exponential explosions during normalization. The rule $\oplus$L is more subtle, since contraction takes place in the right-hand side of the turnstile, and we could not find how to control it yet.

Linear additives are costless formulations of the standard additives, so they can have fruitful applications in the field of ICC. A possible direction could be to improve the implicit characterization of NP established for STA_+ by Marion et al in [6] (see the Introduction), which requires a *specific* reduction strategy to obtain a soundness result. We could replace the sum rule with a non-deterministic version of the linear additives and prove a *Strong* Non-deterministic Polytime Soundness, so not depending on the choice of the reduction strategy.

References

[1] Henk Barendregt, Wil Dekkers & Richard Statman (2013): *Lambda calculus with types*. Cambridge University Press.

[2] Torben Braüner & Valeria De Paiva (1996): *Cut-elimination for full intuitionistic linear logic*. 10, University of Cambridge, Computer Laboratory.

[3] Gianluca Curzi & Luca Roversi (2019): *A type-assignment of linear erasure and duplication*. arXiv preprint arXiv:1912.12837.

[4] Alejandro Díaz-Caro & Gilles Dowek (2013): *Non determinism through type isomorphism*. arXiv preprint arXiv:1303.7334.

[5] Marco Gaboardi & Simona Ronchi Della Rocca (2009): *From light logics to type assignments: a case study*. Logic Journal of the IGPL 17(5), pp. 499–530.

[6] Marco Gaboardi, Jean-Yves Marion & Simona Ronchi Della Rocca (2008): *Soft linear logic and polynomial complexity classes*. Electronic Notes in Theoretical Computer Science 205, pp. 67–87.

[7] Jean-Yves Girard (2017): *Proof-nets: the parallel syntax for proof-theory*. In: Logic and Algebra, Routledge, pp. 97–124.

[8] Jean-Yves Girard & Yves Lafont (1987): *Linear logic and lazy computation*. In: International Joint Conference on Theory and Practice of Software Development, Springer, pp. 52–66.

[9] J Roger Hindley (1989): *BCK-combinators and linear λ-terms have types*. Theoretical Computer Science 64(1), pp. 97–105.

[10] Ross Horne (2019): *The sub-additives: A proof theory for probabilistic choice extending linear logic*. In: 4th International Conference on Formal Structures for Computation and Deduction (FSCD 2019), Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

[11] Harry G. Mairson (2004): *Linear Lambda Calculus and PTIME-completeness*. J. Funct. Program. 14(6), pp. 623–633, doi:10.1017/S0956796804005131. Available at http://dx.doi.org/10.1017/S0956796804005131.

[12] Harry G. Mairson & Kazushige Terui (2003): *On the Computational Complexity of Cut-Elimination in Linear Logic*. In Carlo Blundo & Cosimo Laneve, editors: Theoretical Computer Science, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 23–36.

[13] Satoshi Matsuoka (2004): *Nondeterministic Linear Logic*. arXiv preprint cs/0410029.

[14] François Maurel (2003): *Nondeterministic light logics and NP-time*. In: International Conference on Typed Lambda Calculi and Applications, Springer, pp. 241–255.