Non-Uniform Polytime Computation in the Infinitary Affine Lambda-Calculus

Damiano Mazza

CNRS, UMR 7030, LIPN, Université Paris 13, Sorbonne Paris Cité
Damiano.Mazza@lipn.univ-paris13.fr

Abstract. We give an implicit, functional characterization of the class of non-uniform polynomial time languages, based on an infinitary affine lambda-calculus and on previously defined bounded-complexity subsystems of linear (or affine) logic. The fact that the characterization is implicit means that the complexity is guaranteed by structural properties of programs rather than explicit resource bounds. As a corollary, we obtain a proof of the (already known) P-completeness of the normalization problem for the affine lambda-calculus which mimics in an interesting way Ladner's P-completeness proof of CIRCUIT VALUE (essentially, the argument giving the Cook-Levin theorem). This suggests that the relationship between affine and usual lambda-calculus is deeply similar to that between Boolean circuits and Turing machines.

1 Introduction

Loosely speaking, the aim of implicit computational complexity is to replace clocks (or other explicit resource bounds) with certificates. For example, if we consider polynomial time computation, the idea is to define a structured programming language whose programs guarantee a polynomial dependence of the runtime on the input by construction, *i.e.*, because they satisfy some syntactic condition, not because their execution is artificially stopped after a polynomial number of steps. At the same time, such a programming language must be expressive enough so that every polynomial time function may be somehow implemented. Notable early examples of such methodology are the work of Bellantoni and Cook [2], Leivant and Marion [9], and Jones [5].

We consider here the question of finding an implicit characterization of non-uniform polynomial time, *i.e.*, the class P/poly. Our approach brings together two lines of work, both based on linear logic. The first is the linear-logical take at implicit computational complexity initiated by Girard [4] and reformulated in the λ-calculus, for example, by Asperti and Roversi [1]. The second is the author's work on the infinitary affine λ-calculus [11], previously considered also by Kfoury [6] and Melliès [12].

For our present purposes, the essence of linear logic is in its resource awareness. Linear (or, more precisely, affine) types describe volatile data, which may be accessed only once. Accordingly, the linear (or affine) functional type $A \multimap B$ describes programs producing an output of type B by using their input of type A exactly (or at most) once. Persistent data is described by the type $!A$, which may be understood as volatile access to a bottomless pile of copies of A, thus obtaining unlimited access to A. The usual functional type $A \to B$ may then be expressed by $!A \multimap B$.

In the λ-calculus, which is the prototypical functional language, affinity takes the form of forbidding duplication, which translates into an extremely simple syntactic restriction: each variable must appear at most once in a term. Our previous work [11] shows how affine λ-terms may approximate usual λ-terms arbitrarily well, in a precise topological sense which is compatible with computation (*i.e.*, reduction is continuous). In the limit, usual λ-terms are recovered by considering infinitary affine terms (thus taking quite literally the above idea of "bottomless pile"). However, the limit process (which is just the completion of a uniform space) introduces a host of infinitary terms which do not correspond to any usual λ-term. The reason is easily explained: to act as a persistent memory cell, a datum of type $!A$ must contain infinitely many *identical* copies of a datum of type A. Without further constraints, the infinitary affine λ-calculus allows memory cells whose content changes arbitrarily with each access. This is the "functional gateway" to non-uniform computation.

The technical contribution of this paper is to "tame" the non-uniformity of the unrestricted calculus $\ell\Lambda_\infty$ of [11] so as to keep it within interesting boundaries, namely those of P/poly. Let us give an informal description of what this means. Using (an adaptation of) the standard λ-calculus encodings of binary strings, we may say that a term t *decides* $L \subseteq \{0,1\}^*$ in $\ell\Lambda_\infty$ if, given $w \in \{0,1\}^*$, $t\underline{w} \to^* \underline{b}$ with $b \in \{0,1\}$ according to whether w belongs to L ($\underline{w}$ is the encoding of w and $\to^*$ is the reduction relation of the calculus). Now, t is generally infinite, but we may define a canonical sequence $\lfloor t \rfloor_n$ of approximations of t, which are finite affine terms such that $\lim \lfloor t \rfloor_n = t$. Intuitively, $\lfloor t \rfloor_n$ behaves like t in which every internal memory cell is limited to at most n accesses. We may then appeal to the continuity of reduction, by which, if we let u_n be the normal form of $\lfloor t \rfloor_n \, \underline{w}$, we have that $\underline{b} = \lim u_n$. But our topology is such that pieces of data like $\underline{b}$ are isolated points, so there exists $m \in \mathbb{N}$ such that $u_n = \underline{b}$ for all $n \geq m$. This means that a finite approximation of t suffices to compute $t\underline{w}$. The size of $\lfloor t \rfloor_m$ is linear in m, so the question is: How big is m? Can we relate it to $|w|$? If we can make m be polynomial in $|w|$, the language decided by t is in P/poly: we may use the $\lfloor t \rfloor_m$ as (polynomial) advice and then normalize $\lfloor t \rfloor_m \, \underline{w}$, which may be done in polynomial time in $|w|$ because it is a finite affine term.

There exist several λ-calculus characterizations of P based on linear logic (most notably Girard's [4] and Lafont's [8]) and the naive idea to polynomially bound m would be to reuse the recipes given therein. However, non-uniformity in the λ-calculus is extremely subtle and the approach "take your favorite λ-calculus characterization of P and add non-uniformity" does not necessarily yield P/poly. The most surprising aspect is that polytime non-uniformity seems to refuse the logical principle of contraction (expressed by the formula $!A \multimap !A \otimes !A$): in its presence, m may be exponentially big and we may therefore decide any language (an intuitive explanation is given below). This rules out Girard's approach [4]. Lafont's system [8] does not use contraction but appears to have the opposite problem: we are currently unaware of whether the expressiveness of its non-uniform version reaches P/poly.

The key to our solution is a new structural constraint on terms, which we call *parsimony*. In $\ell\Lambda_\infty$, affinity is enforced by giving a unique integer index to each occurrence of non-linear variable x: intuitively, x_i means "access to the i-th copy of the datum contained in x". In this setting, contraction (corresponding to duplication) is implemented

using "Hilbert's hotel": from an infinite family $(x_i)_{i\in\mathbb{N}}$ representing an argument of type $!A$, we make two infinite families, *e.g.* x_{2i} and x_{2i+1}. Iterating this n times, we obtain a family whose first element is $x_{O(2^n)}$, causing the exponential growth rate of m mentioned above. A very high level description of parsimony is that, when such a reallocation of a family of occurrences is performed, the resulting families may not "waste" indices: each of them contains either finitely many x_i (*i.e.*, it is finite), or almost all of them (*i.e.*, it is co-finite). Parsimony therefore refuses contraction, which necessarily produces infinite co-infinite families. Instead, it allows an asymmetric form of contraction, also known as absorption, expressed by the formula $!A \multimap !A \otimes A$.

Parsimony is coupled with *stratification*, which is a staple of Girard's work [4]. Stratification partitions a program into rigid levels which may not interact and, very roughly speaking, forbids the self-reference that makes the λ-calculus Turing powerful. Alone, it guarantees termination (in elementary time, in the uniform case). Without it, parsimonious terms may diverge and the question of bounding m may not make sense.

For brevity, most of the results are given here without proof. An extended version of this paper, containing the missing proofs, is available on the author's web page.

2 The Affine Lambda-Calculus

Pre-terms. We fix two denumerably infinite disjoint sets of *linear variables*, ranged over by a, b, c, and *non-linear variables*, ranged over by x, y, z. *Patterns* and *pre-terms* are generated by the grammar

$$\mathsf{p}, \mathsf{q} ::= a \mid x \mid \mathsf{p} \otimes \mathsf{q}, \qquad t, u ::= \bot \mid a \mid x_i \mid \lambda\mathsf{p}.t \mid tu \mid t \otimes u \mid \mathbf{u},$$

where $i \in \mathbb{N}$ and $\mathbf{u}$, which we refer to as a *box*, is a finite sequence of pre-terms, which for convenience we identify with a function from $\mathbb{N}$ to pre-terms almost everywhere equal to $\bot$. We also use the explicit notation $\langle \mathbf{u}(0), \dots, \mathbf{u}(n-1)\rangle$, in which we imply that $\mathbf{u}(i) = \bot$ for all $i \geq n$. If a variable a or x appears in a pattern p, we write $a \in \mathsf{p}$ or $x \in \mathsf{p}$. We require that, in $\mathsf{p} \otimes \mathsf{q}$, a variable cannot appear both in p and q. Free and bound variables are defined as customary, the only point worth mentioning is that if $x \in \mathsf{p}$, then *all* occurrences of the form x_i are bound in $\lambda\mathsf{p}.t$. As usual, we identify two pre-terms if they only differ in the names of their bound variables (α-equivalence).

Shallow contexts and *contexts* are defined by the following grammar:

$$S ::= \bullet \mid \lambda\mathsf{p}.S \mid St \mid tS \mid S \otimes t \mid t \otimes S \qquad C ::= S \mid \langle u_0, \dots, C, \dots, u_n\rangle,$$

where $t, u_1, \dots, u_n$ are arbitrary pre-terms. As usual, we denote by $C[t]$ the term obtained by substituting t to $\bullet$ in the context C. We say that u is a *subterm* of t, and we write $u \sqsubseteq t$, if there exists a context C such that $t = C[u]$.

We will find useful to see pre-terms as labelled trees. Intuitively, this is done in the obvious way: a pre-term t induces a function $t : \mathbb{N}^* \longrightarrow \Sigma$, where $\Sigma := \{\bot, a, x_i, \lambda\mathsf{p}, @, \otimes, !\}$ and $\mathbb{N}^*$ is the set of finite sequences of natural numbers, ranged over by α and with the empty sequence denoted by ϵ. Sequences of arbitrary integers are needed because of boxes. The symbol $t[\alpha]$ denotes the kind of constructor at position α in t: $\bot$, a variable (a or x_i), an abstraction (λ), an application ($@$), a tensor ($\otimes$) or a box ($!$). Also, when $u \sqsubseteq t$, we say that u occurs at position α if u is rooted at position α in t. Note that, when the position α does not exist in t, we assume that $t[\alpha] = \bot$.

Terms and reduction. A *term* is a pre-term t such that:

- every linear variable and occurrence of non-linear variable appears at most once in t (*i.e.*, if $x_i, x_j \sqsubseteq t$ occur at different positions, then $i \neq j$);
- whenever $\mathbf{u} \sqsubseteq t$, the free variables of $\mathbf{u}$ are all non-linear.

We denote by $\ell\Lambda$ the set of all terms.

We say that a term t *matches* a pattern p, and write $t \between \mathsf{p}$, when: $t \between a$ for all t; $t \between x$ just if $t = \mathbf{u}$; and if $t \between \mathsf{p}$ and $u \between \mathsf{q}$, then $t \otimes u \between \mathsf{p} \otimes \mathsf{q}$. In case $u \between \mathsf{p}$, we define the *substitution* $t[u/\mathsf{p}]$ as follows: $t[u/a]$ is defined as usual; $t[\mathbf{u}/x]$ is obtained by substituting $\mathbf{u}(i)$ to the unique free occurrence x_i in t, for all $i \in \mathbb{N}$; and $t[u_1 \otimes u_2/\mathsf{p}_1 \otimes \mathsf{p}_2] := t[u_1/\mathsf{p}_1][u_2/\mathsf{p}_2]$.

We define $\to_{\mathrm{s}}$ (*shallow reduction*) and $\rightsquigarrow$ (*unboxing reduction*) as the smallest binary relations $\ell\Lambda$ such that:

- $(\lambda\mathsf{p}.t)u \to_{\mathrm{s}} t[u/\mathsf{p}]$ whenever $u \between \mathsf{p}$;
- if $t \to_{\mathrm{s}} t'$, then $S[t] \to_{\mathrm{s}} S[t']$ for every shallow context S;
- $\mathbf{u} \rightsquigarrow \mathbf{u}(0)$ for every box $\mathbf{u}$.

Note that the unboxing step *is not closed under any context*: it applies only to a term which is itself a box, "extracting" its first subterm. Usually, one allows reduction inside boxes. The non-standard definition adopted here is technically simpler for our purposes.

A *redex* is a term of the form $(\lambda\mathsf{p}.t)u$ and such that $u \between \mathsf{p}$. Each $\to_{\mathrm{s}}$ step is obviously associated with a redex, which we say is *fired* by the reduction step.

Reduction is defined by $\to := \to_{\mathrm{s}} \cup \rightsquigarrow$. If $\dashrightarrow$ is any reduction relation, we denote by $\dashrightarrow^*$ its reflexive-transitive closure and by $\dashrightarrow^l$ the composition of exactly $l \in \mathbb{N}$ steps of $\dashrightarrow$. Reduction is obviously confluent and strongly normalizing. Both points are a consequence of the affinity conditions: no redex is duplicated and the size of terms (*i.e.*, the number of symbols) strictly decreases with reduction.

Uniform structure. As noted above, $\ell\Lambda$ may be seen as a subset of the set of functions $\mathbb{N}^* \longrightarrow \Sigma$. If we equip Σ with the discrete uniformity,[1] then we may endow terms with the *uniformity of uniform convergence on finitely branching trees* (as subsets of $\mathbb{N}^*$). More explicitly, let $(\mathbb{N}^{\mathbb{N}}, \leq, \vee)$ be the join-semilattice of infinite sequences of natural numbers, ranged over by ξ, with the pointwise ordering. We denote by ξ_i the ith element of the sequence ξ and by $n \cdot \xi$ a sequence whose first element is n. We also write $\alpha \prec \xi$ if $\alpha \in \mathbb{N}^*$ is a prefix of ξ. The uniformity of uniform convergence on finitely branching trees is generated by the following basis of entourages, for $\xi \in \mathbb{N}^{\mathbb{N}}$:

$$\mathcal{U}_\xi := \{(t, t') \in \ell\Lambda \times \ell\Lambda \mid \forall \alpha \prec \xi' \leq \xi,\, t[\alpha] = t'[\alpha]\}.$$

The intuition is the following: with each $\xi \in \mathbb{N}^{\mathbb{N}}$ we associate an infinite but finitely branching tree τ_ξ, such that every node at depth i of τ_ξ has exactly $\xi_i + 1$ siblings; then, two terms are ξ-close (*i.e.*, belong to $\mathcal{U}_\xi$) if they coincide on τ_ξ. A basis of open neighborhoods of t for the induced topology is $\mathcal{U}_\xi(t) := \{t' \in \ell\Lambda \mid \forall \alpha \prec \xi' \leq \xi,\, t'[\alpha] = t[\alpha]\}$, for $\xi \in \mathbb{N}^{\mathbb{N}}$, *i.e.*, the terms coinciding with t on τ_ξ. Observe that the

[1] The word "uniformity" takes here its standard topological sense [3], which is essentially a generalization of the concept of metric still allowing one to speak of Cauchy sequences. This, unfortunately, is completely unrelated to the equally standard meaning more common in computer science (and employed, in particular, in the title of this paper).

local basis is uncountable. In fact, one can show that no countable local basis exists for this topology, so the space is not metrizable.

The fundamental result concerning the uniform structure on $\ell\Lambda$ is the Cauchy-continuity of reduction. We remind that a function between uniform spaces is Cauchy-continuous when it preserves Cauchy nets. In particular, it is continuous. Given $\alpha \in \mathbb{N}^*$, we define $R_\alpha : \ell\Lambda \longrightarrow \ell\Lambda$ by $R_\alpha(t) := t'$ if $t \to_{\mathrm{s}} t'$ by reducing a redex at position α, or $R_\alpha(t) := t$ if no such reduction applies. Similarly, we define $U(t) := t'$ if $t \rightsquigarrow t'$ and $U(t) := t$ otherwise.

Proposition 1 (Cauchy-continuity of reduction). *For all $\alpha \in \mathbb{N}^*$, R_α is Cauchy-continuous, and so is U.*

Infinitary terms and approximations. Intuitively, Cauchy sequences in $\ell\Lambda$ are made of terms that coincide on wider and wider trees, such as $\Delta_n := \lambda x.x_0\langle x_1, \ldots, x_n\rangle$. Note, however, that $(\Delta_n)_{n\in\mathbb{N}}$ has no limit in $\ell\Lambda$, showing that the space is not complete. We denote by $\ell\Lambda_\infty$ the completion of $\ell\Lambda$. From now on, the word "term" will refer to an element of $\ell\Lambda_\infty$, whereas the elements of $\ell\Lambda$ will be called *finite terms*.

Indeed, the elements of $\ell\Lambda_\infty$ may be seen as infinitary terms. They still verify the affinity constraints and we apply to them the same terminology and notations as for finite terms (free and bound variable, subterm relation $\sqsubseteq$, etc.). A typical example of infinitary term is $\Delta := \lambda x.x_0\langle x_1, x_2, x_3, \ldots\rangle$, which is the limit of $(\Delta_n)_{n\in\mathbb{N}}$. Apart from being infinitely wide, terms of $\ell\Lambda_\infty$ may also have infinite height, such as $\langle \lambda x.x_0, \lambda x.x_0x_1, \lambda x.x_0(x_1x_2), \ldots\rangle$. Nevertheless, one may show that they are always well-founded.

In fact, we will mostly be interested in infinitary terms of finite height, like Δ above, but knowing that all terms are well-founded is quite useful because it allows reasoning by induction. For example, given $t \in \ell\Lambda_\infty$, we may define a default sequence of finite terms converging to t, its *n-th approximations* $\lfloor t\rfloor_n$, as follows: $\lfloor\bot\rfloor_n := \bot$; $\lfloor a\rfloor_n := a$, $\lfloor x_i\rfloor_n := x_i$; $\lfloor \lambda \mathsf{p}.t\rfloor_n := \lambda \mathsf{p}.\lfloor t\rfloor_n$; $\lfloor tu\rfloor_n = \lfloor t\rfloor_n\lfloor u\rfloor_n$; $\lfloor t \otimes u\rfloor_n := \lfloor t\rfloor_n \otimes \lfloor u\rfloor_n$; $\lfloor \mathbf{u}\rfloor_n := \langle \lfloor \mathbf{u}(0)\rfloor_n, \ldots, \lfloor \mathbf{u}(n)\rfloor_n\rangle$. The definition makes sense because of well-foundedness: technically, what we are saying is that $\lfloor\cdot\rfloor_n$ is a function satisfying the above equalities. One may prove by well-founded induction that such a function is well defined and unique.

Reduction may be defined for terms of $\ell\Lambda_\infty$ in the obvious way, using substitution (which may now require infinitely many substitutions in the case $t[\mathbf{u}/x]$). Nevertheless, we stress that, from a strictly technical point of view, by Proposition 1 we do not need an explicit definition: indeed, Cauchy-continuity is exactly the property guaranteeing that a function on a uniform space uniquely extends to its completion.

It is worthwhile noting that reduction in $\ell\Lambda_\infty$, although still strongly confluent (a topological proof is given in [11]), is no longer normalizing. In fact, if we set $\Omega := \Delta\langle\Delta, \Delta, \Delta, \ldots\rangle$, with Δ as above, we have $\Omega \to_{\mathrm{s}} \Omega$.

Correspondence with a non-linear λ-calculus. For programming purposes, it will be convenient to consider a more standard, non-linear λ-calculus. We use the same sets of linear and non-linear variables as $\ell\Lambda$ (ranged over by a and x, respectively) but for each non-linear variable we also consider a corresponding *§-variable* denoted by $x^{\S}$. Patterns

p are defined as in $\ell\Lambda$, with the addition of §-variables. Terms and *reduction contexts* are defined as follows:

$$M, N ::= a \mid x^{\S} \mid x \mid \lambda\mathsf{p}.M \mid MN \mid M \otimes N \mid \S M \mid !M,$$
$$R ::= \bullet \mid \lambda\mathsf{p}.R \mid RM \mid MR \mid R \otimes M \mid M \otimes R.$$

The affinity constraint is on linear variables and §-variables, which must occur at most once. Non-linear variables may occur arbitrarily many times. Also, the simultaneous presence of $x^{\S}$ and x in a term is excluded. In $!M$ (resp. $\S M$), called *!-box* (resp. *§-box*), we require all free variables to be non-linear (resp. to be non-linear or §-variables). The set of terms thus defined is denoted by Λ.

Matching between terms and patterns is defined as in $\ell\Lambda$, with both $\S M \between x^{\S}$ and $!M \between x^{\S}$, and $!M \between x$. Substitution is also extended in the obvious way: in $M[\S N/x^{\S}]$, $M[!N/x^{\S}]$ and $M[!N/x]$, N is substituted to all free occurrences of x or $x^{\S}$ in M (so several copies of N may be needed). Reduction, denoted by $\to_\beta$, is the union of $\to_{\beta 0}$ and $\rightsquigarrow_\beta$, which are the smallest binary relations on Λ defined as follows:

- $(\lambda\mathsf{p}.M)N \to_{\beta 0} M[N/\mathsf{p}]$ whenever $N \between \mathsf{p}$;
- if $M \to_{\beta 0} M'$, then $R[M] \to_{\beta 0} R[M']$;
- $\S M \rightsquigarrow_\beta M$.

We may represent terms of Λ in $\ell\Lambda_\infty$, as follows. Let $\iota : \mathbb{N}^* \longrightarrow \mathbb{N} \setminus \{0\}$ be an injection. If p is a pattern of Λ, we denote by p^- the pattern of $\ell\Lambda_\infty$ obtained by replacing every $x^{\S}$ with x. Given $\alpha \in \mathbb{N}^*$, we define $[\![M]\!]^\iota_\alpha$ by induction on M: $[\![a]\!]^\iota_\alpha := a$; $[\![x^{\S}]\!]^\iota_\alpha := x_0$; $[\![x]\!]^\iota_\alpha := x_{\iota(\alpha)}$; $[\![\lambda\mathsf{p}.M]\!]^\iota_\alpha := \lambda\mathsf{p}^-.[\![M]\!]^\iota_\alpha$; $[\![MN]\!]^\iota_\alpha := [\![M]\!]^\iota_{0\cdot\alpha}[\![N]\!]^\iota_{1\cdot\alpha}$; $[\![M \otimes N]\!]^\iota_\alpha := [\![M]\!]^\iota_{0\cdot\alpha} \otimes [\![N]\!]^\iota_{1\cdot\alpha}$; $[\![\S M]\!]^\iota_\alpha := \langle [\![M]\!]^\iota_\alpha \rangle$; $[\![!M]\!]^\iota_\alpha := \langle [\![M]\!]^\iota_{0\cdot\alpha}, [\![M]\!]^\iota_{1\cdot\alpha}, [\![M]\!]^\iota_{2\cdot\alpha}, \ldots \rangle$.

We say that $t \in \ell\Lambda_\infty$ *represents* $M \in \Lambda$, and we write $t \blacktriangleleft M$, if there exist α and ι as above such that $[\![M]\!]^\iota_\alpha = t$.

Proposition 2. *Let $t \blacktriangleleft M$ and $M \to_\beta M'$, then $t \to t' \blacktriangleleft M'$.*

Proof. Essentially, this is one direction of the isomorphism of [11]. □

The converse of Proposition 2 fails: let $i > 0$, $t := (\lambda x.x_i)\langle I \rangle$ and $M := (\lambda x.x)\S I$; we have $t \blacktriangleleft M$, yet M is a normal form whereas $t \to_{\mathrm{s}} \bot$. The perfect correspondence of [11] could be recovered by modifying the syntax of $\ell\Lambda$ (and $\ell\Lambda_\infty$) but this is inessential for our purposes.

3 The Parsimonious Stratified Calculus

Stratification. The *box-depth* of a specific occurrence of subterm u in $t \in \ell\Lambda_\infty$, denoted by $\mathrm{d}_u(t)$, is the number of nested boxes of t in which u is contained. For instance, $\mathrm{d}_u(t) = 0$ iff $t = S[u]$ for some shallow context S (this is the reason behind the terminology "shallow"). The *box-depth* of t, denoted by $\mathrm{d}(t)$, is the supremum of the box-depths of its subterms. It is always finite if the height of t is finite, which will be the case of interest to us.

The *binder-relative box-depth* of an occurrence x_i appearing (free or bound) in t, denoted by $\mathrm{rd}_{x_i}(t)$, is: $\mathrm{d}_{x_i}(t)$ if x is free in t; if x is bound, then there exists $\lambda\mathsf{p}.u \sqsubseteq t$

such that $x \in \mathsf{p}$ and x_i appears in u, in which case $\mathrm{rd}_{x_i}(t) := \mathrm{d}_{x_i}(t) - \mathrm{d}_{\lambda \mathsf{p}.u}(t)$ (which is easily seen to be equal to $\mathrm{d}_{x_i}(u)$).

Definition 1 (Stratified term). *A term t is* stratified *if, for all $x_i \sqsubseteq t$, $\mathrm{rd}_{x_i}(t) = 1$. We denote by $\ell\Lambda^{\mathrm{s}}_{\infty}$ the set of all stratified terms and by $\ell\Lambda^{\mathrm{s0}}_{\infty}$ the set of stratified terms having no free non-linear variable.*

Proposition 3 (Reduction and stratification). *1) If $t \in \ell\Lambda^{\mathrm{s}}_{\infty}$ and $t \to_{\mathrm{s}} t'$, then $t' \in \ell\Lambda^{\mathrm{s}}_{\infty}$ and $\mathrm{d}(t') \leq \mathrm{d}(t)$; 2) moreover, if $t \in \ell\Lambda^{\mathrm{s0}}_{\infty}$ and $t \rightsquigarrow t'$, then $t' \in \ell\Lambda^{\mathrm{s0}}_{\infty}$ and $\mathrm{d}(t') = \mathrm{d}(t) - 1$; 3) every $t \in \ell\Lambda^{\mathrm{s0}}_{\infty}$ of finite height is strongly normalizing.*

Parsimony. Given $m, n \in \mathbb{N}$ and $t, t' \in \ell\Lambda_{\infty}$, we write:

- $t \sim_n t'$ if t and t' differ only in the indices of the *bound occurrences* of their non-linear variables, and the indices vary by at most n, *i.e.*, if x_i in t corresponds to x_j in t', then $|i - j| \leq n$.
- $t :<^m t'$ if t' is obtained from t by replacing *every free occurrence* of non-linear variable x_i with x_{i+m}.
- $t :\lesssim^m_n t'$ iff there is u s.t. $t \sim_n u :<^m t'$ iff there is u' s.t. $t :<^m u' \sim_n t'$ (the latter two conditions are equivalent because the relations act on disjoint occurrences).

Definition 2 (Parsimonious term). *A box $\mathbf{u}$ is* parsimonious *if there exist $c, k \in \mathbb{N}$ such that, for all $i \geq j \geq k$, $\mathbf{u}(i) :\lesssim_c^{i-j} \mathbf{u}(j)$. The smallest $k \in \mathbb{N}$ realizing the above definition is called the* non-uniformity factor *of $\mathbf{u}$, or* n.u. factor *for short. A term $t \in \ell\Lambda_{\infty}$ is* parsimonious *if all of its boxes are. We denote by $\ell\Lambda^{\mathrm{p}}_{\infty}$ the set of all parsimonious terms.*

Note that, unlike stratification, parsimony is inherited by subterms. Another difference is that every finite term is parsimonious. In fact, intuitively, the structure of a parsimonious term admits a finite description: in every box $\mathbf{u}$, all $\mathbf{u}(i)$ ultimately have the same "shape". Nonetheless, the term itself may not be finitely describable at all. For instance, if $I_i := \lambda x.x_i$ with $i \in \{0, 1\}$, the term $\langle I_{i_0}, I_{i_1}, I_{i_2}, \ldots \rangle$ is parsimonious (with n.u. factor 0) regardless of the sequence i_n (this will be a key ingredient for encoding infinite binary words). Moreover, we have:

Lemma 1. *1. Every parsimonious term has finite height;*

2. let $\mathbf{u}$ be parsimonious of n.u. factor k and let x have infinitely many free occurrences in $\mathbf{u}$. Then, for all $h \in \mathbb{N}$, there is exactly one free occurrence x_{j_h} in $\mathbf{u}(k+h)$ and $j_h = j_0 + h$.

Proposition 4 (Reduction and parsimony). *If $t \in \ell\Lambda^{\mathrm{p}}_{\infty}$ and $t \to t'$, then $t \in \ell\Lambda^{\mathrm{p}}_{\infty}$.*

Bounds on parsimonious stratified terms. By Propositions 3 and 4, parsimonious stratified terms form a well defined calculus with respect to the reduction relation $\to$.

Definition 3 (The calculus $\ell\Lambda^{\mathrm{ps0}}_{\infty}$). *We define $\ell\Lambda^{\mathrm{ps0}}_{\infty} := \ell\Lambda^{\mathrm{s0}}_{\infty} \cap \ell\Lambda^{\mathrm{p}}_{\infty}$.*

In what follows, δ_d is the Kronecker symbol, equal to 1 if $d = 0$ and to 0 otherwise. Let $t \in \ell\Lambda^{\mathrm{p}}_{\infty}$. The *size of t at box-depth d* is defined as follows. First, we define the size of a pattern by setting $|a| := |x| := 1$, and $|\mathsf{p} \otimes \mathsf{q}| := 1 + |\mathsf{p}| + |\mathsf{q}|$. Then, we set

$|\bot|_d := |a|_d := \delta_d$; $|x_i|_d := (1+i)\delta_d$; $|\lambda \mathsf{p}.t|_d := \delta_d|\mathsf{p}| + |t|_d$; $|tu|_d := |t \otimes u|_d := \delta_d + |t|_d + |u|_d$; $|\mathbf{u}|_0 := 1$ and $|\mathbf{u}|_{d+1} := \sum_{i=0}^{k} |\mathbf{u}(i)|_d$, where k is the n.u. factor of $\mathbf{u}$. Finally, we define the *size* by $|t| = \sum_{j=0}^{\mathrm{d}(t)} |t|_j$.

Lemma 2. *Let $t \in \ell\Lambda^{\mathrm{ps0}}_{\infty}$ and let $t \to^*_{\mathrm{s}} t'$. Then, for all $j \geq 1$, $|t'|_j \leq |t|_1 |t|_j$.*

In what follows, we will make use of the n-th approximations of a term, defined at page 5. We also introduce the following notation: given two terms t, t' and $n \in \mathbb{N}$, $t \backsimeq_n t'$ just if $\lfloor t \rfloor_n = \lfloor t' \rfloor_n$. It is obviously a family of equivalence relations.

Definition 4. *Let $t \in \ell\Lambda_{\infty}$ and let x appear in t. Given $n \in \mathbb{N}$, we define $\mathrm{v}_{x,t}(n) := \sup\{i \in \mathbb{N} \mid x_i \text{ appears in } \lfloor t \rfloor_n\}$.*

Let now $t \to t'$ and $n \in \mathbb{N}$. We define $\mathrm{m}_{t \to t'}(n) \in \mathbb{N}$ as follows. If $t \rightsquigarrow t'$, $\mathrm{m}_{t \to t'}(n) := n$. Otherwise, $t \to_{\mathrm{s}} t'$ by firing a redex $(\lambda \mathsf{p}.u)v$ such that $x^1, \ldots, x^p$ are the non-linear variables appearing in p. Then, we set $\mathrm{m}_{t \to t'}(n) := \max(n, \sup\{\mathrm{v}_{x^1,t}(n), \ldots, \mathrm{v}_{x^p,t}(n)\})$.

Lemma 3. *Let $t \in \ell\Lambda_{\infty}$ and let $t \to t'$. Then, for all $n \in \mathbb{N}$ and for all $u \in \ell\Lambda_{\infty}$, $u \backsimeq_{\mathrm{m}_{t \to t'}(n)} t$ implies $u \to u'$ such that $u' \backsimeq_n t'$.*

Lemma 4. *Let $t \in \ell\Lambda^{\mathrm{p}}_{\infty}$ and let x be a bound variable of t. Then, $\mathrm{v}_{x,t}(n) \leq |t| + n$, for all $n \in \mathbb{N}$.*

Lemma 5. *Let $t \in \ell\Lambda^{\mathrm{ps0}}_{\infty}$ and let $t \to^l t'$. Then:*

1. *$|t'| \leq |t|^{2^{\mathrm{d}(t)}}$ and $l \leq (\mathrm{d}(t)+1)|t|^{2^{\mathrm{d}(t)}} + \mathrm{d}(t)$;*
2. *for all $n \in \mathbb{N}$, there is $m \leq n + l|t|^{2^{\mathrm{d}(t)}}$ such that $\lfloor t \rfloor_m \to^l t'' \backsimeq_n t'$.*

Proof. Point 1 is proved by by induction on the number of $\rightsquigarrow$ steps in the reduction, in a similar way as [4, 1]. In synthesis, Lemma 2 and Proposition 3 give $\mathrm{d}(t)+1$ "rounds" each squaring the size. The result follows.

For point 2, we reason by induction on l. The case $l = 0$ is trivial, so let $t \to^{l'} u \to t'$. The induction hypothesis gives us, for all $m_1 \in \mathbb{N}$, $m(m_1) \leq m_1 + l'|t|^{2^{\mathrm{d}(t)}}$ such that $\lfloor t \rfloor_{m(m_1)} \to^{l'} u' \backsimeq_{m_1} u$. We then apply Lemma 3 to obtain $\lfloor t \rfloor_{m(\mathrm{m}_{u \to t'}(n))} \to^{l'} u' \to t'' \backsimeq_n t'$. To conclude, we need to bound $m(\mathrm{m}_{u \to t'}(n)) \leq \mathrm{m}_{u \to t'}(n) + l'|t|^{2^{\mathrm{d}(t)}}$. By definition, $\mathrm{m}_{u \to t'}(n)$ is either n, in which case we are done, or of the form $\mathrm{v}_{x,u}(n)$ for some x appearing in u. By Lemma 4, $\mathrm{v}_{x,u}(n) \leq |u| + n$. Now, using the size bound of point 1 (which does not depend on l'), we have $|u| \leq |t|^{2^{\mathrm{d}(t)}}$, which allows us to conclude. □

Parsimony and stratification in the non-linear calculus. In Λ, the concepts of box-depth and binder-relative box-depth are defined just as in $\ell\Lambda$, with !- and §-boxes both counting as boxes.

Definition 5 (The uniform calculus Λ^{ps0}). *We denote by Λ^{ps0} the subset of terms of Λ satisfying the following requirements, which correspond to those of $\ell\Lambda^{\mathrm{ps0}}_{\infty}$: 1) occurrences of non-linear variables have binder-relative box-depth 1; 2) every non-linear variable appears in at most one subterm of the form $!M$, in which case it occurs exactly once in M; 3) no §-variable or non-linear variable appears free.*

Proposition 5. *1. If $t \blacktriangleleft M$, then $M \in \Lambda^{\mathrm{ps0}}$ iff $t \in \ell\Lambda^{\mathrm{ps0}}_{\infty}$;*
2. if $M \in \Lambda^{\mathrm{ps0}}$ and $M \to_\beta M'$, then $M' \in \Lambda^{\mathrm{ps0}}$.

Proof. Point 1 is an immediate consequence of the definitions. Point 2 easily follows from point 1, modulo Propositions 3 and 4. □

4 A Characterization of P/poly

Representing basic data and languages. We consider the usual Church encodings of Booleans and binary strings (the members of $\mathbb{W} := \{0,1\}^*$), adapted to $\ell\Lambda^{\mathrm{ps0}}_{\infty}$. For the Booleans, we set $\mathtt{tt} := \lambda a.\lambda b.a$ and $\mathtt{ff} := \lambda a.\lambda b.b$. Given $w = w_1 \cdots w_n \in \mathbb{W}$, we say that a term t is a *Church encoding* of w if $t = \lambda s^0.\lambda s^1.\langle \lambda a.s^{w_1}_{i_1}(\ldots s^{w_n}_{i_n} a \ldots)\rangle$, with $i_1, \ldots, i_n \in \mathbb{N}$ arbitrary as long as affinity is assured. For example, the encodings of 010 are all of the form $\lambda s^0.\lambda s^1.\langle \lambda a.s^0_{i_1}(s^1_j(s^0_{i_2} a))\rangle$, with $i_1 \neq i_2$. We denote by $\underline{w}$ a generic Church encoding of w. Observe that, by choosing the indices as small as possible, every $w \in \mathbb{W}$ admits a Church encoding such that $|\underline{w}| = O(|w|^2)$ (where $|w|$ is the length of the string w). On the other hand, $\mathrm{d}(\underline{w}) = 1$ independently of w and of the Church encoding.

Definition 6 (The class C_∞). *We say that a language $L \subseteq \mathbb{W}$ is* decidable *in $\ell\Lambda^{\mathrm{ps0}}_{\infty}$ ($L \in \mathsf{C}_\infty$) if there exists $t \in \ell\Lambda^{\mathrm{ps0}}_{\infty}$ such that, for all $w \in \mathbb{W}$ and for any one of its Church encodings $\underline{w}$, $t\underline{w} \to^* \mathtt{tt}$ if $w \in L$, and $t\underline{w} \to^* \mathtt{ff}$ otherwise.*

Uniform programming. A good deal of the expressive power of $\ell\Lambda^{\mathrm{ps0}}_{\infty}$ may be shown using the more standard calculus Λ^{ps0}. This is especially convenient because Λ^{ps0} may be provided with a typing discipline which greatly facilitates programming.

The types are second order intuitionistic linear logic formulas, generated by $A, B ::= X \mid A \multimap B \mid A \otimes B \mid \S A \mid !A \mid \forall X.A$, with X ranging over propositional variables. The usual conventions for parentheses are applied ($\multimap$ associates to the right). The typing rules are a decoration of the sequent calculus for a subsystem of intuitionistic linear logic. Typing judgments are of the form $\Gamma \vdash M : A$, where Γ is a finite list of variable assignments of the form $\mathsf{p} : A$. In case $\mathsf{p} = x^{\S}$ (resp. $\mathsf{p} = x$), we require that $A = \S B$ (resp. $A = !B$). The rules are as follows:

$$\frac{}{a : A \vdash a : A}\ \mathsf{ax} \qquad \frac{\Gamma \vdash N : A \quad \Delta, \mathsf{p} : A \vdash M : C}{\Gamma, \Delta \vdash \mathsf{let}\ \mathsf{p} = N\ \mathsf{in}\ M : C}\ \mathsf{cut}$$

$$\frac{\Gamma \vdash M : C}{\Gamma, \mathsf{p} : A \vdash M : C}\ \mathsf{weak} \qquad \frac{\Gamma, x : !A, y^{\S} : \S A \vdash M : C}{\Gamma, x : !A \vdash M[x/y^{\S}] : C}\ \mathsf{asym\ cntr}$$

$$\frac{\Gamma, \mathsf{p} : A \vdash M : B}{\Gamma \vdash \lambda \mathsf{p}.M : A \multimap B}\ \multimap\mathsf{R} \qquad \frac{\Gamma \vdash N : A \quad \Delta, \mathsf{p} : B \vdash M : C}{\Gamma, \Delta, a : A \multimap B \vdash \mathsf{let}\ \mathsf{p} = aN\ \mathsf{in}\ M : C}\ \multimap\mathsf{L}$$

$$\frac{\Gamma \vdash M : A \quad \Delta \vdash N : B}{\Gamma, \Delta \vdash M \otimes N : A \otimes B}\ \otimes\mathsf{R} \qquad \frac{\Gamma, \mathsf{p} : A, \mathsf{q} : B \vdash M : C}{\Gamma, \mathsf{p} \otimes \mathsf{q} : A \otimes B \vdash M : C}\ \otimes\mathsf{L}$$

$$\frac{\overrightarrow{\mathsf{p}} : \overrightarrow{B} \vdash M : A}{\overrightarrow{x}^{\S} : \S\overrightarrow{B} \vdash \S\mathsf{let}\ \overrightarrow{\mathsf{p}} = \overrightarrow{x}^{\S}\ \mathsf{in}\ M : \S A}\ \S \qquad \frac{\overrightarrow{\mathsf{p}} : \overrightarrow{B} \vdash M : A}{\overrightarrow{x} : !\overrightarrow{B} \vdash !\mathsf{let}\ \overrightarrow{\mathsf{p}} = \overrightarrow{x}\ \mathsf{in}\ M : !A}\ !$$

$$\frac{\Gamma \vdash M : A}{\Gamma \vdash M : \forall X.A}\ \forall \mathsf{R}\,(X \notin \mathsf{free}\Gamma) \qquad \frac{\Gamma, \mathsf{p} : A \vdash M : C}{\Gamma, \mathsf{p} : \forall X.A \vdash M : C}\ \forall \mathsf{L}$$

We used the following notational conventions: in the rules cut and $\multimap\,\mathsf{L}$, the notation $\mathsf{let}\,\mathsf{p} = N\,\mathsf{in}\,M$ stands for $M[N/a]$ in case $\mathsf{p} = a$ or $(\lambda \mathsf{p}.M)N$ otherwise (the obvious n-ary generalization of this notation is used in the § and ! rules); in rule $\mathsf{asym\ cntr}$, the substitution $M[x/y^{\S}]$ simply means that the unique occurrence of $y^{\S}$ in M is replaced by x (of which there may already be occurrences); in the § and ! rules, $\overrightarrow{\mathsf{p}} : \overrightarrow{B}$ means that the context is of the form $\mathsf{p}_1 : B_1, \ldots, \mathsf{p}_n : B_n$ and, in the conclusion, $\overrightarrow{x}^{\S} : \S\overrightarrow{B}$ (resp. $\overrightarrow{x} : !\overrightarrow{B}$) means that every $\mathsf{p}_i : B_i$ is replaced by $x^{i\S} : \S B_i$ (resp. $x^i : !B_i$).

The reader may check that if $\mathsf{p}_1 : A_1, \ldots, \mathsf{p}_n : A_n \vdash M : C$ is derivable, then $\lambda \mathsf{p}_1 \ldots \lambda \mathsf{p}_n.M \in \Lambda^{\mathrm{ps0}}$. The system enjoys subject reduction with respect to $\to_{\beta 0}$ but not $\rightsquigarrow_\beta$. This failure is to be expected and is actually rather mild: if $\vdash M : \S A$ and $M \rightsquigarrow_\beta N$, then $\vdash N : A$. This is enough for our purposes; subject reduction in itself is not essential for us, because we never use typing as a means of ensuring properties.

The types of Booleans and Church strings are $\mathsf{Bool} := \forall X.X \multimap X \multimap X$ and $\mathsf{Str} := \forall X.!(X \multimap X) \multimap !(X \multimap X) \multimap \S(X \multimap X)$, which are adaptations of the corresponding standard System F types. Booleans are the same as in $\ell\Lambda^{\mathrm{ps0}}_{\infty}$ and Church strings are obtained by erasing indices. In particular, each string has a unique encoding, *e.g.* $\underline{010} = \lambda s^0.\lambda s^1.\S(\lambda a.s^0(s^1(s^0 a)))$.

Definition 7 (The class C). *A language L is decidable in Λ^{ps0} ($L \in \mathsf{C}$) if there is a derivation $\vdash M : \mathsf{Str} \multimap \S^k\mathsf{Bool}$, with $\S^k A = \S \cdots \S A$ for some $k \in \mathbb{N}$, s.t. $M\underline{w} \to^*_\beta \mathsf{tt}$ if $w \in L$ and $M\underline{w} \to^*_\beta \mathsf{ff}$ if $w \notin L$.*

By Propositions 2 and 5 we have $\mathsf{C} \subseteq \mathsf{C}_\infty$.

A slight variant of the type of binary strings gives us the Church numerals, *i.e.*, unary integers, of type $\mathsf{Nat} := \forall X.!(X \multimap X) \multimap \S(X \multimap X)$. These are of the form $\underline{n} := \lambda s.\S(\lambda a.s(\ldots sa\ldots))$, with n occurrences of s. If $\boldsymbol{a} : \Gamma, c : A \vdash F : A$ and $\boldsymbol{b} : \Delta \vdash Z : A$, we define $\mathsf{it}(F, Z) := (\lambda z^{\S}.\S(z^{\S} Z[\boldsymbol{y}^{\S}/\boldsymbol{b}]))(n!(\lambda c.F[\boldsymbol{x}/\boldsymbol{a}]))$. It is readily verified that $\boldsymbol{x} : !\Gamma, \boldsymbol{y}^{\S} : \S\Delta, n : \mathsf{Nat} \vdash \mathsf{it}(F, Z) : \S A$ and that $\mathsf{it}(F, Z)\,\underline{n} \to^*_\beta (\lambda c.F)(\ldots(\lambda c.F)Z\ldots)$, the n-fold iteration of F on Z. Using iteration, we may define the basic arithmetic functions, including any polynomial, by adapting the usual definitions, much as in [4, 1]. Furthermore, following [1], we may define a type Tur of Turing machine configurations and, for any deterministic transition function, a term of type $\mathsf{Tur} \vdash \mathsf{Tur}$ implementing it. One may also easily implement the function building an initial configuration from a string (of type $\mathsf{Str} \vdash \mathsf{Tur}$), the function telling whether a configuration is accepting (of type $\mathsf{Tur} \vdash \S\mathsf{Bool}$) and the function returning the length of a string (of type $\mathsf{Str} \vdash \mathsf{Nat}$). Composing all these, with the help of iteration and the numerical functions shown above, every deterministic Turing machine with a polynomial clock may be implemented in Λ^{ps0}, showing that $\mathsf{P} \subseteq \mathsf{C}$.

The characterization. We remind that P/poly is the class of languages decided by polytime Turing machines with polynomial advice, or by polynomial-size Boolean circuits.

Theorem 1. $\mathsf{C}_\infty = \mathsf{P}/\mathsf{poly}$ *and* $\mathsf{C} = \mathsf{P}$.

Proof. The inclusion $\mathsf{C}_\infty \subseteq \mathsf{P}/\mathsf{poly}$ is obtained from Lemma 5, as delineated in Sect. 1: if $t\underline{w} \to^l u$, with u the encoding of a Boolean, we have $\lfloor u \rfloor_0 = u$. By Lemma 5, there exists $m \leq (\mathrm{d}(t\underline{w})+1)|t\underline{w}|^{2^{\mathrm{d}(t\underline{w})+1}} + \mathrm{d}(t\underline{w})|t\underline{w}|^{2^{\mathrm{d}(t\underline{w})}}$ such that $\lfloor t\underline{w} \rfloor_m = \lfloor t \rfloor_m \underline{w} \to^* u$. But $|t\underline{w}| = 1 + |t| + |\underline{w}| = O(|w|^2)$ (by choosing the suitable Church encoding) and $\mathrm{d}(t\underline{w}) = \max(\mathrm{d}(t), 1) = O(1)$, so m is polynomial in $|w|$. From this, to prove $\mathsf{C} \subseteq \mathsf{P}$ it is enough to observe that, in case $t = [\![M]\!]$ with M in Λ^{ps0}, the $\lfloor t \rfloor_m$ are in fact polytime computable (they are actually logspace computable, see Sect. 5).

For the converse, we need to encode Turing machines with advice as terms of $\ell\Lambda^{\mathrm{ps0}}_\infty$. We will make the simplifying assumption that the advice strings a_n (where $n \in \mathbb{N}$ is the length of the input) are "cumulative", *i.e.*, for all n, a_n is a polynomially-long prefix of an infinite binary word A. Every polynomial advice may be transformed into a cumulative polynomial advice, so there is no loss of generality. We will show how to encode the infinite string A in $\ell\Lambda^{\mathrm{ps0}}_\infty$ and how a prefix of a given length may be extracted. This is enough to conclude, because the rest is all uniform computation which we already know is representable in $\ell\Lambda^{\mathrm{ps0}}_\infty$ (via Λ^{ps0}): if w is the input string, the prefix of A to be extracted is of length $q(|w|)$ with q a polynomial, and we know that both q and $|\cdot|$ are representable; the resulting advice string is then fed to the encoding of the suitable polynomially-clocked Turing machine, together with a copy of w.

Let A_j be the j-th bit of A, and let

$$\begin{aligned} Z_A &:= \langle\langle \underline{\epsilon} \rangle\rangle \otimes \langle I_{A_0}, I_{A_1}, I_{A_2}, \ldots \rangle, \\ F &:= \lambda w \otimes x.\langle (\lambda f.\lambda y.\langle f_0 y_0 \rangle)(x_0 \langle S_0, S_1 \rangle) w_0 \rangle \otimes \langle x_1, x_2, x_3, \ldots \rangle, \\ \mathsf{extr}_A &:= \lambda n.(\lambda z.\langle (\lambda a \otimes b.a)(z_0 Z_A) \rangle)(n\langle F, F, F, \ldots \rangle), \end{aligned}$$

where, for $i \in \{0,1\}$, $I_i := \lambda x.\langle x_i \rangle$ and S_i represent the two constructors on Church strings (s.t. $S_i\underline{w} \to^* \underline{iw}$). The reader may check that the above terms are all in $\ell\Lambda^{\mathrm{ps0}}_\infty$. The term extr_A takes a Church numeral $\underline{n}$ of $\ell\Lambda^{\mathrm{ps0}}_\infty$ (which is of the form $\lambda s.\langle \lambda a.s_{i_1}(\ldots s_{i_n} a \ldots) \rangle$, with $i_1, \ldots, i_n$ pairwise distinct but otherwise arbitrary) and iterates n times F on Z_A. The result is a pair, of which the first component is taken as the final result. The term Z_A is where we fully exploit non-uniformity, representing A. Finally, if we disregard boxes, F takes a pair (w, iW), composed of a finite and an infinite string, and returns (iw, W). Therefore, $\mathsf{extr}_A\, \underline{q(n)} \to^* \underline{a_n}$ for all $n \in \mathbb{N}$. □

5 Affine Lambda-Terms and Boolean Circuits

Let $L \in \mathsf{P}$. By Theorem 1, we know that L is decided by $M \in \Lambda^{\mathrm{ps0}}$, so deciding whether $w \in L$ amounts to normalizing $M\underline{w}$, which, by Proposition 5, amounts to normalizing $[\![M]\!]\underline{w}$ (with $\underline{w}$ any encoding of w in $\ell\Lambda^{\mathrm{ps0}}_\infty$). But, for this, we know that it is enough to normalize $\lfloor [\![M]\!] \rfloor_m \underline{w}$ with m polynomial in $|w|$. By inspecting the definition of $[\![\cdot]\!]$ and $\lfloor \cdot \rfloor_n$ one may see that building $\lfloor [\![M]\!] \rfloor_m$ from M may be done in logarithmic space (in $|w|$), much like building the circuit representing the computation of a polytime Turing machine from the trace of its execution on w.

We have therefore given an alternative proof of the P-completeness of the normalization problem for the affine λ-calculus (given an affine λ-term, decide whether its

normal form is the Boolean $\mathtt{tt}$). Mairson [10] showed this by encoding Boolean circuits in affine λ-terms. The interest of the above proof is that it is virtually identical to the usual P-completeness proof of CIRCUIT VALUE [7], which is essentially the Cook-Levin theorem and does not rest on the P-completeness of another problem. It is also noteworthy that the "locality of computation" is reflected in the continuity of reduction.

The results of this paper seem to suggest the following "equation":

$$\frac{\text{affine } \lambda\text{-terms}}{\text{(infinitary affine) } \lambda\text{-terms}} = \frac{\text{Boolean circuits}}{\text{Turing machines (with advice)}}$$

The relationship between Boolean circuits and affine calculi was of course already known [10, 13]. However, we are seeing a connection here which is deeper than what was shown by any previous result. An interesting perspective given by the above "equation" is to study the notion of uniformity of families of Boolean circuits via the uniformity of the infinitary affine λ-calculus. This may be defined in a purely algebraic way: the terms t such that $t \blacktriangleleft M$ may be characterized by means of a partial equivalence relation, as in [11]. This might be turned into a notion of uniform family of Boolean circuits which is purely intrinsic, *i.e.*, it depends only on the "shape" of the circuits in the family and does not invoke external algorithms producing the circuits themselves. Investigating such a notion is definitely a topic worth further investigation.

Acknowledgments. We wish to thank Kazushige Terui for discussions which greatly contributed to the development of this work. We acknowledge partial support of ANR projects LOGOI ANR-2010-BLAN-0213-02 and COQUAS ANR-12-JS02-006-01.

References

1. Asperti, A., Roversi, L.: Intuitionistic light affine logic. ACM Trans. Comput. Log. 3(1), 137–175 (2002)
2. Bellantoni, S., Cook, S.A.: A new recursion-theoretic characterization of the polytime functions. Computational Complexity 2, 97–110 (1992)
3. Bourbaki, N.: General Topology: Chapters 1–4. Springer (1998)
4. Girard, J.Y.: Light linear logic. Inf. Comput. 143(2), 175–204 (1998)
5. Jones, N.D.: Logspace and ptime characterized by programming languages. Theor. Comput. Sci. 228(1-2), 151–174 (1999)
6. Kfoury, A.J.: A linearization of the lambda-calculus and consequences. J. Log. Comput. 10(3), 411–436 (2000)
7. Ladner, R.E.: The circuit value problem is log-space complete for P. SIGACT News 6(2), 18–20 (1975)
8. Lafont, Y.: Soft linear logic and polynomial time. Theor. Comput. Sci. 318(1-2), 163–180 (2004)
9. Leivant, D., Marion, J.Y.: Lambda calculus characterizations of poly-time. Fundam. Inform. 19(1/2) (1993)
10. Mairson, H.G.: Linear lambda calculus and ptime-completeness. J. Funct. Program. 14(6), 623–633 (2004)
11. Mazza, D.: An infinitary affine lambda-calculus isomorphic to the full lambda-calculus. In: Proceedings of LICS. pp. 471–480 (2012)
12. Melliès, P.A.: Asynchronous games 2: The true concurrency of innocence. Theor. Comput. Sci. 358(2-3), 200–228 (2006)
13. Terui, K.: Proof nets and boolean circuits. In: Proceedings of LICS. pp. 182–191 (2004)

A Nameless Pre-Terms

We formally introduce *de Bruijn notation* for pre-terms. *Pattern indices* and *nameless pre-terms* are generated by the grammar

$$\mathsf{i} ::= (n,i) \mid \mathsf{i} \otimes \bullet \mid \bullet \otimes \mathsf{i}, \qquad\qquad t,u ::= \bot \mid \mathsf{i} \mid \lambda.t \mid tu \mid t \otimes u \mid \mathbf{u},$$

where $n, i \in \mathbb{N}$ and the remaining notations are identical to the case of pre-terms. An occurrence of index pattern i corresponds to an occurrence of variable. It contains exactly one pair of integers (n,i); n is the actual de Bruijn index: it identifies the λ-binder in the usual way (*i.e.*, n is the number of binders between i and the binder of i); on the other hand, i gives information on the nature of the variable: $i = 0$ means a linear variable, whereas $i > 0$ means that it is the $(i-1)$-th occurrence of a non-linear variable. The structure of i gives information on where the variable is within a larger pattern. With this in mind, every pre-term induces a unique nameless pre-term in the usual way. As an example, the de Bruijn notation for $\lambda x.\lambda a \otimes b.\langle x_0(x_1(b \otimes a))\rangle$ is $\lambda.\lambda.\langle(1,1)((1,2)((\bullet \otimes (0,0)) \otimes ((0,0) \otimes \bullet)))\rangle$.

Although practically unreadable, de Bruijn notation allows us to see pre-terms as labelled trees: if we set $\Sigma := \{\bot, \mathsf{i}, \lambda, @, \otimes, !\}$ (with i ranging over index patterns), then nameless pre-terms are just functions $\mathbb{N}^* \longrightarrow \Sigma$. This is the formal description of terms that is intended to be used in the definition of uniform structure.

B Cauchy-Continuity of Reduction

In the sequel, we denote by $\lambda\mathsf{p}.\ell\Lambda$, $\ell\Lambda \mathbin{@} \ell\Lambda$, $\ell\Lambda \otimes \ell\Lambda$ and $!\ell\Lambda$ the sets of all terms of the form $\lambda\mathsf{p}.t$, tu, $t \otimes u$ and $\mathbf{u}$, respectively. All these sets are endowed with the subspace uniformity. We will also use the following notations:

- $\ell\Lambda \times_{\mathrm{aff}} \ell\Lambda$ is the set of all pairs of terms (t,u) such that tu is also a term (not just a pre-term), endowed with the subspace uniformity of the product uniformity on $\ell\Lambda \times \ell\Lambda$;
- $(\ell\Lambda)^{(\mathbb{N})}$ is the set $!\ell\Lambda$ endowed with the subspace uniformity of the product uniformity on $(\ell\Lambda)^{\mathbb{N}}$ (*i.e.*, the set of *all* sequences of terms).

The maps $\lambda\mathsf{p} : \ell\Lambda \longrightarrow \lambda\mathsf{p}.\ell\Lambda$, $@ : \ell\Lambda \times_{\mathrm{aff}} \ell\Lambda \longrightarrow \ell\Lambda \mathbin{@} \ell\Lambda$ and $\otimes : \ell\Lambda \times_{\mathrm{aff}} \ell\Lambda \longrightarrow \ell\Lambda \otimes \ell\Lambda$, defined by $t \mapsto \lambda\mathsf{p}.t$, $(t,u) \mapsto tu$ and $(t,u) \mapsto t \otimes u$, respectively, are obviously bijective. Furthermore, they and their inverses are all uniformly continuous:

Proposition 6 (Isotropy). *We have the following uniform homeomorphisms:*

1. $\lambda\mathsf{p}.\ell\Lambda \cong \ell\Lambda$ *via* $\lambda\mathsf{p}$*;*
2. $\ell\Lambda \mathbin{@} \ell\Lambda \cong \ell\Lambda \otimes \ell\Lambda \cong \ell\Lambda \times_{\mathrm{aff}} \ell\Lambda$ *via* $@$ *and* $\otimes$*;*
3. $!\ell\Lambda \cong (\ell\Lambda)^{(\mathbb{N})}$ *via the identity.*

Proof. We consider directly point 2, and start with $@^{-1}$. We have to show that for all $\xi', \xi'' \in \mathbb{N}^{\mathbb{N}}$, there exists $\xi \in \mathbb{N}^{\mathbb{N}}$ such that $(tu, t'u') \in \mathcal{U}_\xi$ implies $(t,t') \in \mathcal{U}_{\xi'}$ and $(u,u') \in \mathcal{U}_{\xi''}$. It is easy to check that setting $\xi := 1 \cdot (\xi' \vee \xi'')$ meets the requirement. For $@$, we have $\xi = n \cdot \hat{\xi}$ and must determine ξ', ξ'' to get the converse implication. It is easily seen that setting $\xi' := \xi'' := \hat{\xi}$ is enough. The case of $\otimes$ is virtually identical. Point 1 is also similar.

Before proving point 3, let us remind that the uniformity on $(\ell\Lambda)^{(\mathbb{N})}$ is given by the following basis of entourages: we choose a finite $F \subset \mathbb{N}$, we chose one $\xi_i \in \mathbb{N}^{\mathbb{N}}$ for each $i \in F$, and we define the basic entourage $\mathcal{V}_{(\xi_i)_{i\in F}} := \{(\mathbf{u},\mathbf{u}') \mid \forall i \in F,\ (\mathbf{u}(i),\mathbf{u}'(i)) \in \mathcal{U}_{\xi_i}\}$. Now, in the direction from $!\ell\Lambda$ to $(\ell\Lambda)^{(\mathbb{N})}$, we need to prove that, for all $\{i_1,\dots,i_n\} \subset \mathbb{N}$ and for every $\xi_{i_1},\dots,\xi_{i_n} \in \mathbb{N}^{\mathbb{N}}$, there exists $\xi \in \mathbb{N}^{\mathbb{N}}$ such that $(\mathbf{u},\mathbf{u}') \in \mathcal{U}_\xi$ implies $(\mathbf{u},\mathbf{u}') \in \mathcal{V}_{\xi_{i_1},\dots,\xi_{i_n}}$. We define the integer $k := \max\{i_1,\dots,i_n\}$ and the sequence of integers $\xi' := \bigvee_{1\le j\le n} \xi_{i_j}$. We

invite the reader to check that $\xi := k \cdot \xi'$ meets the requirement. The other direction is easier: we have ξ and we need to determine a finite $F \subset \mathbb{N}$ and $\xi_i \in \mathbb{N}^{\mathbb{N}}$ for each $i \in F$. Let $\xi = n \cdot \xi'$; then, the reader may check that it is enough to set $F := \{0, 1, 2, \dots, n\}$ and $\xi_i := \xi'$ for all $0 \leq i \leq n$. □

In the following, given a net $(t_\iota)_{\iota \in I}$, we say that a property "eventually" holds for t_ι if there exists $\kappa \in I$ such that, for all $\iota \geq \kappa$, t_ι verifies the property. Also, if we define some t'_ι for all $\iota \geq \kappa$ by relying on the fact that t_ι satisfies the property, we speak of the net $(t'_\iota)_{\iota \in I}$ implicitly assuming that t'_ι is fixed arbitrarily for those ι such that t_ι does not satisfy the property.

Corollary 1. *A net $(t_\iota)_{\iota \in I}$ is Cauchy iff one of the following eventually holds:*

1. *$t_\iota = t$, where t is an occurrence of variable or $\bot$;*
2. *$t_\iota \in \lambda\mathsf{p}.\ell\Lambda$ and $(\lambda\mathsf{p}^{-1}(t_\iota))_{\iota \in I}$ is Cauchy in $\ell\Lambda$;*
3. *$t_\iota \in \ell\Lambda \mathbin{@} \ell\Lambda$ and both $(\pi_1(@^{-1}(t_\iota)))_{\iota \in I}$ and $(\pi_2(@^{-1}(t_\iota)))_{\iota \in I}$ are Cauchy;*
4. *$t_\iota \in \ell\Lambda \otimes \ell\Lambda$ and both $(\pi_1(\otimes^{-1}(t_\iota)))_{\iota \in I}$ and $(\pi_2(\otimes^{-1}(t_\iota)))_{\iota \in I}$ are Cauchy;*
5. *$t_\iota = \mathbf{u}_\iota$ and, for all $i \in \mathbb{N}$, $(\mathbf{u}_\iota(i))_{\iota \in I}$ is Cauchy.*

Proof. That all terms are eventually of the same kind (variable, abstraction, application, etc.) is immediate. Then, case 1 is obvious, whereas the other cases follow from the isomorphisms with product uniformities (Proposition 6). □

In the following, we consider the functions $\Pi_i : !\ell\Lambda \longrightarrow \ell\Lambda$, for $i \in \mathbb{N}$, defined by $\Pi_i(\mathbf{u}) := \mathbf{u}(i)$. We also define the set of functions $\mathcal{F} := \{\lambda\mathsf{p}^{-1}, \pi_i \circ @^{-1}, \pi_i \circ \otimes^{-1}, \Pi_j \mid i \in \{1, 2\}, j \in \mathbb{N}\}$. Let $(t_\iota)_{\iota \in I}$, $(t'_\iota)_{\iota \in I}$ be two Cauchy nets. We write $(t_\iota)_{\iota \in I} \sqsubset (t'_\iota)_{\iota \in I}$ just if, eventually, $t_\iota = \varphi(t'_\iota)$ for some $\varphi \in \mathcal{F}$.

Lemma 6 (Well-foundedness). *The relation $\sqsubset$ is well-founded.*

Proof. Assume the contrary, and let $(t^0_\iota)_{\iota \in I} \sqsupset (t^1_\iota)_{\iota \in I} \sqsupset (t^2_\iota)_{\iota \in I} \sqsupset \dots$ be an infinite descending chain. By definition, we have a sequence $(\varphi_n)_{n \in \mathbb{N}}$ of functions in $\mathcal{F}$ such that, for all $n \in \mathbb{N}$, there exists $\kappa_n \in I$ such that, for all $\iota \geq \kappa_n$, $t^{n+1}_\iota = \varphi_n(t^n_\iota)$. Define $\iota_0 := \kappa_0$ and, inductively, let ι_{n+1} be an element of I which is above both κ_{n+1} and ι_n (such an element must exist because I is directed). We set $u_n := t^0_{\iota_n}$. Observe that, by construction, for arbitrary $n \in \mathbb{N}$ we have $\varphi_n \circ \dots \circ \varphi_0(u_{n+1}) = t^{n+1}_{\iota_{n+1}} \neq \bot$. The latter disequality holds because, since $\iota_{n+1} \geq \kappa_{n+1}$, $t^{n+1}_{\iota_{n+1}}$ is in the domain of φ_{n+1}, which never contains $\bot$.

Let us now define $\xi \in \mathbb{N}^{\mathbb{N}}$ by setting ξ_n to be: 0 if $\varphi_n = \lambda\mathsf{p}^{-1}$; $i - 1$ if $\varphi_n = \pi_i \circ @^{-1}$ or $\varphi_n = \pi_i \circ \otimes^{-1}$; and i if $\varphi_n = \Pi_i$. We denote by α_n the prefix of ξ of length $n + 1$. We may prove by induction on n that $\varphi_n \circ \dots \circ \varphi_0(u_{n+1})[\epsilon] = u_{n+1}[\alpha_n]$, for all $n \in \mathbb{N}$. But $(u_n)_{n \in \mathbb{N}}$ is a Cauchy sequence (because it is the subsequence of a Cauchy net) and therefore there exists $k \in \mathbb{N}$ such that, for all $i, j \geq k$ and for every $n \in \mathbb{N}$, $u_i[\alpha_n] = u_j[\alpha_n]$. In particular, $u_k[\alpha_n] = u_{n+1}[\alpha_n] \neq \bot$ for all $n \geq k$, which is absurd, because u_k is a finite term. □

Lemma 7 (Substitution). *Let $(u_\iota)_{\iota \in I}$ be a Cauchy net such that, eventually, $u_\iota \between \mathsf{p}$. In that case, if a net $(t_\iota)_{\iota \in I}$ is Cauchy, then so is $(t_\iota[u_\iota/\mathsf{p}])_{\iota \in I}$.*

Proof. By well-founded induction on $(t_\iota)_{\iota \in I}$, using Corollary 1. □

Given $\alpha \in \mathbb{N}^*$, we define $R_\alpha : \ell\Lambda \longrightarrow \ell\Lambda$ by $R_\alpha(t) := t'$ if $t \to_{\mathrm{s}} t'$ by reducing a redex at position α, or $R_\alpha(t) := t$ if no such reduction applies.

Proposition 7 (Cauchy-continuity of reduction). *For all $\alpha \in \mathbb{N}^*$, R_α is Cauchy-continuous.*

Proof. Let $(t_\iota)_{\iota\in I}$ be a Cauchy net, and let $t'_\iota := R_\alpha(t_\iota)$. We need to show that $(t'_\iota)_{\iota\in I}$ is Cauchy. We do this by induction on the length of α. We start with the inductive case, which is easy. Suppose $\alpha = n \cdot \alpha'$. The proof depends on the value of n, but it is similar in all cases. To give the idea, we let $n = 1$. Now, if we are not in one of cases 3, 4 or 5 of Corollary 1, then eventually $t_\iota[\alpha] = \bot$ and the result is vacuously true. Again, to show the idea, we pick case 4, *i.e.*, eventually $t_\iota = t'_\iota t''_\iota$. By Corollary 1 both $(t'_\iota)_{\iota\in I}$ and $(t''_\iota)_{\iota\in I}$ are Cauchy; by the induction hypothesis, $(R_{\alpha'}(t''_\iota))_{\iota\in I}$ is Cauchy, so $(t'_\iota R_{\alpha'}(t''_\iota))_{\iota\in I}$ is Cauchy (again by Corollary 1). But observe that, for all u, v, $R_\alpha(uv) = uR_{\alpha'}(v)$, hence we are done.

So we only need to prove the Cauchy-continuity of R_ϵ. We start by defining, for every pattern p, a finite $A(\mathsf{p}) \subset \mathbb{N}^*$ as follows. First, we set $A^-(a) := A^-(x) := \{\epsilon\}$ and $A^-(\mathsf{p} \otimes \mathsf{q}) := 0 \cdot A^-(\mathsf{p}) \cup 1 \cdot A^-(\mathsf{q})$; then, we let $A(\mathsf{p}) := 1 \cdot A^-(\mathsf{p})$. Now, observe that t is a redex $(\lambda\mathsf{p}.u)v$ precisely if: $t[\epsilon] = @$; $t[0] = \lambda$; for all $\alpha' \in A(\mathsf{p})$, $t[\alpha']$ has a suitable value depending on p and α' (for instance: if $\mathsf{p} = a$, there is no requirement; if $\mathsf{p} = x$, then we must have $t[1] = !$; for more complex patterns, we have at least $t[1] = \otimes$ and the rest depends on the pattern). The essence of the above discussion is that a pattern p induces $\alpha_1, \dots, \alpha_k \in \mathbb{N}^*$ and $\sigma_1, \dots, \sigma_k \in \Sigma$ such that t is a redex of pattern p iff $t[\alpha_i] = \sigma_i$ for all $1 \leq i \leq k$. Therefore, by extending each α_i arbitrarily to get $\xi_i \in \mathbb{N}^{\mathbb{N}}$ and by setting $\xi := \bigvee_{1\leq i\leq k} \xi_i$, we have $\alpha_i \prec \xi_i \leq \xi$ for all $1 \leq i \leq k$ and, whenever $(t, t') \in \mathcal{U}_\xi$, t is a redex iff t' is.

The above means that, by virtue of the Cauchy property, either eventually t_ι is a redex, or eventually none of t_ι is. In the latter case, $(t'_\iota)_{\iota\in I}$ is trivially Cauchy. In the former case, eventually $t_\iota = (\lambda\mathsf{p}.s_\iota)u_\iota$ and $u_\iota \between \mathsf{p}$. Therefore, eventually $t'_\iota = s_\iota[u_\iota/\mathsf{p}]$. Now, by Corollary 1, $(s_\iota)_{\iota\in I}$ and $(u_\iota)_{\iota\in I}$ are both Cauchy, so Lemma 7 applies, and we are done. □

The Cauchy-continuity of unboxing reduction is trivial: an application of $\rightsquigarrow$ is simply an application of the uniformly continuous map we called Π_0 above (remember that uniform continuity implies Cauchy-continuity).

C Proofs of Sect. 3

C.1 Proof of Proposition 3

Lemma 8. *Let $t, u \in \ell\Lambda_\infty$, let $u \between \mathsf{p}$ and suppose that $x \in \mathsf{p}$ and x free in t imply $\mathrm{d}_{x_i}(t) = 1$ for all $i \in \mathbb{N}$. Then, for all $v \sqsubseteq u$, $\mathrm{d}_v(t[u/\mathsf{p}]) = \mathrm{d}_v(u)$ (provided v is not erased by the substitution).*

Proof. The proof is by induction on p. If $\mathsf{p} = a$, we may assume that a is free in t, in which case it must appear at box-depth 0, so $\mathrm{d}_v(t[u/a]) = \mathrm{d}_v(u)$. If $\mathsf{p} = x$, then $u = \mathbf{u}$ and $v \sqsubseteq \mathbf{u}(i)$ for some $i \in \mathbb{N}$, which means $\mathrm{d}_v(u) = \mathrm{d}_v(\mathbf{u}(i)) + 1$. But, by the hypothesis, $\mathrm{d}_{x_i}(t) = 1$ (again, we may assume that x_i does appear in t), so $\mathbf{u}(i)$ is substituted inside exactly one box of t and $\mathrm{d}_v(t[\mathbf{u}/x]) = \mathrm{d}_v(\mathbf{u}(i)) + 1$, as desired. The inductive case is immediate. □

The second part is an immediate consequence of the definitions, so let us concentrate on the first. Let $x_i \sqsubseteq t'$; it is enough to show that $\mathrm{rd}_{x_i}(t') = 1$. Note that reduction does not create occurrences of variables, so $x_i \sqsubseteq t$ and by hypothesis $\mathrm{rd}_{x_i}(t) = 1$. Let $t = S[(\lambda\mathsf{p}.u)v]$ and $t' = S[u[v/\mathsf{p}]]$. If x_i is in S, then the result is obvious. Otherwise, by α-equivalence we may assume $x \notin \mathsf{p}$ and we have two cases: either $x_i \sqsubseteq u$, in which case the lemma holds regardless of stratification; or $x_i \sqsubseteq v$, in which case, thanks to stratification, we may apply Lemma 8 to obtain $\mathrm{d}_{x_i}(u[v/\mathsf{p}]) = \mathrm{d}_{x_i}(v)$. If x is free in v, this is enough to conclude. Otherwise, we have $\lambda\mathsf{q}.v' \sqsubseteq v$ which binds x and, applying again Lemma 8, we know that $\mathrm{d}_{\lambda\mathsf{q}.v'}(u[v/\mathsf{p}]) = \mathrm{d}_{\lambda\mathsf{q}.v'}(v)$, which allows us to conclude by definition of binder-relative box-depth. □

C.2 Proof of Lemma 1

Point 1 is immediate: by well-foundedness, the only way that a term may have infinite height is if it contains a box $\mathbf{u}$ such that the height of $\mathbf{u}(i)$, for $i \in \mathbb{N}$, is unbounded, contradicting the definition of parsimony.

Let us turn to point 2. First of all, it may be established by a straightforward induction that, for all n, $t \sim_n t'$ implies that t, t' have the same number (including ∞) of occurrences of every free variable. Then, since $\mathbf{u}(k) \sim_h \mathbf{u}(k+h)$, the number of free occurrences of x is the same in all $\mathbf{u}(k+h)$ for all $h \in \mathbb{N}$. Let $x_{j_h^1}$ and $x_{j_h^2}$ be two occurrences in each $\mathbf{u}(k+h)$. Fix $l \in \{1,2\}$ arbitrarily. We claim that the sequence $(j_h^l)_{h\in\mathbb{N}}$ is strictly increasing. First of all, observe that parsimony implies $|j_{h+1}^l - j_h^l| = 1$ (because $j_{h+1}^l = j_h^l$ is impossible by affinity). Then, the sequence is monotonic: if we had something like $j_h^l > j_{h+1}^l < j_{h+2}^l$, we would have $j_h^l = j_{h+2}^l$, contradicting affinity. But then the sequence must be increasing, because it is infinite. Therefore, we have $j_h^l = j_0^l + h$, for all $h \in \mathbb{N}$. Suppose now that $j_0^1 < j_0^2$. We would have $j^1_{j_0^2 - j_0^1} = j_0^1 + j_0^2 - j_0^1 = j_0^2$, contradicting affinity. A similar contradiction is obtained if $j_0^1 > j_0^2$, so $j_0^1 = j_0^2$, which means there is only one occurrence. □

C.3 Proof of Proposition 4

Thanks to the fact that parsimony is inherited by subterms, the result may be proved by a straightforward induction from the following substitution lemma: if $t, u \in \ell\Lambda^{\mathrm{p}}_{\infty}$ and $u \between \mathsf{p}$, then $t[u/\mathsf{p}] \in \ell\Lambda^{\mathrm{p}}_{\infty}$.

To prove the lemma, we start by establishing the following claim. Let $t \sim_n t'$ and let $\mathbf{u}$ be parsimonious with n.u. factor k. Then, if all x_i occurring free in t, t' are such that $i \geq k$, then $t[\mathbf{u}/x] \sim_n t'[\mathbf{u}/x]$. The proof is a straightforward induction, the important case being $t = x_i$ and $t' = x_j$, so that $t[\mathbf{u}/x] = \mathbf{u}(i)$ and $t'[\mathbf{u}/x] = \mathbf{u}(j)$, from which, by hypothesis, we have $\mathbf{u}(i) \sim_{|i-j|} \mathbf{u}(j)$, which implies $\mathbf{u}(i) \sim_n \mathbf{u}(j)$ because the hypothesis $x_i \sim_n x_j$ gives $|i-j| \leq n$.

The proof of the lemma is by induction on p. The base case $\mathsf{p} = a$ and the inductive case are straightforward, so we concentrate on the case $\mathsf{p} = x$, in which $u = \mathbf{u}$. We proceed by induction on t. All cases are immediate except $t = \mathbf{v}$. Let $\mathbf{v}' := \mathbf{v}[\mathbf{u}/x]$. We need to show that $\mathbf{v}'$ is parsimonious. By definition, we have $\mathbf{v}'(i) = \mathbf{v}(i)[\mathbf{u}/x]$. If all the free occurrences of x in $\mathbf{v}$ are concentrated in finitely many $\mathbf{v}(i)$, we conclude trivially by parsimony of $\mathbf{v}$. Otherwise, there necessarily exists $k \in \mathbb{N}$ such that, for all $i \geq k$, whenever x_j appears free in $\mathbf{v}(i)$, j surpasses the n.u. factor of $\mathbf{u}$. Therefore, by the above claim and the parsimony of $\mathbf{v}$, $i, i' \geq k$ implies $\mathbf{v}'(i) \sim_{|i-i'|} \mathbf{v}'(i')$, as desired. □

C.4 Proof of Lemma 2

The proof of this lemma basically mimics the usual proof of the similar result for light linear logic [4], which is done using proof nets. Here, we consider instead a variant of $\ell\Lambda_{\infty}$ with explicit substitutions, following recent work of Accattoli. An *explicit substitution* is of the form $[x := \mathbf{u}]$ where x is a non-linear variable and $\mathbf{u}$ is a box. We use σ to range over finite lists of explicit substitutions (simply called *substitution lists*) and we denote by $\sigma\sigma'$ the concatenation of such lists and by ϵ the empty list. We define $\sigma[u/\mathsf{p}]$ by $\epsilon[u/\mathsf{p}] := \epsilon$ and $([x := \mathbf{u}]\sigma')[u/\mathsf{p}] := [x := \mathbf{u}[u/\mathsf{p}]]\sigma[u/\mathsf{p}]$.

Given a pattern p and a term $u \between \mathsf{p}$, we define $t[u/\mathsf{p}^\ell]$ as follows: $t[u/a^\ell] := t[u/a]$, $t[\mathbf{u}/x^\ell] := t$ and $t[u_1 \otimes u_2/(\mathsf{p}_1 \otimes \mathsf{p}_2)^\ell] := t[u_1/\mathsf{p}_1^\ell][u_2/\mathsf{p}_2^\ell]$. We also define the substitution list $[\![\mathsf{p} := u]\!]$ as follows: $[\![a := u]\!] := \epsilon$; $[\![x := \mathbf{u}]\!] := [x := \mathbf{u}]$; $[\![\mathsf{p}_1 \otimes \mathsf{p}_2 := u_1 \otimes u_2]\!] := [\![\mathsf{p}_1 := u_1]\!][\![\mathsf{p}_2 := u_2]\!]$.

A *configuration* is a pair (t, σ) where $t \in \ell\Lambda_\infty$ and σ is a substitution list. Configurations will be ranged over by c. We define the following rewriting rules on configurations:

- $(S[(\lambda \mathsf{p}.t)u], \sigma\sigma') \to_{\mathrm{m}} (S[t[u/\mathsf{p}^\ell]], \sigma[\![\mathsf{p} := u]\!]\sigma')$ whenever $u \between \mathsf{p}$ and no non-linear variable of p appears in σ',
- $(t, \sigma[x := \mathbf{u}]\sigma') \to_{\mathrm{e}} (t[\mathbf{u}/x], \sigma[\mathbf{u}/x]\sigma')$,

where S is any shallow context. So, when a redex is fired, only the linear part of the substitution is performed immediately, whereas the non-linear part is stored in the substitution list and may be performed later. In fact, we obviously have that $t \to_{\mathrm{s}} t'$ implies $(t, \sigma) \to_{\mathrm{m}} \to_{\mathrm{e}} (t', \sigma)$ for all σ. We set $\to_{\mathrm{c}} := \to_{\mathrm{m}} \cup \to_{\mathrm{e}}$.

A reduction sequence $c \to_{\mathrm{c}}^* c'$ is *normal* if it can be decomposed as $c \to_{\mathrm{m}}^* \to_{\mathrm{e}}^* c'$. It is not hard to verify that, in $\ell\Lambda^{\mathrm{s}}_\infty$, $\to_{\mathrm{e}}$ steps may always be postponed: if $t \in \ell\Lambda^{\mathrm{s}}_\infty$ and $(t, \sigma) \to_{\mathrm{e}} c_1 \to_{\mathrm{m}} c'$, then there exists c_2 such that $(t, \sigma) \to_{\mathrm{m}} c_2 \to_{\mathrm{e}} c'$. Therefore, if $t \in \ell\Lambda^{\mathrm{ps0}}_\infty$ and $t \to_{\mathrm{s}}^* t'$, then $(t, \epsilon) \to_{\mathrm{c}} (t', \epsilon)$ via a normal reduction sequence and it is enough to prove the size bound on t' using such a sequence.

In the sequel, in order to name bound variables unambiguously, we use Barendregt's convention: we assume that all λ's occurring at box-depth 0 in a term t bind distinct variables. Observe that, since the calculus is affine, if $t \to_{\mathrm{s}} t'$ and t adheres to Barendregt's convention, then so does t', without any renaming. This gives an implicit way to track residues during a reduction: an occurrence x_i in t' is necessarily the (unique) residue of x_i in t.

Definition 8 (Arity). *Let x be a bound variable at box-depth 0 of $t \in \ell\Lambda^{\mathrm{p}}_\infty$, i.e., there is $\lambda \mathsf{p}.u \sqsubseteq t$ such that $x \in \mathsf{p}$ and $\mathrm{d}_{\lambda \mathsf{p}.u}(t) = 0$. Let $X := \{i \in \mathbb{N} \mid x_i \text{ is free in } u\}$. The* arity *of x in t, denoted by $\nabla_t(x)$, is defined as follows. If X is finite, then $\nabla_t(x) := \max X$. If X is infinite, then by point 2 of Lemma 1 there is a unique box $\mathbf{u} \sqsubseteq u$ containing all occurrences of x but finitely many; more precisely, it contains (at least) x_{j_0+h} for all $h \in \mathbb{N}$ and for some $j_0 \in \mathbb{N}$ unambiguously determined by the n.u. factor of $\mathbf{u}$; then, we set $\nabla_t(x) := j_0$.*

We write $x \trianglelefteq_\sigma y$ just if $\sigma = \sigma_1[y := \mathbf{v}]\sigma_2[x := \mathbf{u}]\sigma_3$ and there are infinitely many free occurrences of x in $\mathbf{v}$. Given $c := (t, \sigma)$, we define $\mu_c(x) := \sum_{x \trianglelefteq_\sigma y} \nabla_t(y)$. Let, moreover, $\sigma = [x^1 := \mathbf{u}_1] \cdots [x^n := \mathbf{u}_n]$. Then, we define the *potential size* at box-depth $j \geq 1$ of c by $[c]_j := |t|_j + \sum_{k=1}^n \mu_c(x^k)|\mathbf{u}_k|_j$.

One may prove that the order $\trianglelefteq$ is arborescent, which is an easy consequence of the fact that there is at most one $\mathbf{u}$ containing infinitely many occurrences of x. In fact, the order is dual with respect to that of the similar order relation defined in light linear logic [4] (*i.e.*, ours is downward-arborescent, that of light linear logic is upward-arborescent). At the level of the dynamics of reduction, this is the main difference between our system and light linear logic. The other key observation is that, by parsimony, if k is the n.u. factor of $\mathbf{u}$, we have $|\mathbf{u}(k+p)|_{j+1} \leq (1+n)|\mathbf{u}(k)|_{j+1}$, for all $j, p \in \mathbb{N}$. That is, the size of terms in a parsimonious sequence increases linearly, with multiplicative constant 1.

From here, the proof proceeds much as in [4, 1]: we consider a normal sequence $(t, \epsilon) \to_{\mathrm{m}}^* (t_1, \sigma) \to_{\mathrm{e}}^* (t', \epsilon)$ and observe that $|t_1| \leq |t|$; then, we observe that $[(t', \epsilon)]_j = |t'|_j$ and we show that, whenever $c_1 \to_{\mathrm{e}} c_2$, one has $[c_1]_j \geq [c_2]_j$. Therefore, the potential sizes of (t_1, σ) bound the actual sizes at the end of the reduction. One then concludes by observing that the potential size verifies the bound of the statement of the lemma. □

C.5 Proof of Lemma 3

Lemma 9. *Let $t \backsimeq_n t'$. Then:*

1. *if $n' \leq n$, then $t \backsimeq_{n'} t'$;*
2. *if $t = \bot$, $t = a$ or $t = x_i$, then $t' = t$;*

3. *if* $t = \lambda \mathsf{p}.t_1$, *then* $t' = \lambda \mathsf{p}.t_1'$ *and* $t_1' \backsimeq_n t_1$;
4. *if* $t = t_1 t_2$ *(resp.* $t = t_1 \otimes t_2$*), then* $t' = t_1' t_2'$ *(resp.* $t' = t_1' \otimes t_2'$*) and* $t_j' \backsimeq_n t_j$, *for* $j \in \{1, 2\}$;
5. *if* $t = \mathbf{u}$, *then* $t = \mathbf{u}'$ *and, for all* $0 \leq i \leq n$, $\mathbf{u}(i) \backsimeq_n \mathbf{u}'(i)$.

Proof. All points are proved by immediate well-founded inductions. □

First one proves that, for every pattern p containing exactly the non-linear variables $x^1, \ldots, x^p$, for all terms v and w $\between$ p, for all $n \in \mathbb{N}$ and $m \geq \max(n, \sup\{\mathrm{v}_{x^1,v}(n), \ldots, \mathrm{v}_{x^p,v}(n)\})$, we have that $v' \backsimeq_m v$ and $w' \backsimeq_m w$ implies $v'[w'/\mathsf{p}] \backsimeq_n v[w/\mathsf{p}]$. This is done by induction on p, the base cases being proved by well-founded induction on v. The only interesting case is that in which $\mathsf{p} = y$, $v = y_i$ and $w = \mathbf{w}$. In that case, the hypotheses and Lemma 9 give us $v' = y_i$ and $w' = \mathbf{w}'$ such that, for all $0 \leq j \leq m$, $\mathbf{w}'(j) \backsimeq_m \mathbf{w}(j)$. Since $\mathrm{v}_{y,v}(n) = i$ for all $n \in \mathbb{N}$, and since by hypothesis $m \geq i$, we may conclude.

The proof of the lemma has two cases: $t \rightsquigarrow t'$ is immediate and $t \to_{\mathrm{s}} t'$ is a straightforward induction on t, using the above substitution lemma in the key case $t = (\lambda \mathsf{p}.u)v$. □

C.6 Proof of Lemma 4

If there are only finitely many occurrences of x in t, then by definition $\mathrm{v}_{x,t}(n) \leq |t|$ for all $n \in \mathbb{N}$. Otherwise, let k be the n.u. factor of the unique box of t containing $x_{\nabla_t(x)+h}$ for all $h \in \mathbb{N}$. If $n < k$, then again $\mathrm{v}_{x,t}(n) < |t|$; if $n \geq k$, then $\mathrm{v}_{x,t}(n) \leq |t| + n - k$, which also satisfies the requirement. □

C.7 Proof of point 1 of Lemma 5

We reason by induction on the number of $\rightsquigarrow$ steps in the reduction. Let $t \to_{\mathrm{s}}^{l_1} t_1$ be the longest prefix of the reduction such that $t_1 = t'$ or t_1 is a box. By Lemma 2, we have $|t_1|_j \leq |t|_1 |t|_j$ for all $j \geq 1$ and obviously $l_1 \leq |t|$. If $t_1 = t'$, this is enough to conclude. Otherwise, we have $t_1 \rightsquigarrow t_2 \to^{l_2} t'$, with $l = l_1 + l_2 + 1$. Applying the induction hypothesis to the reduction starting with t_2 gives us $|t'| \leq |t_2|^{2^{\mathrm{d}(t_2)}}$. Observe that $\mathrm{d}(t_2) = \mathrm{d}(t_1) - 1 \leq \mathrm{d}(t) - 1$ (using point 1 of Proposition 3). More generally, $|t_2|_j = |t_1|_{j+1}$ for all j, so the bound on the sizes of t_1 gives us $|t_2| \leq |t|^2$. Combined with the above, this allows us to infer the desired size bound for t'. For the bound on l, the induction hypothesis gives $l_2 \leq (\mathrm{d}(t_2) + 1)|t_2|^{2^{\mathrm{d}(t_2)}} + \mathrm{d}(t_2) \leq \mathrm{d}(t)|t|^{2^{\mathrm{d}(t)}} + \mathrm{d}(t) - 1$, hence $l \leq |t| + \mathrm{d}(t)|t|^{2^{\mathrm{d}(t)}} + \mathrm{d}(t)$, from which we conclude. □

D Representing Basic Numerical Functions

Successor, addition and multiplication may be represented as follows:

$$\mathsf{succ} := \lambda s.(\lambda z^{\S}.\S(\lambda a.s(z^{\S}a)))(n!s) \qquad n : \mathsf{Nat} \vdash \mathsf{succ} : \mathsf{Nat}$$

$$\mathsf{add} := \mathsf{it}(\mathsf{succ}, m^{\S}) \qquad m^{\S} : \S\mathsf{Nat}, n : \mathsf{Nat} \vdash \mathsf{add} : \S\mathsf{Nat}$$

$$\mathsf{mul} := \mathsf{it}(\mathsf{add}[k/n], \S\underline{0}) \qquad k : !\mathsf{Nat}, n : \mathsf{Nat} \vdash \mathsf{mul} : \S^2\mathsf{Nat}$$

We may also define *coercions* $n : \mathsf{Nat} \vdash \mathsf{it}(\mathsf{succ}, \underline{0}) : \S^{p+1}!^q\mathsf{Nat}$, with $p, q \in \mathbb{N}$. Coercions compute the identity function (they are iterates of the successor on zero) and are useful for "stripping" modalities from the types of the arguments of another term (thanks to the cut rule) without altering the function computed by it. For instance, we may thus re-type multiplication as

$\mathsf{Nat}, \mathsf{Nat} \vdash \S^3\mathsf{Nat}$ (we omit the term annotations for briefness). From this, by means of a $\S$ rule, we get $\S\mathsf{Nat}, \S\mathsf{Nat} \vdash \S^4\mathsf{Nat}$, then by weak $!\mathsf{Nat}, \S\mathsf{Nat}, \S\mathsf{Nat} \vdash \S^4\mathsf{Nat}$ and, after two applications of $\mathsf{asym\,cntr}$, we get $!\mathsf{Nat} \vdash \S^4\mathsf{Nat}$. Using another coercion, we obtain a derivation of $\mathsf{Nat} \vdash \S^5\mathsf{Nat}$ whose underlying term implements the squaring function. By repeating this processes (via cut) we may program the function $n \mapsto n^k$ for any fixed integer power and, hence, any polynomial with non-negative integer coefficients.

The presence of second order allows us to type other numerical functions which are not representable with simple types, such as the predecessor and subtraction, in the standard way.