Higher-Order Linear Ramified Recurrence*

Ugo Dal Lago[1], Simone Martini[1], and Luca Roversi[2]

[1] Dipartimento di Scienze dell'Informazione, Università di Bologna, Bologna, Italy
{dallago,martini}@cs.unibo.it
[2] Dipartimento di Informatica, Università di Torino, Torino, Italy
roversi@di.unito.it

Abstract. Higher-Order Linear Ramified Recurrence (**HOLRR**) is a linear (affine) λ-calculus — every variable occurs at most once — extended with a recursive scheme on free algebras. Two simple conditions on type derivations enforce both polytime completeness and a strong notion of polytime soundness on typeable terms. Completeness for **PTIME** holds by embedding Leivant's ramified recurrence on words into **HOLRR**. Soundness is established at all types — and not only for first order terms. Type connectives are limited to tensor and linear implication. Moreover, typing rules are given as a simple deductive system.

1 Introduction

The main goal of giving machine-independent characterizations of **PTIME** is to overcome the drawback of conceiving feasible algorithms by thinking directly in terms of low-level machine primitives, like those of Turing machines. The research about this subject has brought forth a wide variety of interesting calculi, that can be classified under two parameters: their originating background, and their expressivity — the ability to naturally express higher-order functions.
Concerning the originating background, proposals range from those which are purely recursion-theoretical to the ones which are purely proof-theoretical.
For example, Bellantoni and Cook's *safe recursion on notation* [1] is of the former kind. Its recursive scheme forbids application of a recursively defined function to the result of a recursive call. The constraint is expressed directly inside the syntax of the recursive schemes, by distinguishing two classes of arguments, namely safe and normal arguments. Another example of a first-order function algebra capturing **PTIME** is Leivant's *ramified recurrence on words* [2, 3], which relies on the notion of tier to control the use of arguments in recursive schemes.
On the other side, purely proof-theoretical systems are logical, deductive systems, usually expressed on a graph language, that of proof-nets. Main examples of this class are *light linear logic* (**LLL**, [4]), *light affine logic* (**LAL**, [5, 6]), and *soft linear logic* (**SLL**, [7]). Boxes are certain regions inside a proof-net, and a box may contain other boxes, in a stratified fashion. The computational core is

* All the authors are partially supported by MIUR COFIN 2002 PROTOCOLLO project. The latter is also supported by DART IST-2001-33477 project.

cut-elimination, whose complexity is controlled by box stratification: the time necessary to normalize a proof-net is a polynomial in the size of the proof, the exponent of the polynomial depending only on the box-nesting depth. This, together with the fact that usual data types can be coded by fixed-depth proofs, implies polytime soundness.

Many interesting systems should be classified in-between these two styles. In these systems, recursion is embedded into typed calculi, and other mechanisms — usually ramification or linearity — are needed to control the computational complexity growth. Typical examples of such systems, here dubbed as type-theoretical, are **HOSLR** [8, 9] and **LT** [10, 11]. In the two systems mentioned, the syntax of Gödel's system **T** is modified to accommodate safe recursion, but a number of additional constraints, a restricted form of linearity *in primis*, are needed to guarantee polynomial soundness. Generalizations of ramified recurrence to higher-order types are presented in [12–14]. In these systems, however, the lack of any linearity constraint prevents from getting a polytime bound. Indeed, at higher types they show either a poly-space or a Kalmar elementary bound. Another related work is [15], where syntactical restrictions on a simply typed calculus with constants and recursion allow to restrict the space of representable functions to relevant complexity classes.

Results. In this paper, we introduce the system of Higher-Order Linear Ramified Recurrence (**HOLRR**). It is a type theoretical system smoothly blending both recursion and proof-theoretic components.

The proof-theoretical core of the system is a linear affine λ-calculus: any variable can be used at most once. Recursion is embedded in the system as a variable binder, whose syntax is inspired by boxes of linear lambda calculi. The types are generated by the usual multiplicative connectives (tensor and arrow). Base types includes denumerably many copies of several free algebras. There is no need for additional type constructs; in particular, there is no explicit modality.

Our principal aim is to obtain results akin to those of Bellantoni, Niggl and Schwichtenberg's **LT** [10, 11], but in a framework with a polynomial bound expressed as a function of specific parameters of the term. Sect. 5 analytically compares the two systems. Here, we stress that: (i) no additional syntactic restriction on terms is needed, besides those induced by typeability; (ii) the degree of the polynomial bounding normalization time of a term M depends only on one parameter of a type derivation for M — its recursion depth.

In particular, we prove a soundness result *à la* **LLL**. Under a given strategy, any term which can be typed satisfying two simple conditions (*word-contextuality* and *ramification*) normalizes in a polynomially bounded time. To be precise, we will prove that, for any (word-contextual and ramified) type derivation π for M, M normalizes in time $O(|M|^h)$, where h depends only on the recursion depth of π. This means that, whenever the recursion depth of type derivations for terms encoding input data is bounded, the defining function is polytime — a similar situation occurring in **LLL** or **LAL**.

Completeness for **PTIME** holds by embedding Leivant's ramified recurrence on words into **HOLRR**.

2 Syntax

A *free algebra* $\mathbb{A}$ is a couple $(\mathcal{C}_\mathbb{A}, \mathcal{R}_\mathbb{A})$ where $\mathcal{C}_\mathbb{A} = \{c_1^\mathbb{A}, \ldots, c_{k(\mathbb{A})}^\mathbb{A}\}$ is a finite set of *constructors* and $\mathcal{R}_\mathbb{A} : \mathcal{C}_\mathbb{A} \to \mathbb{N}$ maps every constructor to its *arity*. A free algebra $\mathbb{A} = (\{c_1^\mathbb{A}, \ldots, c_{k(\mathbb{A})}^\mathbb{A}\}, \mathcal{R}_\mathbb{A})$ is a *word algebra* if

- $\mathcal{R}(c_i^\mathbb{A}) = 0$ for one (and only one) $i \in \{1, \ldots, k(\mathbb{A})\}$;
- $\mathcal{R}(c_j^\mathbb{A}) = 1$ for every $j \neq i$ in $\{1, \ldots, k(\mathbb{A})\}$.

If $\mathbb{A} = (\{c_1^\mathbb{A}, \ldots, c_{k(\mathbb{A})}^\mathbb{A}\}, \mathcal{R}_\mathbb{A})$ is a word algebra, we will assume $c_{k(\mathbb{A})}^\mathbb{A}$ to be the distinguished element of $\mathcal{C}_\mathbb{A}$ whose arity is 0 and $c_1, \ldots, c_{k(\mathbb{A})-1}$ will denote the elements of $\mathcal{C}_\mathbb{A}$ whose arity is 1. $\mathbb{B} = (\{c_1^\mathbb{B}, c_2^\mathbb{B}, c_3^\mathbb{B}\}, \mathcal{R}_\mathbb{B})$ is the word algebra of binary strings. $\mathbb{C} = (\{c_1^\mathbb{C}, c_2^\mathbb{C}\}, \mathcal{R}_\mathbb{C})$, where $\mathcal{R}_\mathbb{C}(c_1^\mathbb{C}) = 2$ and $\mathcal{R}_\mathbb{C}(c_2^\mathbb{C}) = 0$ is the free algebra of binary trees.

$\mathscr{A}$ will be a fixed, finite family $\{\mathbb{A}_1, \ldots, \mathbb{A}_n\}$ of free algebras, where constructor sets $\mathcal{C}_{\mathbb{A}_1}, \ldots, \mathcal{C}_{\mathbb{A}_n}$ are assumed to be pairwise disjoint. We will hereby assume both $\mathbb{B}$ and $\mathbb{C}$ to be in $\mathscr{A}$.

The language $\mathscr{M}_\mathscr{A}$ of **HOLRR** *terms* is defined by the following productions:

$$M ::= x \mid c \mid (M, M) \mid MM \mid \lambda x.M \mid \textbf{let } (x, x) \Leftarrow M \textbf{ in } M \mid \\ \{\!\{M, \ldots, M\}\!\}[x/M, \ldots, x/M]\ M \mid \langle\!\langle M, \ldots, M \rangle\!\rangle[x/M, \ldots, x/M]\ M$$

where c ranges over the constructors for the free algebras in $\mathscr{A}$. An occurrence of a term N inside another term M has *recursion degree* n if it is nested into n terms in the form $\langle\!\langle M, \ldots, M \rangle\!\rangle$ inside M. When we write a term as $\overline{M}$, we are implicitly assuming it to be closed (i.e. to contain no free variables).

The language $\mathscr{T}_\mathscr{A}$ of **HOLRR** types is defined by the following productions:

$$A ::= B_\mathbb{A}^n \mid A \otimes A \mid A \multimap A$$

where n ranges over $\mathbb{N}$ and $\mathbb{A}$ ranges over $\mathscr{A}$. Tensor associates to the left, both in types and terms (that is, pairs). $A \in \mathscr{T}_\mathscr{A}$, define the *lifting* $\#(A) \in \mathscr{T}_\mathscr{A}$ of A:

$$\#(B_\mathbb{A}^n) = B_\mathbb{A}^{n+1}$$
$$\#(A \square B) = \#(A) \square \#(B) \text{ with } \square \in \{\otimes, \multimap\}.$$

The *level* $\mathbb{L}(A) \in \mathbb{N}$ of a type A is defined by induction on A:

$$\mathbb{L}(B_\mathbb{A}^n) = n$$
$$\mathbb{L}(A \otimes B) = \mathbb{L}(A \multimap B) = \max\{\mathbb{L}(A), \mathbb{L}(B)\}.$$

The *index set* $\mathbb{I}(A) \subseteq \mathbb{N}$ of A is defined in a similar way:

$$\mathbb{I}(B_\mathbb{A}^n) = \{n\}$$
$$\mathbb{I}(A \otimes B) = \mathbb{I}(A \multimap B) = \mathbb{I}(A) \cup \mathbb{I}(B).$$

The rules in Fig. 1 define the assignment of types in $\mathscr{T}_\mathscr{A}$ to terms in $\mathscr{M}_\mathscr{A}$. A type derivation π with conclusion $\Gamma \vdash M : A$ will be denoted by $\pi : \Gamma \vdash M : A$. If there is $\pi : \Gamma \vdash M : A$ then we will write $\Gamma \vdash_\mathbf{H} M : A$ and mark M as a *typeable*

$$\frac{}{x : A \vdash x : A}\ A \qquad \frac{\Gamma \vdash M : B}{\Gamma, x : A \vdash M : B}\ W$$

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : A \multimap B}\ I_{\multimap} \qquad \frac{\Gamma \vdash M : A \multimap B \quad \Delta \vdash N : A}{\Gamma, \Delta \vdash MN : B}\ E_{\multimap}$$

$$\frac{\Gamma \vdash M : A \quad \Delta \vdash N : B}{\Gamma, \Delta \vdash (M, N) : A \otimes B}\ I_{\otimes} \qquad \frac{\Gamma \vdash M : A \otimes B \quad \Delta, x : A, y : B \vdash N : C}{\Gamma, \Delta \vdash \ \mathbf{let}(x, y) \Leftarrow M\ \mathbf{in}\ N : C}\ E_{\otimes}$$

$$\frac{n \in \mathbb{N} \quad c \in \mathcal{C}_{\mathbb{A}}}{\vdash c : \underbrace{B^n_{\mathbb{A}} \multimap \ldots \multimap B^n_{\mathbb{A}}}_{\mathcal{R}_{\mathbb{A}}(c)\ \ times} \multimap B^n_{\mathbb{A}}}\ I_c$$

$$\frac{\begin{array}{c} A \equiv B^i_{\mathbb{A}} \qquad \Gamma \equiv x_1 : B_1, \ldots, x_n : B_n \\ \Gamma \vdash M_{c^{\mathbb{A}}_i} : \underbrace{A \multimap \ldots \multimap A}_{\mathcal{R}_{\mathbb{A}}(c^{\mathbb{A}}_i)\ \ times} \multimap C \qquad \Delta_i \vdash N_i : B_i \qquad \Theta \vdash L : A \end{array}}{\Delta_1, \ldots, \Delta_n, \Theta \vdash \{\!\{M_{c_1} \cdots M_{c_k}\}\!\}[x_1/N_1, \ldots, x_n/N_n]\ L : C}\ E^C_{\multimap}$$

$$\frac{\begin{array}{c} A \equiv B^i_{\mathbb{A}} \qquad \Gamma \equiv x_1 : B_1, \ldots, x_n : B_n \\ \Gamma \vdash M_{c^{\mathbb{A}}_i} : \underbrace{A \multimap \ldots \multimap A}_{\mathcal{R}_{\mathbb{A}}(c^{\mathbb{A}}_i)\ \ times} \multimap \underbrace{C \multimap \ldots \multimap C}_{\mathcal{R}_{\mathbb{A}}(c^{\mathbb{A}}_i)\ \ times} \multimap C \\ \Delta_i \vdash N_i : B_i \qquad \Theta \vdash L : A \end{array}}{\Delta_1, \ldots, \Delta_n, \Theta \vdash \langle\!\langle M_{c_1} \cdots M_{c_k} \rangle\!\rangle[x_1/N_1, \ldots, x_n/N_n]\ L : C}\ E^R_{\multimap}$$

Fig. 1. Type assignment rules

HOLRR term. $\mathscr{M}^{\mathbf{H}}_{\mathscr{A}}$ is the set of **HOLRR** typeable terms. A type derivation $\pi : \Gamma \vdash M : A$ is in *standard form* if Γ does not contain variables introduced by rule W.

The *recursion depth* $\mathbb{R}(\pi)$ of a **HOLRR** type derivation $\pi : \Gamma \vdash M : A$ is defined by induction on the structure of π. In particular:

- If π is an instance of rules A or I_c, then $\mathbb{R}(\pi) = 0$.
- If the last rule used in π is $E^R_{\multimap}$, then π has the following shape

$$\frac{\pi_1 \quad \ldots \quad \pi_m \quad \Theta \vdash L : B^i_{\mathbb{A}}}{\Delta, \Theta \vdash \langle\!\langle M_1, \ldots, M_n \rangle\!\rangle[x_1/N_1, \ldots, x_m/N_m]\ L : C}$$

  and $\mathbb{R}(\pi)$ is $i + \max\{\mathbb{R}(\pi_1), \ldots, \mathbb{R}(\pi_m)\}$.
- In all the other cases, π can be written as follows

$$\frac{\pi_1 \quad \ldots \quad \pi_m}{\Gamma \vdash M : A}\ .$$

  We will define $\mathbb{R}(\pi)$ as $\max\{\mathbb{R}(\pi_1), \ldots, \mathbb{R}(\pi_m)\}$.

Proposition 1. *If* $\Gamma \vdash_{\mathbf{H}} M : A$ *and* $\Delta, x : A \vdash_{\mathbf{H}} N : B$, *then* $\Gamma, \Delta \vdash_{\mathbf{H}} N\{M/x\} : B$.

Proof. Induction on the structure of the derivation for $\Delta, x : A \vdash N : B$. □

For every term t of a free algebra $\mathbb{A} \in \mathscr{A}$ and for every natural number n, there is an **HOLRR** type derivation $\pi(t, n) : \vdash t : B^n_{\mathbb{A}}$. This allows to prove:

Proposition 2. *If* $x_1 : A_1, \ldots, x_n : A_n \vdash_{\mathbf{H}} M : B$, *then* $x_1 : \#(A_1), \ldots, x_n : \#(A_n) \vdash_{\mathbf{H}} M : \#(B)$

The reduction rule $\rightarrow$ on $\mathscr{M}_{\mathscr{A}}$ is given in Fig. 2; $\rightsquigarrow$ is the contextual closure of $\rightarrow$. $\rightsquigarrow^*$ is locally confluent and strongly normalizable, property provable by embedding the calculus into system **T**; so, it is Church-Rosser as well.
Redexes in the form $\langle\!\langle M_{c_1}, \ldots, M_{c_k} \rangle\!\rangle [x_1/\overline{N_1}, \ldots, x_n/\overline{N_n}]\ t$ are called *recursive redexes*; those in the form $\{\!\{M_{c_1}, \ldots, M_{c_k}\}\!\}[x_1/N_1, \ldots, x_n/N_n]\ t$ are *conditional redexes*; all the others are called *linear redexes.*

$$(\lambda x.M)N \rightarrow M\{N/x\}$$

$$\mathbf{let}\ (x, y) \Leftarrow (M, N)\ \mathbf{in}\ L \rightarrow L\{M/x, N/y\}$$

$$\begin{array}{l} \langle\!\langle M_{c_1}, \ldots, M_{c_k} \rangle\!\rangle [x_1/\overline{N_1}, \ldots, x_n/\overline{N_n}]\ c_i(t_1, \ldots, t_{\mathcal{R}(c_i)}) \rightarrow \\ \quad M_{c_i}\{\overline{N_1}/x_1, \ldots, \overline{N_n}/x_n\}\ t_1 \cdots t_{\mathcal{R}(c_i)} \\ \qquad \langle\!\langle M_{c_1}, \ldots, M_{c_k} \rangle\!\rangle [x_1/\overline{N_1}, \ldots, x_n/\overline{N_n}]\ t_1 \\ \qquad \cdots \\ \qquad \langle\!\langle M_{c_1}, \ldots, M_{c_k} \rangle\!\rangle [x_1/\overline{N_1}, \ldots, x_n/\overline{N_n}]\ t_{\mathcal{R}(c_i)} \end{array}$$

$$\begin{array}{l} \{\!\{M_{c_1}, \ldots, M_{c_k}\}\!\}[x_1/N_1, \ldots, x_n/N_n]\ c_i(t_1, \ldots, t_{\mathcal{R}(c_i)}) \rightarrow \\ \quad M_{c_i}\{N_1/x_1, \ldots, N_n/x_n\}\ t_1 \cdots t_{\mathcal{R}(c_i)} \end{array}$$

Fig. 2. Normalization on terms

The following proposition will be useful in the following

Proposition 3. *If* $\vdash M : B^n_{\mathbb{A}}$, *then the (unique) normal form of* M *is a free algebra term* t. *Moreover,* t *can be obtained from* M *by successively firing redexes with null recursion degree.*

Proof. By a standard reducibility argument. □

$\mathscr{M}^{\mathbf{H}}_{\mathscr{A}}$ contains terms that cannot be reduced in polynomial time. To enforce this property, we introduce the following two conditions on type derivations:

- a type derivation π is *word-contextual* if every occurrence of B_i in every instance of $E^R_{-\circ}$, inside π, has form $B^m_{\mathbb{W}}$, $\mathbb{W}$ being a *word* algebra.
- a type derivation π is *ramified* if every instance of $E^R_{-\circ}$ inside π satisfies $\mathbb{L}(A) > \mathbb{L}(C)$.

In the following section, we will show that these conditions are both crucial to reach polytime soundness. If $\pi : \Gamma \vdash M : A$, where π is word-contextual and ramified, M is said to be *word-ramified* and we will write $\pi : \Gamma \vdash_{\mathbf{WR}} M : A$. The class of all word-ramified **HOLRR** terms will be denoted as $\mathscr{M}_{\mathscr{A}}^{\mathbf{WR}}$.

3 Polytime Soundness

The goal is to prove polytime soundness for **HOLRR** in the form of

Theorem 1. *There is a sound and complete normalization strategy such that the time required to normalize a term M is $O(|M|^h)$ where h only depends on $\mathbb{R}(\pi)$, $\pi : \Gamma \vdash M : A$ being word-contextual and ramified.*

The reduction strategy we use proceeds by firing the rightmost innermost redex among those with minimum recursion degree, where the firing of a recursive redex corresponds to a complete unfolding, counted as a single step. *Rightmost innermost minimum recursion degree strategy* is the name of such a reduction strategy, and $M \mapsto N$ denotes that M rewrites to N by one of its possible steps. We will prove Theorem 1 studying normalization by way of interaction graphs, which are graphs corresponding to **HOLRR** type derivations. Notice that we will not use interaction graphs as a virtual machine computing normal forms — they are merely a tool facilitating the study of **HOLRR** dynamics.
Let $\mathscr{L}_{\mathscr{A}}$ be the set

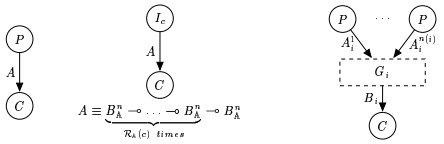
$$\{W, I_{\multimap}, E_{\multimap}, I_{\otimes}, E_{\otimes}, P, C\} \cup \bigcup_{\mathbb{A} \in \mathscr{A}} \bigcup_{c \in \mathcal{C}_{\mathbb{A}}} \{I_c\} \cup \{E^C_{\multimap}, E^R_{\multimap}, P^C, P^R\}.$$

Elements of $\mathscr{L}_{\mathscr{A}}$ either are typing rule names or lie in $\{P, C, P^C, P^R\}$ — they are premises (P), conclusions (C) or limit conditionals (P^C) and recursions (P^R).
An *interaction graph* is a quadruple (V, E, α, β) such that

- (V, E) is a directed graph;
- $\alpha : V \to \mathscr{L}_{\mathscr{A}}$
- $\beta : E \to \mathscr{T}_{\mathscr{A}}$

$\mathscr{G}_{\mathscr{A}}$ is the set of all interaction graphs. We will now introduce a class $\mathscr{G}^{\mathbf{H}}_{\mathscr{A}}$ of interaction graphs corresponding to **HOLRR** type derivations. $\mathscr{G}^{\mathbf{H}}_{\mathscr{A}}$ is defined inductively, mimicking the process of type derivation building. First, the interaction graphs in figure 3(a) lie in $\mathscr{G}^{\mathbf{H}}_{\mathscr{A}}$. Moreover, suppose $G_0, \ldots, G_{k(\mathbb{A})+n} \in \mathscr{G}^{\mathbf{H}}_{\mathscr{A}}$ and they have form as in Fig. 3(b); then all the interaction graphs depicted in Fig. 4 lie in $\mathscr{G}^{\mathbf{H}}_{\mathscr{A}}$, provided the constraints listed next to each graph are satisfied.
To every **HOLRR** type derivation $\pi : \Gamma \vdash M : A$ corresponds an interaction graph $\mathcal{G}(\pi) \in \mathscr{G}^{\mathbf{H}}_{\mathscr{A}}$. Moreover, every instance of rules $I_{\multimap}, E_{\multimap}, I_{\otimes}, E_{\otimes}, I_c, E^C_{\multimap}, E^R_{\multimap}$ in π corresponds to a vertex in $\mathcal{G}(\pi)$ having the same label. In particular, if v corresponds to an instance $\dfrac{}{\vdash c : B^n_{\mathbb{A}} \multimap \ldots \multimap B^n_{\mathbb{A}}}$ of rule I_c, then $\delta(v)$ is the integer n, and, if this occurrence of c has recursion degree m, then $\gamma(v)$ is m.

Lemma 1. *There are two constants $n, m \in \mathbb{Q}$ such that, for every $\pi : \Gamma \vdash M : A$ in standard form, we have $n|M| \leq |V| \leq m|M|$, where $\mathcal{G}(\pi) = (V, E, \alpha, \beta)$.*



(a) (b)

Fig. 3. Some interaction graphs.

Inside a given graph, we call *traps* those subgraphs corresponding to normal form derivations $\pi(t,n) :\vdash t : B^n_{\mathbb{W}}$ (where $\mathbb{W}$ is a word algebra) and ending on the last $E_{\multimap}$ (for instance, Fig. 5 shows the trap corresponding to $\pi(c^{\mathbb{B}}_1 c^{\mathbb{B}}_2 c^{\mathbb{B}}_1 c^{\mathbb{B}}_3, 0)$. We are here interested in certain paths inside interaction graphs: given an interaction graph $G = (V, E, \alpha, \beta)$, an *n-typed path* of G is a sequence $\overline{v} = v_1, \ldots, v_m \in V^+$ such that the two following conditions hold:

- for every $i \in \{1, \ldots, m-1\}$, either $(v_i, v_{i+1}) \in E$ and $n \in \mathbb{I}(\beta(v_i, v_{i+1}))$ or $(v_{i+1}, v_i) \in E$ and $n \in \mathbb{I}(\beta(v_{i+1}, v_i))$, and
- for every $i \in \{1, \ldots, m-1\}$, if v_i is part of a trap, then $v_{i+1}, \ldots, v_m$ must all be part of the same trap.

Intuitively, when a typed path enters a trap, it cannot exit it.
Suppose v is a vertex of G, ϕ is a positive integer and ψ is a nonnegative integer. Then the *weight* $W_{\phi,\psi}(v)$ of v is defined by cases:

$$W_{\phi,\psi}(v) = \begin{cases} 1 & \text{if } \alpha(v) = I_c \wedge \delta(v) \geq \psi \\ 0 & \text{if } \alpha(v) \neq I_c \\ \phi^{\gamma(v)(\psi+1)^{\psi - \delta(v)}} & \text{if } \alpha(v) = I_c \wedge \delta(v) < \psi \end{cases}$$

The weight $W_{\phi,\psi}(\overline{v})$ of a n-typed path $\overline{v} = v_1, \ldots, v_m$ is $\sum_{v \in \{v_1, \ldots, v_m\}} W_{\phi,\psi}(v)$, where every vertex counts once even if it occurs many times in $\overline{v}$. The *n-weight* $W^n_{\phi,\psi}(G)$ of an interaction graph G is just the maximum among $W_{\phi,\psi}(\overline{v})$ over all n-typed paths $\overline{v}$ inside G. The weight of an interaction graph is parametric on ϕ and ψ. The following result, however, holds for every ϕ and ψ.

Lemma 2. *Let $\pi_M : \Gamma \vdash M : A$ and suppose π_M contains a subderivation in the form $\pi(t,n)$. Then $W^n_{\phi,\psi}(\mathcal{G}(\pi_M)) \geq |t|$.*

Remark 1. Basic observations are worth doing to understand the proof of proposition 4 below. The goal is to understand how $W^n_{\phi,\psi}(\mathcal{G}(\pi_N))$ and $W^n_{\phi,\psi}(\mathcal{G}(\pi_M))$ relate each other, when $\pi_M : \Gamma \vdash_{\mathbf{WR}} M : A$, and $\pi_N : \Gamma \vdash_{\mathbf{WR}} N : A$, and M rewrites to N.
Rewriting on terms, as in Fig. 2, is matched by certain transformations on the corresponding graphs, described in Fig. 6. The graph transformations take into account the modifications on the graphs, but for the erasure of sub-terms, which

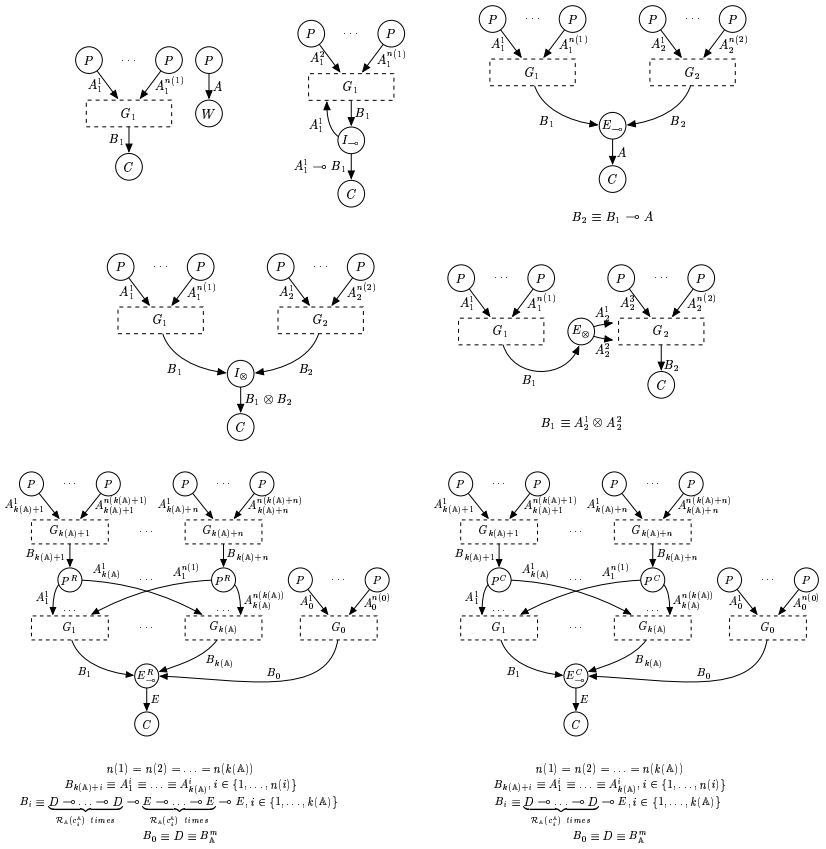


Fig. 4. Inductive cases

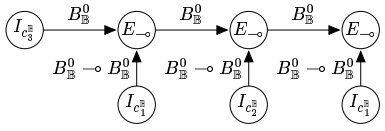


Fig. 5. The trap corresponding to $\pi(c_1^{\mathbb{B}} c_2^{\mathbb{B}} c_1^{\mathbb{B}} c_3^{\mathbb{B}}, 0)$.

is the computational effect of weakening. When describing the modifications on the graphs induced by the firing of a redex, it is always understood that after any transformation as in Fig. 6, one should also perform all those transformations which correspond to the deletion of a sub-term as caused by a substitution for a weakened variable. These transformations can always be written as the one in Fig. 7, where G only depends on the term being deleted. We remark once again

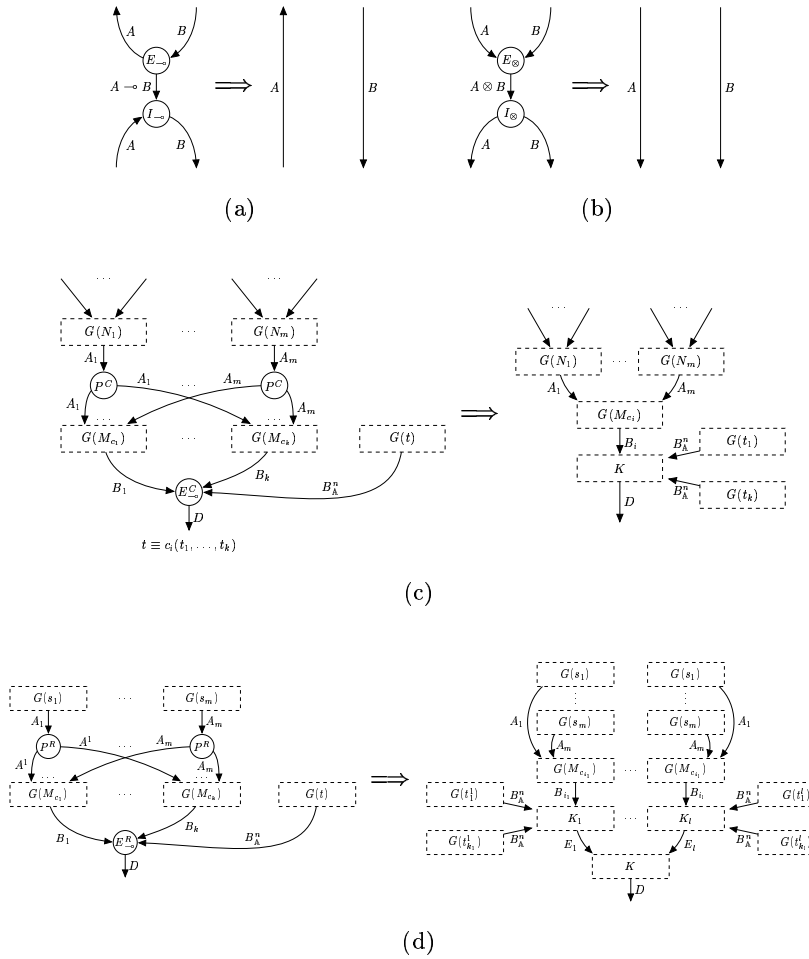


Fig. 6. The graph transformations produced by the firing of a redex.

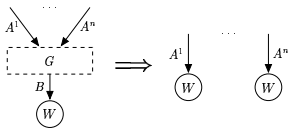


Fig. 7. The graph transformation induced by the substitution for a weakened variable.

that we use graphs as a mere tool for the study of the complexity of reduction, and not as a kind of computational device implementing reduction.

As a first case, assume M yields N by firing a linear redex. Then, $\mathcal{G}(\pi_M)$ transforms to $\mathcal{G}(\pi_N)$ by one between the rules in Fig. 6(a) or Fig. 6(b). Then, $\mathcal{G}(\pi_N)$ has less vertices than $\mathcal{G}(\pi_M)$ and for every n-typed path $\overline{v}$ in $\mathcal{G}(\pi_N)$, there is a corresponding n-typed path $\overline{w}$ in $\mathcal{G}(\pi_M)$, with $W_{\phi,\psi}(\overline{v}) \leq W_{\phi,\psi}(\overline{w})$.
Assume, instead, to fire a conditional redex, namely $\{\!\{M_{c_1}, \ldots, M_{c_k}\}\!\}[x_1/N_1, \ldots, x_m/N_m]\ t$. At the graph level we need to focus on the transformation in Fig. 6(c), where K will contain at most k nodes, all labeled with $E_{\multimap}$, while $t_1, \ldots, t_k$ all are sub-terms of t. Again, $\mathcal{G}(\pi_N)$ has less vertices than $\mathcal{G}(\pi_M)$ and for every n-typed path $\overline{v}$ in $\mathcal{G}(\pi_N)$, there is a corresponding n-typed path $\overline{w}$ in $\mathcal{G}(\pi_M)$, with $W_{\phi,\psi}(\overline{v}) \leq W_{\phi,\psi}(\overline{w})$.
Finally, assume to fire a recursive redex. By proposition 3, it must be in the form $\langle\!\langle M_{c_1}, \ldots, M_{c_k}\rangle\!\rangle[x_1/s_1, \ldots, x_m/s_m]\ t$, where $t, s_1, \ldots, s_m$ are free-algebra terms. At the graph level, the transformation behaves as in Fig. 6(d), where:

- For every $i \in \{1, \ldots, l\}$, there is a constructor $c \in \mathcal{C}_{\mathbb{A}}$ such that $c(t_1^i \ldots t_{k_i}^i)$ is a sub-term of t;
- $K, K_1, \ldots, K_l$ contain nodes v such that $\alpha(v) = E_{\multimap}$;
- $|t|$ bounds both l and the number of vertices in K;
- For every $i \in \{1, \ldots, l\}$, the number of vertices in K_i is bounded by k_i;
- $E_1, \ldots, E_l$ all are in the form $D \multimap \ldots \multimap D$.

If $p \geq n$, then $W^p_{\phi,\psi}(\mathcal{G}(\pi_N)) \leq W^p_{\phi,\psi}(\mathcal{G}(\pi_M))$: for every p-typed path $\overline{v}$ inside $\mathcal{G}(\pi_N)$, there is a corresponding p-typed path $\overline{w}$ inside $\mathcal{G}(\pi_M)$, where $W_{\phi,\psi}(\overline{v}) \leq W_{\phi,\psi}(\overline{w})$. Assume now that $p < n$. Certainly, any p-typed path $\overline{v}$ inside $\mathcal{G}(\pi_N)$ can be mimicked by a p-typed path $\overline{w}$ inside $\mathcal{G}(\pi_M)$ in such a way that a constructor vertex in $\overline{w}$ corresponds to every constructor vertex appearing in $\overline{v}$. This correspondence, however, is not injective. Whenever u is a vertex appearing in $\overline{w}$ and belonging to $G(M_{c_i})$, $\overline{v}$ can contain distinct $u_1, \ldots, u_j$ (where $j \leq l$), all of them being "copies" of u. On the other hand, all the equations $\gamma(u_1) = \ldots = \gamma(u_j) = \gamma(u) - 1$ hold. Notice that, by our definition of a p-typed path, if u belongs to the trap $G(s_i)$, then $\overline{v}$ can only contain *one* copy of u.

The remarks here above lead to the following, crucial, result:

Proposition 4. *There is a function $f : \mathbb{N} \to \mathbb{N}$ such that, for every word-contextual and ramified $\pi_M : \Gamma \vdash M : A$, if $M \mapsto^* N$ and t is a free algebra term appearing in N, then $|t| = O(|M|^{f(\mathbb{R}(\pi_M))})$.*

Proof. Let $\mathcal{G}(\pi_M) = (V, E, \alpha, \beta)$. We will prove that, for every $n, m \in \mathbb{N}$, if $M \mapsto^n N$, then

$$W^m_{|V|,\mathbb{R}(\pi_M)}(\mathcal{G}(\pi_N)) \leq W^m_{|V|,\mathbb{R}(\pi_M)}(\mathcal{G}(\pi_M)) \tag{1}$$

$$W^m_{|V|,\mathbb{R}(\pi_M)}(\mathcal{G}(\pi_M)) \leq \begin{cases} |V| & \text{if } \mathbb{R}(\pi_M) \leq m \\ |V|^{(\mathbb{R}(\pi_M)+1)^{\mathbb{R}(\pi_M)-m+1}} & \text{otherwise} \end{cases} \tag{2}$$

First of all, let us consider the case $n = 0$. N equals M, so (1) is trivially verified. Suppose $\overline{v} = v_1, \ldots, v_p$ be an m-typed path inside $\mathcal{G}(\pi_M)$. If $m \leq \mathbb{R}(\pi)$ then, by

definition, $W_{|V|,\mathbb{R}(\pi_M)}(\overline{v}) \leq |V|$. If $m > \mathbb{R}(\pi)$, then

$$\begin{aligned} W_{|V|,\mathbb{R}(\pi_M)}(\overline{v}) &\leq |V||V|^{\mathbb{R}(\pi_M)(\mathbb{R}(\pi_M)+1)^{\mathbb{R}(\pi_M)-m}} \\ &\leq |V|^{\mathbb{R}(\pi_M)(\mathbb{R}(\pi_M)+1)^{\mathbb{R}(\pi_M)-m}+1} \\ &\leq |V|^{(\mathbb{R}(\pi_M)+1)(\mathbb{R}(\pi_M)+1)^{\mathbb{R}(\pi_M)-m}} \\ &= |V|^{(\mathbb{R}(\pi_M)+1)^{\mathbb{R}(\pi_M)-m+1}} \end{aligned}$$

As a consequence, (2) holds.
Suppose now that $n > 0$ and that the thesis holds for $n-1$. By remarks 1 and by the induction hypothesis, we only have to show that (1) is preserved by recursion unfolding — in other cases, path weights cannot increase. If $m \geq \mathbb{R}(\pi_M)$, then even recursion unfolding do not increase the weight of m-typed paths. If $m < \mathbb{R}(\pi_M)$, let $\overline{w}$ be an m-typed path in $\mathcal{G}(\pi_N)$ and let $\overline{v}$ the m-typed path in $\mathcal{G}(\pi_M)$ that corresponds to $\overline{w}$. As discussed previously, for every vertex u appearing in $\overline{v}$, $\overline{w}$ may contain several distinct vertices $u_1, \ldots, u_j$, all corresponding to u. By lemma 2, $j \leq W^{m+1}_{|V|,\mathbb{R}(\pi_M)}(\mathcal{G}(\pi_M))$ and by the induction hypothesis

$$\begin{aligned} \sum_{i=1}^{j} W_{|V|,\mathbb{R}(\pi_M)}(u_j) &= \sum_{i=1}^{j} |V|^{\gamma(u_i)(\mathbb{R}(\pi_M)+1)^{\mathbb{R}(\pi_M)-m}} \\ &= \sum_{i=1}^{j} |V|^{(\gamma(u)-1)(\mathbb{R}(\pi_M)+1)^{\mathbb{R}(\pi_M)-m}} \\ &\leq \left(|V|^{(\mathbb{R}(\pi_M)+1)^{\mathbb{R}(\pi_M)-m}}\right)\left(|V|^{(\gamma(u)-1)(\mathbb{R}(\pi_M)+1)^{\mathbb{R}(\pi_M)-m}}\right) \\ &= |V|^{\gamma(u)(\mathbb{R}(\pi_M)+1)^{\mathbb{R}(\pi_M)-m}} = W_{|V|,\mathbb{R}(\pi_M)}(u). \end{aligned}$$

This, by lemma 1, concludes the proof. □

Proposition 5. *There is a function $g : \mathbb{N} \to \mathbb{N}$ such that, if $\pi_M : \Gamma \vdash M : A$ is word-contextual and ramified, the number of recursive redexes fired during normalization is $O(|M|^{g(\mathbb{R}(\pi_M))})$.*

Proof. A recursive redex with recursion degree m can be copied $O(|M|^{mf(\mathbb{R}(\pi_M))})$ times during normalization, as can be proved from proposition 4 by induction on m. Now, notice that $m \leq \mathbb{R}(\pi_M)$. As a consequence, the function $g(n) = nf(n) + 1$ is a suitable bound. □

Summing up, proposition 4 gives a bound on the size of free algebra terms appearing inside reducts of a given (word-ramified) term M. This is proved by showing that (for every $n \in \mathbb{N}$) the n-weight of the underlying interaction graph does not increase during normalization. This result, by itself, does not prove anything on the complexity of normalization. Proposition 5, however, exploits it by bounding the total number of recursive redexes fired during normalization of M. So, the proof of Theorem 1 can follow. From proposition 5, the number of recursive redexes the normalization fires is $O(|M|^{g(\mathbb{R}(\pi_M))})$, where $g : \mathbb{N} \to \mathbb{N}$

does not depend on $|M|$. By proposition 4, the time to unfold a recursive redex is itself $O(|M|^{f(\mathbb{R}(\pi_M))})$, where $f : \mathbb{N} \to \mathbb{N}$ does not depend on $|M|$. Finally, notice that, by firing a linear or conditional redex, the underlying interaction graph shrinks. This concludes the proof.

4 Polytime Completeness

This property holds by representing *predicative sorting* into **HOLRR**. Predicative sorting, introduced below, reformulates *ramified recurrence* (or *predicative recursion*) *on words* [2]; given a word algebra $\mathbb{W}$, predicative recursion is a function algebra generating all, and only, the polynomial functions in the form $f : \mathbb{W}^n \to \mathbb{W}$. *Predicative sorting* on $\mathbb{W}$ follows:

1. The function $f_{c^{\mathbb{W}}_{k(\mathbb{W})}} : \mathbb{W}^0 \to \mathbb{W}$ that returns $c^{\mathbb{W}}_{k(\mathbb{W})}$ can be predicatively sorted by $\varepsilon \to n$, for every $n \in \mathbb{N}$, ε being the empty sequence;
2. For every $i \in \{1, \ldots, k(\mathbb{W}) - 1\}$, the function $f_{c^{\mathbb{W}}_i} : \mathbb{W} \to \mathbb{W}$ defined by $f_{c^{\mathbb{W}}_i}(t) = c^{\mathbb{W}}_i\ t$ can be predicatively sorted by $(n) \to n$ for every $n \in \mathbb{N}$;
3. For every $n \in \mathbb{N}$ and $1 \leq i \leq n$, the projection $\pi^n_i : \mathbb{W}^n \to \mathbb{W}$ can be predicatively sorted by $(m_1, \ldots, m_n) \to m$ for every $m, m_1, \ldots, m_n \in \mathbb{N}$, with $m_i = m$;
4. If $f : \mathbb{W}^n \to \mathbb{W}$ can be predicatively sorted by $(m_1, \ldots, m_n) \to m$ and $g_1, \ldots, g_n : \mathbb{W}^p \to \mathbb{W}$ are such that g_i can be predicatively sorted by $(r_1, \ldots, r_p) \to m_i$, then the function $h : \mathbb{W}^p \to \mathbb{W}$ defined by the equation
$$h(t_1, \ldots, t_p) = f(g_1(t_1, \ldots, t_p), \ldots, g_n(t_1, \ldots, t_p)))$$
can be predicatively sorted by $(r_1, \ldots, r_p) \to m$;
5. Suppose for every $i \in \{1, \ldots, k(\mathbb{W}) - 1\}$ there is a function $f_i : \mathbb{W}^{n+1} \to \mathbb{W}$ that can be predicatively sorted by $(l, m_1, \ldots, m_n) \to m$ and that $f_{k(\mathbb{W})} : \mathbb{W}^n \to \mathbb{W}$ can be predicatively sorted by $(m_1, \ldots, m_n) \to m$. Then the function $h : \mathbb{W}^{1+n} \to \mathbb{W}$ defined by
$$h(c^{\mathbb{W}}_i\ t, t_1, \ldots, t_n) = f_i(t, t_1, \ldots, t_n,)$$
$$h(c^{\mathbb{W}}_{k(\mathbb{W})}, t_1, \ldots, t_n) = f_{k(\mathbb{W})}(t_1, \ldots, t_n).$$
can be predicatively sorted by $(l, m_1, \ldots, m_n) \to m$.
6. Suppose for every $i \in \{1, \ldots, k(\mathbb{W}) - 1\}$ there is a function $f_i : \mathbb{W}^{n+2} \to \mathbb{W}$ that can be predicatively sorted by $(l, m_1, \ldots, m_n, m) \to m$ and that $f_{k(\mathbb{W})} : \mathbb{W}^n \to \mathbb{W}$ can be predicatively sorted by $(m_1, \ldots, m_n) \to m$. Then a function $h : \mathbb{W}^{1+n} \to \mathbb{W}$ can be defined recursively:
$$h(c^{\mathbb{W}}_i\ t, t_1, \ldots, t_n) = f_i(t, t_1, \ldots, t_n, h(t, t_1, \ldots, t_n))$$
$$h(c^{\mathbb{W}}_{k(\mathbb{W})}, t_1, \ldots, t_n) = f_{k(\mathbb{W})}(t_1, \ldots, t_n).$$
If $l > m$, then h can be predicatively sorted by $(l, m_1, \ldots, m_n) \to m$.

By the definition here above, if $f : \mathbb{W}^n \to \mathbb{W}$ can be predicatively sorted, then it is *definable by predicative recursion.*

Remark 2. If f can be predicatively sorted by $(m_1, \ldots, m_n) \to m$ and $m_i < m$, then f is independent from its i-th argument (see [8]). We will suppose that, in rule 3, $m_1, \ldots, m_n \geq m$. This ensures that, if f can be predicatively sorted by $(m_1, \ldots, m_n) \to m$, then $m_1, \ldots, m_n \geq m$, simplifying the proof of completeness, without loss of generality.

Theorem 2 (Completeness). *Assume $f : \mathbb{W}^n \to \mathbb{W}$ be predicatively sorted by $(m_1, \ldots, m_n) \to m$. There is a closed term M_f that represents f, whose type can be $B_{\mathbb{W}}^{l_1} \otimes \cdots \otimes B_{\mathbb{W}}^{l_n} \multimap B_{\mathbb{W}}^{l}$, where $l, l_1, \ldots, l_n \in \mathbb{N}$ and, for every $i \in \{1, \ldots, n\}$, either $m_i = m$ and $l_i = l$, or $m_i > m$ and $l_i > l$.*

The proof uses the definition of the terms:

$$\begin{aligned} \mathbf{Coerc} &: B_{\mathbb{W}}^{n} \multimap B_{\mathbb{W}}^{m} \\ \mathbf{Duplicate} &: B_{\mathbb{W}}^{n} \multimap B_{\mathbb{W}}^{m} \otimes B_{\mathbb{W}}^{l} \\ \nabla(M) &: B_{\mathbb{W}}^{i_1} \otimes \cdots \otimes B_{\mathbb{W}}^{i_p} \otimes B_{\mathbb{W}}^{n} \multimap B_{\mathbb{W}}^{n} \end{aligned}$$

such that:

$$\begin{aligned} \mathbf{Coerc}(t) &\rightsquigarrow^* t \\ \mathbf{Duplicate}(t) &\rightsquigarrow^* (t, t) \\ \nabla(M)(t_1, \ldots, t_p, t) &\rightsquigarrow^* M(t_1, \ldots, t_p, t, t) \end{aligned}$$

where $t : B_{\mathbb{W}}^{n}$, $t_j : B_{\mathbb{W}}^{i_j}$ $(j \in \{1, \ldots, p\})$, $M : B_{\mathbb{W}}^{i_1} \otimes \cdots \otimes B_{\mathbb{W}}^{i_p} \otimes B_{\mathbb{W}}^{n} \otimes B_{\mathbb{W}}^{n} \multimap B_{\mathbb{W}}^{n}$, $n \in \mathbb{N}$ and $m, l < n$. In particular:

- **Coerc** is $\lambda x.\langle\!\langle \lambda y.c_1^{\mathbb{W}} y, \ldots, \lambda y.c_{k(A)-1}^{\mathbb{W}} y, c_{k(A)}^{\mathbb{W}} \rangle\!\rangle\, x$;
- **Duplicate** is $\lambda x.\langle\!\langle M_1, \ldots, M_{k(\mathbb{W})} \rangle\!\rangle\, x$, where, for every $i \in \{1, \ldots, k(\mathbb{W}) - 1\}$, M_i is $\lambda y.$ **let** $(z, w) \Leftarrow y$ **in** $(c_i^{\mathbb{W}} z, c_i^{\mathbb{W}} w)$ and $M_{k(\mathbb{W})}$ is $(c_{k(\mathbb{W})}^{\mathbb{W}}, c_{k(\mathbb{W})}^{\mathbb{W}})$;
- $\nabla(M)$ is

  $$\lambda y.\ \mathbf{let}\ (y_1, \ldots, y_n, w) \Leftarrow y\ \mathbf{in}\ \langle\!\langle L, \ldots, L, P \rangle\!\rangle [x_1/y_1, \ldots, x_n/y_n, z/w]\ c_1^{\mathbb{W}} c_{k(\mathbb{W})}^{\mathbb{W}}$$

  where $L = \lambda x.\lambda y.M(x_1, \ldots, x_n, z, y)$ and $P = z$.

5 Comparison with Previous Work

There are a number of type systems with the same goal as **HOLRR** [10, 11, 16, 9]. The most similar is certainly **LT**, introduced in [10] and later refined in [11]. **HOLRR** and **LT** are designed from different starting points. **LT** is basically a *restriction* of Gödel system **T**, extending the ideas of safe recursion [1] to the higher-order. **HOLRR**, on the other hand, is obtained by *endowing* linear affine lambda calculus with constants, conditionals and recursions, somehow being inspired by Leivant's ramified recurrence on words.
Linearity is a key ingredient to control the complexity of normalization, in presence of higher-order recursion. The terms of **LT** are not strictly linear: free variables of ground types can appear more than once. On the other side, any

variable of **HOLRR** occurs at most once in a typeable term, and recursion preserves this constraint. The strict linearity of **HOLRR** fits precisely with the introduction of linear arrows, when discharging an assumption.
Ramification and safety are other tools to get rid of exponential growth. **LT** models safety by distinguishing among complete and incomplete variables and by using two families of arrows and products, with careful constraints on their interplay. **HOLRR** ramification has the same flavor as in the original work on ramified recurrence on words [2, 3], without any major change.
HOLRR accommodates generic free algebras in a uniform way, with just one recursion scheme. **LT** is only about word algebras: the introduction of tree algebras would require to extend the linear discipline to ground variables [9].
In both cases, the system is polytime complete. However, polytime soundness is formulated and proved in two different ways. In **LT**, any term M with free variables $x_1, \ldots, x_n$ is equipped with a polynomial $P_M(y_1, \ldots, y_n)$ in such a way that the time to normalize $M\{N_1/y_1, \ldots, N_n/y_n\}$ is $O(P_M(|N_1|, \ldots, |N_n|))$; this result, however, relies on a number of assumptions: all the terms involved must have linear type, $N_1, \ldots, N_n$ must all be closed and cannot contain complete free variables of higher type, all free variables of $M\{N_1/y_1, \ldots, N_n/y_n\}$ have to be linear and incomplete. This means that there is no evident relation between the structure of M and the degree of P_M. On the contrary, the time needed to compute the normal form of every word-contextual term M of **HOLRR** is $O(|M|^h)$, h only depending on the recursion depth of a type derivation for M. In particular, the recursion depth of any type derivation for any term of any free algebra is null. We claim that our soundness theorem is deeper and more general than the one on **LT**.

6 Relaxing Conditions on Type Derivations

If we drop word-contextuality and ramification, we immediately get outside **PTIME**. For example, if we allow $\mathbb{L}(A)$ to be equal to $\mathbb{L}(C)$ in rule $E^R_{\multimap}$, we can build a term M with $\vdash M : B^0_{\mathbb{B}} \multimap B^0_{\mathbb{B}}$ such that

$$M\ c^{\mathbb{B}}_{i_1} \ldots c^{\mathbb{B}}_{i_n} c^{\mathbb{B}}_3 \rightsquigarrow^* c^{\mathbb{B}}_{i_1} c^{\mathbb{B}}_{i_1} c^{\mathbb{B}}_{i_2} c^{\mathbb{B}}_{i_2} \ldots c^{\mathbb{B}}_{i_n} c^{\mathbb{B}}_{i_n} c^{\mathbb{B}}_3.$$

Iterating M, we easily obtain an exponential behavior. Assume now that, in rule $E^R_{\multimap}$, $B_1, \ldots, B_n$ are arbitrary types, namely that type derivations are not word-contextual. The term $\lambda x.\lambda y.\lambda z.x(yz)$ encoding function composition can be given type $(B^0_{\mathbb{B}} \multimap B^0_{\mathbb{B}}) \multimap (B^0_{\mathbb{B}} \multimap B^0_{\mathbb{B}}) \multimap (B^0_{\mathbb{B}} \multimap B^0_{\mathbb{B}})$; using the obvious generalization of ∇, we obtain a term N with type $(B^0_{\mathbb{B}} \multimap B^0_{\mathbb{B}}) \multimap (B^0_{\mathbb{B}} \multimap B^0_{\mathbb{B}})$ encoding self application. Again, iterating N yields an exponential blow up. The same problem occurs by starting from $c^{\mathbb{C}}_1$ with type $B^0_{\mathbb{C}} \multimap B^0_{\mathbb{C}} \multimap B^0_{\mathbb{C}}$.

7 Conclusions

We provide a higher-order system that embeds, quite naturally, Leivant's ramified recurrence on words.

A final remark about soundness follows. If $\pi_M : \Gamma \vdash M : A$ is a word-contextual ramified derivation, we obtain a bound $O(|M|^{f(\mathbb{R}(\pi_M))})$, for some suitable f. Now, the exponent *does depend* on M. But suppose $\pi_M : \Gamma \vdash M : A \multimap B$ and $\pi_N : \Delta \vdash N : A$ to be word-contextual and ramified. The type derivation $\pi_{MN} : \Gamma, \Delta \vdash MN : B$ is word-contextual and ramified, and $\mathbb{R}(\pi_{MN}) = \max\{\mathbb{R}(\pi_M), \mathbb{R}(\pi_N)\}$. Taking M as a program, the time to compute M on argument N is $O(|MN|^{f(\mathbb{R}(\pi_{MN}))})$ — a polynomial on $|N|$ whenever inputs to M have bounded recursion depth. This includes all the cases where inputs are closed normal forms of a base type. Future work addresses the characterization of higher-order types whose normal forms all have the same recursion depth.

References

1. Bellantoni, S., Cook, S.: A new recursion-theoretic characterization of the polytime functions. Computational Complexity **2** (1992) 97–110
2. Leivant, D.: Stratified functional programs and computational complexity. In: Proceedings of 20th ACM Symposium on Principles of Programming Languages. (1993) 325–333
3. Leivant, D.: Ramified recurrence and computational complexity I: word recurrence and poly-time. In: Feasible Mathematics II. Birkhäuser (1995) 320–343
4. Girard, J.Y.: Light linear logic. Information and Computation **143(2)** (1998) 175–204
5. Asperti, A.: Light affine logic. In: Proceedings of the 13th IEEE Syposium on Logic in Computer Science. (1998) 300–308
6. Asperti, A., Roversi, L.: Intuitionistic light affine logic. ACM Transactions on Computational Logic **3(1)** (2002) 137–175
7. Lafont, Y.: Soft linear logic and polynomial time. Theoretical Computer Science (To appear)
8. Hofmann, M.: Type systems for polynomial-time computation. Habilitationsschrift, Darmstadt University of Technology (1999)
9. Hofmann, M.: Safe recursion with higher types and BCK-algebra. Annals of Pure and Applied Logic **104** (2000) 113–166
10. Bellantoni, S., Niggl, K.H., Schwichtenberg, H.: Higher type recursion, ramification and polynomial time. Annals of Pure and Applied Logic **104** (2000) 17–30
11. Bellantoni, S., Schwichtenberg, H.: Feasible computation with higher types. Marktoberdorf Summer School Proceedings (2001)
12. Leivant, D., Marion, J.Y.: Ramified recurrence and computational complexity IV: Predicative functionals and poly-space. Information and Computation (To appear)
13. Leivant, D., Marion, J.Y.: Ramified recurrence and computational complexity II: Substitution and poly-space. In: Proceedings of 8th International Workshop on Computer Science Logic. (1994) 486–500
14. Leivant, D.: Ramified recurrence and computational complexity III: Higher type recurrence and elementary complexity. Annals of Pure and Applied Logic **96** (1999) 209–229
15. Leivant, D.: Applicative control and computational complexity. In: Proceedings of 13th International Workshop on Computer Science Logic. (1999) 82–95
16. Hofmann, M.: Linear types and non-size-increasing polynomial time computation. In: Proceedings of the 14th IEEE Syposium on Logic in Computer Science. (1999) 464–473