Axiomatic Rewriting Theory II

The $\lambda\sigma$-calculus enjoys finite normalisation cones

Paul-André Melliès
Preuves, Programmes, Systèmes
CNRS EPJ-2025, Université Paris 7
<mellies@pps.jussieu.fr>

Abstract

Every needed strategy is normalising in the $\lambda$-calculus. Here, we extend the result to the $\lambda\sigma$-calculus, a $\lambda$-calculus with explicit substitutions. The extension requires to consider rewriting systems with critical pairs, confluent or non confluent, and to develop for them a satisfactory theory of needed normalisation. Our idea is to count for every term $M$ the number of its normalising paths, up to Lévy permutation equivalence. We deduce from the standardisation theorem in [21] that every needed strategy normalises when this number is finite. The number is zero or one in the $\lambda$-calculus, and we show after [20] that it is finite in the $\lambda\sigma$-calculus.

Introduction: the $\lambda\sigma$-calculus

The $\lambda\sigma$-calculus was introduced in [1] as a bridge between the $\lambda$-calculus and its machine implementations. In contrast with a usual $\lambda$-term, a $\lambda\sigma$-term may contain explicit substitutions which are propagated and substituted according to a series of $\sigma$-rules. An additional rule, the Beta-rule, mimics the $\beta$-rule and produces a new explicit substitution, see figure 1.

Many positive results were proved on the calculus. In particular, the $\sigma$-calculus is shown strongly normalising and confluent; and the whole $\lambda\sigma$-calculus confluent on closed terms, see [6, 25]. However, the author shows in [19] that the simply typed $\lambda\sigma$-calculus is not strongly normalising. One research direction suggested at the end of the paper is to reformulate the calculus and design a strongly normalising typed version. Another direction investigated here is to localize normalising strategies inside the $\lambda\sigma$-calculus. Of course, every normalising strategy of the $\lambda$-calculus defines a normalising $\lambda\sigma$-strategy by translation. But this is not enough: adopting the original philosophy of the $\lambda\sigma$-calculus, we look for normalising $\lambda\sigma$-strategies which do not necessarily mimic the $\lambda$-calculus.

Before any exploration of normalisation in the $\lambda\sigma$-calculus, we should remember the incredible amount of work already devoted to understanding normalisation in the $\lambda$-calculus, and more generally orthogonal systems. The first description of a normalising strategy in the $\lambda$-calculus appears in [5]: the "normal" strategy which reduces at each step the leftmost-outermost redex in the $\lambda$-term. Later, the original scheme was generalised by Lévy: a $\beta$-redex $u$ in a $\lambda$-term $M$ is "needed" when every rewriting path from $M$ to its normal form computes a residual of $u$, and a strategy is "needed" when it reduces at each step a "needed" redex. Lévy proves in [16] that

1
Terms \[ M ::= 1 | MN | \lambda M | M[s] \]
Substitutions \[ s ::= id | \uparrow | M \cdot s | s \circ t \]

Beta \[ (\lambda M)N \rightarrow M[N \cdot id] \]
App \[ (MN)[s] \rightarrow M[s]N[s] \]
Abs \[ (\lambda M)[s] \rightarrow \lambda(M[1 \cdot (s \circ \uparrow)]) \]
Clos \[ M[s][t] \rightarrow M[s \circ t] \]
VarCons \[ 1[M \cdot s] \rightarrow M \]
VarId \[ 1[id] \rightarrow 1 \]
Map \[ (M \cdot s) \circ t \rightarrow M[t] \cdot (s \circ t) \]
IdL \[ id \circ s \rightarrow s \]
Ass \[ (s_1 \circ s_2) \circ s_3 \rightarrow s_1 \circ (s_2 \circ s_3) \]
ShiftCons \[ \uparrow \circ (M \cdot s) \rightarrow s \]
ShiftId \[ \uparrow \circ id \rightarrow \uparrow \]

Figure 1: The \( \lambda \sigma \)-calculus

every needed strategy is normalising in the \( \lambda \)-calculus. The proof makes a significant use of the standardisation theorem.

In this paper, we extend Lévy’s result to the \( \lambda \sigma \)-calculus. The task is not easy for three reasons at least:

- The \( \lambda \sigma \)-calculus is not orthogonal (11 critical pairs, see figure 2). Despite the work of Boudol [4] in the mid-eighties and more recently of Clark and Kennaway [7], our current understanding of non orthogonal dynamics is limited. We take the opportunity here to develop a general theory of neededness in rewriting systems with critical pairs, much broader in scope than \( \lambda \)-calculi with explicit substitutions.

- Huet and Lévy prove in [13] that needed strategies normalise in orthogonal systems, but the property breaks in the presence of critical pairs, as we illustrate with an example in section 3.3. Henceforth a frontier must be traced between the “good” and the “bad” systems, with the prerequisite that orthogonal systems and the \( \lambda \sigma \)-calculus should appear on the same side of the frontier.

- The counter-example in [19] illustrates that the \( \lambda \sigma \)-calculus exhibits unexpected computations, and that the normalisation result we aim at requires a rigorous analysis of its dynamics. Moreover, the model should be as generic as possible to be re-applicable to similar systems, in particular other \( \lambda \)-calculi with explicit substitutions.

The paper is divided in two parts. We develop a theory of normalisation in part I, and apply it to the \( \lambda \sigma \)-calculus in part II. We choose to expose the theory for axiomatic rewriting systems as they appear in [21], so as not to limit ourself to any particular syntactical formulation. Section 1 summarizes the results of [21]. We explain that every axiomatic rewriting system can be seen as a particular 2-category, and express the
standardisation theorem in that perspective. In section 2, we introduce the subcategory of external rewriting paths in any axiomatic system, and study its decomposition properties. Finally, in section 3, we define the notion of needed strategy and establish a needed normalisation theorem starting from the hypothesis of finite normalisation cones.

In part II, we justify the normalisation theorem proved in part I by showing that the \( \lambda \sigma \)-calculus enjoys finite normalisation cones. In section 4, we recall the interpretation method developed by Hardin [10], expressed here as an adjunction property between the rewriting orders of the \( \lambda \)-calculus and the \( \lambda \sigma \)-calculus. Functorial translations between the \( \lambda \sigma \)-calculus and the \( \lambda \)-calculus are discussed. In section 5, we prove that external rewriting paths in the \( \lambda \sigma \)-calculus are projected by \( \sigma \)-normalisation to external rewriting paths in the \( \lambda \)-calculus. In section 6 we deduce from section 5 and a simple argument based on König’s lemma, on strong normalisation of the substitution \( \sigma \)-calculus, and on normalisation of needed strategies in the \( \lambda \)-calculus, that the \( \lambda \sigma \)-calculus enjoys finite normalisation cones.

<table>
<thead>
<tr>
<th>App + Beta</th>
<th>((\lambda \alpha)<a href="%5Ctheta%5Bs%5D">s</a>)</th>
<th>App</th>
<th>((\lambda \alpha \eta)[s])</th>
<th>Beta</th>
<th>(a[t \cdot \text{id}[s]])</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clos + App</td>
<td>((ab)[s \circ t])</td>
<td>Clos</td>
<td>((ab)[s][t])</td>
<td>App</td>
<td>((a<a href="%5Ctheta%5Bs%5D">s</a>)[t])</td>
</tr>
<tr>
<td>Clos + Abs</td>
<td>((\lambda \alpha)[s \circ t])</td>
<td>Clos</td>
<td>((\lambda \alpha)[s][t])</td>
<td>Abs</td>
<td>((\lambda(a[1 \cdot s \circ t]))[t])</td>
</tr>
<tr>
<td>Clos + VarId</td>
<td>(1[id \circ s])</td>
<td>Clos</td>
<td>(1[id][s])</td>
<td>VarId</td>
<td>(1[s])</td>
</tr>
<tr>
<td>Clos + VarCons</td>
<td>(1[(a \cdot s) \circ t])</td>
<td>Clos</td>
<td>(1[a \cdot s][t])</td>
<td>VarCons</td>
<td>(a[t])</td>
</tr>
<tr>
<td>Clos + Clos</td>
<td>(a[s][t \circ t'])</td>
<td>Clos</td>
<td>(a[s][t][t'])</td>
<td>Clos</td>
<td>(a[s \circ t][t'])</td>
</tr>
<tr>
<td>Ass + Map</td>
<td>((a \cdot s) \circ (t \circ t'))</td>
<td>Ass</td>
<td>((a \cdot s) \circ t')</td>
<td>Map</td>
<td>((a[t] \cdot s \circ t) \circ t')</td>
</tr>
<tr>
<td>Ass + IdL</td>
<td>(\text{id} \circ (s \circ t))</td>
<td>Ass</td>
<td>(\text{id} \circ s \circ t)</td>
<td>IdL</td>
<td>(s \circ t)</td>
</tr>
<tr>
<td>Ass + ShiftId</td>
<td>(\circ (\text{id} \circ s))</td>
<td>Ass</td>
<td>((\circ \text{id}) \circ s)</td>
<td>ShiftId</td>
<td>(\circ s)</td>
</tr>
<tr>
<td>Ass + ShiftCons</td>
<td>(\circ ((a \cdot s) \circ t))</td>
<td>Ass</td>
<td>((\circ (a \cdot s)) \circ t)</td>
<td>ShiftCons</td>
<td>(s \circ t)</td>
</tr>
<tr>
<td>Ass + Ass</td>
<td>((s \circ s') \circ (t \circ t'))</td>
<td>Ass</td>
<td>((s \circ s') \circ t')</td>
<td>Ass</td>
<td>((s \circ (s' \circ t)) \circ t')</td>
</tr>
</tbody>
</table>

Figure 2: The eleven critical pairs of the \( \lambda \sigma \)-calculus
Part I: Theory

1 Standardisation

The section motivates axiomatic rewriting systems and their 2-categorical presentation.

1.1 The λ-calculus

Consider the λ-calculus, its reduction graph $G_\lambda$, and the free category $\mathcal{L}$ with λ-terms as objects and finite sequences of β-redexes $M \xrightarrow{\beta} \cdots \xrightarrow{\beta} N$ as morphisms. One may enrich $\mathcal{L}$ with an equivalence $\equiv$ expressing the fact that two morphisms $f, g : M \rightarrow N$ compute the same thing, but in a different order. This is the idea of Lévy in [16], who deduces $\equiv$ from permutation diagrams like in figure 3.

\[
\begin{array}{ccc}
xPQ & \xrightarrow{v} & xP'Q' \\
\downarrow & & \downarrow \\
u & \xrightarrow{u} & u'
\end{array}
\quad \text{(1)}
\quad \text{or}
\quad \begin{array}{ccc}
\Delta P & \xrightarrow{\Delta P'} & P'P' \\
\downarrow & & \downarrow \\
u & \xrightarrow{u} & u'
\end{array}
\quad \begin{array}{c}
P \xrightarrow{\beta} P' \\
Q \xrightarrow{\beta} Q'
\end{array}
\quad \Delta = (\lambda x.xx)
\]

Figure 3: Two redex permutations in the λ-calculus

In [20, 21], we start to orient the permutation equivalence $\equiv$ and express that way the familiar standardisation procedure. The idea is to replace the equivalence $f \equiv g$ by “standardisation cells” $f \Rightarrow g$ indicating that the rewriting path $g$ is more standard than the rewriting path $f$. The cells $f \Rightarrow g$ are constructed using a relation $\triangleright_\lambda$ on rewriting paths expressing local permutation diagrams, and their "orientation".

The relation $\triangleright_\lambda$ is defined as follows. Recall from [16] that a β-redex $M \rightarrow N$ is characterized by the tree-occurrence of the β-pattern $(\lambda x.P)Q$ it contracts in $M$. A β-redex $u : M \rightarrow N$ nests a β-redex $v : M \rightarrow P$ when the occurrence of $u$ is a prefix of the occurrence of $v$. Finally, two rewriting paths $M \xrightarrow{\beta} N$ verify $f \triangleright_\lambda g$ when $f$ and $g$ are two developments of a set $\{u, v\}$ of coinitial and different β-redexes $u$ and $v$, $f$ contracts $v$ first, $g$ contracts $u$ first, and $v$ does not nest $u$:

$$f = M \xrightarrow{u} N \quad \text{and} \quad g = M \xrightarrow{h} N$$

Observe that $f$ is of length 2, with $u'$ the (unique) residual of $u$ after $v$, and that $g$ is of length at least 1, with $h$ a development of the residuals of $v$ after $u$. See [16] for a definition of residual and developments in the λ-calculus.

Many of the dynamical properties of the λ-calculus are retained by the structure $(G_\lambda, \triangleright_\lambda)$. For instance, we may use $\triangleright_\lambda$ to distinguish two kinds of redex permutation in the calculus, keeping in mind that $f \triangleright_\lambda g$ indicates that $g$ is more standard than $f$:

- permutation (1) in figure 3 from $v; u'$ to $u; u'$ is reversible because the two redexes $u$ and $v$ are disjoint ($u$ does not nest $v$ and $v$ does not nest $u$) and therefore the
two paths \(v; u'\) and \(u; u'\) are equivalent from the standardisation perspective. We have \((v; u' \triangleright_{\lambda} u; u')\) and \((u; u' \triangleright_{\lambda} v; u')\).

- permutation (2) in figure 3 from \(v; u'\) to \(u; v_1; v_2\) is irreversible because the redex \(u\) nests \(v\), thus the outside-in path \(u; v_1; v_2\) is strictly more standard than the inside-out path \(v; u'\). We have \((v; u' \triangleright_{\lambda} u; v_1; v_2)\) and \(\neg(u; v_1; v_2 \triangleright_{\lambda} v; u')\).

This motivates the following refinement of traditional abstract rewriting systems, as exposed in [12, 15] for example.

1.2 Axiomatic rewriting systems

Definition 1 (ARS) An axiomatic rewriting system is a pair \((G, \triangleright)\) where \(G\) is a graph and \(\triangleright\) is a binary relation between paths of \(G\). By a graph \(G\), we mean a set \(T\) of vertices (= terms), a set \(R\) of edges (= redexes), and two functions \(\partial_0\) and \(\partial_1\) from \(R\) to \(T\) (= the source and target functions). Given an edge \(e\), we write \(e : M \rightarrow N\) when \(\partial_0(e) = M\) and \(\partial_1(e) = N\). We recall that a path in a graph \(G\) is a sequence \(f = (M_1, r_1, M_2, r_2, ..., M_m, r_m, M_{m+1})\) where \(r_i : M_i \rightarrow M_{i+1}\) for every \(i \in [1, m]\). We write \(f : M_1 \rightarrow M_{m+1}\). The length of \(f\) is \(m\); \(f\) is said to be empty when \(m = 0\).

Two paths \(f : M \rightarrow N\) and \(g : P \rightarrow Q\) are coinitial (resp. coinitial) when \(M = P\) (resp. \(N = Q\)). \(f; g : M \rightarrow Q\) denotes the concatenation of two paths \(f : M \rightarrow P\) and \(g : P \rightarrow Q\) in \(G\).

One benefit of the axiomatic approach is to express various rewriting frameworks under the same elementary form, from Petri nets to the \(\lambda\)-calculus. In [21], we introduce a series of axioms on \((G, \triangleright)\) describing the generic properties of oriented permutations in a rewriting system. We show that the system \((G_{\lambda}, \triangleright_{\lambda})\) associated to the \(\lambda\)-calculus in section 1.1 verifies the axioms. We also define for every first order rewriting term (with or without critical pairs) an axiomatic rewriting system \((G, \triangleright)\) with terms as vertices and redexes as edges, and a permutation relation \(v; u' \triangleright u; h\) precisely when:

- \(u\) and \(v\) are two different and compatible redexes,
- \(v\) does not nest \(u\) w.r.t. the tree nesting ordering,
- the redex \(u'\) is the unique residual of \(u\) after \(v\), and \(h\) is a development of the residuals of \(v\) after \(u\).

We show in [21] that every such system \((G, \triangleright)\) verifies the axioms of the theory. So from now on, we identify a first order rewriting system like the \(\lambda\)-calculus with the axiomatic rewriting system \((G_{\lambda}, \triangleright_{\lambda})\) it defines.

We go one step further. Recall that a 2-category is a category \(C\) where for every two objects \(P\) and \(Q\) of \(C\), the class \(C(P, Q)\) of morphisms \(P \rightarrow Q\) is not only a set, but also a category. This means that we consider morphisms \(\alpha : f \Rightarrow g\) called cells, between morphisms \(f : P \rightarrow Q\) and \(g : P \rightarrow Q\) of the category \(C\). Cells may be composed "vertically" and "horizontally": we write \(\alpha \ast \beta : f \Rightarrow h\) for the vertical composite of \(\alpha : f \Rightarrow g\) and \(\beta : g \Rightarrow h\), and \(\alpha; \beta : f_1; f_2 \Rightarrow g_1; g_2\) for the horizontal composite of
\(\alpha : f_1 \Rightarrow f_2\) and \(\alpha : g_1 \Rightarrow g_2\). Diagrammatically:

\[
\begin{array}{c}
\begin{tikzpicture}
\node (M) at (0,0) {$M$};
\node (N) at (2,0) {$N$};
\node (P) at (4,0) {$P$};
\node (h) at (2,-2) {$h$};
\node (f1) at (0.5,1) {$f_1$};
\node (f2) at (3.5,1) {$f_2$};
\node (f1f2) at (2,2) {$f_1;f_2$};
\node (g1) at (0.5,-1) {$g_1$};
\node (g2) at (3.5,-1) {$g_2$};
\node (g1g2) at (2,-3) {$g_1;g_2$};
\draw[->] (M) to node [above] {$\psi\alpha$} (N);
\draw[->] (N) to node [above] {$\psi\beta$} (P);
\draw[->] (P) to node [above] {$\psi\alpha\beta$} (M);
\draw[->] (M) to node [below] {$h$} (N);
\draw[->] (N) to node [below] {$h$} (P);
\draw[->] (P) to node [below] {$h$} (M);
\end{tikzpicture}
\end{array}
\]

Finally, the interchange law asks that composing four cells \(\alpha : f_1 \Rightarrow g_1\) and \(\beta : f_1 \Rightarrow h_1\), \(\beta : g_1 \Rightarrow h_2\), and \(\alpha : f_2 \Rightarrow g_2\) vertically then horizontally as \((\alpha \ast \alpha_2) \ast (\beta \ast \beta_2) : f_1 ; f_2 \Rightarrow h_1 ; h_2\), or horizontally then vertically as \((\alpha ; \beta) \ast (\alpha_2 ; \beta_2) : f_1 ; f_2 \Rightarrow h_1 ; h_2\) is equivalent.

We interpret any axiomatic rewriting system \((G, \triangleright)\) as a 2-category whose cells \(f \Rightarrow g\) are precisely the “standardisation cells” mentioned in section 1.1.

**Definition 2** \((\Rightarrow^\text{step}, \Rightarrow^R, \Rightarrow^L)\) A standardisation step \(\alpha : d \Rightarrow^\text{step} e\) in \((G, \triangleright)\) is a quadruple \(\alpha = (d_1, f, g, d_2)\) of paths verifying:

\[
d = d_1 \Rightarrow^L d_2 \quad e = d_1 \Rightarrow^R d_2 \quad f \triangleright g
\]

We extend to any \((G, \triangleright)\) the taxonomy observed in \((G, \triangleright^\lambda)\) and declare the step \(\alpha\) reversible when \((g \triangleright f)\), irreversible when \(\neg(g \triangleright f)\). We write

1. \(d \Rightarrow^R e\) when there exists a (possibly empty) sequence of reversible steps

\[
d \Rightarrow^\text{step} \cdots \Rightarrow^\text{step} e
\]

2. \(d \Rightarrow^L e\) when there exists a sequence

\[
d \Rightarrow^\text{step} \cdots \Rightarrow^\text{step} e
\]

containing one irreversible step at least.

**Definition 3** \((\Rightarrow, \sim)\) Any axiomatic rewriting system \((G, \triangleright)\) is interpreted as the 2-category:

1. with underlying category the free category on the graph \(G\),

2. with cell \(\alpha : d \Rightarrow e\) any triple \((d, e, R)\) such that \(d \Rightarrow^R e\), and any triple \((d, e, I)\) such that \(d \Rightarrow^I e\). Two cells \(\alpha = (d, e, X)\) and \(\beta = (e, f, Y)\) compose as \(\alpha \ast \beta = (d, f, X \ast Y)\) where \(R \ast R = R\) and \(I \ast R = R \ast I = I \ast I = I\),
3. the identity cell \( \text{id}^d : d \Rightarrow d \) is defined as \((d, d, R)\).

Recall that an isomorphism in a category \( C \) is a morphism \( f : M \rightarrow N \) with an inverse \( f^{-1} : N \rightarrow M \) such that \( f f^{-1} = \text{id}_M \) and \( f^{-1} f = \text{id}_N \), and that a 2-isomorphism \( \alpha : d \Rightarrow e \) between \( d, e : P \rightarrow Q \) in a 2-category \( C \) is an isomorphism in the hom-category \( \mathcal{C}(P, Q) \). Observe that in \((\mathcal{G}, \triangleright)\), the 2-isomorphisms \( \alpha : d \Rightarrow e \) are precisely the cells \( \alpha = (d, e, R) \) constructed with reversible permutations: we write \( \alpha : d \simeq e \) in that case. Reinterpreting figure 3 in figure 4, permutation (1) becomes a 2-isomorphism and permutation (2) a proper 2-cell in the axiomatic system \((\mathcal{G}_\lambda, \triangleright_\lambda)\).

![Diagram](image)

**Figure 4:** Two standardisation cells in \((\mathcal{G}_\lambda, \triangleright_\lambda)\)

### 1.3 Standardisation theorem

**Definition 4 \((=, \sqsubseteq)\)** We say that a rewriting path \( g : M \rightarrow N \) is more standard than a rewriting path \( f : M \rightarrow N \) when there exists a cell \( f \Rightarrow g \) in \((\mathcal{G}, \triangleright)\). We define the permutation equivalence \( \equiv \) as the least equivalence relation containing two paths \( f \) and \( g \) when \( f \Rightarrow g \). So, the equivalence \( f \equiv g \) between \( f, g : M \rightarrow N \) means that \( f \) and \( g \) are in the same connected component of the hom-category \((\mathcal{G}, \triangleright)\)(\(M, N\)), alternatively, that there is a “zig-zag”

\[
f \Rightarrow h_1 \Leftrightarrow h_2 \Rightarrow \cdots \Leftrightarrow h_n \Rightarrow g
\]

Observe that the definition is equivalent to Lévy’s original definition in the case of \((\mathcal{G}_\lambda, \triangleright_\lambda)\). We say that \( f : M \rightarrow P \) extends \( g : M \rightarrow N \), and write \( g \sqsubseteq f \), when there exists a path \( h : N \rightarrow P \) such that \( f \equiv g; h \).

**Definition 5 \((\text{standard})\)** A rewriting path \( f : M \rightarrow N \) is standard when it cannot be further standardised, meaning that every cell \( \alpha : f \Rightarrow g \) starting from \( f \) is a 2-isomorphism.

In [21], we prove a standardisation theorem for every axiomatic rewriting system \((\mathcal{G}, \triangleright)\). The theorem states that:

1. for every rewriting path \( d : M \rightarrow N \), there exists a standard path \( \downarrow_d : M \rightarrow N \) and a unique cell \( \alpha : d \Rightarrow \downarrow_d \),

2. two paths \( d, e : M \rightarrow N \) equal modulo \( \equiv \) have their standard path \( \downarrow_d, \downarrow_e : M \rightarrow N \) equal modulo \( \simeq \):

\[
\forall M \overset{d}{\rightarrow} N, \quad d \equiv e \Rightarrow \downarrow_d \simeq \downarrow_e
\]
Property 1. and 2. correspond to what is called existence and uniqueness of standardisation in [16, 4, 9, 20]. By property 2, there exists a unique standard path (unique modulo \( \simeq \)) in each Lévy equivalence class \( \equiv \). By property 1, this path \( d \) is characterised by the equivalence \( d \simeq \downarrow d \). We illustrate this in figure 5, where the rewriting path \( v_3 u' : \Delta(Ia) \to aa \) enjoys two standard paths \( u; v_1; v'_2 \) and \( u; v_2; v'_1 \) which are equal modulo \( \simeq \).

In [21], we prove also that every \((G, \triangleright)\) verifies properties A and B:

A. two rewriting paths \( d \) and \( e \) are standard when their composite \( d; e \) is standard.

This means that for every pair \( M \xrightarrow{d} N \xrightarrow{e} P \) of composable paths

\[
d; e \simeq \downarrow d; e \Rightarrow d \simeq \downarrow d \text{ and } e \simeq \downarrow e
\]

B. The standardisation algorithm introduced in [9, 20, 21] computes the standard path \( \downarrow d \) by extracting recursively an “outermost” redex in the rewriting path \( d \).

The definition of the algorithm implies that for every triple \( M \xrightarrow{d} N \xrightarrow{e} P \xrightarrow{f} Q \) of composable paths:

\[
d; \downarrow e; f \simeq \downarrow d; e; f \Rightarrow d; \downarrow e \simeq \downarrow d; e
\]

We prefer to keep the axiomatics in [21] implicit and unless explicitly mentioned (in section 3) limit ourself to the following definition of an axiomatic system \((G, \triangleright)\): it is a 2-category equipped with an operator \( \downarrow \) on morphisms (the categorician will recognize a lax endofunctor \( \downarrow : (G, \triangleright) \to (G, \triangleright) \)) verifying properties 1, 2, A and B.

2 The category \( E \) of external rewriting paths

2.1 Motivation and definition

Standard rewriting paths do not compose generally. In the \( \lambda \)-calculus for instance, the two \( \beta \)-redexes

\[
(\lambda x. y)(Ia) \xrightarrow{\beta} (\lambda x. y)a \quad (\lambda x. y)a \xrightarrow{\beta} y
\]
are standard but their composite \( u; v \) is not standard because of the irreversible permutation:

\[
((\lambda x. y)(Ia) \rightarrow (\lambda x. y)a \rightarrow y) \Rightarrow ((\lambda x. y)(Ia) \rightarrow y)
\]

This suggests to strengthen the notion of standardness and define a subcategory of "external computations" expressing in any \((G, \triangleright)\) the idea of leftmost-outermost computation in the \(\lambda\)-calculus. For example, we look for a syntax-free explanation for the "externality" of a \(\beta\)-redex like

\[
r : (\lambda x. \lambda y. x)MP \rightarrow (\lambda y. MP)P
\]

In [20], we suggest that \( r \) is external because any path \( r; f \) obtained by extending \( r \) with a standard path \( f : (\lambda y. MP)P \rightarrow Q \) is standard. This leads to the following definition in any axiomatic system \((G, \triangleright)\):

**Definition 6 (external)** A rewriting path \( e : M \rightarrow N \) is **external** in \((G, \triangleright)\) when for every right-composable rewriting path \( f : N \rightarrow P \), we have:

\[
e; \downarrow_f \simeq \downarrow_{ecf}
\]

The first observation is that, contrarily to standard paths, external paths do compose.

**Lemma 7** The class \( \mathcal{E} \) of external paths defines a subcategory of \((G, \triangleright)\).

**Proof** Identity paths \( id_P : P \rightarrow P \) are external. Moreover, given two external paths \( d : M \rightarrow N \) and \( e : N \rightarrow P \), and any path \( f : P \rightarrow Q \):

\[
d; e; \downarrow_f \simeq d; \downarrow_{ecf} \quad \text{by } e \in \mathcal{E} \text{ and left composition of 2-isomorphisms,}
\]

\[
\simeq \downarrow_{d; e; f} \quad \text{by } d \in \mathcal{E}
\]

We conclude that the composite \( d; e \) is external.

The following definition appears already in Hillen's analysis of long-\(\beta\eta\)-normal forms in the \(\lambda\)-calculus, see [11].

**Definition 8 (split monic, normal form)** A path \( f : M \rightarrow N \) is **split monic** when there exists a path \( g : N \rightarrow M \) such that \( f; g \simeq id_M \). A term \( M \) is **normal** when every path \( f : M \rightarrow N \) is a split monic. We call also \( M \) a **normal form**. A path \( d : M \rightarrow N \) is called **normalising** when \( N \) is normal.

**Lemma 9** Every standard and normalising path is external in \((G, \triangleright)\).

**Proof** Let \( e : M \rightarrow N \) be standard and normalising, and \( f : N \rightarrow P \) be any right-composable path. By definition of a normal form, there exists a path \( g : P \rightarrow N \) such that \( f; g \equiv id_N \). We claim that \( id_N \) is standard. Indeed, it follows from property 2 and \( id_N \equiv id_N; \downarrow_{id_N} \) that \( \downarrow_{id_N} \simeq \downarrow_{id_N; id_N} \). We have \( id_N; \downarrow_{id_N} \simeq \downarrow_{id_N; id_N} \) by definition of \( id_N \), and our claim that \( id_N \simeq \downarrow_{id_N} \) follows by property A.

We obtain the series of equivalences:

\[
e; \downarrow_{f; g} \simeq e; \downarrow_{id_N} \quad \text{by } f; g \equiv id_N \text{ and property 2,}
\]

\[
\simeq e \quad \text{by } id_N \simeq \downarrow_{id_N},
\]

\[
\simeq \downarrow_e \quad \text{because } e \text{ is standard,}
\]

\[
\simeq \downarrow_{ecfg} \quad \text{by } e \equiv e; f; g \text{ and property 2.}
\]

Here, we apply property B. and deduce the equivalence \( e; \downarrow_f \simeq \downarrow_{ecf} \). We conclude that \( e \) is external.

\[\Box\]
2.2 Decomposition properties

In lemma 7, we show that external paths compose well. We study now how they decompose. Let us observe first that

**Lemma 10** Every external path is standard in \((\mathcal{G}, \triangleright)\).

**Proof** Suppose that \(e : M \rightarrow N\) is external. It follows from \(e \equiv e; \downarrow_{\text{id}_N}\) and property 2 that \(e; \downarrow_{\text{id}_N} \simeq \downarrow_{\text{id}_N} \simeq \downarrow_e \simeq \downarrow_{e; \text{id}_N}\). We deduce that

\[ e; \downarrow_{\text{id}_N} \simeq \downarrow_e \downarrow_{\text{id}_N} \simeq \downarrow_e \simeq \downarrow_{e; \text{id}_N}\]

by externality of \(e\). We apply property A and conclude that \(e \simeq \downarrow_e\).

**Lemma 11 (right decomposition)** The path \(N \overset{e}{\rightarrow} P\) in a pair \(M \overset{d}{\rightarrow} N \overset{f}{\rightarrow} P\) is always external when the composite \(M \overset{df}{\rightarrow} P\) is external.

**Proof** Given a pair \(M \overset{d}{\rightarrow} N \overset{f}{\rightarrow} P\) with external composite \(M \overset{df}{\rightarrow} P\), and any right-composable path \(f : P \rightarrow Q\), we have:

\[ d; e; \downarrow_f \simeq \downarrow_{de;f}\]

because \(d; e\) is external

\[ \simeq \downarrow_{de;\downarrow_f}\]

by \(d; e; f \equiv d; e; f; \downarrow_f\) and standardisation.

It follows from property A, that \(e; \downarrow_f \simeq \downarrow_e; \downarrow_f\), then from \(e; f \equiv e; \downarrow_f\) and standardisation that \(\downarrow_e; \downarrow_f \simeq \downarrow_e; \downarrow_f\). We conclude that \(e\) is external in \((\mathcal{G}, \triangleright)\).

**Lemma 12 (left decomposition)** Suppose that for every two cofinal paths \(f : M \rightarrow P\) and \(g : M \rightarrow Q\) in \((\mathcal{G}, \triangleright)\), we have two cofinal paths \(f' : Q \rightarrow N\) and \(g' : P \rightarrow N\) such that \(f; g' \equiv g; f'\). In that case, the path \(M \overset{f}{\rightarrow} N \overset{g'}{\rightarrow} P\) is external when the composite \(M \overset{df}{\rightarrow} P\) is external.

**Proof** We prove that \(d; \downarrow_f \simeq \downarrow_{df}\) for any path \(f : N \rightarrow Q\) right-composable with \(d\), and conclude that \(d\) is external. By hypothesis, there are two paths \(e'\) and \(f'\) such that \(f; e' \equiv e; f'\). Property B and the series of equivalences

\[ \downarrow_{de;f'} \simeq \downarrow_{de;f'}\]

by \(e; f' \equiv f; e'\),

\[ \simeq \downarrow_{de;\downarrow_f}\]

by \(d; e \in \mathcal{E}\),

\[ \simeq \downarrow_{d; e; \downarrow_f}\]

by \(e \in \mathcal{E}\),

\[ \simeq \downarrow_{d; e; \downarrow_f}\]

by \(e; f' \equiv f; e'\).

implies that \(d; \downarrow_f \simeq \downarrow_{df}\). We conclude.

Let \((\mathcal{G}, \triangleright)\) be the category obtained by quotienting the 2-category \((\mathcal{G}, \triangleright)\) with the relation \(\equiv\). Huet and Lévy prove in [13] that the quotient category \((\mathcal{G}, \triangleright)\)/\(\equiv\) has pushouts when the calculus is orthogonal, see [22] for a discussion. This means that for every pair \(P \overset{f}{\rightarrow} M \overset{g}{\rightarrow} Q\) of cofinal rewriting paths, there exists a pair \(P \overset{g'}{\rightarrow} N \overset{f'}{\rightarrow} Q\) of cofinal rewriting paths making the diagram (1) commute modulo Lévy equivalence:

\[
\begin{array}{ccc}
M & \overset{g}{\rightarrow} & Q \\
\downarrow_{f} & \equiv & \downarrow_{f'} \\
\downarrow_{g'f} & & \downarrow_{g'f'} \\
\downarrow_{P} & \overset{g'}{\rightarrow} & \downarrow_{N} \\
\end{array}
\]

\[1\]
Furthermore, for every two rewriting paths $h_1 : P \to O$ and $h_2 : Q \to O$ verifying $f \circ h_1 \equiv g \circ h_2$, there exists a rewriting path $h : N \to O$ unique modulo $\equiv$, making the diagram (2) commute in $(\mathcal{G}, \triangleright)/\equiv$:

\[
\begin{array}{c}
M \xrightarrow{g} Q \\
\downarrow^f \quad \downarrow^{g'} \quad \downarrow^{h_2} \\
P \xrightarrow{h_3} N \quad \xrightarrow{h_2} O
\end{array}
\]

(2)

This algebraic strengthening of the Church-Rosser property insures that the premises of lemma 12 are verified. Because every external path decomposes on the right and on the left by lemma 11 and 12, a rewriting path is external if and only if every redex it contracts is external.

This is not true anymore in a rewriting system with critical pairs: an external path may also contract non external redexes. For instance, Boudol introduces in [4], section 5.1, the following first-order rewriting system:

\[
\begin{align*}
F(A, B, x) & \to C & F(A, A, A) & \to C & \Omega & \to A \\
F(B, x, A) & \to C & F(B, B, B) & \to C & \Omega & \to B \\
F(x, A, B) & \to C
\end{align*}
\]

He observes that the term $F(\Omega, \Omega, \Omega)$ is confluent, normalises to $C$, but does not contain any external redex (Boudol speaks about “needed” and “strongly needed” redexes. Every strongly needed or “necessary” redex in his sense is external in our sense, but the converse is not true.) This is precisely the fact that $F(\neg, \neg, \neg)$ describes a non sequential function, the so-called Gustave function appearing in [3]. For instance, the redex

\[
r : F(\Omega, \Omega, \Omega) \to C
\]

is not external because the path

\[
F(\Omega, \Omega, \Omega) \xrightarrow{t} F(A, \Omega, \Omega) \to F(A, A, \Omega) \to F(A, A, B) \xrightarrow{F} C
\]

is not standard. However, the redex $r$ may be extended to an external path

\[
F(\Omega, \Omega, \Omega) \xrightarrow{t} F(A, \Omega, \Omega) \to F(A, B, \Omega)
\]

which cannot be computed as a sequence of external redexes. Another illustration appears in the $\lambda\sigma$-calculus, equipped with the usual tree nesting ordering, A simple argument shows that for every $M, N, s, t$, the path

\[
e : (\lambda M)\lambda N[s][t] \xrightarrow{\text{beta}} M[N \cdot id][s][t] \xrightarrow{\text{Clos}} M[(N \cdot id) \circ s][t]
\]

is external. However, the first redex it contracts

\[
(\lambda M)\lambda N[s][t] \xrightarrow{\text{beta}} M[N \cdot id][s][t]
\]

11
is not external because of the irreversible permutation

\[ \text{Beta1; Clos2} \triangleright_{\lambda \sigma} \text{Clos1; Beta2} \]

in the diagram:

\[
\begin{array}{c}
\text{Beta1; Clos2} \triangleright_{\lambda \sigma} \text{Clos1; Beta2} \\
((\lambda M)N)[s][t] \xrightarrow{\text{Beta1}} M[N \cdot \text{id}][s][t] \xrightarrow{\text{Clos2}} M[(N \cdot \text{id}) \circ s][t] \\
\text{Clos1} \downarrow \quad \Downarrow \quad \text{Clos2} \quad \Downarrow \\
((\lambda M)N)[s \circ t] \xrightarrow{\text{Beta2}} M[N \cdot \text{id}][s \circ t]
\end{array}
\]

It would be easy to show that if we choose \( M, N, s \) and \( t \) in such a way that \( P = ((\lambda M)N)[s][t] \) has a normal form \( Q \), and extend \( e \) to a normalising path \( e; f : P \rightarrow Q \), the normalising path \( e; f \) cannot be reorganised modulo permutation as a sequence of external reducts. We conclude that external reducts do not provide a complete description of the normalisation space. This explains why we need to introduce the notion of external path instead of external reduct when we decide to analyse the \( \lambda \sigma \)-calculus, and more generally rewriting systems with critical pairs.

### 2.3 Another definition of \( E \)

We discuss briefly another definition of externality appearing in [20]. In the \( \lambda \)-calculus, an external \( \beta \)-reduct like

\[ r : (\lambda x. \lambda y. x)MP \rightarrow (\lambda y. M)P \]

is neither duplicated or erased by any rewriting path \( d : (\lambda x. \lambda y. x)MP \rightarrow Q \). Thus, whenever \( r \sqsubseteq d \), the reduct \( r \) appears “preserved” by the path \( d \) in the sense that \( d \) factors as \( d = D; r'; E \) where \( r' \) is the (unique) residual of \( r \) through \( D \). This may be expressed otherwise for every two rewriting paths \( f : (\lambda y. M)P \rightarrow Q \) and \( d : (\lambda x. \lambda y. x)MP \rightarrow Q \),

\[ d \equiv r; f \Rightarrow (\exists g, \exists \alpha, f \equiv g \text{ and } \alpha : d \Rightarrow r; g) \]

Here, \( g \) is constructed as the path \( D'; E \) where \( D' \) is a path residual of \( D \) after \( r \) in the \( \lambda \)-calculus (see [16, 13, 4] for a definition). We show that the characterization extends to any axiomatic rewriting system \((G, \triangleright)\):

**Lemma 13** A rewriting path \( e : M \rightarrow N \) is external \((G, \triangleright)\) if and only if for every right-composable path \( f : N \rightarrow P \) and for every path \( d : M \rightarrow P \):

\[ d \equiv e; f \Rightarrow (\exists g : N \rightarrow P, \exists \alpha, f \equiv g \text{ and } \alpha : d \Rightarrow e; g) \]  \hspace{1cm} (3)

**Proof** (only if) take \( g = \downarrow f \). (if) we choose a path \( f : N \rightarrow P \) and apply property (3) to \( d = \downarrow e; f \). We deduce that there exists a morphism \( g \equiv f \) and a cell \( \alpha \) such that \( \alpha : \downarrow e; f \Rightarrow e; g \). Composing \( \alpha \) with \( \beta : g \Rightarrow \downarrow f \), we obtain a cell \( \downarrow e; f \Rightarrow e; \downarrow f \). By standardness of \( \downarrow e; f \), we conclude that \( \downarrow e; f \simeq e; \downarrow f \). The equivalence is verified for every path \( f : N \rightarrow P \) right-composable with \( e : M \rightarrow N \), which is therefore external.
3 Finite normalisation cones

3.1 On neededness in the presence of critical pairs

Our next task is to characterise the needed paths of an axiomatic system \((G, \triangleright)\). We start from the definition of Maranget in [18] for orthogonal rewriting systems: A redex \(r : M \rightarrow N\) is needed when it has a residual (at least) after any rewriting path \(f : M \rightarrow P\), unless \(f\) contracts at some step a residual of \(r\). We want to extend the definition to rewriting systems with critical pairs. Consider a rewriting system like:

\[ A \rightarrow B \quad A \rightarrow C \]

The critical pair in \(A\) implies that the redex \(r : A \rightarrow B\) has no residual after the reduction \(f : A \rightarrow C\). Should we conclude that \(r\) is not needed? Not really. Maybe needed is not the right word to qualify \(r\) and should be replaced by unavoidable, since every path from \(A\) either contracts a residual of \(r\), or a redex forming a critical pair with a residual of \(r\). But if we choose to keep the word needed, then we should definitely consider \(r\) as needed, and adapt Maranget’s definition accordingly: We say that a redex \(r : M \rightarrow N\) is needed when every rewriting path \(f : M \rightarrow P\) extending \(r\) \((r \sqsubseteq f)\) contracts a residual of \(r\) at least. Check that the definition is equivalent with the previous one in an orthogonal rewriting system, and that the redex \(r : A \rightarrow B\) is needed in our example.

The definition is fine, but not adapted to our current setting: remember that we never use or define the notion of residual in our axiomatic systems. This choice is not ornamental: we believe that \(\triangleright\) is the fundamental structure, and that the residual relation is one possible way to reconstruct \(\triangleright\) in various settings. So, we need to characterize needed paths another way.

Definition 14 (\(|-|\)) We write \(|f|\) the length of a rewriting path \(f : M \rightarrow N\), which is the number of redexes it contracts.

In [21], we extend to any axiomatic rewriting system \((G, \triangleright)\) a property established by Lévy in the \(\lambda\)-calculus:

C. Given any pair of composable paths \(M \xrightarrow{f} N \xrightarrow{g} P\) in \((G, \triangleright)\), the length \(|\downarrow_{fg}|\) is bounded by the length \(|\downarrow_f|\):

\[ \forall (M, N, P), \forall M \xrightarrow{f} N \xrightarrow{g} P : \quad |\downarrow_{fg}| \geq |\downarrow_f| \]

The reason for the inequality is that every redex contracted in \(\downarrow_g\) corresponds to a unique redex contracted in \(\downarrow_{fg}\). Observe now that the inequality becomes strict when the path \(f : M \rightarrow N\) is a needed redex \(r : M \rightarrow N\) because one residual of \(r\) at least appears contracted in \(\downarrow_{rg}\). We use the property to characterise the needed paths of \((G, \triangleright)\):

Definition 15 (needed path) A rewriting path \(f : M \rightarrow N\) is properly needed in an axiomatic rewriting system \((G, \triangleright)\) when:

\[ \forall P, \forall N \xrightarrow{g} P, \quad |\downarrow_{fg}| > |\downarrow_f| \]

We say that \(f : M \rightarrow N\) is needed when \(f\) is properly needed, or when \(f\) is split monic.
We will prove several results on the class of needed paths in \((G, \rhd)\). Again, we are careful to limit ourself to five properties of \(\| - \|\): it associates an ordinal to every rewriting path \(f\) and verifies properties C, D, E, and F:

D. In every axiomatic rewriting system, two paths \(f\) and \(g\) involved in a reversible permutation \(f \rhd g\) have lengths 2. By definition of \(\simeq\), this implies:

\[
\forall (P, Q), \forall P \xrightarrow{d} Q : \quad d \simeq e \Rightarrow \|d\| = \|e\|
\]

E. In every axiomatic rewriting system, the length of a path is defined as its number of rewriting steps. We deduce:

\[
\forall (M, N, P), \forall M \xrightarrow{f} N \xrightarrow{g} Q : \quad \|f; g\| \geq \|f\|
\]

F. Suppose that two paths \(f; \downarrow_g\) and \(\downarrow_g\) have the same length in \((G, \rhd)\). Then, the path \(f\) is empty and a fortiori split monic.

\[
\forall (M, N, P), \forall M \xrightarrow{f} N \xrightarrow{g} Q : \quad \|f; \downarrow_g\| = \|\downarrow_g\| \Rightarrow f \text{ is split monic.}
\]

We prove that:

**Lemma 16** The class \(N\) of needed paths defines a subcategory of \((G, \rhd)\). Moreover, for every pair of rewriting paths \(d : L \rightarrow M\) and \(f : P \rightarrow Q\), the path \(d \rhd_\downarrow f\) is properly needed when \(e : M \rightarrow P\) is properly needed.

**Proof** The last assertion follows by definition of properly needed paths, and by property C. We deduce that \(N\) is a category by observing that all identities are needed, and that split monics compose well. ■

**Lemma 17** Every external path is needed in \((G, \rhd)\).

**Proof** We claim that any external path \(e : M \rightarrow N\) is properly needed when it is not split monic. Indeed, we have for any right-composable path \(f : N \rightarrow P\):

\[
\|\downarrow_f\| \leq \|\downarrow_{e; f}\| \leq \|e; \downarrow_f\| \leq \|\downarrow_f\| \quad \text{by property C,}\n\]

\[
\|\downarrow_f\| = \|e; \downarrow_f\| \quad \text{by property D, and externality of } e,\n\]

\[
\|\downarrow_f\| \neq \|\downarrow_f\| \quad \text{by property F.}\n\]

The strict inequality \(\|\downarrow_f\| < \|\downarrow_{e; f}\|\) follows. We conclude that every external path \(e\) is either split monic or properly needed — thus in both cases needed. ■

### 3.2 Needed strategies

We choose a definition of strategy which generalizes the traditional notion of “one step” strategy, as it appears in for instance in [2, 13]. In particular, the definition of \(S\) as a total function is not really restrictive because the equality \(S(P) = \text{id}_P\) is always possible.
**Definition 18 (strategy)** A strategy $S$ in $(G, \triangleright)$ is a function which associates to every term $P$ a rewriting path

$$S(P) : P \rightarrow Q$$

The indexed family of paths $S^n(P) : P \rightarrow Q_n$ is constructed by induction on $n \in \mathbb{N}$:

- $S^0(P) = \text{id}_P$
- $S^{n+1}(P) = S^n(P) ; S(Q_n)$ when $S^n(P) : P \rightarrow Q_n$.

We say that a rewriting path $d : P \rightarrow Q$ follows a strategy $S$ when there exists $n \in \mathbb{N}$ such that $d = S^n(P)$.

**Definition 19 (normalising strategy)** A strategy $S$ is declared normalising when for every term $P$ with a normal form, there exists $n \in \mathbb{N}$ such that $S^n(P) : P \rightarrow Q$ computes a normal form $Q$.

**Definition 20 (needed strategy)** A strategy $S$ is declared needed when, for every term $P$:

1. the path $S(P) : P \rightarrow Q$ is needed,
2. the path $S(P) : P \rightarrow Q$ is properly needed when there exists a properly needed path $P \rightarrow R$ outgoing $P$,
3. $Q$ has a normal form when $P$ has a normal form and $S(P) : P \rightarrow Q$.

Condition 3 could be easily replaced by a similar condition on needed paths, asking that a term $Q$ has a normal form whenever $P$ has a normal form and $f : P \rightarrow Q$ is needed. This is a matter of taste: by keeping condition 3 here, we ensure Lemma 17. Condition 2 forbids $S$ to stop on a term for no reason.

### 3.3 Finite normalisation cone

Huet and Lévy prove in [13] that every needed strategy normalises in an orthogonal rewriting system. This is not true any more in the presence of critical pairs. Consider the term $A$ in the first order rewriting system:

$$A \rightarrow A \quad A \rightarrow B$$

Observe that the redex $r : A \rightarrow A$ is external and a fortiori needed, and that the term $A$ has normal form $B$. However, the needed strategy $S$ defined by $S(A) = r : A \rightarrow A$ and $S(B) = \text{id}_B$ does not normalise $A$.

At that point, we introduce a criterion to detect *well-behaved* systems among rewriting systems with critical pairs. The idea is to count the number of normalising paths from a term, modulo Lévy equivalence $\equiv$.

- The number is infinite from $A$ to $B$ in the last example because every rewriting path $A \rightarrow A \cdots A \rightarrow B$ defines its own permutation class modulo $\equiv$.  

15
• the number is at most one in any orthogonal rewriting system. This follows the existence of pushouts in the category \((G, \triangleright) / \equiv\), see [16, 13]. Indeed, suppose that there exists two different normalising paths \(M \xrightarrow{f} P\) and \(M \xrightarrow{g} Q\). By existence of pushouts (see section 2.2 for a definition), there exist two rewriting paths \(f' : Q \to N\) and \(g' : P \to N\) such that \(f;g' \equiv g;f'\). But \(P\) and \(Q\) are normal forms, and \(f'\) and \(g'\) are therefore split monics. We conclude that \(f \equiv g\) in a setting like [16, 13, 4, 21] where the identities are the only split monics. Otherwise, an elementary categorical reasoning shows that there exists two paths \(h : P \to Q\) and \(h^{-1} : Q \to P\) verifying \(h;h^{-1} = \text{id}_P\) and \(h^{-1};h = \text{id}_Q\), such that \(f;h \equiv g\) and \(g;h^{-1} \equiv f\). This implies that the normalisation cone is at most singleton in definition 21 (it is empty when the term is not normalising).

• We prove in part II that the number is finite from any closed λσ-term.

This leads to the following definition:

**Definition 21 (normalisation cone)** We call normalisation cone from a term \(M\) any cone \((e_i : M \to P_i)_{i \in I}\) of normalising paths such that every normalising path \(f : M \to Q\) factors through a unique path \(e_i : M \to P_i\) modulo \(\equiv\):

\[
\exists! i \in I, \exists P_i \xrightarrow{h} Q, \quad M \xrightarrow{f} Q \equiv M \xrightarrow{e_i} P_i \xrightarrow{h} Q
\]

An axiomatic rewriting system \((G, \triangleright)\) enjoys finite normalisation cones when from any term \(M\) there exists a finite normalisation cone \((e_i : M \to P_i)_{i \in I}\).

### 3.4 Normalisation theorem

The main motivation of definition 21 is theorem 1.

**Theorem 1 (normalisation)** Suppose that the axiomatic system \((G, \triangleright)\) enjoys finite normalisation cones. Then, every needed strategy \(S\) is normalising in \((G, \triangleright)\).

**Proof** By hypothesis, we may assign to any term \(M\) a finite normalisation cone \((e_i^M : M \to P_i)_{i \in I_M}\), and choose each \(e_i^M\) standard. We define \(\text{prof}(M)\) as:

1. \(\text{prof}(M) = \infty\) when the cone \((e_i^M : M \to P_i)_{i \in I_M}\) is empty, or equivalently, when \(M\) has no normal form,

2. otherwise, \(\text{prof}(M)\) is the length \(\|e_i^M\|\) of the longest path \(e_i^M : M \to P_i\) in the cone \((e_i^M : M \to P_i)_{i \in I_M}\):

\[
\text{prof}(M) = \max\{\|e_i^M\| \mid i \in I_M\}
\]

We use the universality condition in the definition of a normalisation cone, and associate to any rewriting path \(f : M \to N\) the function:

\[
f^* : I_N \to I_M
\]

which maps each index \(j \in I_N\) to the unique index \(i \in I_M\) such that

\[
e_i^M \equiv f;e_j^N
\]
Now, we claim that $||\psi_f|| = ||\psi_g||$ whenever two normalising paths $M \overset{F}{\rightarrow} P$ and $M \overset{G}{\rightarrow} Q$ verify $f \subseteq g$. Indeed, by definition of $f \subseteq g$, there exists a path $h : P \rightarrow Q$ such that $f; h \equiv g$. Because $P$ is normal, the path $h$ is split monic, which means that there exists a path $h'$ such that $h; h' \equiv id_P$. The equivalence $g; h' \equiv f$ follows. We apply uniqueness of standardisation (property 2) and externality of $\psi_f$ and $\psi_g$ (lemma 9) to obtain the series of equivalence

$$\psi_f \overset{\sim}{\Rightarrow} \psi_f; h \overset{\sim}{\Rightarrow} \psi_f; h \overset{\sim}{\Rightarrow} \psi_g \overset{\sim}{\Rightarrow} \psi_g; h' \overset{\sim}{\Rightarrow} \psi_f$$

We apply property D. and E. and conclude that $||\psi_f|| \leq ||\psi_g|| \leq ||\psi_f||$.

We apply immediately this result to (4) and deduce:

$$\forall j \in I_N, \quad ||e^N_{f(j)}|| = ||\psi_f; e^N_{f(j)}||$$

(5) We claim (α) that $\text{proof}(M) > \text{proof}(N)$ when $f : M \rightarrow N$ is properly needed and $N$ has a normal form. By definition of a properly needed path, we have:

$$\forall j \in I_N, \quad ||\psi_f; e^N_{f(j)}|| > ||e^N_j||$$

We apply property (5) and deduce that

$$\forall j \in I_N, \quad ||e^M_{f(j)}|| > ||e^N_j||$$

The inequality applies in particular to the index $j \in I_N$ defining $\text{proof}(N)$ as $||e^N_j||$. We conclude that $\text{proof}(M) > \text{proof}(N)$.

We claim (β) that a term $M$ is normal whenever there exists a split monic path $f : M \rightarrow P$ to a normal form $P$. Indeed, by definition of split monic, there exists a path $g : P \rightarrow N$ such that $f; g \equiv id_M$. Now, the path $g$ is split monic too because $P$ is normal. It follows that $M$ and $P$ are isomorphic in the category $(G, \triangleright) / \equiv$. We conclude that $M$ is a normal form.

Let $S$ be a needed strategy. We prove that $\text{proof}(M) > \text{proof}(N)$ whenever $M$ is non normal and has a normal form. Indeed, let $f : M \rightarrow P$ be a normalising path from $M$. By lemma 9, the path $f$ is external, and by lemma 17, it is needed. By definition of needed, $f$ is either split monic or properly needed, but split monic is forbidden because $M$ is non normal and claim (β). We conclude that $f$ is properly needed. By condition 2 and 3 in the definition of a needed strategy, the path $S(M) : M \rightarrow N$ is properly needed, and $N$ has a normal form. We conclude that $\text{proof}(M) > \text{proof}(N)$ by claim (α).

We have just established that $\text{proof}(M) > \text{proof}(N)$ and that $N$ has a normal form whenever $M$ is non normal and has a normal form. We conclude by well-foundedness that the strategy $S$ is normalising. ■

### 3.5 Fair and hyper normalisation

The reader interested in *fair* normalisation should observe that theorem 1 adapts easily to that setting. The idea appearing with the definition of “eventually outmost” in [8] is to mix fairly efficient and inefficient computations, and prove a normalisation result. Suppose for instance that a strategy $S$ verifies for every term $P$ that:
1. there exists $n \in \mathbb{N}$ such that $S^n(P) : P \rightarrow Q$ is needed,

2. there exists $n \in \mathbb{N}$ such that $S^n(P) : P \rightarrow Q$ is properly needed when there exists a properly needed path $P \rightarrow R$ outgoing $P$,

3. $Q$ has a normal form when $P$ has a normal form and $S(P) : P \rightarrow Q$.

We may translate any such “eventually needed” or “hyper needed” strategy $S$ as the strategy $S'$ which associates to any term $P$ the path

$$S'(P) = S^n(P) : P \rightarrow Q$$

where $n \in \mathbb{N}$ is chosen such that:

- $S^n(P) : P \rightarrow Q$ is needed,

- $S^n(P) : P \rightarrow Q$ is properly needed when there exists a properly needed path $P \rightarrow R$ outgoing $P$.

The strategy $S'$ is needed. Moreover, normalisation of $S'$ implies normalisation of $S$. By theorem 1 on $S'$, we deduce that the strategy $S$ is normalising when the rewriting system $(G, \triangleright)$ enjoys finite normalisation cones.
Part II: Application to the $\lambda\sigma$-calculus

We devote this part II to a single proof: the fact that the $\lambda\sigma$-calculus enjoys finite normalisation cones. As part I (theorem 1) explains, this implies that every needed strategy of the $\lambda\sigma$-calculus normalises.

4 Remarks on the interpretation method

We recall the so-called interpretation method developed by Hardin in [10]. Every (possibly open) $\lambda$-term $M$ may be seen as a (closed) $\lambda\sigma$-term $U(M)$ by de Bruijn translation. This translation is a functor $U : \lambda \rightarrow \lambda\sigma$ between the two order categories $(\lambda, \leq_{\lambda})$ and $(\lambda\sigma, \leq_{\lambda\sigma})$: the objects of the first category are the (possibly open) $\lambda$-terms, with $M \leq_{\lambda} N$ when $M \beta$-reduces to $N$; the objects of the second category are the (closed) $\lambda\sigma$-terms, with $M \leq_{\lambda\sigma} N$ when $M \lambda\sigma$-reduces to $N$. Conversely, every (closed) $\lambda\sigma$-term may be interpreted as a (possibly open) $\lambda$-term by $\sigma$-normalisation, this defining a functor $\sigma : \lambda\sigma \rightarrow \lambda$. The two functors appear to be adjoint in fact, $U : \lambda \rightarrow \lambda\sigma$ right adjoint to $\sigma : \lambda\sigma \rightarrow \lambda$:

$$\forall M \in \lambda\sigma, \forall N \in \lambda, \quad \sigma(M) \leq_{\lambda} N \iff M \leq_{\lambda\sigma} U(N)$$

(6)

Functionality breaks when we move from preorders $(\lambda, \leq_{\lambda})$ and $(\lambda\sigma, \leq_{\lambda\sigma})$ to the free categories $\lambda^{\approx}$ and $\lambda\sigma^{\approx}$ derived from the reduction graphs of (possibly open) $\lambda$-terms and (closed) $\lambda\sigma$-terms respectively. On one hand, there is no canonical "forgetful" functor $U : \lambda^{\approx} \rightarrow \lambda\sigma^{\approx}$ because of the different strategies to $\sigma$-normalise a $\lambda\sigma$-term; on the other hand, no "interpretation" functor $\sigma : \lambda\sigma^{\approx} \rightarrow \lambda^{\approx}$ because generally $\sigma$-normalisation duplicates Beta-redoxes.

A solution is to consider rewriting paths $M \rightarrow N$ modulo reversible permutations, in the categories $\lambda^{\approx} = (\lambda^{\rightarrow}/\approx)$ and $\lambda\sigma^{\approx} = (\lambda\sigma^{\rightarrow}/\approx)$. First, there is a forgetful functor $U : \lambda^{\rightarrow} \rightarrow \lambda\sigma^{\approx}$ obtained by contracting at each $\beta$-step $M \rightarrow N$ the corresponding Beta-redox $U(M) \rightarrow P$, followed by a standard $\sigma$-normalisation path $P \rightarrow U(N)$ avoiding the Clus and Ass-rules: there is always such a path $P \rightarrow \sigma(P)$, and all choices are equivalent modulo $\approx$ by standardness and absence of clinical pair in that fragment of the $\sigma$-calculus, see figure 2. Conversely, there is an interpretation functor $\sigma : \lambda\sigma^{\rightarrow} \rightarrow \lambda^{\approx}$ obtained by contracting "in parallel" for each Beta-redox $M \rightarrow N$ the $\beta$-redexes obtained after $\sigma$-normalisation in $\sigma(M)$. Because every two such $\beta$-redexes are disjoint, the different computations of the $\beta$-redexes are equal modulo $\approx$.

We obtain the commutative diagrams:

$$\begin{array}{ccc}
\lambda^{\approx} & \xrightarrow{U} & \lambda\sigma^{\approx}
\end{array} \quad \begin{array}{ccc}
\lambda\sigma^{\approx} & \xrightarrow{\sigma} & \lambda^{\approx}
\end{array}$$

(7)

The existence of a functor $\lambda^{\approx} \rightarrow \lambda\sigma^{\approx}$ making the first diagram commute is a consequence of the fact that two disjoint $\beta$-redexes induce disjoint Beta and $\sigma$-normalisation procedures in $\lambda\sigma$. As for the second diagram, there exists no functor $\lambda\sigma^{\approx} \rightarrow \lambda^{\approx}$ making it commute because two disjoint Beta-redexes in a $\lambda\sigma$-term $M$ may have nested $\sigma$-projections in $\sigma(M)$, for instance when $M = P[P \cdot id]$ and $P = (\lambda x)1 = (\lambda x[x])1$. 

19
5 From standard $\lambda\sigma$-paths to standard $\lambda$-paths

Starting from the two functors discussed in section 4

$$U : \lambda^\omega \rightarrow \lambda\sigma^\omega \quad \sigma : \lambda\sigma^\rightarrow \rightarrow \lambda^\omega$$

we observe that $\sigma$ does not preserve standardness in general, for instance when it translates the standard rewriting $\lambda\sigma$-path

$$(\lambda111)[(\lambda1)1 \cdot id] \rightarrow ((\lambda111)[1[1 \cdot id] \cdot id] \rightarrow (11)[1 \cdot id][1[1 \cdot id] \cdot id]$$

as an inside-out computation in the $\lambda$-calculus

$$(\lambda x.x)(\lambda y.y)1 \rightarrow (\lambda x.x)1 \rightarrow 11$$

We develop in section 7.2 a simple solution to that mismatch by considering that redexes of $s$ are nested redexes of $M$ in the $\lambda\sigma$-term $M[\sigma]$. This “dynamical” ordering prevents the path $(8)$ to be standard, and we check in section 7.2 that the new and tighter notion of standardness is preserved by interpretation $\sigma$.

Here, we adopt a much simpler approach and prove that $\sigma$ preserves traditional standardness when one restricts to paths $f : M \rightarrow U(N)$ ending on the de Bruijn translation $U(N)$ of a $\lambda$-term $N$.

**Lemma 22** Every standard rewriting path $f : M \rightarrow U(N)$ in $\lambda\sigma^\rightarrow$ is interpreted as a standard path $\sigma(f) : \sigma(M) \rightarrow N$ in $\lambda^\omega$.

We start the proof with a definition and two lemmas.

**Definition 23 (enshrine, annihilate, preserve)** An occurrence $x$ enshrines a redex $u : P \rightarrow Q$ when $x$ is a strict prefix of the occurrence of $u$. A redex $u$ annihilates an occurrence $x$ when the occurrence of $u$ is a prefix of $x$. A path $u_1; \cdots; u_n$ preserves an occurrence $x$ when none of the redexes $u_i$ annihilates $x$.

**Lemma 24** Suppose that an occurrence $x$ enshrines a redex $r : P \rightarrow Q$. For every rewriting path $\lambda\sigma$-path $f = r_1 g$,

1. either $f$ preserves the occurrence $x$,

2. or $f$ is equal modulo $\simeq$ to a rewriting path $h_1; u_2; v; h_2$ where the path $h_1$ preserves the occurrence $x$; the occurrence $x$ enshrines the redex $u$; the redex $v$ annihilates the occurrence $x$.

**Proof** Let $u_1; \cdots; u_j : P \rightarrow R$ be a $\lambda\sigma$-path which preserves the occurrence $x$, and contracts a redex enshrined in $x$. Let $u_k$ be the last redex enshrined in $x$. If $k < j$, the redex $u_{k+1}$ which is not enshrined in $x$ may be permuted reversibly before $u_k$:
Observe that the resulting path \( u_2 \cdots ; u_k \cdots ; u_{j+1} \cdots ; u_j \) preserves \( x \), and contracts the redex \( u_{k+1} \) enshrined in \( x \). Repeating the process \( j-k \) times, one constructs a path \( v_1 \cdots ; v_j \cdots ; u_j \) whose last redex \( v_j \) is enshrined in \( x \).

Let \( f = r_1 \cdots ; r_n \) be a path and \( x \) an occurrence which enshrines \( r_1 \). We suppose that \( f \) does not preserve \( x \). Let \( r_{j+1} \) be the first redex in \( f = r_1 \cdots ; r_n \) which annihilates \( x \). By our previous result, there exists a path \( v_1 \cdots ; v_j \) equal to \( r_1 \cdots ; r_j \) modulo \( \simeq \) whose last redex \( u = v_j \) is enshrined in \( x \). We conclude. ■

Given a pair of composable \( \lambda \sigma \)-redexes \( M \xrightarrow{\sigma} N \xrightarrow{\sigma} P \), we say that the redex \( u \) creates the redex \( v \) when \( v \) is not the residual after \( u \) of any redex in \( M \). We illustrate the definition with two examples:

\[
(\lambda M)N[s] \xrightarrow{\sigma} M[N \cdot id][s] \xrightarrow{\sigma} M[(N \cdot id) \circ s]
\]

or

\[
(\lambda M)[s]N \xrightarrow{\sigma} (\lambda M[1 \cdot (id \circ \uparrow)])N \xrightarrow{\sigma} M[1 \cdot (id \circ \uparrow)][N \cdot id]
\]

(9)

**Lemma 25** Suppose that an occurrence \( x \) enshrines a redex \( r : P \rightarrow Q \). Every standard path \( f = r_\gamma g \) preserves the occurrence \( x \) when:

- \( x \) is a cons-node \( (M \cdot s) \),
- \( x \) is a \( \lambda \)-node \( \lambda M \),
- \( x \) is an application node \( MN \) and the path \( f \) is a \( \sigma \)-path.

**Proof** Suppose that an occurrence \( x \) annihilated by \( v \) enshrines \( u \) in a pair \( M \xrightarrow{\sigma} N \xrightarrow{\sigma} P \). Clearly, the path \( u_x v \) cannot be standard, unless \( u \) creates \( v \). Inspecting the eleven rewriting rules of the \( \lambda \sigma \)-calculus in figure 1, we see that creation is forbidden in three cases at least:

1. When the occurrence \( x \) is a cons-node \( M \cdot s \) because \( M \cdot s \) is "impervious" to any interaction with \( M \) or \( s \),
2. When the occurrence \( x \) is a \( \lambda \)-node \( \lambda M \) because \( \lambda M \) is "impervious" to any interaction with \( M \),
3. When the occurrence \( x \) is an application node \( MN \) and \( v \) is a \( \sigma \)-redex; the only possible pattern of creation appears in (9) where \( u \) is an Abs-redex and \( v \) is a *Beta*-redex.

We apply lemma 24 on the standard rewriting path \( f = r_\gamma g \) and the occurrence \( x \) enshrining \( r \). By cases 1—3, the second alternative cannot hold when \( x \) is a cons-node or a \( \lambda \)-node, or when \( x \) is an application node and \( f \) does not contain any *Beta*-redex. We conclude.

**Proof of Lemma 22.** By standard path in \( \lambda \omega \), we mean the equivalence class of a standard rewriting path in \( \lambda \omega \). The main thing to observe is that no \( \lambda \sigma \)-redex contracted in the standard path \( f : M \rightarrow U(N) \) ever appears enshrined in a cons-node \( M \cdot s \). We proceed by contradiction. Suppose that there exists a redex \( r_i \) contracted in \( f = r_1 \cdots ; r_n \) enshrined in the occurrence \( x \) of a cons-node. We reach a contradiction between two facts: \( U(N) \) does not contain any cons-node, and by lemma 25 the path \( r_i \cdots ; r_n : P \rightarrow U(N) \) preserves \( x \). We conclude.

21
It follows that no Beta-redex contracted in \( f \) ever occurs inside a substitution \( s \). Of course, the remark does not extend to the \( \sigma \)-redexes contracted in \( f \), as the following standard path \( f : M \rightarrow U(N) = P \) illustrates:

\[
1(1 \cdot id) \circ \sigma (P \cdot id) \overset{\lambda \rightarrow}{\mapsto} 1[1 \cdot id] \circ \lambda (P \cdot id) \overset{\text{Var} \text{Cons}}{\rightarrow} 1[P \cdot id] \overset{\text{Var} \text{Cons}}{\rightarrow} P
\]

Anyhow, the property implies that every Beta-redex contracted in \( f \) is translated as a unique \( \beta \)-redex in \( \sigma(f) \). This means that whenever \( f : M \rightarrow U(N) \) is standard, we may consider \( \sigma(f) \) as a proper rewriting path in \( \lambda \), not just an equivalence class of \( \simeq \) in \( \lambda \). We show now that any standardisation cell \( \alpha : \sigma(f) \Rightarrow h \) in \( \lambda \) may be mirrored as a standardisation cell \( \beta : f \Rightarrow g \) in \( \lambda \), in the sense that \( \sigma(g) = h \).

Let us write \( \sigma(f) = R_1; \ldots; R_p \) and \( \phi : \{1, \ldots, p\} \rightarrow \{1, \ldots, n\} \) the function which associates to any \( \beta \)-redex \( R_k \) in \( \sigma(f) \) the corresponding Beta-redex \( R_{\phi(k)} \) in \( f \).

Let \( R_k \) and \( R_{k+1} \) be two consecutive \( \beta \)-redexes in \( \sigma(f) \). The \( \lambda \)-path \( r_i; \ldots; r_j = r_{\phi(k+1)}; \ldots; r_{\phi(k+1)-1} \) between \( R_k \) and \( R_{\phi(k+1)} \) contracts only \( \sigma \)-redexes.

Suppose that the two \( \beta \)-redexes \( R_k \) and \( R_{k+1} \) are permuted in \( \sigma(f) \) using a reversible permutation \( \mathcal{R}_k : R_{k+1} \simeq R_k \). We construct \( g \) such that \( \sigma(g) = R_1; \ldots; R'_k; R'_{k+1}; \ldots; R_p \). By lemma 25, the path \( r_i; \ldots; r_j \) preserves any application node enshrining \( r_{\phi(k)} \), and in particular, the lowest application node appearing above \( R_k : P \rightarrow Q \) and \( R_{k+1} \) in the \( \lambda \)-term \( \sigma(P) \). It is straightforward then that \( r_{\phi(k)}; r_i; \ldots; r_j; r_{\phi(k+1)} \) may be reorganised modulo \( \simeq \) as a rewriting path \( g \) with projection \( \sigma(g) = R_k; R'_k; R_{k+1} \).

We claim also that the two \( \beta \)-redexes \( R_k \) and \( R_{k+1} \) cannot be permuted using an irreversible permutation \( \mathcal{R}_k : R_k \Rightarrow R'_k; R_{k+1} \). We proceed by contradiction. By lemma 25, the path \( r_i; \ldots; r_n \) preserves every \( \lambda \)-node enshrining \( r_{\phi(k)} \). Among these \( \lambda \)-nodes, the \( \lambda \)-node involved in \( R_{k+1} \) after \( \sigma \)-normalisation is annihilated by \( r_{\phi(k+1)} \). We reach a contradiction and conclude.

**Lemma 26** Every external rewriting path \( e : M \rightarrow N \) in \( \lambda \) is interpreted as an external path \( \sigma(e) : \sigma(M) \rightarrow \sigma(N) \) in \( \lambda \).

**Proof** Because \( \sigma : \lambda \rightarrow \lambda \simeq \lambda \) respects the permutation equivalence \( \simeq \), we may reformulate lemma 22 as:

\[
\sigma(\downarrow f) = \downarrow \sigma(f)
\]

for any \( \lambda \)-path \( f : M \rightarrow U(N) \).

Now, let \( e : M \rightarrow N \) be an external \( \lambda \)-path in the \( \lambda \)-calculus, and \( f \) any \( \lambda \)-path \( f : \sigma(N) \rightarrow P \). We call \( h : N \rightarrow U(\sigma(N)) \) any \( \sigma \)-normalisation path from \( N \).

\[
\downarrow \sigma(e); f = \downarrow \sigma(e) \downarrow \sigma(h); \sigma(U(f)) \overset{\text{by } \sigma(U(f)) = f, \text{ functionality of } \sigma, \text{ and } \downarrow \sigma(h; U(f))}{\downarrow \sigma(e) \downarrow \sigma(h; U(f))} = \sigma(e); \sigma(h; U(f)) \overset{\text{by } \sigma(h; U(f)), \text{ functionality of } \sigma, \text{ and } \sigma(e); \downarrow \sigma(h; U(f))}{\downarrow \sigma(e); \sigma(h; U(f))} \overset{\text{by } \sigma(h; U(f)), \text{ functionality of } \sigma, \text{ and } \downarrow \sigma(e); \sigma(h; U(f))}{\downarrow \sigma(e); \sigma(h; U(f))} \overset{\text{by } \sigma(h; U(f)), \text{ functionality of } \sigma, \text{ and } \downarrow \sigma(e); \sigma(h; U(f))}{\downarrow \sigma(e); \sigma(h; U(f))} \overset{\text{by } \sigma(h; U(f)), \text{ functionality of } \sigma, \text{ and } \downarrow \sigma(e); \sigma(h; U(f))}{\downarrow \sigma(e); \sigma(h; U(f))} \overset{\text{by } \sigma(h; U(f)), \text{ functionality of } \sigma, \text{ and } \downarrow \sigma(e); \sigma(h; U(f))}{\downarrow \sigma(e); \sigma(h; U(f))} \overset{\text{by } \sigma(h; U(f)), \text{ functionality of } \sigma, \text{ and } \downarrow \sigma(e); \sigma(h; U(f))}{\downaward}{\downarrow f} \overset{\text{by } \sigma(h; U(f)), \text{ functionality of } \sigma, \text{ and } \downarrow \sigma(e); \sigma(h; U(f))}{\downarrow f}
\]

We conclude.
6 Main result

We prove that

**Theorem 2** Every closed λσ-term enjoys a finite normalisation cone.

**Proof** The proof is by contradiction. Suppose that there exists a closed λσ-term \( M \) with an infinite number of normalising λσ-path \( M \rightarrow N \), modulo Lévy permutation equivalence \( \equiv \).

**First step: König's lemma.** We construct the infinite tree \( T \) whose nodes are the standard paths \( M \rightarrow N \) which may be extended to a standard normalising path \( M \rightarrow P \rightarrow N \). The nodes are ordered by prefix ordering on paths. Because every λσ-term contains at most a finite number of λσ-redexes, we may apply König's lemma and deduce that there exists an infinite path \( f_\infty \) in the tree.

**Second step: strong σ-normalisation.** By strong normalisation of the substitution σ-calculus established in [6, 25], the infinite path \( f_\infty \) contracts an infinite number of Beta-redexes.

**Third step: lemma 26.** By definition of \( T \), every finite prefix \( f : M \rightarrow P \) of \( f_\infty \) may be extended to a standard normalising path \( M \rightarrow P \rightarrow N \). By lemma 9, the path \( f;g \) is external. By lemma 26, its projection \( \sigma(f;g) \) is external. Every Beta-redex \( r : P \xrightarrow{\beta} Q \) contracted in \( f \) projects as a β-redex \( \sigma(r) : \sigma(P) \rightarrow \sigma(Q) \) contracted in the external λ-path \( \sigma(f;g) \). The β-redex \( \sigma(r) \) is external by the two decomposition lemmas 11 and 12 applied on \( \sigma(f;g) \).

**Last step: needed normalisation in the λ-calculus.** We conclude from step 2 and 3, that projecting \( f_\infty \) constructs an infinite external λ-path from \( \sigma(M) \). We reach a contradiction between three facts: the λ-term \( \sigma(M) \) has a normal form \( \sigma(N) \), external paths are needed by lemma 17, and needed strategies are normalising in the λ-calculus. We conclude.

We apply theorem 1 and deduce that:

**Corollary 27** Every needed strategy is normalising in the closed λσ-calculus.

We do not prove that the calculus of open λσ-terms enjoys finite normalisation cones. This, we do not know. It should be possible however to extend the proof to the λσ-calculus of open terms with no substitution variable.

To conclude, let us stress that λσ-neededness does not depend on neededness in the λ-calculus: it is an internal notion provided by definition 15. We may illustrate it as follows. The *vertebra* of a λσ-term \( M \) is defined inductively as a λσ-term vertebra(M) with extra constant \(-\):

1. vertebra(1) = 1, 3. vertebra(PQ) = (vertebra(P))−,
2. vertebra(λP) = λ(vertebra(P)), 4. vertebra(P[s]) = (vertebra(P))[−].
The spine \( \text{spine}(M) \) of \( M \) is defined as \( \text{vertebra}(M) \) when \( M \) is not of the form \( M = \lambda \cdots \lambda (i_1 \cdots i_n) \) for a de Bruijn number \( i = 1[\Downarrow \circ \cdots \circ \Downarrow] \), and as
\[
\text{spine}(M) = \lambda \cdots \lambda (i \text{ spine}(M_1) \cdots \text{spine}(M_n))
\]
when \( M = \lambda \cdots \lambda (i_1 \cdots i_n) \). Call spine every \( \lambda \sigma \)-redex \( u : M \to N \) whose occurrence appears not replaced by the constant \( \bot \) in the spine of \( M \). An easy argument shows that every spine redex is properly needed. We apply corollary 27 and deduce that every \( \lambda \sigma \)-strategy contracting a spine redex at each step (or eventually) is normalising.

7 Appendix

7.1 Head-normalisation

Theorem 2 may be extended to consider values instead of just normal forms. We start with the axiomatic definition of a value-set appearing in [23], adapted from Glauert and Khasidashvili’s definition in [14]. Given an axiomatic rewriting system \( (G, \Rightarrow) \), we say that a redex \( u \) nests another redex \( v \), and write \( u \prec v \), when there exists an irreversible permutation
\[
(M \Downarrow u \Rightarrow Q \downarrow \Rightarrow N) \Rightarrow (M \Downarrow u \Rightarrow P \downarrow \Rightarrow N)
\]
We say that two redexes \( u \) and \( v \) are disjoint, and write \( u \parallel v \), when there exists a reversible permutation
\[
(M \Downarrow u \Rightarrow Q \downarrow \Rightarrow N) \Rightarrow (M \Downarrow P \downarrow \Rightarrow N)
\]
Two redexes \( u : M \to P \) and \( v : M \to Q \) are declared compatible when \( u \prec v \), \( u \parallel v \) or \( v \prec u \). The axiomatics of [21] insures that the three cases \( u \parallel v \), \( u \prec v \) and \( v \prec u \) are exclusive.

A set \( V \) of vertices is declared open-stable in \( (G, \Rightarrow) \) when the three properties below are verified by every permutation diagram \( v; u \Rightarrow u; f \):
\[
\begin{array}{ccc}
\alpha & \downarrow & \beta \\
M & \Downarrow v & Q \\
\Downarrow & \Rightarrow & \Downarrow \\
\Downarrow & \Rightarrow & \Downarrow \\
P & \Downarrow f & N \\
\end{array}
\]
[head] if \( u \prec v \) and \( Q \in V \), then \( M \in V \),
[stable] if \( u \parallel v \) and \( P, Q \in V \), then \( M \in V \),
[closed] if \( M \to N \) and \( M \in V \), then \( N \in V \)

We prove in [23] that from any term \( M \in G \), there exists a cone \( (e_i : M \to V_i)_{i \in I} \) of minimal rewriting paths to the set \( V \). By minimal, we mean that each \( V_i \in V \), and that for every path \( f : M \to V \) with \( V \in V \), there exists one and only one \( i \in I \) such that \( f \) factors as \( f \equiv e_i ; h \) for some path \( h \).

\[
\begin{array}{ccc}
M & \Downarrow e_i & V_i \\
\Downarrow f & \Rightarrow & \Downarrow h \\
\Downarrow & \Rightarrow & \Downarrow \\
V & & \\
\end{array}
\]
Moreover, the rewriting path \( h \) is unique modulo \( \equiv \). Here, we establish that the “head-normalisation” cone \( (e_i : M \rightarrow V_i)_{i \in I} \) is finite for any reasonable notion of value-set \( V \) in the \( \lambda \sigma \)-calculus. By reasonable, we mean the three following properties of \( V \) in \((G_{\lambda \sigma}, \triangleright_{\lambda \sigma})\):

1. \( V \) is open-stable in \((G_{\lambda \sigma}, \triangleright_{\lambda \sigma})\),
2. its projection \( \sigma(V) \) is open-stable in \((G_{\lambda}, \triangleright_{\lambda})\),
3. for every Beta-redex \( u : M \rightarrow N \) between two \( \lambda \sigma \)-terms \( M \) and \( N \) with projections \( \sigma(M) \) and \( \sigma(N) \) in \( \sigma(V) \), and every normalisation \( \sigma \)-path \( f : N \rightarrow U(\sigma(N)) \) inducing a standard path \( M \triangleright_{\lambda \sigma} N \triangleright_{\lambda \sigma} U(\sigma(N)) \), there exists a path \( M \triangleright_{\lambda \sigma} V \triangleright_{\lambda \sigma} U(\sigma(N)) \) verifying:

\[
M \triangleright_{\lambda \sigma} N \triangleright_{\lambda \sigma} U(\sigma(N)) \simeq M \triangleright_{\lambda \sigma} V \triangleright_{\lambda \sigma} U(\sigma(N))
\]

such that \( f_1 \) is a \( \sigma \)-path and \( V \) is an element of \( V \).

Another way to state property 3 is to ask that every path \( e_i \) in the cone \((e_i : M \rightarrow V_i)_{i \in I}\) from \( M \) to \( V \) is a \( \sigma \)-path when \( M \) is a \( \lambda \sigma \)-term with interpretation \( \sigma(M) \) in \( \sigma(V) \). As an illustration, we call \( \lambda \sigma \)-head-normal form any closed \( \lambda \sigma \)-term written as \( \lambda \sigma \lambda(i_1, \ldots, i_n) \) where \( i = 1[\uparrow \circ \cdots \circ \uparrow] \) is a de Bruijn number, and \( i_1, \ldots, i_n \) are \( \lambda \sigma \)-terms. An easy syntactical reasoning shows that the set \( V_{\lambda \sigma} \) of \( \lambda \sigma \)-head-normal forms verifies the three properties 1—3.

**Theorem 3** Let \( V \) be a set of closed \( \lambda \sigma \)-terms verifying properties 1—3, and \( M \) a closed \( \lambda \sigma \)-term. The normalisation cone \((e_i : M \rightarrow V_i)_{i \in I}\) from \( M \) to \( V \) is finite, and every path \( e_i : M \rightarrow V_i \) in the cone projects as the unique path \( e : \sigma(M) \rightarrow V \) in the normalisation cone from \( \sigma(M) \) to \( \sigma(V) \).

**Proof** We show that the cone \((e_i : M \rightarrow V_i)_{i \in I}\) is finite. By \( \sigma \)-strong normalisation, we only need to prove that the number of \( \text{Beta-redex} \) \( e_i \) is bounded. This will follow from the fact that every path \( e_i \) projects as the canonical path \( e : \sigma(M) \rightarrow V \) defined by the open-stable set \( \sigma(V) \) in the \( \lambda \sigma \)-calculus. Let \( e_i : M \rightarrow V_i \) be any path in the cone, and \( f : V_i \rightarrow U(\sigma(V_i)) \) a \( \sigma \)-path normalising \( V_i \). Recall from our stability theorem in [23] that \( e_i \) is external. This implies here that \( g = e_i \upharpoonright f \) is standard. By lemma 22, its projection \( g = \sigma(g) \) is standard. We claim that \( g \) is precisely the canonical path \( e : M \rightarrow V \), modulo \( \simeq \). We proceed by contradiction, and suppose that \( \neg(\sigma(g) \simeq e) \). The proof of lemma 22 informs us that to every \( \lambda \sigma \)-path \( h \) equal to \( g \) modulo \( \simeq \), there corresponds a \( \lambda \sigma \)-path \( h \simeq g \) with projection \( \sigma(h) = h \). We may reorganise \( g = M \triangleright_{\lambda \sigma} V_i \upharpoonright_{\lambda \sigma} U(\sigma(V_i)) \) as \( h = M \triangleright_{\lambda \sigma} P \triangleright_{\lambda \sigma} Q \triangleright_{\lambda \sigma} U(\sigma(V_i)) \) modulo \( \simeq \), in such a way that \( r \) is a \( \text{Beta-redex} \), that \( h_2 \) a \( \sigma \)-path, and that \( \sigma(P) \in \sigma(V) \). By property 3, the standard path \( r; h_2 \) may be reorganised modulo \( \simeq \) as a \( \sigma \)-path \( h \triangleright_{\lambda \sigma} W \in V \) followed by a \( \lambda \sigma \)-path \( W \triangleright_{\lambda \sigma} U(\sigma(V_i)) \). We obtain the series of equivalence:

\[
M \triangleright_{\lambda \sigma} V_i \upharpoonright_{\lambda \sigma} U(\sigma(V_i)) \simeq M \triangleright_{\lambda \sigma} P \triangleright_{\lambda \sigma} Q \triangleright_{\lambda \sigma} U(\sigma(V_i)) \\
\simeq M \triangleright_{\lambda \sigma} P \triangleright_{\lambda \sigma} W \triangleright_{\lambda \sigma} U(\sigma(V_i))
\]
It follows from $W \in \mathcal{V}$ and the definition of $(e_i : M \to V_i)_{i \in I}$ that $e_i \sqsubseteq h_1; h_3$. By externality of $e_i$ and standardness of $h_1; h_3$, there exists a path $h_4 : V_i \to W$ verifying

$$e_i; h_4 \simeq h_1; h_3$$

This contradicts the fact that $e_i$ contracts strictly more Beta-redexes than $h_1; h_3$. We conclude. ■

### 7.2 A dynamical order

We relax the assumption that $f : M \to U(N)$ in lemma 22. Observe that the tree nesting ordering on $\lambda\sigma$-terms is not completely satisfactory: a substitution $s$ and a term $M$ are disjoint in the $\lambda\sigma$-term $M[s]$ despite the fact that the substitution $s$ appears nested in $M$ after projection $\sigma$. So, we introduce another more “dynamical” ordering $\preceq\star$ defined as the least order on $\lambda\sigma$-redexes extending the usual tree ordering, closed under context, and verifying:

1. $u \preceq\star v$ whenever the $\lambda\sigma$-redex $u$ occurs in $M$ and the $\lambda\sigma$-redex $v$ occurs in $s$ in a term $M[s]$,

2. $u \preceq\star v$ whenever the $\lambda\sigma$-redex $u$ occurs in $s$ and the $\lambda\sigma$-redex $v$ occurs in $t$ in a substitution $s \circ t$.

We define the system $(G\star, \succ \star)$ as $G\star = G\lambda\sigma$ and $v; u' \succ \star v; f$ precisely when $v; u' \succ \lambda\sigma u; f$ and $- (v \preceq\star u)$. By lack of space, we do not prove here that $(G\star, \succ \star)$ defines an axiomatic rewriting system on $\lambda\sigma$-redexes. The alteration of $\preceq$ into $\preceq\star$ allows to improve diagram (7) in section 4.

**Lemma 28** We have the following commutative diagrams:

\[
\begin{array}{ccc}
\lambda\sigma \simeq & \downarrow & \lambda\sigma \simeq \\
\lambda \to & \mathcal{U} \star & \lambda \to \\
\end{array}
\]

\[
\begin{array}{ccc}
\lambda \simeq & \downarrow & \lambda \simeq \\
\lambda \sigma \to & \mathcal{U} \star & \lambda \sigma \to \\
\end{array}
\]

where $\lambda\sigma \simeq = \lambda\sigma / \simeq\star$ denotes the free category $\lambda\sigma \star$ quotiented by the relation $\simeq\star$ derived from $\succ \star$.

**Proof** The construction of the two functors $\mathcal{U} \star$ and $\sigma\star$ and the completion of the first diagram are straightforward. Here, we complete the second diagram and prove that $\sigma\star$ respects the relation $\simeq$. This reduces to the claim that two Beta-redexes disjoint wrt. $\preceq\star$ are interpreted as a set of disjoint $\beta$-redexes after $\sigma$-normalisation. Since Beta-redexes usually disappear and reappear in the process of $\sigma$-normalisation, we switch to Beta-patterns, defined as any pattern of the form $(\lambda M)[s_1] \cdots [s_m])N$ for $m \geq 0$ in a $\lambda\sigma$-term $P$. Any Beta-redex is a Beta-pattern, and two Beta-patterns $u$ and $v$ are declared disjoint when their application nodes are disjoint wrt. the tree nesting ordering. By residual of a Beta-pattern in $P$ through a $\lambda\sigma$-redex $r : P \to Q$, we mean the obvious syntactical notion, keeping in mind that a $\beta$-redex obtained by $\sigma$-normalisation from a Beta-pattern should be its residual. Finally, given any two
disjoint Beta-patterns $u$ and $v$ in $P$, we write $u \land v$ the longest occurrence above $u$ and $v$ in $P$ wrt. the tree nesting ordering. This node $u \land v$ is either a node $MN$, $M[s]$, $M \ast s$ or $s \circ t$.

Let us observe that:

1. two different residuals $u'$ and $u''$ of a Beta-pattern $u$ after a $\lambda \sigma$-redex $r : P \to Q$ are necessarily disjoint, with $u' \land u''$ an application or a cons node. The $\sigma$-redex $r$ is an App-redex in the first case, a Map-redex in the second case, and $u$ always lies in the duplicated substitution,

2. let $u$ and $v$ be two disjoint Beta-patterns such that $u \land v$ is an application or cons node. Let $u'$ and $v'$ be residuals of $u$ and $v$ respectively after a $\sigma$-redex $r : P \to Q$. An easy inspection of the ten possible $\sigma$-rules for $r$ shows that $u'$ and $v'$ are disjoint and that $u' \land v'$ is an application or cons node.

We conclude that two disjoint $\beta$-redexes $u$ and $v$ in a $\lambda \sigma$-term $P$ project by $\sigma$-normalisation as a set of disjoint $\beta$-redexes in $\sigma(P)$.

Lemma 28 does not provide any remarkable adjunction property like (6), even extended to a polyadjunction property à la Taylor [24], see also [23]. However, it verifies that:

**Lemma 29** Every standard rewriting path $f : M \to N$ in $(G \star, \triangleright \ast)$ projects through $\sigma_\ast$ as a standard $\lambda$-path (more exactly: $\simeq$-class) in $\lambda^\omega$.

**Proof.** The whole argument is based on an elementary property of standardisation in the $\lambda$-calculus. Given any path $d : M \to N$, and a vector $f_i : P_i \to Q_i$ of paths, we may define the path

$$d[f_i/x_i] : M[P_i/x_i] \to N[Q_i/x_i]$$

obtained by applying first the rewriting path $d : M \to N$ on $M[P_i/x_i]$, then each rewriting path $f_i : P_i \to Q_i$ on $N[P_i/x_i]$. The construction is valid modulo $\simeq$, and $d[f_i/x_i]$ is standard whenever $d$ and each of the $f_i$'s is standard.

We define similarly $i[f] : i[s] \to i[t]$ for every $\lambda \sigma$-path $f : s \to t$ between substitutions $s$ and $t$, by applying $f : s \to t$ in front of the de Bruijn number $i = 1[\uparrow \circ \cdots \circ \uparrow]$.

We prove that:

- the projection $\sigma_\ast(f)$ is standard for every standard path $f : M \to N$,

- the projection $\sigma_\ast(i[f])$ is standard for every standard path $f : s \to t$ and every number $i \geq 1$,

by induction on the length of $f$, then on the size of $M$. When the path $f = u; g$ contracts a $\sigma$-redex $u$ at first step, the two paths $f$ and $g$ define by $\sigma_\ast$ the same $\simeq$-class in $\lambda^\omega$, which is standard by induction hypothesis applied to $g$. So, we suppose from now on that the first redex $u$ contracted by $f$ is a Beta-redex. We proceed by induction on the size of $M$ and $s$.

1. Suppose that $M = PQ$ or $M = P[s]$, and that $f$ does not preserve the root node. Because $f$ does not preserve any occurrence in $M$, by lemma 25, the Beta-redex $u$ is not enshrined by any cons-node $L \cdot t$ or $\lambda$-node $\lambda L$ in $M$. Because application nodes $KL$ are “impervious” to interactions with $L$, we could very well extend
lemma 25 and establish that \( u \) cannot appear in any subterm \( L \) inside a subterm \( KL \) of \( M \). All in all, we obtain that the Beta-redex \( u \) is translated in \( \lambda^\omega \) as an external \( \beta \)-redex, and conclude by induction hypothesis.

2. Suppose that \( M = PQ \) and that \( f : M \to N \) preserves the root node \( PQ \). The path \( f \) decomposes as two paths \( f_P : P \to P' \) and \( f_Q : Q \to Q' \) which are standard in \((G_{\beta}, d, \star)\). By induction hypothesis, their projection is standard, and we conclude that the projection of \( f \) is standard.

3. Suppose that \( M = P[s] \) and that \( f : M \to N \) preserves the root node \( P[s] \). The way we choose to define \( \sigma_\star \) implies that \( f \) is the sequential composite \( f = f_P \circ f_s \) of a path \( f_P : P \to Q \) and a path \( f_s : s \to t \). By induction hypothesis, \( \sigma_\star(f_P) \) and each \( \sigma_\star(i[f_s]) \) is standard in \( \lambda^\omega \). We deduce standardness of \( \sigma_\star(f) \) from the equality in \( \lambda^\omega \):

\[
\sigma_\star(f) = \sigma_\star(f_P)[\sigma_\star(i[f_s])]
\]

4. The case \( s = M \cdot s' \) is trivial. We treat the other case \( s = s_1 \circ s_2 \). The first redex contracted by \( f : s \to t \) is a Beta-redex and therefore enshrined in a cons-node. The path \( f \) which preserves the cons-node preserves a fortiori the root node \( s_1 \circ s_2 \). Moreover, the choice of \( \sigma \) implies that \( f \) is the sequential composite \( f = f_1 \cdot f_2 \) of a path \( f_1 : s_1 \to t_1 \) and a path \( f_2 : s_2 \to t_2 \). We reason as in case 2, and deduce that each \( i[f] : i[s] \to i[t] \) projects as a standard morphism in \( \lambda^\omega \).

\[ \square \]

Acknowledgements

The author would like to thank Pierre-Louis Curien and Thérèse Hardin for their enthusiasm and support.

References


