A Normalisation Result for Higher-Order Calculi with Explicit Substitutions

Eduardo Bonelli[1,2]

[1] LIFIA, Facultad de Informática, Universidad Nacional de La Plata, 50 y 115,
La Plata (1900), Buenos Aires, Argentina.
[2] Department of Computer Science, Stevens Institute of Technology,
Hoboken, NJ 07030, USA.
ebonelli@cs.stevens-tech.edu

Abstract. Explicit substitutions (ES) were introduced as a bridge between the theory of rewrite systems with binders and substitution, such as the λ-calculus, and their implementation. In a seminal paper P.-A. Melliès observed that the dynamical properties of a rewrite system and its ES-based implementation may not coincide: he showed that a strongly normalising term (i.e. one which does not admit infinite derivations) in the λ-calculus may lose this status in its ES-based implementation. This paper studies normalisation for the latter systems in the general setting of higher-order rewriting: Based on recent work extending the theory of needed strategies to non-orthogonal rewrite systems we show that needed strategies normalise in the ES-based implementation of any orthogonal pattern higher-order rewrite system.

1 Introduction

This paper studies normalisation for calculi of explicit substitutions (ES) implementing higher-order term rewrite systems (HORS). The latter are rewrite systems in which binders and substitution are present, the λ-calculus [Bar84] being a typical example. A recent approach to the implementation of HORS is the use of ES [ACCL91, BBLRD96, KR95, DG01]. ES were introduced as a bridge between HORS and their concrete implementations. Their close relation with abstract reduction machines [ACCL91, HMP96, BBLRD96] allows us to speak of ES-based *implementations* of HORS. The idea behind these implementations is that the complex notion of substitution is promoted to the object-level by introducing new operators into the language in order to compute substitutions explicitly. This allows HORS to be expressed as more fine-grained (first-order) rewrite systems in which no complex substitution nor binders are present. Such a process may be applied to any HORS [BKR01]. As an example Fig. 1 shows the rules of $\lambda\sigma$ [ACCL91], a calculus of ES implementing the λ-calculus (based on de Bruijn indices notation [dB72] in order to discard annoying issues related to the renaming of variables[1]).

[1] Variables in terms are represented as positive integers. Eg. $\lambda x.x$ is represented as $\lambda 1$, $\lambda x.\lambda y.x$ as $\lambda\lambda 2$, and $\lambda x.y$ as $\lambda 2$.

A.D. Gordon (Ed.): FOSSACS 2003, LNCS 2620, pp. 153–168, 2003.

$$\begin{array}{ll} \text{Terms} & X := n \mid XX \mid \lambda X \mid X[s] \\ \text{Substitutions} & s := id \mid {\uparrow} \mid X \cdot s \mid s \circ s \end{array}$$

$$\begin{array}{lll@{\qquad}lll}
(\lambda X)\,Y & \rightarrow_{Beta} & X[Y \cdot id] & & & \\
(X\,Y)[s] & \rightarrow_{App} & X[s]\,Y[s] & (X \cdot s) \circ t & \rightarrow_{Map} & X[t] \cdot (s \circ t) \\
(\lambda X)[s] & \rightarrow_{Lam} & \lambda X[1 \cdot (s \circ {\uparrow})] & id \circ s & \rightarrow_{IdL} & s \\
X[s][t] & \rightarrow_{Clos} & X[s \circ t] & (s_1 \circ s_2) \circ s_3 & \rightarrow_{Ass} & s_1 \circ (s_2 \circ s_3) \\
1[X \cdot s] & \rightarrow_{VarCons} & X & {\uparrow} \circ (X \cdot s) & \rightarrow_{ShiftCons} & s \\
1[id] & \rightarrow_{VarId} & 1 & {\uparrow} \circ id & \rightarrow_{ShiftId} & {\uparrow}
\end{array}$$

Fig. 1. The $\lambda\sigma$ calculus

An obstacle which arises when using ES for implementing HORS is that results on normalisation which hold in the higher-order rewriting setting may not be preserved in its implementation. A well-known example of this mismatch is due to Melliès [Mel95]: he exhibited a strongly normalising (typed) term in the λ-calculus for which the $\lambda\sigma$-calculus introduces an infinite derivation. However, the problem is not confined to the setting of λ-calculus but rather affects any HORS. For example the following well-typed Haskell program:

```
map (\x → (map id [ map id [true] ])) [ map id [true] ]
```

(where `id` abbreviates `\x → x`) is easily seen to be strongly normalising (and reduces to `[[[true]]]`), however its ES-based implementation (cf. Ex. 2) may introduce infinite derivations for it in a similar way to [Mel95] (see [Bon03]).

This mismatch calls for careful consideration of normalising strategies in the context of ES-based implementations of HORS. This paper studies normalisation in the latter systems based on *needed* strategies, a notion introduced in [HL91]. Needed strategies are those which rewrite redexes which are "needed" (cf. Section 2.2) in order to attain a normal form, assuming it exists. For eg. the underlined redex in $1[2 \cdot \underline{(\lambda 1)\,1} \cdot id]$ is not "needed" in order to achieve a normal form since there is derivation, namely $1[2 \cdot (\lambda 1)\,1 \cdot id] \rightarrow_{VarCons} 2$, that never reduces it. In fact the infinite $\lambda\sigma$-derivation of the aforementioned Haskell program takes place inside a substitution s in a term of the form $1[X \cdot s]$.

The literature on needed strategies for HORS has required the systems to be *orthogonal* [HL91, Mar92, GKK00][2]. A system is orthogonal if no conflicts (overlap) between redexes may arise. Neededness for orthogonal systems does not suffice in our setting since HORS (even orthogonal ones) are implemented as *non-orthogonal* systems in the ES-based approach. For eg., although the λ-calculus is orthogonal, $\lambda\sigma$ is not, as witnessed by the critical pair: $(\lambda X)[s]\,Y[s] \leftarrow_{App} ((\lambda X)\,Y)[s] \rightarrow_{Beta} X[Y \cdot id][s]$. However, recently an extension has been introduced for non-orthogonal systems [Mel96, Mel00]. Motivated by this work on needed derivations for non-orthogonal systems we prove the following new result: all needed strategies in ES-based implementations of arbitrary

[2] An exception is [Oos99] however only weakly orthogonal systems are studied.

orthogonal pattern HORS normalise. This extends the known result [Mel00] that needed strategies in $\lambda\sigma$ normalise.

As an example of (one of) the issues which must be revisited in the extended setting of HORS is a result [Mel00, Lem.6.4] which states that if the first redex in a *standard* $\lambda\sigma$-derivation occurs under a "λ" symbol, then the whole derivation does so too. A standard derivation is one in which computation takes place in an outside-in fashion (Def. 1). What makes "λ" so special in this regard in the $\lambda\sigma$-calculus is that creation of redexes above "λ", from below it, is not possible. We introduce the notion of *contributable symbol* in order to identify these special symbols in the ES-based implementation of arbitrary higher-order rewrite systems (under our definition "λ" is uncontributable), and prove an analogous result.

Structure of the paper. Section 2 reviews the ERS_{db} higher-order rewriting formalism and the theory of neededness for orthogonal systems together with Melliès' extension to non-orthogonal ones. Section 3 identifies the Standard-Projection Proposition as the only requirement for an orthogonal pattern HORS to verify normalisation of needed strategies. Section 4 is devoted to verifying that the latter holds for HORS. We then conclude and suggest possible future research directions.

2 Setting the Scene

2.1 The ERS_{db} Formalism and Its ES-based Implementations

The ERS_{db} formalism. ERS_{db} is a higher-order rewriting formalism based on de Bruijn indices notation [BKR00]. Rewrite rules are constructed from metaterms; metaterms are built from: de Bruijn *indices* $1, 2, \ldots$, *metavariables* $X_l, Y_l, Z_l, \ldots$ where l is a label (i.e. a finite sequence of symbols) over an alphabet of *binder indicators* $\alpha, \beta, \ldots$, *function symbols* $f, g, h, \ldots$ equipped with an arity n with $n \geq 0$, *binder symbols* $\lambda, \mu, \nu, \xi, \ldots$ equipped with a positive arity, and a *metasubstitution* operator $M[\![N]\!]$. *Terms* are metaterms without occurrences of metavariables nor metasubstitution. A *rewrite rule* is a pair of metaterms $L \to R$ such that: the head symbol of L is either a function or a binder symbol, all metavariables occurring in R also occur in L (disregarding labels), and there are no metasubstitutions in L. An ERS_{db} $\mathcal{R}$ is a set of rewrite rules. The $\lambda\sigma$-calculus of Fig. 1 is an ERS_{db} (as well as all first-order rewrite systems). Two other examples are:

Example 1 (ERS_{db} rewrite rules).

$app(\lambda X_\alpha, Y_\epsilon)$	$\to_{\beta_{db}}$	$X_\alpha[\![Y_\epsilon]\!]$
$map(\xi X_\alpha, nil)$	$\to_{map.1}$	nil
$map(\xi X_\alpha, cons(Y_\epsilon, Z_\epsilon))$	$\to_{map.2}$	$cons(X_\alpha[\![Y_\epsilon]\!], map(\xi X_\alpha, Z_\epsilon))$

A *rewrite step* is obtained by instantiating rewrite rules with valuations; the latter result from extending assignments (mappings from metavariables to terms) to the full set of metaterms and computing metasubstitutions $M[\![N]\!]$ by the usual de Bruijn substitution [BKR00]. The labels in metavariables are used to restrict the set of valuations to "good" ones. For example, the classical η_{db} rule is written $\lambda(\mathit{app}(X_\alpha, 1)) \rightarrow_{\eta_{db}} X_\epsilon$. The X_α and X_ϵ indicate that a valuation is good if it assigns X_α and X_ϵ some term t in which 1-level indices do not occur free, for otherwise this index would be bound on the *LHS* and free on the *RHS*. Another example is $\mathit{imply}(\exists\forall X_{\alpha\beta}, \forall\exists X_{\beta\alpha}) \rightarrow_{\mathit{Comm}} \mathit{true}$ where a good valuation must verify that if t is assigned to $X_{\alpha\beta}$ then the term resulting from t by interchanging 1 with 2-level indices must be assigned to $X_{\beta\alpha}$ [BKR00].

A *redex* r is a triple consisting of a term M, a valuation, and a position[3] p in M such that M at position p is an instance of the *LHS* of a rewrite rule via the valuation; the induced rewrite step is written $M \rightarrow_r N$. Letters $r, s, t, \ldots$ stand for redexes. We use $\rightarrow_{\mathcal{R}}$ for the rewrite step relation induced by an ERS_{db} $\mathcal{R}$ and $\twoheadrightarrow_{\mathcal{R}}$ for its reflexive-transitive closure. A *derivation* is a sequence of rewrite steps; we use $|\phi|$ for the length (number of rewrite steps) of a derivation ϕ; letters $\phi, \varphi, \ldots$ stand for derivations. If $r_1, \ldots, r_n$ are composable rewrite steps then $r_1; \ldots; r_n$ is the derivation resulting from composing them. Derivations that start (resp. end) at the same term are called coinitial (resp. cofinal).

ES-based implementations of HORS. Any ERS_{db} may be implemented as a first-order rewrite system with the use of ES [BKR01, Bon01]. The implementation process (cf. Rem. 1) goes about dropping labels in metavariables and replacing metasubstitution operators $\bullet[\![\bullet]\!]$ in rewrite rules with explicit substitutions $\bullet[\bullet \cdot \mathit{id}]$. Also, new rules - the substitution calculus - are added in order to define the behavior of the new explicit substitutions; roughly, this calculus is in charge of propagating substitutions until they reach indices and then discarding the substitutions or replacing the indices. In this paper we use the σ-calculus [ACCL91] as substitution calculus[4], its rules have been presented in Fig. 1 (disregarding *Beta*); it is confluent [ACCL91] and strongly normalising [CHR92]. If $\mathcal{R}$ is an ERS_{db} then we write $\mathcal{R}^{\mathsf{ES}}_\sigma$ for its ES-based implementation and refer to it as an *implementation* (of $\mathcal{R}$).

Example 2.

$(\beta_{db})^{\mathsf{ES}}_\sigma = \lambda\sigma$		
$\mathit{map}^{\mathsf{ES}}_\sigma = \mathit{map.1}^{\mathsf{ES}} \cup \mathit{map.2}^{\mathsf{ES}} \cup \sigma$ where:		
$\mathit{map}(\xi X, \mathit{nil})$	$\rightarrow_{ES(\mathit{map.1})}$	nil
$\mathit{map}(\xi X, \mathit{cons}(Y, Z))$	$\rightarrow_{ES(\mathit{map.2})}$	$\mathit{cons}(X[Y \cdot \mathit{id}], \mathit{map}(\xi X, Z))$

Two basic properties of ES-based implementations of HORS are *Simulation* (if $M \rightarrow_{\mathcal{R}} N$ then for some M', $M \rightarrow_{\mathcal{R}^{\mathsf{ES}}} M' \twoheadrightarrow_\sigma \sigma(N)$, where $\sigma(N)$ denotes the σ-normal form of N) and *Projection* ($M \rightarrow_{\mathcal{R}^{\mathsf{ES}}_\sigma} N$ then $\sigma(M) \twoheadrightarrow_{\mathcal{R}} \sigma(N)$). For eg. $\mathit{map}(\xi(\mathit{cons}(1, \mathit{nil})), \mathit{cons}(2, \mathit{nil})) \rightarrow_{\mathit{map.2}} \mathit{cons}(\mathit{cons}(2, \mathit{nil}), \mathit{map}(\xi(\mathit{cons}(1, \mathit{nil})), \mathit{nil}))$ may be simulated in its ES-based implementation as:

[3] As usual, positions are paths in (terms represented as) trees [DJ90, BN98].

[4] The rules *App* and *Lam* in Fig. 1 are present because $\lambda\sigma$ implements β_{db}; in the general case we would have $f(X_1, \ldots, X_n)[s] \rightarrow_{\mathit{Func}_f} f(X_1[s], \ldots, X_n[s])$ for each function symbol f and $\xi(X_1, \ldots, X_n)[s] \rightarrow_{\mathit{Bind}_\xi} \xi(X_1[1 \cdot (s \circ \uparrow)], \ldots, X_n[1 \cdot (s \circ \uparrow)])$ for each binder symbol ξ.

$$\begin{array}{ll}
 & \mathit{map}(\xi(\mathit{cons}(1, \mathit{nil})), \mathit{cons}(2, \mathit{nil})) \\
\rightarrow_{ES(map.2)} & \mathit{cons}(\mathit{cons}(1, \mathit{nil})[2 \cdot id], \mathit{map}(\xi(\mathit{cons}(1, \mathit{nil})), \mathit{nil})) \\
\rightarrow_{Func_{cons}} & \mathit{cons}(\mathit{cons}(1[2 \cdot id], \mathit{nil}[2 \cdot id]), \mathit{map}(\xi(\mathit{cons}(1, \mathit{nil})), \mathit{nil})) \\
\rightarrow_{VarCons} & \mathit{cons}(\mathit{cons}(2, \mathit{nil}[2 \cdot id]), \mathit{map}(\xi(\mathit{cons}(1, \mathit{nil})), \mathit{nil})) \\
\rightarrow_{Func_{nil}} & \mathit{cons}(\mathit{cons}(2, \mathit{nil}), \mathit{map}(\xi(\mathit{cons}(1, \mathit{nil})), \mathit{nil}))
\end{array}$$

Remark 1. We restrict attention to a subset of ERS_{db}, namely the class of *Pattern* $ERS_{db}(PERS_{db})$. This corresponds to the usual requirement that *LHS* of rules be higher-order patterns in other rewrite formalisms [Mil91, Nip91, KOvR93]. $\mathcal{R}$ is defined to be a pattern ERS_{db} if its ES-based implementation does not contain explicit substitutions on the *LHS*. For example, η_{db} is translated to $\lambda(app(X[\uparrow], 1)) \rightarrow_{ES(\eta_{db})} X$. And *Comm* to $imply(\exists\forall X, \forall\exists X[2 \cdot 1 \cdot (\uparrow \circ \uparrow)]) \rightarrow true$ [BKR01]. Thus η_{db} and *Comm* are not pattern ERS_{db}; those of Ex. 1 are. The former exhibit the fact that when translating to an ES-based setting, higher-order matching may not always be coded as syntactic matching. The "occurs check" imposed by η_{db} or the "commutation of indices check" imposed by *Comm* are complex features of higher-order matching that require further machinery (matching *modulo* the calculus of ES) in order to be mimicked in a first-order setting. This is why we consider pattern ERS_{db}.

2.2 Standardisation and Neededness

The notion of standard derivations in rewriting[5] may be formalised using the *redex-permutation* approach [Klo80, Mel96]. We revisit this approach *very* briefly.

Given two non-overlapping redexes r, s in some term M we define the notion of a redex *residual* of r after contracting s (written r/s) with an example. In $M = \underline{(\lambda\lambda(2\,2))\,((\lambda 1)\,2)}\,3 \rightarrow_{\beta_{db}} (\lambda((\lambda 1)\,3)\,((\lambda 1)\,3))\,3 = N$, the residuals of $r = ((\lambda 1)\,2)$ in M after contracting the underlined redex s are the two copies of $(\lambda 1)\,3$ in N. Note that s has no residuals in N (i.e. for any s, $s/s = \emptyset$), and also that r/s is a finite set of redexes. The outermost redex in N is said to be *created* since it is not the residual of any redex in M. If U_M is a finite set of non-overlapping redexes ($r, s \in U_M$ implies r does not overlap s) in M and s is a redex in M, then $U_M/s = \{v | \exists u \in U_M, v \in u/s\}$. The residuals of r after a derivation $r_1; \ldots; r_n$ coinitial with it is defined as $((r/r_1)/r_2) \ldots /r_n$. A *development* of U_M is a derivation $\phi = r_1; \ldots; r_n$ s.t. $r_i \in U_M/(r_1; \ldots; r_{i-1})$ for $i \in 1..n$ and $U_M/\phi = \emptyset$ (if this last condition is not satisfied we say ϕ is a *partial development* of U_M). A well-known result called Finite Developments states that all partial developments are finite [Bar84, Oos94]. This allows one to prove the following:

Proposition 1 (Parallel Moves or Basic Tile Lemma [Bar84, HL91]). *Given two non-overlapping redexes r, s in some term M, the divergence resulting from contracting r and s may be settled by developing their corresponding residuals (Fig. 2(a)). Moreover, for any u in M, $u/(r; s/r)$=$u/(s; r/s)$.*

[5] In the sequel we restrict attention to left-linear rewrite systems: variables occur at most once in *LHS*s.

$$\begin{tikzcd}
 & N_1 \arrow[dr, two heads, "s/r"] & \\
M \arrow[ur, "r"] \arrow[dr, "s"'] & & O \\
 & N_2 \arrow[ur, two heads, "r/s"'] &
\end{tikzcd}$$

(a)

$$\begin{tikzcd}
K\,1\,a \arrow[r, "r_1"] & (\lambda 2)\,a \arrow[r, "r_2"] & 1 \arrow[d, equal] \\
K\,1\,(I a) \arrow[u, "s_1"] \arrow[r, "r_3"'] & (\lambda 2)\,(I a) \arrow[u, "s2"'] \arrow[r, "r4"'] & 1
\end{tikzcd}$$

(b)

$$\begin{tikzcd}
 & N_1 \arrow[dr, "s/r"] \arrow[dd, Leftrightarrow] & \\
M \arrow[ur, "r"] \arrow[dr, "s"'] & & O \\
 & N_2 \arrow[ur, two heads, "r/s"'] &
\end{tikzcd}$$

(c: disjoint case)

$$\begin{tikzcd}
 & N_1 \arrow[dr, "s/r"] \arrow[dd, Rightarrow] & \\
M \arrow[ur, "r"] \arrow[dr, "s"'] & & O \\
 & N_2 \arrow[ur, two heads, "r/s"'] &
\end{tikzcd}$$

(d: nested case - s nests r)

Fig. 2. Tiles

Fig. 2(b) shows two basic tiles in the λ-calculus, where $K = \lambda\lambda 2$. As depicted, these basic tiles may be "glued" in order to construct tilings between some coinitial and cofinal derivations ϕ and φ, in which case we write $\phi \equiv \varphi$ and say that ϕ and φ are Lévy-Permutation Equivalent [Lév78, Klo80, Mel96]. For example, $s_1; r_1; r_2 \equiv r_3; s_2; r_2$, however $I(\underline{I\,1}) \to_{\beta_{db}} I\,1 \not\equiv \underline{I(I\,1)} \to_{\beta_{db}} I\,1$, where $I = \lambda 1$. Moreover, by comparing the relative positions of r and s ((1) disjoint, left tile in Fig. 2(b) and (2) nested - a redex s nests r when the position of s is a prefix of the position of r - right tile in Fig. 2(b)) we can orient these tiles as indicated in Fig. 2(c,d) and define standard derivations.

Definition 1 (Standard derivation).

1. *Let r, s be cointial redexes. If they are disjoint, then $r; s/r \Diamond s; r/s$ models a reversible tile; if s nests r, then $r; s/r \rhd s; r/s$ models an irreversible one. A reversible step ($\overset{\mathtt{r}}{\Rightarrow}$) is defined as $\phi_1; r; s/r; \phi_2 \overset{\mathtt{r}}{\Rightarrow} \phi_1; s; r/s; \phi_2$ where $r; s/r \Diamond s; r/s$; an irreversible one ($\overset{\mathtt{i}}{\Rightarrow}$) as $\phi_1; r; s/r; \phi_2 \overset{\mathtt{i}}{\Rightarrow} \phi_1; s; r/s; \phi_2$ where $r; s/r \rhd s; r/s$. $\Rightarrow$ denotes the least reflexive-transitive closure of $\overset{\mathtt{r}}{\Rightarrow} \cup \overset{\mathtt{i}}{\Rightarrow}$ and $\simeq$ is the least equivalence relation containing $\overset{\mathtt{r}}{\Rightarrow}$.*
2. *Two coinitial and cofinal derivations ϕ, φ are Lévy-permutation equivalent, if $\phi \equiv \varphi$ where $\equiv$ is the least equivalence relation containing $\Rightarrow$.*
3. *A derivation ϕ is standard if it is minimal in the following sense: there is no sequence of the form $\phi = \phi_0 \overset{\mathtt{r}}{\Rightarrow} \ldots \overset{\mathtt{r}}{\Rightarrow} \phi_{k-1} \overset{\mathtt{i}}{\Rightarrow} \phi_k$, where $k \geq 1$.*

For example $r_3; r_4$ is standard, but $r_3; s_2; r_2$ is not. The standardisation theorem states that every derivation may be standardised into a unique standard derivation by oriented tiling:

Theorem 1 (Standardisation [Lév78, Bar84, HL91, Mel96]).

1. *(Existence) For any ϕ there exists a standard derivation φ s.t. $\phi \Rightarrow \varphi$.*
2. *(Unicity) If $\varphi_1 \equiv \phi$ and $\varphi_2 \equiv \phi$ and $\varphi_{1,2}$ are standard, then $\varphi_1 \simeq \varphi_2$.*

Let $\mathtt{std}(\phi)$ stand for the (unique modulo $\simeq$) standard derivation in the $\equiv$-equivalence class of ϕ. For example, $\mathtt{std}(s_1; r_1; r_2) = \mathtt{std}(r_3; s_2; r_2) = r_3; r_4$ in Fig. 2(b).

Standard derivations are used to show that needed strategies are normalising in *orthogonal systems*. A rewrite system is orthogonal if any pair of coinitial redexes r, s do not overlap. Define r in M to be a *needed redex* if it has at least one residual in any coinitial derivation ϕ, unless ϕ contracts a residual of r [Mar92]. For example, for the rewriting system $\{f(X_\epsilon, b) \to c,\ a \to b\}$ the right occurrence of a in $f(a, a)$ is needed but not the left one. A needed rewrite strategy is one that only selects needed redexes. By defining a measure $|M|$ as "the length of the unique standard derivation (modulo $\simeq$) to M's normal form" it may be shown [HL91, Mel96] that if $M \to_r N$ for some needed redex r, then $|M|>|N|$; hence needed strategies normalise in orthogonal rewrite systems. This measure is well-defined since in orthogonal systems any two coinitial derivations to (the unique) normal form may be tiled [HL91, Mel96].

In the case of non-orthogonal systems the notion of needed redex requires revision. Indeed, in $\mathcal{R} = \{a \to_r a,\ a \to_s b\}$ the derivation to normal form $\phi : a \to_s b$ leaves no residual of r. However one cannot conclude that r is not needed since although r is not reduced in ϕ a redex which overlaps with r has. Thus the notion of needed redex is extended to needed derivations as follows:

Definition 2 (Needed derivations in non-orthogonal systems [Mel00]). *$\phi : M \twoheadrightarrow N$ is needed in a non-orthogonal rewrite system if $|\mathtt{std}(\phi;\psi)|>|\mathtt{std}(\psi)|$ for any term P and any derivation $\psi : N \twoheadrightarrow P$.*

Note that now $a \to_r a$ is needed in the aforementioned example. The concept of needed redexes is extended to that of derivations since, in contrast to orthogonal systems, terms in non-orthogonal ones may not have needed redexes. For example, in $\{xor(true, X_\epsilon) \to_L true, xor(X_\epsilon, true) \to_R true, \Omega \to_\Omega true\}$ the term $xor(\Omega, \Omega)$ has no needed redexes [Klo92].

Needed derivations get us "closer" to a normal form, however the aforementioned measure for orthogonal systems is no longer well-defined: there may be two or more $\equiv$-distinct normalising derivations. Eg. $\phi_1 : a \to_r a \to_s b$, $\phi_2 : a \to_r a \to_r a \to_s b$, $\phi_3 : a \to_r a \to_r a \to_r a \to_s b$, $\ldots$, etc. are $\equiv$-distinct normalising derivations since each r-step creates a new copy of a. In [Mel00] such badly-behaved systems are discarded by requiring the following property to be fulfilled: A *normalisation cone*[6] for a term M is a family $\{\psi_i : M \twoheadrightarrow N \mid i \in I_M\}$ of normalising derivations such that every normalising derivation $\phi : M \twoheadrightarrow N$ is Lévy-permutation equivalent to a unique derivation ψ_i (i.e. $\exists! i \in I_M$ s.t. $\phi \equiv \psi_i$). A rewrite system enjoys *finite normalisation cones* (*FNC*) when there exists a *finite* normalisation cone for any term M.

Redefining the measure of a term $|M|$ to be "the length of the longest standard derivation in M's cone to M's normal form" allows one to show [Mel00] that

[6] This definition differs slightly from [Mel00, Def.4.8] since we make use of the fact that ES-based implementations of orthogonal HORS are confluent [BKR01].

if $\phi : M \twoheadrightarrow N$ is a needed derivation, then $|M|{>}|N|$; hence needed strategies normalise in non-orthogonal rewrite systems satisfying *FNC*.

3 *FNC* for ES-based Implementations of HORS

It is therefore of interest to identify conditions guaranteeing that ES-based implementations of HORS enjoy finite normalisation cones. In [Mel00] the *FNC* property is shown for $\lambda\sigma$; this result relies on two conditions: (1) if ϕ is a standard derivation in $\lambda\sigma$ ending in a σ-normal form (cf. Sect. 4), then $\sigma(\phi)$ is standard in the λ-calculus, and (2) needed strategies are normalising for the λ-calculus; $\sigma(\phi)$ is obtained by mapping each $\lambda\sigma$-rewrite step in ϕ to its "corresponding" or "projected", if any, rewrite step in λ (see Stage 2, Sect. 4). Eg. if $\phi : (1\,1)[(\lambda 1)\,2 \cdot id] \to_{Beta} (1\,1)[1[2 \cdot id] \cdot id]$ then $\sigma(\phi)$ takes the form:

$$((\lambda 1)\,2)\,((\lambda 1)\,2) \to_{\beta_{db}} 2\,((\lambda 1)\,2) \to_{\beta_{db}} 2\,2$$

In this paper we show that the *FNC* property holds for the ES-based implementation of *arbitrary* orthogonal $PERS_{db}$. This generalizes [Mel00, Thm.7.1], proved for the λ-calculus. The proof follows the same lines as in [Mel00]; it relies on our meeting requirement (1), namely

Proposition 2 (Std-Projection Proposition). *Let $\mathcal{R}$ be a left-linear $PERS_{db}$. Every standard derivation $\phi : M \twoheadrightarrow N$ in $\mathcal{R}_\sigma^{\mathsf{ES}}$ with N in σ-normal form is projected onto a standard derivation $\sigma(\phi) : \sigma(M) \twoheadrightarrow N$ in $\mathcal{R}$.*

Here is how *FNC* follows from the Std-Projection Proposition:

Proposition 3. *The ES-based implementation of any orthogonal $PERS_{db}$ $\mathcal{R}$ verifying the Std-Projection Proposition enjoys FNC: every closed $\mathcal{R}_\sigma^{\mathsf{ES}}$-term has FNC.*

Proof. Suppose, on the contrary, that there exists a closed $\mathcal{R}_\sigma^{\mathsf{ES}}$-term with an infinite number of normalising $\mathcal{R}_\sigma^{\mathsf{ES}}$-derivations, modulo Lévy-permutation equivalence. We may construct an infinite tree whose nodes are the standard derivations $M \twoheadrightarrow N$ which may be extended to normalising derivations $M \twoheadrightarrow N \twoheadrightarrow P$ where nodes are ordered by the prefix ordering, and by König's Lemma (since every $\mathcal{R}_\sigma^{\mathsf{ES}}$ term contains finite redexes) deduce the existence of an infinite derivation ϕ_∞. Moreover, since σ is strongly normalising we know that ϕ_∞ has an infinite number of $\mathcal{R}^{\mathsf{ES}}$-steps.

Let ϕ_∞ be of the form $M \to M_1 \to M_2 \to \ldots$. Every finite prefix $\phi_i : M \twoheadrightarrow M_i$ of ϕ_∞ may be extended to a standard normalising path $\chi_i : M \twoheadrightarrow M_i \twoheadrightarrow N$ (see below, left). And, by Prop. 2, each $\sigma(\chi_i) : \sigma(M) \twoheadrightarrow \sigma(M_i) \twoheadrightarrow \sigma(N) = N$ is a standard and normalising $\mathcal{R}$ derivation.

Since $\mathcal{R}$ is orthogonal all the normalising derivations $\sigma(\chi_i) : \sigma(M) \twoheadrightarrow \sigma(N) = N$ must be Lévy-permutation equivalent. Thus we have: $\sigma(\chi_1) \equiv \sigma(\chi_2) \equiv \sigma(\chi_3) \equiv \sigma(\chi_4) \equiv \ldots$. And from Thm. 1(2) and the fact that $\phi \simeq \varphi$ implies $|\phi|{=}|\varphi|$ we deduce that: $|\sigma(\chi_1)|{=}|\sigma(\chi_2)|{=}|\sigma(\chi_3)|{=}|\sigma(\chi_4)|{=}\ldots$

We reach a contradiction from the fact that there are an infinite number of $\mathcal{R}^{\mathsf{ES}}$ redexes in ϕ_∞ and that Prop. 2 projects $\mathcal{R}^{\mathsf{ES}}$ redexes to unique $\mathcal{R}$-redexes (see Stage 2, Sect. 4): For every $i > 0$ there is a $j > i$ such that $|\sigma(\chi_j)|>|\sigma(\chi_i)|$. See below, right, where the squiggly arrow $\sigma(r_i)$ is the identity if r_i is a σ redex and is an $\mathcal{R}$ redex if r_i is an $\mathcal{R}^{\mathsf{ES}}$ redex.

$$\begin{tikzcd}
M \arrow[r, "r_1"] \arrow[dr, two heads, "\mathcal{R}^{\mathsf{ES}}_\sigma"'] & M_1 \arrow[r, "r_2"] \arrow[d, two heads, "\mathcal{R}^{\mathsf{ES}}_\sigma"'] & M_2 \arrow[r, "r_3"] \arrow[dl, two heads, "\mathcal{R}^{\mathsf{ES}}_\sigma"] & \cdots \\
 & N & & \cdots
\end{tikzcd}
\qquad
\begin{tikzcd}
\sigma(M) \arrow[r, squiggly, "\sigma(r_1)"] \arrow[dr, two heads, "\mathcal{R}"'] & \sigma(M_1) \arrow[r, squiggly, "\sigma(r_2)"] \arrow[d, two heads, "\mathcal{R}"'] \arrow[d, phantom, "\equiv" near start, shift right=3ex] \arrow[d, phantom, "\equiv" near start, shift left=3ex] & \sigma(M_2) \arrow[r, squiggly, "\sigma(r_3)"] \arrow[dl, two heads, "\mathcal{R}"] & \cdots \\
 & \sigma(N) = N & & \cdots
\end{tikzcd}$$

4 The Std-Projection Proposition

We now concentrate on the proof of the Std-Projection Proposition which proceeds by contradiction and is developed in three stages. Before continuing however, we remark that it is non-trivial. In fact, in the general case in which N is not required to be a σ-normal form it fails:

$$\chi : ((\lambda(11))1)[\underline{(\lambda 1)c} \cdot id] \rightarrow_{Beta} \underline{((\lambda(11))1)}[1[c \cdot id] \cdot id] \rightarrow_{Beta} (11)[1 \cdot id][1[c \cdot id] \cdot id]$$

χ is a standard $\lambda\sigma$-derivation, however $\sigma(\chi) : (\lambda(11))\underline{((\lambda 1)c)} \rightarrow_\beta \underline{(\lambda(11))c} \rightarrow_\beta cc$ is not standard in the λ-calculus.

Let χ be any standard $\mathcal{R}^{\mathsf{ES}}_\sigma$ derivation. The idea of the proof, inspired from [Mel00], is to show that every reversible (resp. irreversible) step in the projection $\sigma(\chi)$ of χ may be mimicked by 1 or more reversible (resp. reversible steps followed by 1 or more irreversible) steps in χ, the ES-based derivation. Hence we may conclude by reasoning by contradiction. Stage 1 of the proof shows why the projection of an $\mathcal{R}^{\mathsf{ES}}$ step results in a unique $\mathcal{R}$ step if χ ends in a σ-normal form, Stage 2 uses this fact to prove that reversible steps may be mimicked as explained above and Stage 3 considers irreversible steps.

Stage 1 (Substitution Zones)

First of all, note that χ consists of two kinds of rewrite steps: $\mathcal{R}^{\mathsf{ES}}$ steps and σ steps. We argue that it is not possible for a $\mathcal{R}^{\mathsf{ES}}$ step to take place inside a substitution if χ ends in a σ-normal form. The reason is that in that case the $\mathcal{R}^{\mathsf{ES}}$-redex would occur inside some term P in $P \cdot s$ and hence under the "$\cdot$" symbol. Since χ is standard, redexes reduced below a "$\cdot$" symbol cannot create redexes above it, and since N is a pure term we arrive at a contradiction. We formalise this argument below (Lemma 2).

Definition 3. *Given an implementation $\mathcal{R}^{\mathsf{ES}}_\sigma$ with Γ the set of function and binder symbols, we define $g \in \Gamma$ of arity n as uncontributable in $\mathcal{R}^{\mathsf{ES}}_\sigma$ if*

1. *either, g does not occur on the LHS of any rule in $\mathcal{R}^{\mathsf{ES}}_\sigma$,*
2. *or, g occurs on the LHS of a rule in $\mathcal{R}^{\mathsf{ES}}_\sigma$ only under the form $g(X_1, .., X_n)$ (i.e. it occurs applied to metavariables).*

A symbol in Γ which is not uncontributable is called contributable.

Example 3. The λ-symbol is an example of an uncontributable symbol in the $\lambda\sigma$-calculus, i.e. in $(\beta_{db})_{\sigma}^{\mathsf{ES}}$. Whereas, the application symbol is contributable in $\lambda\sigma$ due to the *Beta*-rule. Also, for any $PERS_{db}$ $\mathcal{R}$ the cons-symbol "$\cdot$" is uncontributable in $\mathcal{R}_{\sigma}^{\mathsf{ES}}$ since it is only the rules of σ that govern the behaviour of "$\cdot$".

The notion of uncontributable symbol attempts to capture those symbols from which reduction below it cannot create/erase redexes above it. Note that, although similar, this concept does not coincide with that of constructor symbol in a constructor TRS [Klo92]: given a constructor TRS there may be constructor symbols which are contributable (e.g. "s" in $f(s(s(x))) \to x$) and likewise there may be uncontributable symbols that are not constructor symbols (e.g. "f" in $f(x) \to a$).

We say that a derivation $r_1; \ldots; r_n$ *preserves* a position p when none of the redexes r_i is above p.

Lemma 1. *Let $\mathcal{R}_{\sigma}^{\mathsf{ES}}$ implement $\mathcal{R}$. Suppose that a position p is strictly above a redex $P \to_r Q$. Every standard derivation $\phi = r; \psi$ preserves p when:*

1. *either, p is a g-node for g an uncontributable symbol in $\mathcal{R}_{\sigma}^{\mathsf{ES}}$,*
2. *or, p is a g-node for g a function or binder symbol in $\mathcal{R}_{\sigma}^{\mathsf{ES}}$ and ψ is a σ-derivation.*

Proof. Given the standard derivation $\phi = r; \psi$ and the position p strictly above r two cases may arise: either ϕ preserves p (in which case we are done) or otherwise ϕ may be reorganized modulo $\simeq$ into a derivation $\phi_1; u; v; \phi_2$ such that ϕ_1 preserves the position p, the position p is strictly above a redex u, and a redex v is above p. We shall see that the latter case results in a contradiction. Note that the derivation $u; v$ cannot be standard, unless u creates v. Now, in at least the following two cases creation is not possible:

1. When p is the position of an uncontributable symbol in $\mathcal{R}_{\sigma}^{\mathsf{ES}}$. This follows from the fact that contraction of a redex below an uncontributable symbol may not create a redex above it.
2. When the position p is a function or binder symbol node and u is a σ-redex then an $\mathcal{R}^{\mathsf{ES}}$-redex must have been created, in other words, the only possible pattern of creation is when u is a $Func_f$-redex for some function symbol f or a $Bind_{\xi}$-redex for some binder symbol ξ and v is an $\mathcal{R}^{\mathsf{ES}}$-redex. For example, the pair $M = g(h(c)[id]) \to_{Func_h} g(h(c[id])) \to c$ where $\mathcal{R} = \{g(h(X)) \to c\}$, $p = \epsilon$ in M, the position at which v occurs in N is ϵ and the position at which u occurs in M is 1. Note that it is not possible for u to be an $\mathcal{R}_{\sigma}^{\mathsf{ES}}$-redex and v a σ-redex since σ-redexes above function or binder symbols cannot be created from below them.

The following key lemma states that the left argument of a "$\cdot$" symbol determines an "enclave" in a standard $\mathcal{R}_{\sigma}^{\mathsf{ES}}$-derivation to σ-normal form.

Lemma 2. *Let $\mathcal{R}_\sigma^{\mathsf{ES}}$ implement $\mathcal{R}$ and let $\phi : M \twoheadrightarrow N$ be a standard $\mathcal{R}_\sigma^{\mathsf{ES}}$-derivation with N in σ-normal form. Then no $\mathcal{R}_\sigma^{\mathsf{ES}}$-redex ever appears in the left argument of a substitution $P \cdot s$.*

Proof. By contradiction. Suppose there exists an r_i contracted in $\phi = r_1; \ldots ; r_n$ inside the left argument P of a substitution $P \cdot s$. Since the "$\cdot$" symbol is uncontributable, then by Lemma 1(1) the derivation $r_i; \ldots ; r_n$ preserves p. Since N is a pure term (i.e. has no explicit substitutions) we arrive at a contradiction.

Stage 2 (Reversible Steps)

So now we know that every $\mathcal{R}^{\mathsf{ES}}$ redex contracted in χ does not occur inside a substitution and hence that it has a unique *correspondent* $\mathcal{R}$-redex in $\sigma(\chi)$. This means that if $\sigma(\chi) = R_1; \ldots ; R_o$, then there is a function $\rho : \{1, \ldots , o\} \to \{1, \ldots , n\}$ which associates to any $\mathcal{R}$-redex R_k in $\sigma(\chi)$ the unique $\mathcal{R}^{\mathsf{ES}}$-redex $r_{\rho(k)}$ in $\chi = r_1; \ldots ; r_n$ to which it corresponds.

It should be mentioned that there are a number of subtle issues regarding the notion of "correspondence" that must be considered. Due to lack of space these issues have been relegated to the manuscript [Bon03], however we comment on them briefly.

- First of all, observe that it does not coincide with that of the residual relation since $\mathcal{R}^{\mathsf{ES}}$-redexes may be lost when traversed by substitutions: For example, the *Beta*-redex in M is lost in the following σ-derivation: $M = ((\lambda P)\, Q)[id] \to_{App} (\lambda P)[id]\, Q[id] \to_{Lam} (\lambda(P[1 \cdot id \circ \uparrow]))\, Q[id]$. Therefore, an appropriate notion of correspondent for tracing $\mathcal{R}^{\mathsf{ES}}$ redexes through σ derivations must be defined.
- Secondly, since σ rewriting may duplicate terms it must be proved that the correspondents of redexes not occurring inside substitutions are indeed unique.
- Finally, the following parametricity property for σ showing the absence of "syntactical coincidences" [HL91] must be verified: if $r_{\rho(i)}$ is a redex in M_i whose correspondent is R_i in $\sigma(M_i)$ then the definition of correspondent should not depend on any particular σ-derivation taking M_i to $\sigma(M_i)$.

Let R_k and R_{k+1} be two consecutive $\mathcal{R}$-redexes in $\sigma(\chi)$. Note that the $\mathcal{R}_\sigma^{\mathsf{ES}}$-derivation $r_i; \ldots ; r_j = r_{\rho(k)+1}; \ldots ; r_{\rho(k+1)-1}$ between $r_{\rho(k)}$ and $r_{\rho(k+1)}$ contracts only σ-redexes, as depicted below:

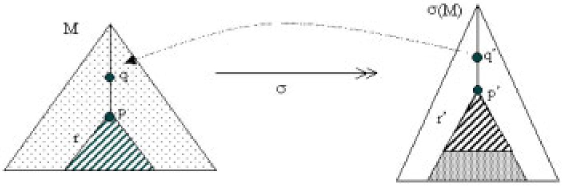
$$\begin{tikzcd}
M_{i-1} \arrow[r, "r_{\rho(k)}"'] \arrow[d, two heads, "\sigma"'] & M_i \arrow[r, two heads, "\sigma", "r_i;\ldots;r_j"'] \arrow[d, two heads, "\sigma"'] & M_j \arrow[r, "r_{\rho(k+1)}"'] \arrow[d, two heads, "\sigma"'] & M_{j+1} \arrow[d, two heads, "\sigma"'] \\
\sigma(M_{i-1}) \arrow[r, "R_k"'] & \sigma(M_i) \arrow[r, equal] & \sigma(M_j) \arrow[r, "R_{k+1}"'] & \sigma(M_{j+1})
\end{tikzcd}$$

We now show that every reversible standardisation step $\sigma(\chi) \overset{\mathtt{r}}{\Rightarrow} \omega$ in $\mathcal{R}$ may be mirrored as a non-empty series of reversible standardisation steps $\chi \overset{\mathtt{r}}{\Rightarrow} \ldots \overset{\mathtt{r}}{\Rightarrow} \phi$

in $\mathcal{R}_\sigma^{\mathsf{ES}}$, where $\sigma(\phi) = \omega$. It suffices to show that if R_k and R_{k+1} can be permuted using a reversible tile $(R_k; R_{k+1} \diamond R'_k; R'_{k+1})$, then a number of reversible steps may be applied to $r_{\rho(k)}; r_i; \ldots; r_j; r_{\rho(k+1)}$ yielding ϕ s.t $\sigma(\phi) = R'_k; R'_{k+1}$. However, first we need the notion of descendent of a position.

In the same way as the notion of residuals allow us to keep track of redexes along derivations, the notion of *descendent* allows us to keep track of positions (hence symbols) along derivations. Here is an example: if $\mathcal{R} = \{f(X_\epsilon, b) \to c\}$ and consider the rewrite step $M = f(f(a,b),c) \to f(c,c) = N$. The "$f$" at the root in M descends to the "f" at the root in N and the inner "f" in M (and the "b") descends to the "c" in N. The "a" in M, however, has no descendents in N. The inverse of the descendent relation is called the ancestor relation.

Remark 2 (Origins of positions outside substitutions). Let r be an $\mathcal{R}^{\mathsf{ES}}$-redex in M occurring at a position p not inside a substitution. As already noted, it has a unique corresponding $\mathcal{R}$-redex R in $\sigma(M)$ occurring at some position p'. Moreover, the following property on the ancestors of positions not inside substitutions holds: for every position q' in $\sigma(M)$ with $q' < p'$ (where $<$ is the prefix order), we have $q < p$ where q is a position in M of the same symbol and is the (unique) ancestor of q' as illustrated below. The fact that q' is unique follows from the observation that σ may not create new function or binder symbols.



Let us now turn to the proof of this stage. Suppose two $\mathcal{R}$-redexes R_k and R_{k+1} can be permuted using a reversible tile, that is, $R_k; R_{k+1} \diamond R'_k; R'_{k+1}$. We construct an $\mathcal{R}_\sigma^{\mathsf{ES}}$-derivation ϕ s.t. $\chi \simeq \phi$ and $\sigma(\phi) = R_1; \ldots; R'_k; R'_{k+1}; \ldots; R_p$. By Lemma 1(2), the derivation $r_i; \ldots; r_j$ preserves the position of any function or binder symbol strictly above $r_{\rho(k)}$. And, in particular, the lowest function or binder symbol g appearing above $R_k : \sigma(P) \to \sigma(Q)$ and R_{k+1} in the term $\sigma(P)$ which, by Remark 2, is strictly above $r_{\rho(k)}$ in P. Then the derivation $\psi = r_{\rho(k)}; r_i; \ldots; r_j; r_{\rho(k+1)}$ may be reorganized modulo $\simeq$ into a derivation ψ' such that $\sigma(\psi') = R'_k; R'_{k+1}$ as follows: let p be the position of this occurrence of g in P and assume $P|_p = g(N_1, \ldots, N_m)$ and suppose $r_{\rho(k)}$ occurs in N_{l_1} and the head symbol of $r_{\rho(k+1)}$ occurs in N_{l_2} for $l_1, l_2 \in 1..m$ and $l_1 \neq l_2$:

1. First contract all the redexes in $r_i; \ldots; r_j$ prefixed by $p.l_2$
2. Second contract $r_{\rho(k+1)}$,
3. Third contract the (unique) residual of $r_{\rho(k)}$,
4. Finally contract the remaining redexes of $r_i; \ldots; r_j$, i.e. those prefixed by $p.1, \ldots, p.l_2 - 1, p.l_2 + 1, \ldots, p.m$.

Stage 3 (Irreversible Steps)

Finally, we show that also irreversible standardisation steps in $\mathcal{R}$ may be mimicked in the implementation: every irreversible standardisation step $\sigma(\chi) \overset{\mathtt{i}}{\Rightarrow} \omega$ in $\mathcal{R}$ may be mirrored as a non-empty series of standardisation steps $\chi \overset{\mathtt{r}}{\Rightarrow} \ldots \overset{\mathtt{r}}{\Rightarrow} \phi' \overset{\mathtt{i}}{\Rightarrow} \ldots \overset{\mathtt{i}}{\Rightarrow} \phi$ with at least one irreversible step in $\mathcal{R}^{\mathsf{ES}}_{\sigma}$, where $\sigma(\phi) = \omega$.

Hence the proof of Prop. 2 concludes by reasoning by contradiction since every standardisation step acting on the projected HO rewrite derivation may be mimicked by projection-related standardisation steps of the same nature (reversible/irreversible) over derivations in the implementation.

Suppose two $\mathcal{R}$-redexes $R_k : \sigma(P) \to \sigma(Q)$ and R_{k+1} can be permuted using an irreversible tile $R_k; R_{k+1} \rhd R'_k; \psi$. Observe the following:

Observation: Let r be a redex in M at some position p, instance of a rule $L \to R$. The *pattern* of r is the subterm of M at position p where the arguments of r (i.e. the terms substituted for the variables of L) are replaced by holes. Similarly to Rem. 2, all the symbols in the pattern of (the σ-ancestor of) R_{k+1} strictly above R_k in $\sigma(P)$ are present in P above the occurrence of $r_{\rho(k)}$. Moreover, none of these symbols occurs embraced by a substitution operator.

This follows from two facts:

1. first, by Lemma 1, the derivation $r_i; \ldots; r_j$ preserves all these symbols (in particular the lowest one), and
2. second, $r_{\rho(k+1)}$ is an $\mathcal{R}^{\mathsf{ES}}$-redex for $\mathcal{R}$ a *PERS*$_{db}$ (cf. Rem. 1) hence its *LHS* contains no occurrences of the substitution operator $\bullet[\bullet]$.

We consider two cases for Stage 2, reasoning by contradiction in each one: (A) The redex $r_{\rho(k)}$ in P occurs under an uncontributable symbol g belonging to the pattern of R_{k+1}, (B) All symbols above $r_{\rho(k)}$ in P belonging to the pattern of R_{k+1} are contributable. Note that the second case is not possible in the lambda calculus since the β-redex pattern always has the uncontributable symbol λ.

A. By Lemma 1(1) the derivation $r_i; \ldots; r_n$ preserves every uncontributable symbol strictly above $r_{\rho(k)}$. Among these symbols is the symbol g involved in the pattern of R_{k+1}. The redex $r_{\rho(k+1)}$ is above the position of this symbol. We reach a contradiction.
B. Suppose that the two $\mathcal{R}$-redexes R_k and R_{k+1} can be permuted using an irreversible tile $R_k; R_{k+1} \rhd R'_k; \psi$; we shall arrive at a contradiction. Let p be the occurrence of the unique σ-ancestor of the head symbol g of R_{k+1} in P, and $P|_p = g(M_1, .., M_m)$. By Lemma 1(2) the derivation $r_i; ...; r_j$ preserves p. Let $l \in 1..m$ such that $r_{\rho(k)}$ occurs in M_l. We may then reorganize modulo $\simeq$ the derivation $\psi = r_{\rho(k)}; r_i; \ldots; r_j; r_{\rho(k+1)}$ obtaining ψ', as follows:
   (a) First rewrite all redexes in $r_i; \ldots; r_j$ prefixed by $p.1$,$p.2$,...,$p.l-1$,$p.l+1$,..,$p.m$ in turn (i.e. first all those prefixed by $p.1$, then those by $p.2$, and so on) and those disjoint to p.
   (b) Second, rewrite all redexes prefixed by $p.l$ but disjoint to the (unique) residual of $r_{\rho(k)}$. At this moment the redex $r_{\rho(k+1)}$ must have emerged since

- by the Observation there are no substitution symbols in M_l between p and the position of $r_{\rho(k)}$, and
- $r_{j+1} = r_{\rho(k+1)}$, it is an $\mathcal{R}^{\mathsf{ES}}$-redex and hence its *LHS* contains no occurrences of the substitution operator $\bullet[\bullet]$.

(c) Thirdly, rewrite the (unique) residual of $r_{\rho(k)}$, say $r'_{\rho(k)}$, followed by the redexes in $r_i; \ldots; r_j$ prefixed by the occurrence of $r_{\rho(k)}$, i.e. those not rewritten in Step 1 or Step 2.

(d) Finally, rewrite $r_{\rho(k+1)}$.

Note that $\psi' = \psi'_1; \psi'_2$ where ψ'_1 consists solely of σ-rewrite steps (see Step 1 and 2, above) and $\psi'_2 = r'_{\rho(k)}; r_{k_1}; \ldots; r_{k_m}; r_{\rho(k+1)}$ for some $m \leq j - i$. Applying $m + 1$ irreversible standardisation steps starting from ψ'_2 we may obtain $\psi'_3 = r'_{\rho(k+1)}; \psi_{r_{\rho(k)}}; \psi_{r_{k_1}}; \ldots; \psi_{r_{k_m}}$. Finally, setting $\psi = \psi'_1; \psi'_3$ we may conclude.

This concludes the proof of Prop. 2. As a consequence we have:

Theorem 2. *Let $\mathcal{R}$ be any orthogonal pattern ERS_{db}. All needed derivations normalise in the ES-based implementation $\mathcal{R}^{\mathsf{ES}}_{\sigma}$ of $\mathcal{R}$.*

Remark 3. Although our interest is in normalisation we would like to point out that Prop. 2 may be seen as reducing standardisation for HORS to that of first-order systems. Given a derivation $\chi : M \twoheadrightarrow N$ in an orthogonal pattern ERS_{db} $\mathcal{R}$, we recast χ in the ES-based implementation of $\mathcal{R}$, then we standardise the resulting derivation in the first-order setting [Bou85] and finally we project back to the HO-setting. The resulting $\mathcal{R}$ derivation ϕ shall not be just any standard derivation from M to N, but also Lévy-permutation equivalent to χ, in other words, $\phi \equiv \chi$. This may be verified by proving that $\varphi \Rightarrow \phi$ implies $\sigma(\varphi) \equiv \sigma(\phi)$.

5 Conclusions

We have addressed normalisation by needed reduction in the ES-based approach to the implementation of HORS. Melliès [Mel95] observed that the implementation of a higher-order rewrite system by means of calculi of explicit substitution may change its normalisation properties fundamentally; indeed a term possessing no infinite derivations in the λ-calculus may lose this property when shifting to the $\lambda\sigma$-calculus. Based on an extension of the theory of needed redexes to overlapping systems [Mel00] we have shown that all needed derivations normalise in the ES-based implementation of any orthogonal pattern HORS; the latter result has been established in the setting of the ERS_{db} formalism for higher-order rewriting. The key property that has been addressed in order to apply the aforementioned theory is to show that standard derivations in the ES-based implementation of a HORS project to standard derivations in the higher-order setting (Std-Projection Proposition). The fact that this key property is all that is required owes to a simplified proof of [Mel00, Thm.7.1] (Prop. 3).

In [BKR01] the ES-based implementation of HORS does not fix a particular calculus of explicit substitutions. Instead a macro-based presentation encompassing a wide class of calculi of ES is used. The study of the abstract properties that make the proof of the Std-Projection Proposition go through would allow the results presented here to be made independent of σ, the calculus of ES which we have dealt with in this paper.

Further in this line, it would be interesting to insert the work presented here into the axiomatic setting of Axiomatic Rewrite Systems as developed in [Mel96, Mel00]. This would require a formulation of calculi of ES based on abstract axiomatic properties, work which we are currently undertaking.

Acknowledgements. To Delia Kesner, Alejandro Ríos and the referees for providing valuable suggestions.

References

[ACCL91] M. Abadi, L. Cardelli, P-L. Curien, and J-J. Lévy. Explicit substitutions. *Journal of Functional Programming*, 4(1):375–416, 1991.

[Bar84] H.P. Barendregt. *The Lambda Calculus: its Syntax and Semantics.* Studies in Logic and the Foundations of Mathematics 103. North-Holland, Amsterdam, revised edition, 1984.

[BBLRD96] Z. Benaissa, D. Briaud, P. Lescanne, and J. Rouyer-Degli. $\lambda\upsilon$, a calculus of explicit substitutions which preserves strong normalisation. *Journal of Functional Programming*, 6(5):699–722, 1996.

[BKR00] E. Bonelli, D. Kesner, and A. Ríos. A de Bruijn notation for Higher-Order Rewriting. In *Proceedings of the 11th RTA*, number 1833 in LNCS. Springer-Verlag, 2000.

[BKR01] E. Bonelli, D. Kesner, and A. Ríos. From Higher-Order to First-Order Rewriting. In *Proceedings of the 12th RTA*, number 2051 in LNCS. Springer-Verlag, 2001.

[BN98] F. Baader and T. Nipkow. *Term Rewriting and All That.* Cambridge University Press, 1998.

[Bon01] E. Bonelli. *Term rewriting and explicit substitutions.* PhD thesis, Université de Paris Sud, November 2001.

[Bon03] E. Bonelli. A normalisation result for higher-order calculi with explicit substitutions, 2003. Full version of this paper. `http://www-lifia.info.unlp.edu.ar/~ eduardo/`.

[Bou85] G. Boudol. Computational semantics of term rewrite systems. In M. Nivat and J.C. Reynolds, editors, *Algebraic methods in Semantics.* Cambridge University Press, 1985.

[CHR92] P-L. Curien, T. Hardin, and A. Ríos. Strong normalization of substitutions. In *Proceedings of Mathematical Foundations of Computer Science*, number 629 in LNCS, pages 209–217. Springer-Verlag, 1992.

[dB72] N.G. de Bruijn. Lambda calculus notation with nameless dummies, a tool for automatic formula manipulation with application to the church-rosser theorem. *Indag. Mat.*, 5(35), 1972.

[DG01] R. David and B. Guillaume. A λ-calculus with explicit weakening and explicit substitutions. *Mathematical Structures in Computer Science*, 11(1), 2001.

[DJ90] N. Dershowitz and J-P. Jouannaud. Rewrite systems. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 243–309. North-Holland, 1990.

[GKK00] J. Glauert, R. Kennaway, and Z. Khasidashvili. Stable results and relative normalization. *Journal of Logic and Computation*, 10(3), 2000.

[HL91] G. Huet and J-J. Lévy. Computations in orthogonal rewriting systems. In J.L. Lassez and G.D. Plotkin, editors, *Computational Logic; Essays in honor of Alan Robinson*, pages 394–443. MIT Press, 1991.

[HMP96] Th. Hardin, L. Maranget, and P. Pagano. Functional back-ends within the lambda-sigma calculus. In *Proceedings of the International Conference on Functional Programming*, LNCS. Springer-Verlag, 1996.

[Klo80] J.W. Klop. *Combinatory Reduction Systems.* PhD thesis, CWI, Amsterdam, 1980. Mathematical Centre Tracts n.127.

[Klo92] J.W. Klop. Term rewriting systems. *Handbook of Logic in Computer Science*, 2:1–116, 1992.

[KOvR93] J.W. Klop, V. van Oostrom, and F. van Raamsdonk. Combinatory Reduction Systems: introduction and survey. *Theoretical Computer Science*, 121(1–2):279–308, 1993.

[KR95] F. Kamareddine and A. Ríos. A λ-calculus à la de Bruijn with explicit substitutions. In *Proceedings of the International Symposium on Programming Language Implementation and Logic Programming (PLILP)*, number 982 in LNCS. Springer Verlag, 1995.

[Lév78] J-J. Lévy. *Réductions correctes et optimales dans le lambda-calcul.* PhD thesis, Université Paris VII, 1978.

[Mar92] L. Maranget. *La stratégie paresseuse.* PhD thesis, Université Paris VII, 1992.

[Mel95] P-A. Melliès. Typed λ-calculi with explicit substitutions may not terminate. In *Proceedings of Typed Lambda Calculi and Applications*, number 902 in LNCS. Springer-Verlag, 1995.

[Mel96] P-A. Melliès. *Description Abstraite des Systèmes de Réécriture.* PhD thesis, Université Paris VII, 1996.

[Mel00] P-A. Melliès. Axiomatic Rewriting Theory II: The $\lambda\sigma$-calculus enjoys finite normalisation cones. *Journal of Logic and Computation*, 10(3):461–487, 2000.

[Mil91] D. Miller. A logic programming language with lambda-abstraction, function variables, and simple unification. In P. Schroeder-Heister, editor, *Proceedings of the International Workshop on Extensions of Logic Programming, FRG, 1989*, number 475 in Lecture Notes in Artificial Intelligence. Springer-Verlag, December 1991.

[Nip91] T. Nipkow. Higher-order critical pairs. In *Proceedings of the Sixth Annual IEEE Symposium on Logic in Computer Science.* IEEE Computer Society Press, July 1991.

[Oos94] V. van Oostrom. *Confluence for Abstract and Higher-order Rewriting.* PhD thesis, Vrije University, 1994.

[Oos99] V. van Oostrom. Normalization in weakly orthogonal rewriting. In *Proceedings of the 10th International Conference on Rewriting Techniques and Applications*, number 1631 in LNCS. Springer-Verlag, 1999.