Refinements of Lazy Narrowing for Left-Linear Fully-Extended Pattern Rewrite Systems

Mircea Marin[1], Taro Suzuki[2], and Tetsuo Ida[1]

[1] Institute of Information Sciences and Electronics
University of Tsukuba, Tsukuba 305-8573, Japan
{mmarin,ida}@score.is.tsukuba.ac.jp
[2] Department of Computer Software
University of Aizu, Aizu Wakamatsu 965-8580, Japan
taro@u-aizu.ac.jp

ISE-TR-01-180

Abstract. Lazy narrowing is a general E-unification procedure for equational theories presented by confluent term rewriting systems. It has been deeply studied in the first order case and various higher-order extensions have been proposed in an attempt to improve its expressive power. Such extensions suffer from huge search space in guessing the solutions of variables of functional type. For practical purposes, the need to reduce the search space of solutions is of paramount importance.
In this paper we introduce HOLN, a higher-order lazy narrowing calculus for E-unification in theories presented by pattern rewrite systems. The calculus is designed to deal with both oriented and unoriented equations, and keeps track of the variables which are to be bound to normalized solutions. We discuss the operating principle of HOLN, its main properties, and propose refinements to reduce its search space for solutions. Our refinements are defined for classes of left-linear fully-extended pattern rewrite systems which are widely used in higher-order functional logic programming.

1 Introduction

Lazy narrowing is a general E-unification procedure for equational theories that are presented by confluent term rewriting systems. It has been extensively studied in the first-order case (see, e.g., [10, 11]) and serves as computational model of many functional logic programming (FLP) languages. Motivated by functional programming, many higher-order extensions of the FLP programming style have been proposed. Naturally, these extensions impose suitable generalizations of the underlying computational model to the higher-order case. Since higher-order constructs provide support for concise and natural formulations of many real-world problems, this research field has attracted much interest in recent years (see, e.g., [4, 7, 8, 13, 15]). Of particular interest is the framework suggested in [15] for E-unification in theories presented by pattern rewrite systems. Among the main benefits of adopting this framework, we mention:

1. the expressive power of FLP is extended with lambda abstractions and variables of functional type,
2. many higher-order generalizations of the properties of first-order lazy narrowing depend on the properties (confluence, determinism, termination, etc.) of the underlying rewrite relation. Rewriting with pattern rewrite systems preserves properties of first-order term rewriting which are crucial in lifting the essential properties of first-order lazy narrowing to the higher-order case.

The main difficulty in the design of a suitable computational model for higher-order FLP is harnessing the high nondeterminism of guessing the solutions of variables of functional type without loosing completeness. There are at least two ways to overcome this difficulty: (a) we restrict to certain classes of rewrite systems, and (b) we restrict to certain classes of goals to be solved. It is important to identify restrictions which preserve the possibility to formulate large classes of problems in an easy and convenient way.

In this paper we are concerned with a calculus inspired by the calculus LN proposed by Prehofer [15]. We call this calculus *higher-order lazy narrowing calculus* (HOLN for short). HOLN differs from LN in the following respects:

1. LN is designed to solve goals consisting of oriented equations. HOLN can solve goals made of both oriented and unoriented equations.
2. LN regards all equations between λ-terms with free variable at head position as constraints. We call these equations *flex/flex equations*. Since solving flex/flex equations is highly nondeterministic, LN doesn't solve them. This decision is motivated by the fact that flex/flex equations are always solvable, and this information is often sufficient (e.g., in theorem proving). By contrast, HOLN solves certain flex/flex equations without increasing the nondeterminism of computation. As a consequence, HOLN can compute more detailed answers.
3. The inference rules of LN depend only on the syntactic structure of goals and rewrite rules. In addition, HOLN takes into account the fact that certain goal variables must be bound to normalized solutions. This additional

information allows us to reduce the high nondeterminism of guessing bindings of normalized variables. We claim that the runtime overhead of keeping track of the normalized variables in the goal pays off in comparison with the reduction of nondeterminism enabled by this information.

The structure of the paper is as follows. In Section 2 we introduce our main notions and notations. In Section 3 we formally introduce HOLN together with its main properties, i.e. soundness and completeness. In Section 4 we introduce four refinements of HOLN for left-linear fully-extended PRSs (LEPRSs for short). Finally, in Section 5 we draw some conclusions and directions of future work.

2 Preliminaries

We first describe the meta-language of simply-typed λ-calculus. The notation is roughly consistent with [2, 9, 15].

2.1 The Simply-Typed λ-Calculus

Starting with a fixed set of base types $\mathcal{B}$, the set of all types $\mathcal{T}$ is the closure of $\mathcal{B}$ under the function space constructor $\rightarrow$. The letter τ ranges over types. Function types associate to the right, i.e., we parse $\tau_1 \rightarrow \tau_2 \rightarrow \tau_3$ as $\tau_1 \rightarrow (\tau_2 \rightarrow \tau_3)$. We write $\overline{\tau_n} \rightarrow \tau$ instead of $\tau_1 \rightarrow \ldots \rightarrow \tau_n \rightarrow \tau$, if τ is a base type.

Terms are generated as usual, by λ-abstraction and application, from a set of *typed variables* $\mathcal{V} = \bigcup_{\tau \in \mathcal{T}} \mathcal{V}_\tau$ and a set of *typed function symbols* $\mathcal{F} = \bigcup_{\tau \in \mathcal{T}} \mathcal{F}_\tau$, where $\mathcal{V}_\tau \cap \mathcal{V}_{\tau'} = \mathcal{F}_\tau \cap \mathcal{F}_{\tau'} = \emptyset$ if $\tau \neq \tau'$. We assume that $\mathcal{V}_\tau$ is countable for any type τ. We denote the *application* of two terms s, t by $(s\ t)$, and the *abstraction* of a term t over a variable x by $\lambda x.t$. An occurrence of a variable x in a term t is *bound* if it occurs below a binder for x, i.e., the occurrence of x is in a subterm $\lambda x.t'$ of t. Otherwise it is *free*. Variables with free and bound occurrences in a term t will be denoted by $\mathcal{FV}(t)$ and $\mathcal{BV}(t)$ respectively.

A *type judgement* that a term t is of type τ is written as $t : \tau$. The following inference rules inductively define the set of simply-typed λ-terms:

$$\frac{a \in \mathcal{F}_\tau \cup \mathcal{V}_\tau}{a : \tau} \quad \frac{s : \tau \rightarrow \tau' \quad t : \tau}{(s\ t) : \tau'} \quad \frac{x : \tau \quad s : \tau'}{(\lambda x.s) : \tau \rightarrow \tau'}.$$

In the sequel we consider only simply-typed λ-terms. We denote by $\mathcal{T}(\mathcal{F}, \mathcal{V})$ the set of simply-typed λ-terms, and by *type*(t) the type of a simply-typed λ-term t.

The following naming conventions are used in the sequel:

sets of finite variables in $\mathcal{V}$:	V, W
variables or function symbols:	a
bound variables or function symbols:	v
simply-typed λ-terms:	l, r, s, t, u
constants in $\mathcal{F}$:	f, g
bound variables:	x, y, z
free variables:	X, Y, Z, H
non-negative integers:	i, j, k, m, n, N

To ease the notation, we also adopt the following abbreviations:

$\overline{ob_n}$	for a sequence of syntactic objects $ob_1, \ldots, ob_n$ where $n \geq 0$; the symbol $\Box$ denotes the empty sequence
$\lambda\overline{x_n}.s$	for $\lambda x_1.\ldots.\lambda x_n.s$
$a(\overline{s_n})$	for $((\cdots(a\ s_1)\cdots)\ s_n)$

For instance, $\lambda\overline{x_m}.f(\overline{s_n})$ stands for $\lambda x_1 \ldots \lambda x_m.((\cdots(f\ s_1)\cdots)\ s_n)$. The subscripts m and n will be omitted when irrelevant or understood from the context.

Let $s[t/X]$ denote the result of replacing each free occurrence of X in s by t. The *conversion rules* in λ-calculus are defined as follows:

(α-conversion) If $y \notin \mathcal{FV}(t) \cap \mathcal{BV}(t)$ and $type(y) = type(x)$ then $\lambda x.t \succ_\alpha \lambda y.(t[y/x])$,
(β-conversion) $(\lambda x.s)\ t \succ_\beta s[t/x]$,
(η-conversion) If $x \notin \mathcal{FV}(t)$ then $(\lambda x.(t\ x)) \succ_\eta t$.

If we denote by $t[l]$ a λ-term with a distinguished occurrence of a subterm l, then let $t[r]$ denote the result of replacing the single subterm l by the term r, where $type(l) = type(r)$. We define the *α-reduction* relation $\longrightarrow_\alpha$ as

$$t[l] \longrightarrow_\alpha t[r] \text{ iff } l \succ_\alpha r.$$

The *β-reduction relation* $\longrightarrow_\beta$ and *η-reduction relation* $\longrightarrow_\eta$ are defined similarly. We define $\longrightarrow_{\beta\eta}$ as $\longrightarrow_\beta \cup \longrightarrow_\eta$. For each of these reduction relations, we also define the symmetric closure $\longleftrightarrow_\phi$, the transitive closure $\longrightarrow^+_\phi$, the reflexive-transitive closure $\longrightarrow^*_\phi$, and the reflexive-symmetric-transitive closure $\longleftrightarrow^*_\phi$ in the obvious fashion ($\phi \in \{\alpha, \beta, \eta\}$). The relations $\longleftrightarrow^*_\beta$, $\longleftrightarrow^*_\eta$, and $\longleftrightarrow^*_{\beta\eta}$ are called β-, η-, and *$\beta\eta$-equivalence* respectively. Since the simply-typed λ-calculus is confluent and terminating with respect to β-reduction (respectively η-reduction) [1], every term t has a normal form which is denoted by $t{\downarrow_\beta}$ (respectively $t{\downarrow_\eta}$). The *β-normal form* (respectively *η-normal form*) of a term t is denoted by $t{\downarrow_\beta}$ (respectively $t{\downarrow_\eta}$). Let t be in β-normal form (i.e., $t = t{\downarrow_\beta}$). Then t is of the form $\lambda\overline{x_m}.a(\overline{s_n})$, where $a \in \mathcal{F} \cup \mathcal{V}$ is called the *head* of t, denoted by head(t). The *η-expanded form* of $t = \lambda\overline{x_m}.a(\overline{s_n})$ is recursively defined by

$$t{\uparrow_\eta} = \lambda\overline{x_{m+k}}.a(\overline{s_n{\uparrow_\eta}}, x_{n+1}{\uparrow_\eta}, \ldots, x_{n+k}{\uparrow_\eta})$$

where $t : \overline{\tau_{m+k}} \to \tau$ and $x_{m+1}, \ldots, x_{m+k} \notin \mathcal{FV}(\overline{s_n})$. We call $t{\downarrow_\beta}{\uparrow^\eta}$ the *long $\beta\eta$-normal form* of a term t, also written $t{\updownarrow^\eta_\beta}$. A term t is in *long $\beta\eta$-normal form* if $t = t{\updownarrow^\eta_\beta}$.

A term t in long $\beta\eta$-normal form is called *flex* if head(t) is a free variable, and *rigid* otherwise.

We will in general assume that terms are in long $\beta\eta$-normal form and that the transformation of a term into its long $\beta\eta$-normal form is an implicit operation, e.g., when applying a substitution to a term (see next). We will also identify α-equivalent terms and assume that bound variables with different binders have different names. This identification can be achieved at syntactic level if we adopt the de Bruijn representation of λ-terms [3]. Since $s \leftrightarrow_{\alpha\beta\eta} t$ iff $s\updownarrow_{\beta}^{\eta} \leftrightarrow_{\alpha} t\updownarrow_{\beta}^{\eta}$ [5], we can detect the $\alpha\beta\eta$-equivalence of two terms by comparing the de Bruijn representations of their long $\beta\eta$-normal forms.

The *size* $|t|$ of a term t in long $\beta\eta$-normal form is the number of symbols occurring in t, not counting binders. Formally, $|\lambda x.t| = |t|$ and $|a(\overline{t_n})| = 1+\Sigma_{i=1}^{n}|t_i|$.

A *position* is a sequence of natural numbers identifying a subterm in a term. The set $Pos(t)$ of positions in a term t is defined inductively as follows: $Pos(a) = \{\epsilon\}$ if $a \in \mathcal{V} \cup \mathcal{F}$; $Pos(\lambda x.t) = \{\varepsilon\} \cup \{1{\cdot}q \mid q \in Pos(t)\}$; and $Pos(a(\overline{s_n})) = \{\epsilon\} \cup \bigcup_{i=1}^{n}\{i{\cdot}q \mid q \in Pos(s_i)\}$. Here ε denotes the empty sequence. If $p \in Pos(t)$, then $t_{|p}$ denotes the subterm of s at position p, and $t[s]_p$ denotes the term obtained from t by replacing its subterm at position p by a term s of appropriate type.

Let $p \in Pos(s)$. The sequence $\mathcal{BV}(s,p)$ of λ-abstracted variables on the path to p in s is defined inductively as:

- $\mathcal{BV}(s,\varepsilon) = \square$,
- $\mathcal{BV}(a(\overline{s_n}), i{\cdot}p) = \mathcal{BV}(s_i,p)$,
- $\mathcal{BV}(\lambda x.t, 1{\cdot}p) = x, \mathcal{BV}(t,p)$.

A *substitution* is a map $\theta : \mathcal{V} \to \mathcal{T}(\mathcal{F},\mathcal{V})$ such that:

(a) $type(\theta(X)) = type(X)$ for all $X \in \mathcal{V}$,
(b) $Dom(\theta) := \{X \in \mathcal{V} \mid X \neq \theta(X)\}$, called the *domain* of θ, is finite.

We frequently identify θ with the set $\{X \mapsto \theta(X) \mid X \in Dom(\theta)\}$ of *variable bindings*. We denote the set $\bigcup_{X\in Dom(\theta)} \mathcal{FV}(\theta(X))$ of free variables *introduced* by θ by $Ran(\theta)$. We also denote the *codomain* $\{\theta(X) \mid X \in Dom(\theta)\}$ of θ by $Cod(\theta)$. The empty substitution is denoted by ϵ, and the set of all substitutions by $Subst(\mathcal{F},\mathcal{V})$. Two substitutions θ_1 and θ_2 are *equal on* V, notation $\theta_1 = \theta_2\ [V]$, if $\theta_1(X) = \theta_2(X)$ for all $X \in V$. The *restriction of a substitution* θ to V, denoted by $\theta\restriction_V$, is defined by $\theta\restriction_V(X) = \theta(X)$ if $X \in V$, and $\theta\restriction_V(X) = X$ otherwise.

The *application* of a substitution $\theta = \{X_1 \mapsto t_1, \ldots, X_n \mapsto t_n\}$ to a term t, denoted by $t\theta$, is defined as $[t_n/X_n]\ldots[t_1/X_1]t$. This notation is extended to other syntactic constructs over λ-terms (e.g., sequences of terms, equations, etc.) in the obvious way. For example, if $\overline{t_n}$ is a sequence of terms, then $\overline{t_n}\theta$ denotes the sequence of terms $t_1\theta, \ldots, t_n\theta$.

The *composition* $\theta_1\theta_2$ of two substitutions θ_1, θ_2 is defined as $\theta_1\theta_2(X) := (X\theta_1)\theta_2$. A substitution θ_1 is *more general than* a substitution θ_2 over a set of variables V, notation $\theta_1 \leq \theta_2\ [V]$, if $\theta_1\gamma = \theta_2\ [V]$ for some substitution γ. θ_1 and θ_2 are incomparable over V if neither $\theta_1 \leq \theta_2\ [V]$ nor $\theta_2 \leq \theta_1\ [V]$.

Two terms s and t are called *unifiable* is there exists a substitution θ such that $s\theta = t\theta$. Such a θ is called a *unifier* of terms s and t.

Let $V \in \mathcal{P}_{fin}(\mathcal{V})$. A *renaming* away from V is a map $\rho : \mathcal{V} \to \mathcal{T}(\mathcal{F},\mathcal{V})$ with:

- $Dom(\rho) := \{X \in \mathcal{V} \mid X \neq \rho(X)\} \in \mathcal{P}_{fin}(\mathcal{V})$,
- $\{t\downarrow_\eta \mid t \in Cod(\rho)\} \subset \mathcal{V}$, where $Cod(\rho) := \{\rho(X) \mid X \in Dom(\rho)\}$,
- $\rho(X) \neq \rho(Y)$ for all $X, Y \in Dom(\rho)$ with $X \neq Y$, and
- $Ran(\rho) \cap V = \emptyset$, where $Ran(\rho) := \bigcup_{X\in Dom(\rho)} \mathcal{FV}(\rho(X))$.

E.g., $\rho = \{X \mapsto \lambda x.H_1(x), Y \mapsto H_2\}$ is a renaming away from $V = \{X,Y\}$.

2.2 Pattern Rewrite Systems

The following subclass of simply-typed λ-terms was introduced by Miller [12] and is often called higher-order pattern in the literature.

Definition 1 (Pattern [12]). *A* higher-order pattern *(*pattern *for short) is a term t in which every subterm of the form $X(\overline{u_n})$ with $X \in \mathcal{FV}(t)$ has $\overline{u_n\downarrow_\eta}$ a sequence of distinct bound variables.*

For example, the terms $\lambda x, y, z.X(z, x)$ and $\lambda x, y, z.f(X(x, z), Y(x))$ are patterns. The terms $\lambda x, y.X(x, y, x)$ and $\lambda x, y.f(X(x, g))$ are not patterns.

Patterns have the remarkable property that unification is unitary. Moreover, if two patterns are unifiable then a most general unifier can be computed in linear time [16]. This result shows that unification with patterns behaves similar to the first-order case.

Definition 2. *A* fully extended pattern *is a pattern t such that for all $p \in Pos(t)$, if $t|_p = X(\overline{u_n})$ with $X \in \mathcal{FV}(t)$ then $\overline{u_n\downarrow_\eta}$ is a permutation of $\mathcal{BV}(t, p)$.*

For instance, the pattern $\lambda x, y, z.X(x, z, y)$ is fully-extended, but the pattern $\lambda x, y, z.f(X(x, y))$ is not.

Definition 3 (Pattern rewrite system). *A* pattern rewrite system *(*PRS *for short) is a set $\mathcal{R}$ of pairs $l \to r$ such that*

(c_1) *l and r are λ-terms of the same base type,*
(c_2) *$\mathcal{FV}(r) \subseteq \mathcal{FV}(l)$,*
(c_3) *l is a pattern of the form $f(\overline{l_n})$.*

A fully extended pattern rewrite system *(*EPRS *for short) is a pattern rewrite system $\mathcal{R}$ which satisfies the additional condition:*

(c_4) *$\forall(l \to r) \in \mathcal{R}$, l is a fully extended pattern.*

In the sequel we assume given a PRS $\mathcal{R}$. We regard $\mathcal{F}$ as the disjoint union $\mathcal{F}_d \uplus \mathcal{F}_c$, where $\mathcal{F}_d = \{f \in \mathcal{F} \mid \exists(f(\overline{l_n}) \to r) \in \mathcal{R}\}$, and $\mathcal{F}_c = \mathcal{F} \setminus \mathcal{F}_d$. The elements of $\mathcal{F}_d$ are called *defined symbols*, and the elements of $\mathcal{F}_c$ are called *(data) constructors.*

Definition 4 (Rewriting). *If $(l \to r) \in \mathcal{R}$ and $p \in Pos(s)$, we define a* rewrite step *from s to t as*

$$s \to_{p,\theta}^{l\to r} t :\Leftrightarrow s|_p = l\theta \wedge t = s[r\theta]_p.$$

We often omit some of the parameters $p, \theta, l \to r$ and may write $s \to_\mathcal{R} t$ instead. The relation $\to_\mathcal{R}$ is called the rewrite relation *induced by $\mathcal{R}$ on $\mathcal{T}(\mathcal{F}, \mathcal{V})$.*

We mention below an equivalent definition of rewriting which takes into account the free variables in $s|_p$ which were bound in s. This new definition is based on the notion of *lifter.*

Definition 5 (Lifter). *An $\overline{x_k}$-*lifter *of a term t (respectively rewrite rule $l \to r$)* away from V *is a substitution $\sigma = \{X \mapsto \rho(X)(\overline{x_k}) \mid X \in \mathcal{FV}(t)\}$ where ρ is a renaming with $Dom(\rho) = \mathcal{FV}(t)$ (respectively $Dom(\rho) = \mathcal{FV}(l)$), $Ran(\rho) \cap V = \emptyset$ and $\rho(X) : \overline{\tau_{k+m}} \to \tau$ if $x_1 : \tau_1, \ldots, x_k : \tau_k$ and $X : \tau_{k+1} \to \ldots \tau_{k+m} \to \tau$.*

For example, $\{X \mapsto Y(x)\}$ is an x-lifter of $f(X)$ away from any set $V \subseteq \mathcal{V} \backslash \{Y\}$.

An *$\overline{x}$-lifted rewrite rule* of a rewrite rule $l \to r$ away from V is an expression of the form $l\sigma \to r\sigma$ where σ is an $\overline{x}$-lifter of $l \to r$.

The following definition of rewriting can be proved to be equivalent to Definition 4.

Definition 6 (Rewriting). *If $(l \to r) \in \mathcal{R}$ and $p \in Pos(s)$, we define a* rewrite step *from $\lambda\overline{x_j}.s$ to $\lambda\overline{x_j}.t$ as*

$$\lambda\overline{x_j}.s \to_{p,\theta}^{l \to r} \lambda\overline{x_j}.t :\Leftrightarrow \lambda\overline{x_k}.(s|_p) = \lambda\overline{x_k}.l\theta \wedge \lambda\overline{x_j}.t = \lambda\overline{x_j}.s[r\theta]_p,$$

where $\overline{x_k} = \mathcal{BV}(\lambda\overline{x_j}.s, p)$ and $l \to r$ is an $\overline{x_k}$-lifted rewrite rule away from $\mathcal{FV}(s)$.

We denote by $\to_{\mathcal{R}}^*$ the reflexive-transitive closure of $\to_{\mathcal{R}}$, and by $\leftrightarrow_{\mathcal{R}}^*$ the reflexive-symmetric-transitive closure of $\to_{\mathcal{R}}$. Two terms s and t are *$\mathcal{R}$-joinable*, notation $s \downarrow_{\mathcal{R}} t$, if there exists a term u such that $s \to_{\mathcal{R}}^* u$ and $t \to_{\mathcal{R}}^* u$. $\mathcal{R}$ is *confluent* if whenever $s \to_{\mathcal{R}}^* l$ and $s \to_{\mathcal{R}}^* r$, we have $l \downarrow_{\mathcal{R}} r$. $\mathcal{R}$ is *left-linear* if there is no rewrite rule $(l \to r) \in \mathcal{R}$ with multiple occurrences of a free variable in l.

A term s is *$\mathcal{R}$-normalized* if there is no rewrite step $s \to_{\mathcal{R}} t$. A substitution θ is $\mathcal{R}$-normalized if $\theta(X)$ is $\mathcal{R}$-normalized for any $X \in Dom(\theta)$.

$\mathcal{R}$ induces an equivalence relation $=_{\mathcal{R}}$ on $\mathcal{T}(\mathcal{F}, \mathcal{V})$, which is the least equivalence relation induced by the following axioms and inference rules:

$$\frac{}{t =_{\mathcal{R}} t} \quad \frac{s =_{\mathcal{R}} t}{t =_{\mathcal{R}} s} \quad \frac{s =_{\mathcal{R}} t \quad t =_{\mathcal{R}} u}{s =_{\mathcal{R}} u} \quad \frac{s =_{\mathcal{R}} t}{\lambda x.s =_{\mathcal{R}} \lambda x.t} \quad \frac{s =_{\mathcal{R}} s' \quad t =_{\mathcal{R}} t'}{(s\ t) =_{\mathcal{R}} (s'\ t')}$$

$$\frac{(l \to r) \in \mathcal{R}}{l =_{\mathcal{R}} r} \quad \frac{s \leftrightarrow_{\beta\eta}^* t}{s =_{\mathcal{R}} t}$$

It has been shown [18] that $=_{\mathcal{R}}$ coincides with the model-theoretical semantics for higher-order equational logic. Moreover, we have the following relationship between rewriting and equational logic [9]:

$$s =_{\mathcal{R}} t \Leftrightarrow s\updownarrow_{\beta}^{\eta} \leftrightarrow_{\mathcal{R}}^* t\updownarrow_{\beta}^{\eta}. \tag{1}$$

An equation is a pair (s, t) of terms of the same type. We distinguish between *oriented equations*, written as $s \rhd t$, and *unoriented* equations, written as $s \approx t$. We denote by $Eq(\mathcal{F}, \mathcal{V})$ the set of equations over variables $\mathcal{V}$ and function symbols $\mathcal{F}$. A *flex/flex equation* is an equation between flex terms. For example, $\lambda x.X(x, x) \approx \lambda x.Y$ is a flex/flex equation, but $\lambda x.X(x, x) \approx \lambda x.f(Y)$ is not.

A substitution γ is a *pattern substitution* if $\gamma(X)$ is a pattern for any $X \in Dom(\gamma)$. γ is an *$\mathcal{R}$-solution* of $s \approx t$, notation $\gamma \in \mathcal{U}_{\mathcal{R}}(s \approx t)$, if $s\gamma \leftrightarrow_{\mathcal{R}}^* t\gamma$. Since $s\gamma$ and $t\gamma$ are assumed to be in long $\beta\eta$-normal form, relation (1) implies that $\gamma \in \mathcal{U}_{\mathcal{R}}(s \approx t)$ iff $s\gamma =_{\mathcal{R}} t\gamma$. Under the additional assumption that $\mathcal{R}$ is confluent, this condition is equivalent to $s\gamma \downarrow_{\mathcal{R}} t\gamma$, or to the existence of a rewrite derivation of the form $s\gamma \approx t\gamma \rightarrow^*_{\mathcal{R}} u \approx u$ for some $u \in \mathcal{T}(\mathcal{F}, \mathcal{V})$. Such a rewrite derivation is called a *rewrite proof* of $\gamma \in \mathcal{U}_{\mathcal{R}}(s \approx t)$.

A substitution γ is an *$\mathcal{R}$-solution* of $s \rhd t$, notation $\gamma \in \mathcal{U}_{\mathcal{R}}(s \rhd t)$, if $s\gamma \rightarrow^*_{\mathcal{R}} t\gamma$. This condition is equivalent to the existence of a rewrite derivation $s\gamma \rhd t\gamma \rightarrow^*_{\mathcal{R}} t\gamma \approx t\gamma$ which rewrites only the left-hand side. Such a rewrite derivation is called a *rewrite proof* of $\gamma \in \mathcal{U}_{\mathcal{R}}(s \rhd t)$.

Let $Proof_{\mathcal{R}}(e, \gamma)$ denote the set of rewrite proofs of $\gamma \in \mathcal{U}_{\mathcal{R}}(e)$.

In the sequel we assume $\mathcal{R}$ to be a confluent PRS. Let $E = \overline{e_N}$ be a sequence of N equations. The notions of $\mathcal{R}$-solution and rewrite proof are extended to sequences of equations in the obvious way. Formally, we say that γ is an $\mathcal{R}$-solution of E, notation $\gamma \in \mathcal{U}_{\mathcal{R}}(E)$, if $\gamma \in \mathcal{U}_{\mathcal{R}}(e_i)$ for all $i \in \{1, \ldots, N\}$. A *rewrite proof* of $\gamma \in \mathcal{U}_{\mathcal{R}}(E)$ is a map $\rho : \{1, \ldots, N\} \rightarrow \bigcup_{i=1}^{N} Proof_{\mathcal{R}}(e_i, \gamma)$ such that $\rho(i) \in Proof_{\mathcal{R}}(e_i, \gamma)$ for all $i \in \{1, \ldots, N\}$. We denote by $Proof_{\mathcal{R}}(E, \gamma)$ the set of rewrite proofs of $\gamma \in \mathcal{U}_{\mathcal{R}}(E)$.

In general we are interested in computing $\mathcal{R}$-solutions which are $\mathcal{R}$-normalized w.r.t. some set of variables. Therefore, we adopt the following notions of goal and $\mathcal{R}$-solution.

Definition 7 (Goal, $\mathcal{R}$-solution). *A* goal *is a pair $E\lfloor_W$ where E is a sequence of equations and $W \in \mathcal{P}_{fin}(\mathcal{V})$. $E\lfloor_W$ is a* flex *goal if E is a sequence of flex/flex equations.*

A substitution γ is an $\mathcal{R}$-solution of $E\lfloor_W$, *notation $\gamma \in \mathcal{U}_{\mathcal{R}}(E\lfloor_W)$, if $\gamma\upharpoonright_W$ is an $\mathcal{R}$-normalized substitution and $\gamma \in \mathcal{U}_{\mathcal{R}}(E)$.*

For a given set of function symbols $\mathcal{F}$ and set of variables $\mathcal{V}$ we denote the set of goals by *Goal*$(\mathcal{F}, \mathcal{V})$, the set of flex goals by *Goal*$_f(\mathcal{F}, \mathcal{V})$, and define the set $\mathcal{FV}(E\lfloor_W)$ of variables of a goal $E\lfloor_W$ by $\mathcal{FV}(E\lfloor_W) = \mathcal{FV}(E) \cup W$.

Definition 8. *A set A is a* complete set of $\mathcal{R}$-solutions *of a goal $E\lfloor_W$, notation $A \in CSU_{\mathcal{R}}(E\lfloor_W)$, if it satisfies the following conditions:*

soundness: $A \subseteq \mathcal{U}_{\mathcal{R}}(E\lfloor_W)$,
completeness: $\forall \gamma \in \mathcal{U}_{\mathcal{R}}(E\lfloor_W), \exists \theta \in A. \theta \leq \gamma\ [\mathcal{FV}(E\lfloor_W)]$.

A is a minimal complete set of $\mathcal{R}$-solutions *of a goal $E\lfloor_W$, notation $A \in MCSU_{\mathcal{R}}(E\lfloor_W)$, if $A \in CSU_{\mathcal{R}}(E\lfloor_W)$ and any two substitutions $\theta_1, \theta_2 \in A$ are incomparable over $\mathcal{FV}(E)$.*

In general, computing a complete set of $\mathcal{R}$-solutions is highly intractable, mainly because solving flex/flex equations is highly nondeterministic. Moreover, higher-order unification is known to be nullary [6]. This means that $MCSU_{\emptyset}(E\lfloor_W)$ may not exist. Obviously, this implies that $MCSU_{\mathcal{R}}(E\lfloor_W)$ may not exist. The good news is that there are many applications in which it is sufficient to decide the existence of an $\mathcal{R}$-solution for a goal (i.e., if $CSU_{\mathcal{R}}(E\lfloor_W)$ is empty or not), and it is known that $CSU_{\mathcal{R}}(E\lfloor_W) \neq \emptyset$ whenever E is a sequence of flex/flex equations. For such applications, it is sufficient to be able to compute a complete set of so called *partial $\mathcal{R}$-solutions* of a goal.

Definition 9. *Let* $E\lfloor_W$ *be a goal. A* complete set of partial $\mathcal{R}$-solutions *of* $E\lfloor_W$ *is a set* A *of pairs* $\langle\theta, F\lfloor_{W'}\rangle$ *which satisfy the following conditions:*

(c_1) $F\lfloor_{W'}$ *is a flex goal,*
(c_2) $\forall\langle\theta, F\lfloor_{W'}\rangle \in A, \forall\gamma \in \mathcal{U}_\mathcal{R}(F).\theta\gamma \in \mathcal{U}_\mathcal{R}(E)$,
(c_3) $\forall\gamma \in \mathcal{U}_\mathcal{R}(E\lfloor_W), \exists\langle\theta, F\lfloor_{W'}\rangle \in A, \exists\gamma' \in \mathcal{U}_\mathcal{R}(F\lfloor_{W'}) : \ \gamma = \theta\gamma' \ [\mathcal{FV}(E\lfloor_W)]$.

Intuitively, every goal $F\lfloor_{W'}$ of a partial $\mathcal{R}$-solution $\langle\theta, F\lfloor_{W'}\rangle$ represents the goal that should be solved to reach an $\mathcal{R}$-solution of $E\lfloor_W$, but we don't solve $F\lfloor_{W'}$ because of high nondeterminism. Condition (c_3) corresponds to the completeness condition of Definition 8, whereas condition (c_2) corresponds to a weakened form of the soundness condition of Definition 8.

Note that if A satisfies conditions $(c_1), (c_2), (c_3)$, then the set

$$Ext(A) = \{\theta\gamma \mid \langle\theta, F\lfloor_{W'}\rangle \in A \text{ and } \gamma \in \mathcal{U}_\mathcal{R}(F)\}$$

is a complete set of $\mathcal{R}$-solutions of E, and contains a subset $A' \in CSU_\mathcal{R}(E\lfloor_W)$. However, we may have $Ext(A) \notin CSU_\mathcal{R}(E\lfloor_W)$.

Since a complete set of partial $\mathcal{R}$-solutions may be infinite, we are interested in the design of calculi $\mathcal{C}$ which enumerate a set $Ans_\mathcal{R}^\mathcal{C}(E\lfloor_W)$ of partial $\mathcal{R}$-solutions of $E\lfloor_W$ for which conditions $(c_1), (c_2), (c_3)$ hold.

Definition 10. *Let* $\mathcal{C}$ *be a calculus which computes a set of pairs*

$$Ans_\mathcal{R}^\mathcal{C}(E\lfloor_W) \subseteq Subst(\mathcal{F}, \mathcal{V}) \times Goal_f(\mathcal{F}, \mathcal{V})$$

for any given goal $E\lfloor_W$. *We say that* $\mathcal{C}$ *is*

sound *if* $Ans_\mathcal{R}^\mathcal{C}(E\lfloor_W)$ *satisfies condition* (c_2) *of Definition 9.*
complete *if* $Ans_\mathcal{R}^\mathcal{C}(E\lfloor_W)$ *satisfies condition* (c_3) *of Definition 9.*

3 Lazy Narrowing Calculi

In this section we will present several higher-order lazy narrowing calculi designed to compute complete sets of partial $\mathcal{R}$-solutions.

Such a calculus $\mathcal{C}$ will be described by a finite set of labelled inference rules. The inference rules are binary relations on goals of the form

$$(E_1, e, E_2)\lfloor_W \Rightarrow_{\alpha,e,\theta} (E_1\theta, E, E_2\theta)\lfloor_{W'}$$

where α is the label of the inference rule, e is the selected equation, θ is the substitution computed in the inference step, $W' = \mathcal{FV}(W\theta)$, and E is a sequence of equations whose elements are called the *descendants* of e. If e' is an equation in E_1 or E_2, then e' has only one descendant in the inference step, namely the corresponding equation $e'\theta$ in $E_1\theta$ or $E_2\theta$. We often omit some of the subscripts α, e, θ of an inference step when they are irrelevant or understood from the context.

We call $\mathcal{C}$*-step* an inference step of a calculus $\mathcal{C}$. We will denote by *step*$(\mathcal{C})$ the set of inference steps of a calculus $\mathcal{C}$. A $\mathcal{C}$*-derivation* is a (possibly empty) sequence of $\mathcal{C}$-steps

$$E\lfloor_W = E_0\lfloor_{W_0} \Rightarrow_{\alpha_1,\theta_1} E_1\lfloor_{W_1} \Rightarrow_{\alpha_2,\theta_2} \cdots \Rightarrow_{\alpha_n,\theta_n} E_n\lfloor_{W_n}$$

abbreviated $E_0\lfloor_{W_0} \Rightarrow^n_\theta E_n\lfloor_{W_n}$ or simply $E_0\lfloor_{W_0} \Rightarrow^*_\theta E_n\lfloor_{W_n}$, where $\theta = \theta_1 \ldots \theta_n$. A $\mathcal{C}$*-refutation* is a $\mathcal{C}$-derivation $E\lfloor_W \Rightarrow^*_\theta F\lfloor_{W'}$ for which there is no $\mathcal{C}$-step starting with $F\lfloor_{W'}$. We define

$$Ans^{\mathcal{C}}_{\mathcal{R}}(E\lfloor_W) = \{\langle\theta, F\lfloor_{W'}\rangle \mid \exists\ \mathcal{C}\text{-refutation } E\lfloor_W \Rightarrow^*_\theta F\lfloor_{W'}\}.$$

In the sequel se adopt the following naming conventions:

equations:	$e, e', \ldots, e_1, e_2, \ldots$
sequences of equations:	$E, E', \ldots, E_1, E_2, \ldots$
sequences of flex equations:	F
$\mathcal{C}$-steps:	$\pi, \pi', \ldots, \pi_1, \pi_2, \ldots$
$\mathcal{C}$-derivations:	$\Pi, \Pi', \ldots, \Pi_1, \Pi_2, \ldots$

In the sequel we will introduce several higher-order lazy narrowing calculi and analyze their main properties. To simplify their presentation, we adopt the following conventions:

- $s \approx^{-1} t$ stands for $t \approx s$, and $s \rhd^{-1} t$ stands for $t \rhd s$,
- for any binary symbol $\bowtie$, we abbreviate by $\overline{u_n \bowtie v_n}$ a sequence of expressions $u_1 \bowtie v_1, \ldots, u_n \bowtie v_n$. For example, $\overline{X_n \mapsto t_n}$ denotes the sequence of variable bindings $X_1 \mapsto t_1, \ldots, X_n \mapsto t_n$, whereas $\overline{s_n \rhd t_n}$ denotes the sequence of equations $s_1 \rhd t_1, \ldots, s_n \rhd t_n$.
- whenever convenient, we relax the convention of writing terms in long $\beta\eta$-normal form, but keep the convention that all written terms are β-normal forms,
- $H, H_1, H_2, \ldots$ denote distinct fresh variables; also, the sequences $\overline{y_m}$, $\overline{y_n}$, $\overline{y'_n}$ and $\overline{z_p}$ are assumed to consist of distinct bound variables.

3.1 The Calculus HOLN

HOLN consists of three groups of inference rules: *preunification rules, narrowing rules,* and *rules for removal of flex/flex equations.*

Preunification rules

[i] **Imitation.**
If $\simeq \in \{\approx, \approx^{-1}, \rhd, \rhd^{-1}\}$ then

$$(E_1, \lambda\overline{x}.X(\overline{s_m}) \simeq \lambda\overline{x}.g(\overline{t_n}), E_2)\lfloor_W \Rightarrow_{[i],\theta} (E_1, \overline{\lambda\overline{x}.H_n(\overline{s_m}) \simeq \lambda\overline{x}.t_n}, E_2)\theta\lfloor_{W'}$$

where $\theta = \{X \mapsto \lambda\overline{y_m}.g(\overline{H_n(\overline{y_m})})\}$.

[p] **Projection.**
If $\lambda\overline{x}.t$ is rigid and $\simeq \in \{\approx, \approx^{-1}, \rhd, \rhd^{-1}\}$ then

$$(E_1, \lambda\overline{x}.X(\overline{s_m}) \simeq \lambda\overline{x}.t, E_2)\lfloor_W \Rightarrow_{[\mathsf{p}],\theta} (E_1, \lambda\overline{x}.X(\overline{s_m}) \simeq \lambda\overline{x}.t, E_2)\theta\lfloor_{W'}$$

where $\theta = \{X \mapsto \lambda\overline{y_m}.y_i(\overline{H_n(\overline{y_m})})\}$

[d] **Decomposition.** If $\simeq \in \{\approx, \rhd\}$ then

$$(E_1, \lambda\overline{x}.v(\overline{s_n}) \simeq \lambda\overline{x}.v(\overline{t_n}), E_2)\lfloor_W \Rightarrow_{[\mathsf{d}],\epsilon} (E_1, \overline{\lambda\overline{x}.s_n \simeq \lambda\overline{x}.t_n}, E_2)\lfloor_{W'}$$

Lazy narrowing rules

[on] **Outermost narrowing at nonvariable position.**
If $\simeq \in \{\approx, \approx^{-1}, \rhd\}$ and $f(\overline{l_n}) \rightarrow r$ is a fresh[1] $\overline{x}$-lifted rewrite rule of $\mathcal{R}$ then

$$(E_1, \lambda\overline{x}.f(\overline{s_n}) \simeq \lambda\overline{x}.t, E_2)\lfloor_W \Rightarrow_{[\mathsf{on}],\epsilon} (E_1, \overline{\lambda\overline{x}.s_n \rhd \lambda\overline{x}.l_n}, \lambda\overline{x}.r \simeq \lambda\overline{x}.t, E_2)\lfloor_{W'}$$

[ov] **Outermost narrowing at variable position.**
If $\simeq \in \{\approx, \approx^{-1}, \rhd\}$, $f(\overline{l_n}) \rightarrow r$ is a fresh $\overline{x}$-lifted rewrite rule of $\mathcal{R}$, and either $\lambda\overline{x}.X(\overline{s_m})$ is not a pattern or $X \notin W$ then

$$\begin{aligned}(E_1, \lambda\overline{x}.X(\overline{s_m}) \simeq \lambda\overline{x}.t, E_2)\lfloor_W \Rightarrow_{[\mathsf{ov}],\theta} (&E_1\theta, \overline{\lambda\overline{x}.H_n(\overline{s_m\theta}) \rhd \lambda\overline{x}.l_n},\\ &\lambda\overline{x}.r \simeq \lambda\overline{x}.t\theta, E_2\theta)\lfloor_{W'}\end{aligned}$$

where $\theta = \{X \mapsto \lambda\overline{y_m}.f(\overline{H_n(\overline{y_m})})\}$.

Rules for removal of flex/flex equations

[t] **Trivial equation.**
If $\simeq \in \{\approx, \rhd\}$ then

$$(E_1, t \simeq t, E_2)\lfloor_W \Rightarrow_{[\mathsf{t}],\epsilon} (E_1, E_2)\lfloor_W$$

[fs] **Flex/flex same.**
If $\simeq \in \{\approx, \rhd\}$ and $X \in W$ then

$$(E_1, \lambda\overline{x}.X(\overline{y_n}) \simeq \lambda\overline{x}.X(\overline{y'_n}), E_2)\lfloor_W \Rightarrow_{[\mathsf{fs}],\theta} (E_1, E_2)\theta\lfloor_{W'}$$

where $\theta = \{X \mapsto \lambda\overline{y_n}.H(\overline{z_p})\}$ with $\{\overline{z_p}\} = \{y_i \mid y_i = y'_i, 1 \le i \le n\}$.

[fd] **Flex/flex different.**
If $X, Y \in W$ then

$$(E_1, \lambda\overline{x}.X(\overline{y_m}) \approx \lambda\overline{x}.Y(\overline{y'_n}), E_2)\lfloor_W \Rightarrow_{[\mathsf{fd}],\theta} (E_1, E_2)\theta\lfloor_{W'}.$$

If $X \in W$ then

$$(E_1, \lambda\overline{x}.X(\overline{y_m}) \rhd \lambda\overline{x}.Y(\overline{y'_n}), E_2)\lfloor_W \Rightarrow_{[\mathsf{fd}],\theta} (E_1, E_2)\theta\lfloor_{W'}.$$

In both situations, $\theta = \{X \mapsto \lambda\overline{y_m}.H(\overline{z_p}), Y \mapsto \lambda\overline{y'_n}.H(\overline{z_p})\}$ with $\{\overline{z_p}\} = \{\overline{y_m}\} \cap \{\overline{y'_n}\}$.

[1] This means that $f(\overline{l_n}) \rightarrow r$ is an $\overline{x}$-lifted rewrite rule away from the finite set of free variables which occurred in the preceding part of the computation.

Remark 1. The calculus HOLN restricts the application of inference rule [ov] by taking into account the information that certain variables must be bound to $\mathcal{R}$-normalized values. This information is also used for solving certain flex/flex equations in a deterministic way. Keeping track of the $\mathcal{R}$-normalized variables of a goal and employing this information in the solving process is a novel feature which distinguishes HOLN from the other higher-order lazy narrowing calculi proposed so far in the literature.

3.2 Main Properties of HOLN

It is obvious that $Ans_{\mathcal{R}}^{HOLN}(E\lfloor_W) \subseteq Subst(\mathcal{F},\mathcal{V}) \times Goal_f(\mathcal{F},\mathcal{V})$ whenever $E\lfloor_W \in Goal(\mathcal{F},\mathcal{V})$. In this section we prove that HOLN is a sound and complete calculus. The following trivial lemmata will be instrumental in our proofs.

Lemma 1. *Let $\theta, \gamma_1, \gamma_2 \in Subst(\mathcal{F},\mathcal{V})$ and $V, V' \in \mathcal{P}(\mathcal{V})$ such that $V' \subseteq \mathcal{FV}(V\theta)$. If $\gamma_1 = \gamma_2\ [V]$ then $\theta\gamma_1 = \theta\gamma_2\ [V']$.*

Lemma 2. *Let $\theta, \gamma \in Subst(\mathcal{F},\mathcal{V})$ and $V, V' \in \mathcal{V}$ such that $V' \subseteq \mathcal{FV}(V\theta)$. If θ is a pattern substitution and $\theta\gamma\restriction_V$ is $\mathcal{R}$-normalized, then $\gamma\restriction_{V'}$ is $\mathcal{R}$-normalized.*

First, we prove that HOLN is sound. The following theorem is instrumental in our proof of soundness.

Theorem 1. *Let $\pi : E\lfloor_W \Rightarrow_{\alpha,e,\theta} E'\lfloor_{W'}$ be an arbitrary HOLN-step. If $\gamma' \in \mathcal{U}_{\mathcal{R}}(E')$ then $\theta\gamma' \in \mathcal{U}_{\mathcal{R}}(E)$.*

Proof. The proof is by case distinction on the label of the inference step.

– If $\alpha = [\mathrm{i}]$ then π is of the form

$$(E_1, \underbrace{\lambda\overline{x}.X(\overline{s_m}) \simeq \lambda\overline{x}.g(\overline{t_n})}_{e}, E_2)\lfloor_W \Rightarrow_{[\mathrm{i}],e,\theta} (E_1, \overline{\lambda\overline{x}.H_n(\overline{s_m}) \simeq \lambda\overline{x}.t_n}, E_2)\theta\lfloor_{W'}$$

where $\theta = \{X \mapsto \lambda\overline{y_m}.g(\overline{H_n(\overline{y_m})})\}$. Obviously, $\theta\gamma' \in \mathcal{U}_{\mathcal{R}}(E_1, E_2)$, and we only have to check that $\theta\gamma' \in \mathcal{U}_{\mathcal{R}}(e)$. From $\gamma' \in \mathcal{U}_{\mathcal{R}}(E')$ we learn that $\gamma' \in \mathcal{U}_{\mathcal{R}}(\overline{\lambda\overline{x}.H_n(\overline{s_m\theta}) \simeq \lambda\overline{x}.t_n})$.
This implies that $\gamma' \in \mathcal{U}_{\mathcal{R}}(\lambda\overline{x}.g(\overline{H_n(\overline{s_m\theta})}) \simeq \lambda\overline{x}.g(\overline{t_n})) = \mathcal{U}_{\mathcal{R}}(e\theta)$, and thus $\theta\gamma' \in \mathcal{U}_{\mathcal{R}}(e)$.

– If $\alpha = [\mathrm{on}]$, there is a fresh $\overline{x}$-lifted rewrite rule $f(\overline{l_n}) \to r$ of $\mathcal{R}$ such that π is of the form

$$(E_1, \underbrace{\lambda\overline{x}.f(\overline{s_n}) \simeq \lambda\overline{x}.t}_{e}, E_2)\lfloor_W \Rightarrow_{[\mathrm{on}],e,\epsilon} (E_1, \overline{\lambda\overline{x}.s_n \rhd \lambda\overline{x}.l_n}, \lambda\overline{x}.r \simeq \lambda\overline{x}.t, E_2)\lfloor_{W'}$$

and we only have to check that $\gamma' \in \mathcal{U}_{\mathcal{R}}(e)$. From $\gamma' \in \mathcal{U}_{\mathcal{R}}(E')$ we learn that

$$\begin{array}{ll} \lambda\overline{x}.s_j\gamma' \to_{\mathcal{R}}^* \lambda\overline{x}.l_j\gamma' & \text{for } 1 \le j \le n \\ \lambda\overline{x}.r\gamma' \to_{\mathcal{R}}^* \lambda\overline{x}.t'\gamma' & \text{if } e \text{ is an oriented equation} \\ \lambda\overline{x}.r\gamma' \leftrightarrow_{\mathcal{R}}^* \lambda\overline{x}.t\gamma' & \text{if } e \text{ is an unoriented equation} \end{array}$$

These relations imply that

$$\lambda\overline{x}.f(\overline{s_n})\gamma' \to^*_{\mathcal{R}} \lambda\overline{x}.t\gamma' \text{ if } e \text{ is an oriented equation}$$
$$\lambda\overline{x}.f(\overline{s_n})\gamma' \leftrightarrow^*_{\mathcal{R}} \lambda\overline{x}.t\gamma' \text{ if } e \text{ is an unoriented equation}$$

i.e., γ' is an $\mathcal{R}$-solution of e.

– If $\alpha =$[ov] then there is a fresh $\overline{x}$-lifted rewrite rule $f(\overline{l_n}) \to r$ of $\mathcal{R}$ such that

$$\pi : (E_1, \underbrace{\lambda\overline{x}.X(\overline{s_m}) \simeq \lambda\overline{x}.t}_{e}, E_2) \downharpoonright_W \Rightarrow_{[\text{ov}],e,\theta} (E_1\theta, \overline{\lambda\overline{x}.H_n(\overline{s_m\theta}) \rhd \lambda\overline{x}.l_n}, \lambda\overline{x}.r \simeq \lambda\overline{x}.t\theta, E_2\theta) \downharpoonright_{W'}$$

where $\theta = \{X \mapsto \lambda\overline{y_m}.f(\overline{H_n(\overline{y_m})})\}$. In this case we only have to check that $\theta\gamma' \in \mathcal{U}_{\mathcal{R}}(e)$. From $\gamma' \in \mathcal{U}_{\mathcal{R}}(E')$ we learn that

$$\begin{array}{ll} \lambda\overline{x}.H_j(\overline{s_m\theta})\gamma' \to^*_{\mathcal{R}} \lambda\overline{x}.l_j\gamma' & \text{for } 1 \le j \le n \\ \lambda\overline{x}.r\gamma' \to^*_{\mathcal{R}} \lambda\overline{x}.t\theta\gamma' & \text{if } e \text{ is an oriented equation} \\ \lambda\overline{x}.r\gamma' \leftrightarrow^*_{\mathcal{R}} \lambda\overline{x}.t\theta\gamma' & \text{if } e \text{ is an unoriented equation} \end{array}$$

As a consequence, we have

$$\lambda\overline{x}.X(\overline{s_m})\theta\gamma' = \lambda\overline{x}.f(\overline{H_n(\overline{s_m\theta})})\gamma' \to^*_{\mathcal{R}} \lambda\overline{x}.f(\overline{l_n\gamma'}) \to_{\mathcal{R}} \lambda\overline{x}.r\gamma'$$

and therefore:

$$\lambda\overline{x}.X(\overline{s_m})\theta\gamma' \to^*_{\mathcal{R}} \lambda\overline{x}.t\theta\gamma' \text{ if } e \text{ is an oriented equation}$$
$$\lambda\overline{x}.X(\overline{s_m})\theta\gamma' \leftrightarrow^*_{\mathcal{R}} \lambda\overline{x}.t\theta\gamma' \text{ if } e \text{ is an unoriented equation}$$

Hence, $\theta\gamma' \in \mathcal{U}_{\mathcal{R}}(e)$.

– the cases when $\alpha \in \{$[p],[d],[t],[fs],[fd]$\}$ are straightforward. □

Corollary 1 (Soundness). *HOLN is sound.*

Proof. Let $E\downharpoonright_W$ be an arbitrary goal, $\gamma' \in \mathcal{U}_{\mathcal{R}}(F)$, and

$$E\downharpoonright_W = E_0\downharpoonright_{W_0} \Rightarrow_{\alpha_1,\theta_1} E_1\downharpoonright_{W_1} \Rightarrow_{\alpha_2,\theta_2} \cdots \Rightarrow_{\alpha_n,\theta_n} E_n\downharpoonright_{W_n} = F\downharpoonright_{W'}$$

an HOLN-refutation, abbreviated $E\downharpoonright_W \Rightarrow^n_\theta F\downharpoonright_{W'}$. We can apply n times Lemma 1 to infer that

$$\begin{array}{l} \gamma' \in \mathcal{U}_{\mathcal{R}}(F) = \mathcal{U}_{\mathcal{R}}(E_n) \Rightarrow \theta_n\gamma' \in \mathcal{U}_{\mathcal{R}}(E_{n-1}), \\ \ldots, \\ \theta_2 \ldots \theta_n\gamma' \in \mathcal{U}_{\mathcal{R}}(E_2) \Rightarrow \theta_1(\theta_2 \ldots \theta_n\gamma') \in \mathcal{U}_{\mathcal{R}}(E_0) = \mathcal{U}_{\mathcal{R}}(E). \end{array}$$

Thus, $\theta\gamma' \in \mathcal{U}_{\mathcal{R}}(E)$. □

The following definition will be used in the completeness proof of HOLN and of its further refinements.

Definition 11 (Configuration). *A* configuration *is a tuple* $\langle E\downharpoonright_W, \gamma, \rho\rangle$ *with* $\gamma \in \mathcal{U}_{\mathcal{R}}(E\downharpoonright_W)$ *and* $\rho \in \mathit{Proof}_{\mathcal{R}}(E, \gamma)$. *We denote the set of configurations by Cfg.*

Our main idea of proving completeness of a higher-order lazy narrowing calculus $\mathcal{C}$ is to identify a poset $(\mathcal{A}, \succeq)$ with $\mathcal{A} \subseteq \mathit{Cfg}$, $\succ$ a well-founded ordering on $\mathcal{A}$, and a partial function

$$\Phi_{\mathcal{C}} : \mathcal{A} \times \mathcal{P}_{fin}(\mathcal{V}) \times \mathit{Eq}(\mathcal{F}, \mathcal{V}) \to \mathit{step}(\mathcal{C}) \times \mathcal{A}$$

which satisfy the following conditions:

(a) $\forall \gamma \in \mathcal{U}_{\mathcal{R}}(E\lfloor_W), \exists \rho \in \mathit{Proof}_{\mathcal{R}}(E, \gamma).\langle E\lfloor_W, \gamma, \rho\rangle \in \mathcal{A}$,
(b) If $T = \langle E\lfloor_W, \gamma, \rho\rangle \in \mathcal{A}$, $V \in \mathcal{P}_{fin}(\mathcal{V})$ with $\mathcal{FV}(E\lfloor_W) \subseteq V$, and $e \in E$ can be selected in a $\mathcal{C}$-step, then $\Phi_{\mathcal{C}}(T, V, e) = \langle \pi, T'\rangle$ with $\pi : E\lfloor_W \Rightarrow_{e,\theta} E'\lfloor_{W'}$, $T' = \langle E'\lfloor_{W'}, \gamma', \rho'\rangle$, $T \succ T'$, and $\gamma = \theta\gamma'$ $[V]$.

Under these assumptions, proving completeness of $\mathcal{C}$ proceeds as follows. Let $E_0\lfloor_{W_0} \in \mathit{Goal}(\mathcal{F}, \mathcal{V})$ and $\gamma_0 \in \mathcal{U}_{\mathcal{R}}(E_0\lfloor_{W_0})$. By (a), there exists a triple $T_1 = \langle E_0\lfloor_{W_0}\gamma_0, \rho_0\rangle \in \mathcal{A}$. Let $V_1 = \mathcal{FV}(E_0) \cup W_0$.

If $\Phi_{\mathcal{C}}(T_1, V_1, e)$ is undefined for all $e \in E$ then the derivation $E_0\lfloor_{W_0} \Rightarrow^0_\epsilon E_0\lfloor_{W_0}$ is a $\mathcal{C}$-refutation, thus $\epsilon \in \mathit{Ans}^{\mathcal{C}}_{\mathcal{R}}(E_0\lfloor_{W_0})$. Since, $\epsilon \leq \gamma$ $[\mathcal{FV}(E_0) \cup W_0]$ for any $\gamma \in \mathcal{U}_{\mathcal{R}}(E_0\lfloor_{W_0})$, we conclude that $\mathcal{C}$ is complete.

Otherwise, let $e_1 \in E_0$ be an equation for which $\Phi_{\mathcal{C}}(T_1, V_1, e_0)$ is defined, and let $\langle \pi_1, T_2\rangle = \Phi_{\mathcal{C}}(T_1, V_1, e)$. We assume that $\pi_1 : E_0\lfloor_{W_0} \Rightarrow_{e_1,\theta_1} E_1\lfloor_{W_1}$, $T_2 = \langle E_1\lfloor_{W_1}, \gamma_1, \rho_1\rangle$, and choose $V_2 = \mathcal{FV}(V_1\theta_1)$.

We can now repeat the above construction by starting from T_2, as we did for T_1. The construction is depicted below.

$\gamma_0 \in \mathcal{U}_{\mathcal{R}}(E_0\lfloor_{W_0}) \Rightarrow \exists T_1 = \langle E_0\lfloor_{W_0}, \gamma_0, \rho_0\rangle \in \mathcal{A}$ (by (a))
Choose $V_1 = \mathcal{FV}(E_0) \cup W_0$,
$e_1 \in E_0$ for which $\Phi_{\mathcal{C}}(T_1, V_1, e_1)$ is defined.
$\Downarrow$
Let $\langle \pi_1, T_2\rangle = \Phi_{\mathcal{C}}(T_1, V_1, e_1)$
where $\pi_1 : E_0\lfloor_{W_0} \Rightarrow_{e_1,\theta_1} E_1\lfloor_{W_1}$,
$T_2 = \langle E_1\lfloor_{W_1}, \gamma_1, \rho_1\rangle$.
Choose $V_2 = \mathcal{FV}(V_1\theta_1)$
$e_2 \in E_1$ for which $\Phi_{\mathcal{C}}(T_2, V_2, e_2)$ is defined
$\Downarrow$
$\vdots$
Choose $V_N = \mathcal{FV}(V_{N-1}\theta_{N-1})$,
$e_2 \in E_1$ for which $\Phi_{\mathcal{C}}(T_2, V_2, e_2)$ is defined.
$\Downarrow$
Let $\langle \pi_N, T_{N+1}\rangle = \Phi_{\mathcal{C}}(T_N, V_N, e_N)$
where $\pi_N : E_{N-1}\lfloor_{W_{N-1}} \Rightarrow_{e_N,\theta_N} E_N\lfloor_{W_N}$
$T_{N+1} = \langle E_N\lfloor_{W_N}, \gamma_N, \rho_N\rangle$

Let Π be the $\mathcal{C}$-derivation obtained by concatenating the $\mathcal{C}$-steps $\pi_1, \ldots, \pi_N$ in this order. By property (b), we have $T_1 \succ \ldots \succ T_{N+1}$. Since $\succ$ is well-founded, we will eventually reach a triple $T_{N+1} = \langle E_N\lfloor_{W_N}, \gamma_N, \rho_N\rangle$ with $E_N = F$ consisting of flex equations which can not be selected in a $\mathcal{C}$-step starting with $F\lfloor_{W_N}$. Thus, Π is a $\mathcal{C}$-refutation.

It remains to show that $\gamma_0 = \theta_1 \dots \theta_N \gamma_N \; [\mathcal{FV}(E_0) \cup W_0]$. First, we prove by induction on k that

$$\forall k \in \{0, \dots, N-1\}. \gamma_{N-k-1} = \theta_{N-k} \dots \theta_N \gamma_N \; [V_{N-k}]. \tag{2}$$

The base case for $k = 0$ holds by condition (b) for π_N. If (2) holds for $k < N-1$ then, since $V_{N-k} = \mathcal{FV}(V_{N-k-1}\theta_{N-k-1})$, we learn by Lemma 1 that

$$\theta_{N-k-1}\gamma_{N-k-1} = \theta_{N-k-1}\theta_{N-k} \dots \theta_N \gamma_N \; [V_{N-k-1}].$$

By condition (b) for π_{N-k-1}, we know that $\gamma_{N-k-2} = \theta_{N-k-1}\gamma_{N-k-1} \; [V_{N-k-1}]$. Thus, $\gamma_{N-k-2} = \theta_{N-k-1}\theta_{N-k} \dots \theta_N \gamma_N \; [V_{N-k-1}]$ and this concludes our inductive proof of (2). In particular, for $k = N-1$, we have that $\gamma_0 = \theta_1 \dots \theta_N \gamma_N \; [V_1]$. □

Theorem 2 (Completeness). *If $\mathcal{R}$ is confluent then* HOLN*is complete.*

To prove that HOLN is complete, we must identify a poset $(\mathcal{A}, \succeq)$ with $\mathcal{A} \subseteq$ *Cfg*, and a partial function

$$\Phi_{\mathrm{HOLN}} : \mathcal{A} \times \mathcal{P}_{fin}(\mathcal{V}) \times Eq(\mathcal{F}, \mathcal{V}) \rightarrow step(\mathcal{C}) \times \mathcal{A}$$

which satisfy conditions (a) and (b) of our generic completeness proof of a higher-order lazy narrowing calculus $\mathcal{C}$. First, we introduce some useful notions and prove some auxiliary results.

Given an equation e, we define the *size* of e by $|e| := |s| + |t|$ if $e = s \approx t$ or $e = s \rhd t$. The *size* of $E = \overline{e_N}$ is defined by $|E| := \{|e_i| \mid 1 \leq i \leq N\}_m$, where $\{\}_m$ denotes the multiset constructor. The *length* of a rewrite proof ρ for $\gamma \in \mathcal{U}_{\mathcal{R}}(E\lfloor_W)$ is defined by $|\rho| := \Sigma_{i=1}^{N} |\rho(i)|$, where $\rho(i)$ denotes the number of rewrite steps of $\rho(i)$.

Let $\succeq$ be the lexicographic combination of the orderings $\succeq_A, \succeq_B, \succeq_C$, where:

- $\langle E\lfloor_W, \gamma, \rho\rangle \succeq_A \langle E'\lfloor_{W'}, \gamma', \rho'\rangle$ iff $|\rho| \geq |\rho'|$,
- $\langle E\lfloor_W, \gamma, \rho\rangle \succeq_B \langle E'\lfloor_{W'}, \gamma', \rho'\rangle$ iff $\{|X\gamma| \mid X \in Dom(\gamma)\} \geq_{mul} \{|X'\gamma'| \mid X' \in Dom(\gamma')\}$,
- $\langle E\lfloor_W, \gamma, \rho\rangle \succeq_C \langle E'\lfloor_{W'}, \gamma', \rho'\rangle$ iff $|E\gamma| \geq_{mul} |E'\gamma'|$.

Let $=_{A,B,C} := \succeq \cap \succeq^{-1}$. Then $\succ$ is obviously well-founded. Since $\mathcal{R}$ is confluent, condition (a) is obviously satisfied. It remains to show how Φ_{HOLN} can be defined in a way which satisfies condition (b).

The following six lemmata are crucial to justify the correctness of our definition of Φ_{HOLN}. If not stated otherwise, we assume that $\simeq \in \{\approx, \approx^{-1}, \rhd, \rhd^{-1}\}$.

Lemma 3. *Let $E = \overline{e_N}$, $T = \langle E\lfloor_W, \gamma, \rho\rangle \in$ Cfg with $e_k = \lambda\overline{x}.v(\overline{s_n}) \simeq \lambda\overline{x}.v(\overline{t_n}) \in E$, and $V \in \mathcal{P}_{fin}(\mathcal{V})$. Assume $\rho(k)$ has no rewrite steps at the head positions of the equational sides. We define*

$$E' = (\overline{e_{k-1}}, \overline{\lambda\overline{x}.s_n \simeq \lambda\overline{x}.t_n}, e_{k+1}, \dots, e_N) \quad \textit{and} \quad \pi : E\lfloor_W \Rightarrow_{[\mathsf{d}], e_k, \epsilon} E'\lfloor_W.$$

Then π is a valid HOLN-step and there exists $\rho' \in Proof_{\mathcal{R}}(E', \gamma')$ such that $T' = \langle E'\lfloor_W, \gamma, \rho'\rangle \in$ Cfg and $T \succ T'$. We denote the pair $\langle \pi, T'\rangle$ by $\Phi_{[\mathsf{d}]}(T, V, e_k)$.

Proof. Since $\rho(k)$ has no rewrite steps at the head positions of the equational sides, we have:

$$\rho(k) : \lambda\overline{x}.v(\overline{s_n\gamma}) \simeq \lambda\overline{x}.v(\overline{t_n\gamma}) \to_{\mathcal{R}}^{*} \lambda\overline{x}.v(\overline{u_n}) \simeq \lambda\overline{x}.v(\overline{u_n})$$

and we can rearrange the rewrite steps of $\rho(k)$ into a sequence of rewrite derivations $R_1, R_2, \ldots, R_n$ where each R_j $(1 \leq j \leq n)$ is of the form

$$\begin{aligned} R_j : \lambda\overline{x}.v(\overline{u_{j-1}}, s_j\gamma, \ldots, s_n\gamma) &\simeq \lambda\overline{x}.v(\overline{u_{j-1}}, t_j\gamma, \ldots, t_n\gamma) \\ \to_{\mathcal{R}}^{*} \lambda\overline{x}.v(\overline{u_{j-1}}, u_j, s_{j+1}\gamma, \ldots, s_n\gamma) &\simeq \lambda\overline{x}.v(\overline{u_{j-1}}, u_j, t_{j+1}\gamma, \ldots, t_n\gamma) \end{aligned}$$

by rewriting only the j-th subterms of the sides of the equation. Then $|\rho(k)| = \Sigma_{j=1}^{n}|R_j|$ and we can extract from R_j a corresponding rewrite derivation

$$R'_j : \lambda\overline{x}.s_j\gamma \simeq \lambda\overline{x}.t_j\gamma \to_{\mathcal{R}}^{*} \lambda\overline{x}.u_j \simeq \lambda\overline{x}.u_j$$

with $|R'_j| = |R_j|$. We define $\rho'(i)$ for $i \in \{1, \ldots, N+n-1\}$ by

$$\rho'(i) = \begin{cases} \rho(i) & \text{if } i < k, \\ R'_{i-k+1} & \text{if } k \leq i < k+n, \\ \rho(i-n+1) & \text{if } k+n \leq i \leq N+n-1. \end{cases}$$

It is easy to see that $\rho' \in \mathit{Proof}_{\mathcal{R}}(E', \gamma)$, $T' = \langle E'\lfloor_W, \gamma, \rho'\rangle \in \mathit{Cfg}$ and $T =_A T'$, $T =_B T'$, $T \succ_C T'$. Thus $T \succ T'$. □

Lemma 4. *Let $E = \overline{e_N}$, $T = \langle E\lfloor_W, \gamma, \rho\rangle \in \mathit{Cfg}$, $e_k = \lambda\overline{x}.X(\overline{s}) \simeq \lambda\overline{x}.X(\overline{s}) \in E$, and $V \in \mathcal{P}_{fin}(\mathcal{V})$. We define:*

$$\begin{aligned} &E' = (\overline{e_{k-1}}, e_{k+1}, \ldots, e_N), \\ &\rho' : \{1, \ldots, N-1\} \to \textstyle\bigcup_{i \in \{1,\ldots,N\}-\{k\}} \mathit{Proof}_{\mathcal{R}}(e_i, \gamma),\ \rho'(i) = \begin{cases} \rho(i) & \text{if } 1 \leq i < k, \\ \rho(i+1) & \text{if } k \leq i < N \end{cases} \\ &T' = \langle E'\lfloor_W, \gamma, \rho'\rangle,\ \pi : E\lfloor_W \Rightarrow_{[\mathrm{t}], e_k, \epsilon} E'\lfloor_W. \end{aligned}$$

Then π is a valid HOLN-step, $\rho' \in \mathit{Proof}_{\mathcal{R}}(E', \gamma)$, $T' = \langle E'\lfloor_W, \gamma, \rho'\rangle \in \mathit{Cfg}$, and $T \succ T'$. We denote the pair $\langle \pi, T'\rangle$ by $\Phi_{[\mathrm{t}]}(T, V, e_k)$.

Proof. Straightforward. □

Lemma 5. *Let $E = \overline{e_N}$, $T = \langle E\lfloor_W, \gamma, \rho\rangle \in \mathit{Cfg}$, $e_k = \lambda\overline{x}.X(\overline{s_m}) \simeq \lambda\overline{x}.t \in E$ with $\lambda\overline{x}.t$ rigid, and $V \in \mathcal{P}_{fin}(\mathcal{V})$ such that $\mathcal{FV}(E\lfloor_W) \subseteq V$.*

(i) Assume head$(X\gamma) \in \mathcal{F}$ *and $\rho(k)$ has no rewrite steps at the head positions of the equational sides. Let π be the HOLN-step $\pi : E\lfloor_W \Rightarrow_{[\mathrm{i}], e_k, \theta} E'\lfloor_{W'}$. There exists $\rho' \in \mathit{Proof}_{\mathcal{R}}(E'\lfloor_{W'})$ such that $T' = \langle E'\lfloor_{W'}, \gamma', \rho'\rangle \in \mathit{Cfg}$, $T \succ T'$ and $\gamma = \theta\gamma'$ $[V]$. We denote the pair $\langle \pi, T'\rangle$ by $\Phi_{[\mathrm{i}]}(T, V, e_k)$.*

(ii) Assume $X\gamma = \lambda\overline{y_m}.y_i(\overline{u_p})$. Then there exist

- *a HOLN-step $\pi : E\lfloor_W \Rightarrow_{[\mathrm{p}], e_k, \theta} E'\lfloor_{W'}$ and*
- *$T' = \langle E'\lfloor_{W'}, \gamma', \rho'\rangle \in \mathit{Cfg}$ such that $T \succ T'$ and $\gamma = \theta\gamma'$ $[V]$.*

We denote the pair $\langle \pi, T'\rangle$ by $\Phi_{[\mathrm{p}]}(T, V, e_k)$.

Proof. First, we prove Lemma 5.(i). Assume head$(X\gamma) = f \in \mathcal{F}$ and $\rho(k)$ has no rewrite steps at the head positions of the equational sides. In this case $\lambda\overline{x}.t$ must be of the form $\lambda\overline{x}.f(\overline{t_n})$ and we can write $\gamma = \theta\gamma'\ [\mathit{Dom}(\gamma) \setminus \{\overline{H_n}\}]$, where $\theta = \{X \mapsto \lambda\overline{y_m}.f(\overline{H_n(\overline{y_m})})\}$, $\mathit{Dom}(\gamma') = (\mathit{Dom}(\gamma) \setminus \{X\}) \cup \{\overline{H_n}\}$, and $\overline{H_n}$ is a sequence of distinct fresh variables. Therefore, $\{\overline{H_n}\} \cap V = \emptyset$, and thus $\gamma = \theta\gamma'\ [V]$. In this case, $\rho(k)$ is of the form

$$\lambda\overline{x}.f(\overline{H_n(\overline{s_m})\gamma'}) \simeq \lambda\overline{x}.f(\overline{t_n\theta\gamma'}) \rightarrow^*_{\mathcal{R}} \lambda\overline{x}.f(\overline{u_n}) \simeq \lambda\overline{x}.f(\overline{u_n})$$

with no rewrite steps at the head positions of the equational sides. Let $E'' = E\theta$. Then $\gamma' \in \mathcal{U}_{\mathcal{R}}(E'')$ and $\rho \in \mathit{Proof}_{\mathcal{R}}(E'', \gamma')$. By Lemma 2 we learn that $\gamma' \in \mathcal{U}_{\mathcal{R}}(E''\lfloor_{W'})$. Note that $T'' = \langle E''\lfloor_{W'}, \gamma', \rho\rangle \in \mathit{Cfg}$ and $T \succ T''$. We can apply Lemma 3 to construct $\rho' \in \mathit{Proof}_{\mathcal{R}}(E', \gamma')$ such that $T' = \langle E'\lfloor_{W'}, \gamma', \rho'\rangle \in \mathit{Cfg}$ and $T'' \succ T'$. We conclude $T \succ T'$ from the transitivity of $\succ$.

Next, we prove Lemma 5.(ii). Assume $X\gamma = \lambda\overline{y_m}.y_i(\overline{u_p})$. Then we can write $\gamma = \theta\gamma'\ [\mathit{Dom}(\gamma) \setminus \{\overline{H_p}\}]$, where $\theta = \{X \mapsto \lambda\overline{y_m}.y_i(\overline{H_p(\overline{y_m})})\}$, $\mathit{Dom}(\gamma') = (\mathit{Dom}(\gamma) \setminus \{X\}) \cup \{\overline{H_p}\}$, and $\overline{H_p}$ is a sequence of distinct fresh variables. Therefore, $\{\overline{H_p}\} \cap V = \emptyset$, and thus $\gamma = \theta\gamma'\ [V]$. Let

$$\pi : E\lfloor_W \Rightarrow_{[\mathrm{p}], e_k, \theta} E'\lfloor_{W'}$$

Obviously, $\gamma' \in \mathcal{U}_{\mathcal{R}}(E')$ and $\rho \in \mathit{Proof}_{\mathcal{R}}(E', \gamma')$. By Lemma 2, we learn that $\gamma' \in \mathcal{U}_{\mathcal{R}}(E'\lfloor_{W'})$. We define $T' := \langle E'\lfloor_{W'}, \gamma', \rho\rangle$. Then $T' \in \mathit{Cfg}$ and $T =_A T'$, $T \succ_B T'$. Thus, $T \succ T'$. This concludes our proof of Lemma 5.(ii). □

Lemma 6. *Let $E = \overline{e_N}$, $T = \langle E\lfloor_W, \gamma, \rho\rangle \in \mathit{Cfg}$, $e_k \in E$, and $V \in \mathcal{P}_{fin}(\mathcal{V})$ such that $\mathcal{FV}(E\lfloor_W) \subseteq V$.*

(i) If $e_k = \lambda\overline{x}.s \simeq \lambda\overline{x}.t$ with head$((\lambda\overline{x}.s)\gamma) = f \in \mathcal{F}$, *$\simeq \in \{\approx, \rhd\}$, and $\rho(k)$ has a rewrite step at the head position of the left-hand side, then there exist*
- *an $\overline{x}$-lifter $f(\overline{l_n}) \rightarrow r$ of a rewrite rule of $\mathcal{R}$ and*
- *$T' = \langle \underbrace{(\overline{e_{k-1}}, \lambda\overline{x}.s \rhd \lambda\overline{x}.f(\overline{l_n}), \lambda\overline{x}.r \simeq \lambda\overline{x}.t, e_{k+1}, \ldots, e_N)}_{E'}\lfloor_W, \gamma', \rho'\rangle \in \mathit{Cfg}$*

such that $T \succ T'$, $\gamma = \gamma'\ [V]$, and $\rho'(k)$ has no rewrite steps at the head position of the left-hand side.
In this case we define $\Phi_{[\mathrm{o}]}(T, V, e_k) := \langle \pi, T'\rangle$ where $\pi : E\lfloor_W \Rightarrow E'\lfloor_W$.

*(ii) If $e_k = \lambda\overline{x}.s \approx \lambda\overline{x}.t$,*head$((\lambda\overline{x}.t)\gamma) \in \mathcal{F}$, *and $\rho(k)$ has a rewrite step at the head position of the right-hand side, then there exist*
- *an $\overline{x}$-lifter $f(\overline{l_n}) \rightarrow r$ of a rewrite rule of $\mathcal{R}$ and*
- *$T' = \langle \underbrace{(\overline{e_{k-1}}, \lambda\overline{x}.s \rhd \lambda\overline{x}.f(\overline{l_n}), \lambda\overline{x}.s \simeq \lambda\overline{x}.r, e_{k+1}, \ldots, e_N)}_{E'}\lfloor_W, \gamma', \rho'\rangle \in \mathit{Cfg}$*

such that $T \succ T'$, $\gamma = \gamma'\ [V]$, and $\rho'(k)$ has no rewrite steps at the head position of the left-hand side.
In this case we define $\Phi_{[\mathrm{o}]}(T, V, e_k) := \langle \pi, T'\rangle$ where $\pi : E\lfloor_W \Rightarrow E'\lfloor_W$.

Proof. We prove only Lemma 6.(i) because Lemma 6.(ii) has a similar proof. Under the given assumptions, we can assume $\rho(k)$ is of the form

$$e_k\gamma \rightarrow^*_{\mathcal{R}} \lambda\overline{x}.f(\overline{s'_n}) \simeq \lambda\overline{x}.t' \xrightarrow[p,\delta]{f(\overline{l_n})\rightarrow r} \lambda\overline{x}.r\delta \simeq \lambda\overline{x}.t' \rightarrow^*_{\mathcal{R}} \lambda\overline{x}.u \simeq \lambda\overline{x}.u$$

where p is the head position of the left-hand side, $f(\overline{l_n}) \to r$ is a fresh $\overline{x}$-lifter of a rewrite rule in $\mathcal{R}$ such that $\lambda\overline{x}.f(\overline{l_n})\delta = \lambda\overline{x}.f(\overline{s'_n})$, and the depicted rewrite step is the first one at position p. Then $(\lambda\overline{x}.s)\gamma = \lambda\overline{x}.f(\overline{s_n})$, and we can decompose $\rho(k)$ into two rewrite derivations:

$$\begin{aligned} R_1 &: e\gamma = \lambda\overline{x}.f(\overline{s_n}) \simeq \lambda\overline{x}.t\gamma \to^*_{\mathcal{R}} \lambda\overline{x}.f(\overline{l_n})\delta \simeq \lambda\overline{x}.t' \\ R_2 &: \lambda\overline{x}.f(\overline{l_n})\delta \simeq \lambda\overline{x}.t' \to^{f(\overline{l_n})\to r}_{p,\delta} \lambda\overline{x}.r\delta \simeq \lambda\overline{x}.t' \to^*_{\mathcal{R}} \lambda\overline{x}.u \simeq \lambda\overline{x}.u \end{aligned}$$

Let R'_1 be the rewrite derivation obtained from R_1 by rearranging the rewrite steps such that we first rewrite the left-hand sides and next the right-hand sides. This means that R'_1 can be decomposed into 2 rewrite derivations:

$$\begin{aligned} R''_1 &: e\gamma = \lambda\overline{x}.f(\overline{s_n}) \simeq \lambda\overline{x}.t\gamma \to^*_{\mathcal{R}} \lambda\overline{x}.f(\overline{l_n})\delta \simeq \lambda\overline{x}.t\gamma \\ R''_2 &: \lambda\overline{x}.f(\overline{l_n})\delta \simeq \lambda\overline{x}.t\gamma \to^{f(\overline{l_n})\to r}_{p,\delta} \lambda\overline{x}.r\delta \simeq \lambda\overline{x}.t\gamma \to^*_{\mathcal{R}} \lambda\overline{x}.r\delta \simeq \lambda\overline{x}.t' \end{aligned}$$

such that R''_1 has no rewrite steps at the head position of the left-hand side, and $|R''_1| + |R''_2| = |R'_1| = |R_1|$.

$f(l_n) \to r$ is a fresh $\overline{x}$-lifter, and thus $\mathcal{FV}(\lambda\overline{x}.f(\overline{l_n})) \cap \mathit{Dom}(\gamma) = \emptyset$. Since $\mathit{Dom}(\delta) \subseteq \mathcal{FV}(\lambda\overline{x}.f(\overline{l_n}))$, we conclude that $\gamma' := \gamma \cup \delta$ is a well-defined substitution and $\gamma' \in \mathcal{U}_{\mathcal{R}}(E'\lfloor_W)$. Note that $E' = \overline{e'_{N+1}}$ with

- $e'_i = e_i$ and $\rho(i) \in \mathit{Proof}_{\mathcal{R}}(e'_i, \gamma')$, if $i < k$,
- $e'_k = \lambda\overline{x}.s \rhd \lambda\overline{x}.f(\overline{l_n})$ and $R''_1 \in \mathit{Proof}_{\mathcal{R}}(e'_k, \gamma')$,
- $e'_{k+1} = \lambda\overline{x}.r \simeq \lambda\overline{x}.t$ and $(R''_2, R_2) \in \mathit{Proof}_{\mathcal{R}}(e'_{k+1}, \gamma')$,
- $e'_i = e_{i-1}$ and $\rho(i-1) \in \mathit{Proof}_{\mathcal{R}}(e'_i, \gamma')$ if $i > k+1$.

Therefore, we can define $\rho' \in \mathit{Proof}_{\mathcal{R}}(E', \gamma')$ by $\rho'(i) = \begin{cases} \rho(i) & \text{if } i < k, \\ R''_1 & \text{if } i = k, \\ (R''_2, R_2) & \text{if } i = k+1, \\ \rho(i-1) & \text{if } i > k+1. \end{cases}$

Then obviously $T' \in \mathit{Cfg}$ and $T \succ_A T'$. Thus $T \succ T'$. □

Lemma 7. *Let $E = \overline{e_N}$, $T = \langle E\lfloor_W, \gamma, \rho\rangle \in \mathit{Cfg}$, $e_k = \lambda\overline{x}.X(\overline{y_n}) \simeq \lambda\overline{x}.X(\overline{y'_n}) \in E$ with $X \in W$, and $V \in \mathcal{P}_{fin}(\mathcal{V})$ such that $\mathcal{FV}(E\lfloor_W) \subseteq V$.*

We define the HOLN-step $\pi : E\lfloor_W \Rightarrow_{[\mathsf{fs}], e_k, \theta} E'\lfloor_{W'}$.

Then there exists $T' = \langle E'\lfloor_{W'}, \gamma', \rho'\rangle \in \mathit{Cfg}$ such that $T \succ T'$ and $\gamma = \theta\gamma'$ $[V]$. We denote the pair $\langle \pi, T'\rangle$ by $\Phi_{[\mathsf{fs}]}(T, V, e_k)$.

Proof. From $\gamma \in \mathcal{U}_{\mathcal{R}}(E\lfloor_W)$ and $X \in W$ results that $X\gamma$ is an $\mathcal{R}$-normal form. This implies that $\lambda\overline{x}.X(\overline{y_n})\gamma$ and $\lambda\overline{x}.X(\overline{y'_n})\gamma$ are $\mathcal{R}$-normal forms too, and thus $\rho(k)$ has no rewrite steps. Hence, γ is a unifier $\lambda\overline{x}.X(\overline{y})$ and $\lambda\overline{x}.X(\overline{y'_n})$. On the other hand, it is well-known that the substitution θ computed by π is a mgu [14] of $\lambda\overline{x}.X(\overline{y_n})$ and $\lambda\overline{x}.X(\overline{y'_n})$ over $\{X\}$. From the freshness condition on the variables introduced by θ, we conclude that $V \cap \mathit{Ran}(\theta) = \emptyset$, and thus $\theta \leq \gamma$ $[V]$. Therefore there exists γ' such that $\gamma = \theta\gamma'$ $[V]$. Together with Lemma 2, this implies that $\gamma' \in \mathcal{U}_{\mathcal{R}}(E'\lfloor_{W'})$ and that the map ρ' defined by

$$\rho'(i) = \begin{cases} \rho(i) & \text{if } 1 \leq i < k, \\ \rho(i+1) & \text{if } k \leq i < N \end{cases}$$

is a rewrite proof of $\gamma' \in \mathcal{U}_{\mathcal{R}}(E'\lfloor_{W'})$. This implies that $T' = \langle E'\lfloor_{W'}, \gamma', \rho' \rangle \in Cfg$. Since $T =_A T'$, $T \succeq_B T'$ and $T \succ_C T'$, we conclude that $T \succ T'$. □

Lemma 8. *Let $E = \overline{e_N}$, $T = \langle E\lfloor_W, \gamma, \rho \rangle \in Cfg$, $e_k \in E$, and $V \in \mathcal{P}_{fin}(\mathcal{V})$ such that $\mathcal{FV}(E\lfloor_W) \subseteq V$. Assume* $\begin{cases} e_k = \lambda\overline{x}.X(\overline{y_m}) \approx \lambda\overline{x}.Y(\overline{y'_n}) \text{ and } X, Y \in W, \\ \text{or} \\ e_k = \lambda\overline{x}.X(\overline{y_m}) \rhd \lambda\overline{x}.Y(\overline{y'_n}) \text{ and } X \in W. \end{cases}$
and let π be the HOLN-step $\pi : E\lfloor_W \Rightarrow_{[\mathrm{fd}],e_k,\theta} E'\lfloor_{W'}$.

Then there exists $T' = \langle E'\lfloor_{W'}, \gamma', \rho' \rangle \in Cfg$ such that $T \succ T'$ and $\gamma = \theta\gamma'$ $[V]$. We denote the pair $\langle \pi, T' \rangle$ by $\Phi_{[\mathrm{fd}]}(T, V, e_k)$.

Proof. From $\gamma \in \mathcal{U}_{\mathcal{R}}(E\lfloor_W)$ and $X \in W$ results that $X\gamma$ is an $\mathcal{R}$-normal form. This implies that $\lambda\overline{x}.X(\overline{y_m})\gamma$ is an $\mathcal{R}$-normal form. Also, if e_k is an unoriented equation, we learn from $Y \in W$ and $\gamma \in \mathcal{U}_{\mathcal{R}}(E\lfloor_W)$ that $Y\gamma$ is an $\mathcal{R}$-normal form, and thus $\lambda\overline{x}.Y(\overline{y'_n})\gamma$ is an $\mathcal{R}$-normal form too.

These observations imply that $\rho(k)$ has no rewrite steps, redardless whether e_k is an oriented equation or not. This implies that γ is a unifier of $\lambda\overline{x}.X(\overline{y_m})$ and $\lambda\overline{x}.X(\overline{y'_n})$. On the other hand, it is well-known that the substitution θ computed by π is a mgu [14] of $\lambda\overline{x}.X(\overline{y_m})$ and $\lambda\overline{x}.Y(\overline{y'_n})$ over $\{X, Y\}$. From the freshness condition on the variables introduced by θ, we conclude that $V \cap Ran(\theta) = \emptyset$, and thus $\theta \leq \gamma$ $[V]$. Therefore there exists γ' such that $\gamma = \theta\gamma'$ $[V]$. Together with Lemma 2, this implies that $\gamma' \in \mathcal{U}_{\mathcal{R}}(E'\lfloor_{W'})$ and the map ρ' defined by

$$\rho'(i) = \begin{cases} \rho(i) & \text{if } 1 \leq i < k, \\ \rho(i+1) & \text{if } k \leq i < N \end{cases}$$

is a rewrite proof of $\gamma' \in \mathcal{U}_{\mathcal{R}}(E'\lfloor_{W'})$. Thus $T' = \langle E'\lfloor_{W'}, \gamma', \rho' \rangle \in Cfg$. Since $T =_A T'$ and $T \succ_B T'$, we conclude that $T \succ T'$. □

We are ready now to define Φ_{HOLN}. Let $E = \overline{e_N}$, $E\lfloor_W \in Goal(\mathcal{F}, \mathcal{V})$, $V \in \mathcal{P}_{fin}(\mathcal{V})$ with $\mathcal{FV}(E\lfloor_W) \subseteq V$, and $e_k \in E$ an equation which can be selected in an HOLN-step starting with $E\lfloor_W$.

We choose $(\mathcal{A}, \succeq) = (Cfg, \succeq)$ where $\succeq$ is the lexicographic combination of $\succeq_A, \succeq_B, \succeq_C$, and distinguish the following cases:

- $e_k = \lambda\overline{x}.X(\overline{s_n}) \simeq \lambda\overline{x}.X(\overline{s_n})$. Then we define $\Phi_{\mathrm{HOLN}}(T, V, e_k) := \Phi_{[\mathrm{t}]}(T, V, e_k)$. In this case, Φ_{HOLN} satisfies condition (b) of our generic completeness proof, because of Lemma 4.
- Otherwise, assume $e_k = \lambda\overline{x}.v(\overline{s_n}) \simeq \lambda\overline{x}.v(\overline{t_n})$ and $\rho(k)$ has no rewrite steps at the head positions of the equational sides. Then we define $\Phi_{\mathrm{HOLN}}(T, V, e_k) := \Phi_{[\mathrm{d}]}(T, V, e_k)$. In this case, Φ_{HOLN} satisfies condition (b) of our generic completeness proof, because of Lemma 3.
- Otherwise, assume $e_k = \lambda\overline{x}.X(\overline{s_m}) \simeq \lambda\overline{x}.t$ with $\mathrm{head}(X\gamma) \in \mathcal{F}$, $\lambda\overline{x}.t$ rigid and $\rho(k)$ has no rewrite steps at the head positions of the equational sides. Then we define $\Phi_{\mathrm{HOLN}}(T, V, e_k) := \Phi_{[\mathrm{i}]}(T, V, e_k)$. In this case, Φ_{HOLN} satisfies condition (b) of our generic completeness proof, because of Lemma 5.(i).

- Otherwise, assume $e_k = \lambda\overline{x}.X(\overline{s_m}) \simeq \lambda\overline{x}.t$ with $X\gamma = \lambda\overline{y_m}.y_i(\overline{u_p})$ and $\lambda\overline{x}.t$ rigid. Then we define $\Phi_{\text{HOLN}}(T, V, e_k) := \Phi_{[\text{p}]}(T, V, e_k)$. In this case, Φ_{HOLN} satisfies condition (b) of our generic proof because of Lemma 5.(ii).
- Otherwise, assume $e_k = \lambda\overline{x}.s \simeq \lambda\overline{x}.t$ with $\text{head}((\lambda\overline{x}.s)\gamma) \in \mathcal{F}$, $\simeq \in \{\approx, \rhd\}$, and $\rho(k)$ has a rewrite step at the head position of the left-hand side. Then there exists $\langle \pi_1, T'' \rangle = \Phi_{[\text{o}]}(T, V, e_k)$ where T'' is a configuration of the form

$$T'' = \langle \underbrace{(\overline{e_{k-1}}, \lambda\overline{x}.s \rhd \lambda\overline{x}.f(\overline{l_n}), \lambda\overline{x}.r \simeq \lambda\overline{x}.t, e_{k+1}, \ldots, e_N)}_{E''}\downarrow_W, \gamma'', \rho'' \rangle$$

and $\rho''(k)$ has no rewrite steps at the head position of the left-hand side. Let $E'' = \overline{e''_{N+1}}$. Then $e''_k = \lambda\overline{x}.s \rhd \lambda\overline{x}.f(\overline{l_n})$ and we can determine $\langle \pi_2, T' \rangle = \Phi_{\text{HOLN}}(T'', V, e''_k)$ by appeal to the previous cases. Note that $\gamma = \gamma''\ [V]$ and $T \succ T'$ because $T \succ T''$ by Lemma 6, and $T'' \succ T'$ by property (b) of Φ_{HOLN} defined so far. We can write

$$\pi_2 : E''\downarrow_W \Rightarrow_{\alpha, e''_k, \theta} E'\downarrow_{W'}$$

and $T' = \langle E'\downarrow_{W'}, \gamma', \rho' \rangle$. It is easy to check that $\alpha \in \{[\text{i}], [\text{d}]\}$. We have $\gamma'' = \theta\gamma'\ [V]$ by property (b) of Φ_{HOLN} defined so far. Thus, $\gamma = \theta\gamma'\ [V]$. Also, note that the relation

$$\pi : E\downarrow_W \Rightarrow_{\alpha, e_k, \theta} E'\downarrow_{W'}$$

is an [ov]-step if $\alpha = [\text{i}]$ and an [on]-step if $\alpha = [\text{d}]$. Hence, in this case we can define $\Phi_{\text{HOLN}}(T, V, e_k) := \langle \pi, T' \rangle$.
- Otherwise, assume $e_k = \lambda\overline{x}.X(\overline{y_n}) \simeq \lambda\overline{x}.X(\overline{y'_n})$ with $X \in W$. Then we define $\Phi_{\text{HOLN}}(T, V, e_k) := \Phi_{[\text{fs}]}\ (T, V, e_k)$. In this case, Φ_{HOLN} satisfies condition (b) of our generic proof, because of Lemma 7.
- Otherwise, assume $\begin{cases} e_k = \lambda\overline{x}.X(\overline{y_m}) \approx \lambda\overline{x}.Y(\overline{y'_n}) \text{ and } X, Y \in W \\ \text{or} \\ e_k = \lambda\overline{x}.X(\overline{y_m}) \rhd \lambda\overline{x}.Y(\overline{y'_n}) \text{ and } X \in W. \end{cases}$
Then we define $\Phi_{\text{HOLN}}(T, V, e_k) := \Phi_{[\text{fd}]}\ (T, V, e_k)$. In this case, Φ_{HOLN} satisfies condition (b) of our generic completeness proof, because of Lemma 8.

Since these are all the possibilities which can be satisfied by a selected equation e_k in an HOLN-step, we conclude that our definition of Φ_{HOLN} satisfies condition (b) of our generic proof. □

Remark 2. We have actually proved a stronger result: if $\mathcal{R}$ is a confluent PRS then HOLN is *strongly complete*, i.e., completeness is independent of the choice of the equation in the current goal.

4 Refinements

There are two sources of nondeterminism in computations with HOLN-derivations: the choice of the inference rule of HOLN, and the choice of the rewrite rule of $\mathcal{R}$ when narrowing steps are performed. In the sequel we will investigate the possibility to make the computation with HOLN-derivations more deterministic by reducing the choices of inference rules.

4.1 HOLN$_1$: a Refinement of HOLN for Left-Linear EPRSs

Programs restricted to left-linear (term or pattern) rewrite systems are widely accepted in functional logic programming. As we will see later, the notion of left-linear EPRS (LEPRS for short) extends the notion of left-linear term rewriting system to the higher-order case in a way which preserves many properties of their first-order counterpart which are relevant to our investigation.

In this subsection we study the behavior of HOLN when $\mathcal{R}$ is a LEPRS.

First, we confine our attention to a particular class of oriented equations produced upon outermost narrowing steps, the class of parameter-passing descendants.

Definition 12. *A* parameter-passing equation *of a goal $E'\lfloor_{W'}$ in an* HOLN-*derivation $\Pi : E\lfloor_W \Rightarrow^*_\theta E'\lfloor_{W'}$ is either*

(a) an equation $\lambda\overline{x}.s_i \rhd \lambda\overline{x}.l_i$ $(1 \le i \le n)$ if the last step of Π is of the form:

$$(E_1, \lambda\overline{x}.f(\overline{s_n}) \simeq \lambda\overline{x}.t, E_2)\lfloor_W \Rightarrow_{[\mathrm{on}],\epsilon} (E_1, \overline{\lambda\overline{x}.s_n \rhd \lambda\overline{x}.l_n}, \lambda\overline{x}.r \simeq \lambda\overline{x}.t, E_2)\lfloor_{W'}$$

(b) an equation $\lambda\overline{x}.H_i(\overline{s_m\theta}) \rhd \lambda\overline{x}.l_i$ $(1 \le i \le n)$ if the last step of Π is of the form:

$$(E_1, \lambda\overline{x}.X(\overline{s_m}) \simeq \lambda\overline{x}.t, E_2)\lfloor_W \Rightarrow_{[\mathrm{ov}],\theta} (E_1\theta, \overline{\lambda\overline{x}.H_n(\overline{s_m\theta}) \rhd \lambda\overline{x}.l_n}, \lambda\overline{x}.r \simeq \lambda\overline{x}.t\theta, E_2\theta)\lfloor_{W'}.$$

A parameter-passing descendant *of a goal $E'\lfloor_{W'}$ in an* HOLN-*derivation Π : $E\lfloor_W \Rightarrow^*_\theta E'\lfloor_{W'}$ is either a parameter-passing equation in Π or a descendant of a parameter-passing equation in Π.*

Note that parameter-passing descendants are always oriented equations. To distinguish them, we will write $s \blacktriangleright t$ instead of $s \rhd t$.

The following lemma characterizes the HOLN-derivations when $\mathcal{R}$ is a (fully-extended) left-linear PRS, and can be proved by induction on the length of the HOLN-derivation.

Lemma 9. *Let $\mathcal{R}$ be a left-linear PRS and*

$$\Pi : E\lfloor_W \Rightarrow^*_\theta (E_1, s \blacktriangleright t, E_2)\lfloor_{W'}$$

an HOLN-derivation such that $s \blacktriangleright t$ is a parameter-passing descendant of $(E_1, s \blacktriangleright t, E_2)\lfloor_{W'}$ in Π. Then

(i) t is a linear pattern,
(ii) $(\mathcal{FV}(E_1, s) \cup \mathcal{FV}(E\theta) \cup W') \cap \mathcal{FV}(t) = \emptyset$,
(iii) if $\mathcal{R}$ is an LEPRS then t is a fully-extended pattern.

An important theoretical result which is relevant to our investigation is the validity of the standardization theorem for confluent LEPRS [17]. In the sequel we give a brief account to this theoretical result.

Definition 13. *A position p is a* pattern position *of a term t if $p \in Pos(t)$ and* $\mathrm{head}(t|_q) \notin \mathcal{FV}(t)$ *for all $q \leq p$. We denote by $Pat(t)$ the set of pattern positions of a term t.*

For example, if $t = f(a, X(\lambda x.x))$ then ϵ and 1 and 2 are pattern positions of t, but 2 and $2{\cdot}1$ are not.

Definition 14. *Let $E = \overline{e_N}$ and $\gamma \in \mathcal{U}_{\mathcal{R}}(E\lfloor_W)$. A rewrite proof $\rho \in Proof_{\mathcal{R}}(E, \gamma)$ is* outside-in *if for any $k \in \{1, \ldots, N\}$ we have:*

(i) $\rho(k)$ is an outside-in *rewrite derivation, that is, if $\rho(k)$ is of the form*

$$e_k\gamma \rightarrow_{p_1}^{l_1 \rightarrow r_1} \ldots \rightarrow_{p_n}^{l_n \rightarrow r_n} u \simeq u$$

then the following condition is satisfied for all $1 \leq i \leq n-1$: if there exists j with $i < j$ such that $p_i = p_j + q$ then $q \in Pat(l_j)$ for the least such j,

(ii) If $e_k = s \blacktriangleright t \in E$ and $\rho(k)$ has a rewrite step at position $1{\cdot}p$ such that no later rewrite steps take place above position $1{\cdot}p$ then $p \in Pat(t)$.

A configuration $\langle E\lfloor_W, \gamma, \rho\rangle$ is outside-in *if ρ is an outside-in rewrite proof. We denote by Cfg^{oi} the set of outside-in configurations.*

Lemma 10. *Let $\mathcal{R}$ be a confluent LEPRS, $\overline{e_N}\lfloor_W$ a goal without parameter-passing descendants, and $\gamma \in \mathcal{U}_{\mathcal{R}}(\overline{e_N}\lfloor_W)$. Then there exists an outside-in rewrite proof ρ of $\gamma \in \mathcal{U}_{\mathcal{R}}(\overline{e_N}\lfloor_W)$.*

Proof. By the standardization theorem for LEPRSs [17], there exists a rewrite proof ρ of $\gamma \in \mathcal{U}_{\mathcal{R}}(\overline{e_N}\lfloor_W)$ such that for any $k \in \{1, \ldots, N\}$, $\rho(k)$ is an outside-in rewrite proof of $\gamma \in \mathcal{U}_{\mathcal{R}}(e_k\lfloor_W)$. Since $E\lfloor_W$ has no parameter-passing descendants, we conclude that $T := \langle \overline{e_N}\lfloor_W, \gamma, \rho\rangle$ is an outside-in configuration. □

We are ready now to state our main theorem.

Theorem 3. *Let $E = \overline{e_N}$, $T = \langle E\lfloor_W, \gamma, \rho\rangle \in Cfg^{oi}$, $V \in \mathcal{P}_{fin}(\mathcal{V})$, and $e_k \in E$ an equation which can be selected in an HOLN-step π starting from $E\lfloor_W$. If $\Phi_{\mathrm{HOLN}}(T, V, e_k) = \langle \pi, T'\rangle$ then $T' \in Cfg^{oi}$.*

Proof. Let $\pi : E\lfloor_W \Rightarrow_{\alpha, e_k, \theta} E'\lfloor_{W'}$ and $T' = \langle E'\lfloor_{W'}, \gamma', \rho'\rangle$ with $E' = \overline{e'_{N'}}$. To prove that $T' \in Cfg^{oi}$, we must show that for all $i \in \{1, \ldots, N'\}$:

(i) $\rho'(i)$ is an outside-in rewrite derivation, and
(ii) If $e'_i = s \blacktriangleright t$ and $\rho'(i)$ has a rewrite step at position $1{\cdot}p$ such that no later rewrite steps take place above position $1{\cdot}p$ then $p \in Pat(t)$.

We distinguish two cases, whether e'_i is a descendant of e_k in π or not.
Case 1. Assume e'_i is a descendant of e_k in π. Then $\alpha \in \{$[i], [p], [d], [on], [ov]$\}$.

If $\alpha =$[p] then $i = k$, $e'_i = e_k\theta$ and $\rho'(i) = \rho(k)$. Since $\rho(k)$ is outside-in, $\rho'(i)$ is outside-in too. If e'_i is a parameter-passing descendant then $e_k = s \blacktriangleright t$ and $e'_i = s\theta \blacktriangleright t\theta$. Assume $\rho'(i)$ has a rewrite step at position $1{\cdot}p$ and no later rewrite steps take place above $1{\cdot}p$. Since $\rho'(i) = \rho(k)$ and $\rho(k)$ satisfies condition (ii), we learn that $1{\cdot}p \in \mathit{Pat}(t)$. Since $\mathit{Pat}(t) \subseteq \mathit{Pat}(t\theta)$, we learn that $1{\cdot}p \in \mathit{Pat}(t\theta)$. Thus, $\rho'(i)$ satisfies condition (ii).

If $\alpha \in \{$[i],[d]$\}$ then, by the definition of Φ_{HOLN}, $\rho(k)$ is an outside-in rewrite derivation without rewrite steps at the head positions of the equational sides. By construction, $\rho'(i)$ is outside-in too. Moreover, if e'_i is a parameter-passing descendant, then e_k is also a parameter-passing descendant. Since $T \in \mathit{Cfg}^{oi}$, we learn that $\rho(k)$ satisfies condition (ii). This implies that $\rho'(i)$ satisfies condition (ii) too.

Assume $\alpha \in \{$[on],[ov]$\}$. Then π is of the form

$$\pi : E\lfloor_W \Rightarrow_{\alpha,e_k,\theta} (\overline{e_{k-1}\theta}, \overline{\lambda\overline{x}.s_n \blacktriangleright \lambda\overline{x}.l_n}, \lambda\overline{x}.r \simeq \lambda\overline{x}.t\theta, e_{k+1}\theta, \ldots, e_N\theta)\lfloor_{W'}$$

where $e_k = \lambda\overline{x}.s \simeq \lambda\overline{x}.t$ with $\simeq \in \{\approx, \approx^{-1}, \rhd\}$ and $\lambda\overline{x}.s\theta = \lambda\overline{x}.f(\overline{s_n})$. If $e'_i = e'_{k+n} = \lambda\overline{x}.r \simeq \lambda\overline{x}.t\theta$ then from the definition of Φ_{HOLN} (see proof of Lemma 3.2) results that we can assume w.l.o.g. that $\rho'(k+n)$ is obtained from $\rho(k)$ by removing an initial sequence of rewrite steps. Since $\rho(k)$ satisfies conditions (i) and (ii), we conclude that $\rho'(k+n)$ satisfies them too. Otherwise, $e'_i = e'_{k+j-1} = \lambda\overline{x}.s_j \blacktriangleright \lambda\overline{x}.l_j$. Let's assume $\rho'(k+j-1)$ is a rewrite derivation of the form

$$e'_{k+j-1}\gamma' = \lambda\overline{x}.s_j\gamma' \blacktriangleright \lambda\overline{x}.l_j\gamma' \rightarrow^{l'_1\rightarrow r_1}_{1\cdot p_1,\delta_1} \ldots \rightarrow^{l'_m\rightarrow r_m}_{1\cdot p_m,\delta_m} \lambda\overline{x}.l_j\gamma' \blacktriangleright \lambda\overline{x}.l_j\gamma'$$

which successively rewrites at positions $1{\cdot}p_1, \ldots, 1{\cdot}p_m$. By the construction of $\rho'(k+j-1)$ from $\rho(k)$ results that $\rho(k)$ has a rewrite sub-derivation of the form

$$\begin{array}{l}\lambda\overline{x}.f(\overline{s_{j-1}}, s_j, s_{j+1}, \ldots, s_n)\gamma' \simeq \lambda\overline{x}.t\gamma \rightarrow^{l'_1\rightarrow r_1}_{1\cdot q\cdot j\cdot q_1,\delta_1} \cdots \\ \rightarrow^{l'_m\rightarrow r_m}_{1\cdot q\cdot j\cdot q_m,\delta_1} \lambda\overline{x}.f(\overline{s_{j-1}}, l_j, s_{j+1}, \ldots, s_n)\gamma' \simeq \lambda\overline{x}.t\gamma \rightarrow^*_{\mathcal{R}} \lambda\overline{x}.u \simeq \lambda\overline{x}.u\end{array}$$

which starts by rewriting successively at positions $1{\cdot}q{\cdot}j{\cdot}q_1, \ldots, 1{\cdot}q{\cdot}j{\cdot}q_m$. In addition, $q{\cdot}q_i = p_i$ for all $i \in \{1, \ldots, m\}$. This implies that if $\rho'(k+j-1)$ is not outside-in then the displayed sub-derivation of $\rho(k)$ is not outside-in, which contradicts with the fact that $\rho(k)$ is outside-in. Thus, $\rho'(k+j-1)$ is outside-in. It remains to prove that $\rho'(k+j-1)$ satisfies property (ii). Assume $\rho'(k+j-1)$ has a rewrite step at position $1{\cdot}p$ such that no later rewrite steps take place above $1{\cdot}p$. In the outside-in rewrite sub-derivation of $\rho(k)$ depicted above, the first rewrite step above position $1{\cdot}q{\cdot}j{\cdot}q_k$ is at position $1{\cdot}q$ with $f(\overline{l_n}) \rightarrow r$. This implies that $p_k = q{\cdot}q_k \in \mathit{Pat}(l_k)$, i.e., (ii) holds.

Case 2. If e' is not a descendant of e in π then $e'_i = e_i\theta$ for some $i \in \{1, \ldots, N\} - \{k\}$, and $\rho'(i) = \rho(i)$. Since $T \in \mathit{Cfg}^{oi}$, we have that $\rho(i)$ is outside-in, and thus $\rho'(i)$ is outside-in too.

Next, we prove that $\rho'(i)$ satisfies condition (ii). Let's assume e'_i be a parameter-passing descendant. Then e'_i is of the form $s\theta \blacktriangleright t\theta$ where $e_i = s \blacktriangleright t$. Suppose $\rho'(i)$ has a rewrite step at position $1{\cdot}p$ such that no later rewrite steps take place above position $1{\cdot}p$. Since $\rho'(i) = \rho(i)$, we conclude that $p \in \mathit{Pat}(t)$. Since $\mathit{Pat}(t) \subseteq \mathit{Pat}(t\theta)$, we conclude that $\rho'(i)$ satisfies condition (ii). □

We are ready now to define our first refinement of the calculus HOLN.

Definition 15 (HOLN$_1$). HOLN$_1$ *is the calculus defined by* HOLN$_1 \cup \{[\mathrm{v}]\}$ *be the calculus obtained from HOLN by dropping the application of inference rule [on] to selected equations of the form* $\lambda\overline{x}.f(\overline{s_n}) \blacktriangleright \lambda\overline{x}.X(\overline{y_k})$ *where* $f \in \mathcal{F}_d$.

Main properties of HOLN$_1$

First, we prove that if $\mathcal{R}$ is a confluent LEPRS then HOLN$_1$ is strongly complete.

Theorem 4 (Strong completeness). *Let* $\mathcal{R}$ *be a confluent LEPRS. Then* HOLN$_1$ *is strongly complete.*

Proof. Let $V \in \mathcal{P}_{fin}(\mathcal{V})$ with $\mathcal{FV}(E_0 \lfloor_{W_0}) \subseteq V$, and $\gamma \in \mathcal{U}_{\mathcal{R}}(E_0 \lfloor_{W_0})$. We choose $\mathcal{A} := \mathit{Cfg}^{oi}$, $\succeq$ is the lexicographic combination of $\succeq_A$, $\succeq_B$, $\succeq_C$, and $\Phi_{\mathrm{HOLN}_1} := \Phi_{\mathrm{HOLN}}$. To make our generic proof outlined in Section 3.2 work for these choices, we only have to check that:

(a) there exists a configuration $T_0 = \langle E_0 \lfloor_{W_0}, \gamma_0, \rho\rangle \in \mathit{Cfg}^{oi}$. To prove this, we first observe that $E_0 \lfloor_{W_0}$ has no parameter-passing descendants because it is the initial goal. Therefore, by Lemma 10, we conclude that there exists $T_0 = \langle E_0 \lfloor_{W_0}, \gamma_0, \rho\rangle \in \mathit{Cfg}^{oi}$.

(b) If $E = \overline{e_N}$, $T = \langle E\lfloor_W, \gamma, \rho\rangle \in A$, $V \in \mathcal{P}_{fin}(\mathcal{V})$ with $\mathcal{FV}(E\lfloor_W) \subseteq V$, $e_k \in E$ is selectable in an HOLN-step, and $\Phi_{\mathrm{HOLN}}(T, V, e_k) = \langle \pi, T'\rangle$ then $T' \in A$ and $\pi \in \mathit{step}(\mathrm{HOLN}_1)$. The fact that $T' \in A$ follows from Lemma 3.

Thus, it remains to check that $\pi \in \mathit{step}(\mathrm{HOLN}_1)$, i.e., that if $e_k = \lambda\overline{x}.f(\overline{s_n}) \blacktriangleright \lambda\overline{x}.X(\overline{y_n})$ with $f \in \mathcal{F}_d$ then π is not an [on]-step.

Let's assume, by contrary, that $e_k = \lambda\overline{x}.f(\overline{s_n}) \blacktriangleright \lambda\overline{x}.X(\overline{y_n})$ with $f \in \mathcal{F}_d$ and π is an [on]-step. From the definition of Φ_{HOLN} we learn that $\rho(k)$ has a rewrite step at the head position of the left-hand side. Since $\rho(k)$ can not have rewrite steps above the head position of the left-hand side and $\langle E_k \lfloor_{W_k}, \gamma_k, \rho_k\rangle \in \mathit{Cfg}^{oi}$, by property (ii) we learn that the head position of $\lambda\overline{x}.X(\overline{y_n})$ should be a pattern position, which is a contradiction. Hence, π is not an [on]-step. □

Another important property of HOLN$_1$ is soundness, and this is an immediate consequence of soundness of HOLN.

Theorem 5 (Soundness). HOLN$_1$ *is sound.*

4.2 Eager Variable Elimination

In this subsection we present a further refinement of HOLN$_1$ to reduce the nondeterminism of solving parameter-passing descendants of the form $s \blacktriangleright t$ where t is a flex term. The refinement is defined under the assumption that $\mathcal{R}$ is a left-linear EPRS.

First, we note that, by Lemma 9, if $e = s \blacktriangleright t$ is a parameter-passing descendant in a HOLN$_1$ derivation, then t is of the form $\lambda\overline{x}.X(\overline{y})$ where $\overline{y}$ is a permutation of $\overline{x}$ and $X \notin \mathcal{FV}(s)$. This implies that $\theta := \{X \mapsto \lambda\overline{y}.s\}$ is a well defined substitution and $\theta \in \mathcal{U}(e) \subseteq \mathcal{U}_{\mathcal{R}}(e)$.

Definition 16 (HOLN$_2$). *We define the calculus* HOLN$_2$:= HOLN$_1$ ∪ {[v]}, *where* [v] *is a new inference rule defined by:*

[v] **Variable elimination.**

$$(E_1, s \blacktriangleright \lambda\overline{x}.X(\overline{y}), E_2)\!\downharpoonright_W \Rightarrow_{[v],\theta} (E_1, E_2)\theta\!\downharpoonright_{W'}$$

where $\theta = \{X \mapsto \lambda\overline{y}.s\}$, *and assume that* [v] *has the highest priority.*

The calculus HOLN$_2$ is called *higher-order lazy narrowing with eager variable elimination* because it addresses the possibility to eagerly eliminate the free variable occurring in the right-hand side of a parameter-passing descendant by binding it to left-hand side of the equation.

Proving soundness of HOLN$_2$ is trivial.

Theorem 6 (Soundness). *The calculus* HOLN$_2$ *is sound.*

In the rest of this subsection we will prove that HOLN$_2$ is strongly complete. First, we prove the following auxiliary lemmata.

Lemma 11. *Let* $\mathcal{R}$ *be a LEPRS and* $\Pi : E\!\downharpoonright_W \Rightarrow^*_\theta (E_1, s \blacktriangleright t, E_2)\!\downharpoonright_{W'}$ *be an* HOLN$_2$*-derivation. Then*

(i) t *is a linear fully-extended pattern,*
(ii) $(\mathcal{FV}(E_1, s) \cup W') \cap \mathcal{FV}(t) = \emptyset$.

Proof. By Lemma 9, we know that properties (i) and (ii) hold if Π is a HOLN$_1$-derivation. Thus, we only have to check that properties (i) and (ii) are preserved by [v]-steps.

Let $E = (E_1, s \blacktriangleright \lambda\overline{x}.X(\overline{y}), E_2)$ and $\pi : E\!\downharpoonright_W \Rightarrow_{[v],s\blacktriangleright\lambda\overline{x}.X(\overline{y}),\theta} E'\!\downharpoonright_{W'}$ be a [v]-step. Assume that all parameter-passing descendants $s' \blacktriangleright t' \in E$ satisfy the conditions

(a) t' is a linear fully-extended pattern,
(b) If $E = (E', s' \blacktriangleright t', E'')$ then $(\mathcal{FV}(E', s') \cup W) \cap \mathcal{FV}(t') = \emptyset$.

Let $E' = (E''_1, s'' \blacktriangleright t'', E''_2)$. We must show that

(i) t'' is a linear fully-extended pattern,
(ii) $(\mathcal{FV}(E''_1, s'') \cup W') \cap \mathcal{FV}(t'') = \emptyset$.

By the definition of [v], $s'' \blacktriangleright t''$ is a descendant of an equation $s' \blacktriangleright t' \in (E_1, E_2)$. We can write $E = (E'_1, s' \blacktriangleright t', E'_2)$. By (b), we have $X \notin \mathcal{FV}(t'')$, thus $t'' = t'$. Since (a) implies that t' is a linear fully-extended pattern, we learn that (i) holds. To prove (ii), we distinguish two cases:

1. $s \blacktriangleright \lambda\overline{x}.X(\overline{y}) \in E'_1$. Then $Ran(\theta) \subset \mathcal{FV}(E'_1)$, and therefore $\mathcal{FV}(E''_1, s'') \subset \mathcal{FV}(E'_1, s')$. Thus, $\mathcal{FV}(t'') \cap \mathcal{FV}(E''_1, s'') = \mathcal{FV}(t') \cap \mathcal{FV}(E''_1, s'') \subseteq \mathcal{FV}(t') \cap \mathcal{FV}(E'_1, s') = \emptyset$. Also, by (i) we have $X \notin W$, therefore $W' = W$, therefore $W' \cap \mathcal{FV}(t'') = W \cap \mathcal{FV}(t') = \emptyset$. Hence, (ii) holds.

2. $s \blacktriangleright \lambda\overline{x}.X(\overline{y}) \in E_2'$. Then $X \notin \mathcal{FV}(E_1', s' \blacktriangleright t') \cup W$, and thus $E_1'' = E_1'$, $s'' = s'$, $t'' = t'$, $W = W'$. In this case (ii) follows from (b). $\square$

Lemma 12. *Let* $E = (\overline{e_{k-1}}, s \blacktriangleright \lambda\overline{x}.X(\overline{y}), e_{k+1}, \ldots, e_N)$, $T = \langle E\lfloor_{\underline{W}, \gamma, \rho}\rangle \in$ *Cfg*^{oi}, $V \in \mathcal{P}_{fin}(\mathcal{V})$ *with* $\mathcal{FV}(E\lfloor_W) \subseteq V$, *and* $\pi : E\lfloor_W \Rightarrow_{[v],s\blacktriangleright\lambda\overline{x}.X(\overline{y}),\theta} \overline{e'_{N-1}}\lfloor_{W'}$. *We define*

$$\gamma' = \gamma\restriction_{Dom(\gamma)-\{X\}},$$
$$\rho' : \{1,\ldots,N-1\} \to \bigcup_{i=1}^{N-1} \mathit{Proof}_{\mathcal{R}}(e_i', \gamma'),\ \rho'(i) = \begin{cases} \rho(i) & \text{if } i < k, \\ \rho(i+1) & \text{if } k \leq i < N \end{cases}$$
$$T' = \langle \overline{e'_{N-1}}\lfloor_{W'}, \gamma', \rho'\rangle$$

Then $T' \in$ *Cfg*^{oi} *and* $T \succ T'$. *We denote the pair* $\langle \pi, T'\rangle$ *by* $\Phi_{[v]}(T, V, e_k)$.

Proof. From $T \in$ *Cfg*^{oi} we learn that $\rho(k)$ is an empty rewrite proof of $\gamma \in \mathcal{U}_{\mathcal{R}}(s \blacktriangleright \lambda\overline{x}.X(\overline{y}))$. This implies that $X\gamma = \lambda\overline{y}.s\gamma$, therefore $\rho' \in \mathit{Proof}_{\mathcal{R}}(\overline{e'_{N-1}}, \gamma')$ and $T' \in$ *Cfg*^{oi}. Obviously, $T =_A T'$, $T \succeq_B T'$ and $T \succ_c T'$, thus $T \succ T'$. $\square$

We are ready now to claim that the calculus HOLN$_2$ is strongly complete.

Theorem 7 (Strong completeness). *Let* $\mathcal{R}$ *be a confluent LEPRs. Then* HOLN$_2$ *is strongly complete.*

Proof. We choose $(\mathcal{A}, \succeq)$ as in the proof of strong completeness of HOLN$_1$ and define the partial function

$$\Phi_{\text{HOLN}_2} : \mathit{Cfg}^{oi} \to \mathcal{P}_{fin}(V) \times \mathit{Eq}(\mathcal{F}, \mathcal{V}) \to \mathit{step}(\text{HOLN}_2) \times \mathit{Cfg}^{oi}$$
$$\Phi_{\text{HOLN}_2}(T, V, e) = \begin{cases} \Phi_{[v]}(T, V, e) & \text{if } \Phi_{[v]}(T, V, e) \text{ is defined,} \\ \Phi_{\text{HOLN}_1}(T, V, e) & \text{otherwise.} \end{cases}$$

$\square$

4.3 Lazy Narrowing with Confluent Constructor LEPRSs

In this subsection we present a refinement of HOLN for confluent constructor LEPRSs. This refinement has been inspired by a similar refinement of the calculus LNC with leftmost equation selection strategy for left-linear constructor TRSs [10], and addresses the possibility to avoid the generation of parameter-passing descendants of the form $s \blacktriangleright t$ with $t \notin \mathcal{T}(\mathcal{F}_c, \mathcal{V})$. The effect of this behavior is that the nondeterminism between inference rules [on] and [d] disappears for parameter-passing descendants.

Definition 17. *A C-derivation respects strategy* $\mathcal{S}_{left}$ *if every C-step is applied to the leftmost selectable equation.*

It has been shown [10] that the calculus LNC with leftmost equation selection strategy $\mathcal{S}_{\text{left}}$ does not generate parameter-passing descendants with defined symbols in the right-hand side. It can be shown that the calculus HOLN$_2$ has this property too.

Lemma 13. *Let $\mathcal{R}$ be a confluent constructor LEPRS. If Π is an* HOLN_2*-derivation which respects strategy $\mathcal{S}_{\text{left}}$ then all equations $s \blacktriangleright t$ in Π satisfy the condition $t \in \mathcal{T}(\mathcal{F}_c, \mathcal{V})$.*

Proof. Because $\mathcal{R}$ is a constructor LEPRS, the only way to generate equations $s \blacktriangleright t$ with $t \notin \mathcal{T}(\mathcal{F}_c, \mathcal{V})$ is via α-steps of the form

$$(E_1, \lambda\overline{x_n}.s' \blacktriangleright \lambda\overline{x_n}.X(\overline{y_n}), E, \underbrace{\lambda\overline{z_k}.X(\overline{t_n}) \simeq \lambda\overline{z_k}.u}_{e}, E_2)\lfloor_W \Rightarrow_{\alpha,e,\{X \mapsto \lambda\overline{x_n}.f(\ldots)\}} \cdots$$

with $X \in \mathcal{FV}(t')$, $f \in \mathcal{F}_d$ and $\alpha \in \{[\text{i}],[\text{ov}]\}$. Such an α-step can not occur in the HOLN_2-derivation because it would not respect strategy $\mathcal{S}_{\text{left}}$: the equation $\lambda\overline{x_n}.s' \blacktriangleright \lambda\overline{x_n}.X(\overline{y_n})$ is more to the left than e, and it can be selected in a [v]-step. □

Corollary 2. *Let Π be an* HOLN_2*-refutation which respects strategy $\mathcal{S}_{\text{left}}$. Then Π has no* [on]*-steps applied to parameter-passing descendants.*

Proof. Immediate consequence of Lemma 13. □

We are ready now to define our refinement for confluent constructor LEPRS.

Definition 18 (HOLN_3). HOLN_3 *is the calculus obtained from* HOLN_2 *by dropping the application of rule* [on] *to parameter-passing descendants.*

Theorem 8 (Soundness). HOLN_3 *is sound.*

Proof. Immediate consequence from the soundness property of HOLN_2. □

Theorem 9 (Completeness). *The calculus* HOLN_3 *with leftmost equation selection strategy is complete.*

Proof. Immediate consequence of Corollary 2. □

Remark 3. In our previous work of studying higher-order lazy narrowing with confluent constructor LEPRSs, we have proposed another calculus to avoid the generation of parameter-passing descendants $s \blacktriangleright t$ with defined symbol occurrences in the left-hand side. Our previous calculus differs from HOLN_3 by replacing the inference rule [v] with another rule called [c], which is shown below.

[c] **Constructor propagation.**
If $\exists s \blacktriangleright \lambda\overline{x_k}.X(\overline{y_k}) \in E_1$ and $s' = \lambda\overline{y_k}.s(\overline{x_k})$ then

$$(E_1, \lambda\overline{z_n}.X(\overline{t_k}) \simeq \lambda\overline{z_n}.u, E_2)\lfloor_W \Rightarrow_{[\text{c}],\epsilon} (E_1, \lambda\overline{z_n}.s'(\overline{t_k}) \simeq \lambda\overline{z_n}.u, E_2)\lfloor_{W'}$$

By giving to rule [c] the highest priority, it can be shown that this calculus with leftmost equation selection strategy is strongly complete. Also, this calculus is sound for goals consisting of unoriented equations. □

4.4 HOLN$_4$: a Refinement to Detect Redundant Equations

In this subsection we investigate the possibility to detect and eliminate equations which do not contribute to the computation of an answer substitution. We call such equations *redundant.* By detecting and eliminating such useless equations we save computing time.

Definition 19. *Let* $(E_1, e, E_2)\lfloor_W$ *be a goal. The equation* $e \in E$ *is called* redundant *in* $E\lfloor_W$ *if* $e = \lambda\overline{x_k}.s \blacktriangleright \lambda\overline{x_k}.X(\overline{y_k})$ *such that the following conditions are satisfied:*

(a) $\overline{y_k}$ *is a permutation of the sequence* $\overline{x_k}$*,*
(b) $X \notin \mathcal{FV}(E_1, \lambda\overline{x_k}.s, E_2)$.

We define the calculus HOLN$_4$:= HOLN $\cup$ {[rm]} where [rm] is a new inference rule defined as follows:

[rm] Removal of redundant equations.
If $e = \lambda\overline{x_k}.s \blacktriangleright \lambda\overline{x_k}.X(\overline{y_k})$ is redundant in $(E_1, e, E_2)\lfloor_W$ then

$$(E_1, e, E_2)\lfloor_W \Rightarrow_{[\mathrm{rm}],e,\epsilon} (E_1, E_2)\lfloor_W$$

To give the calculus HOLN$_4$ a more deterministic computational behavior, we may assume that the inference rule [rm] has the highest priority, i.e., that when [rm] is applicable to a selected equation, then we discard the application of the other inference rules.

Main properties.
We argue that HOLN$_4$ is sound and strongly complete.

Theorem 10. HOLN$_4$ *is sound.*

Proof. It is enough to check that if there is an HOLN$_4$-derivation

$$\Pi_1 : E\lfloor_W \Rightarrow^*_\theta (E_1, e, E_2)\lfloor_{W'}$$

with e redundant in $(E_1, e, E_2)\lfloor_{W_1}$ then the HOLN$_4$-derivation

$$\Pi_2 : E\lfloor_W \Rightarrow^*_\theta (E_1, e, E_2)\lfloor_{W'} \Rightarrow_{[\mathrm{rm}],\epsilon} (E_1, E_2)\lfloor_{W'}$$

obtained by extending Π_1 with an [rm]-step, satisfies the condition:

$$\{\theta\gamma\restriction_{\mathcal{FV}(E\lfloor_W)} \mid \gamma \in \mathcal{U}_\mathcal{R}(E_1, e, E_2)\} = \{\theta\gamma'\restriction_{\mathcal{FV}(E\lfloor_W)} \mid \gamma' \in \mathcal{U}_\mathcal{R}(E_1, E_2)\}.$$

By Lemma 1, it is sufficient to prove that

$$\{\gamma\restriction_V \mid \gamma \in \mathcal{U}_\mathcal{R}((E_1, e, E_2)\lfloor_{W'})\} = \{\gamma'\restriction_V \mid \gamma' \in \mathcal{U}_\mathcal{R}((E_1, E_2)\lfloor_{W'})\}$$

where $V = \mathcal{FV}(E\theta) \cup W'$. To prove this fact, it is sufficient to note that $X \notin V$. This follows by an easy induction proof on the length of Π_1. □

Theorem 11. *Let $\mathcal{R}$ be a confluent PRS. Then* HOLN$_4$ *is strongly complete.*

Proof. We choose $\mathcal{A} = Cfg$ and the poset $\langle \mathcal{A}, \succeq \rangle$ where $\succeq$ is the lexicographic combination of orderings $\succeq_A$, $\succeq_B$, $\succeq_C$ used in the completeness proof of HOLN.

Let $E = \overline{e_N}$, $E{\downharpoonleft}_W \in Goal(\mathcal{F}, \mathcal{V})$, $V \in \mathcal{P}_{fin}(\mathcal{V})$ with $\mathcal{FV}(E{\downharpoonleft}_W) \subseteq V$, and $T = \langle E{\downharpoonleft}_W, \gamma, \rho \rangle \in Cfg$. For any equation $e_k \in E$ which is selectable in an HOLN$_4$-step we define:

$$\Phi_{\text{HOLN}_4}(T, V, e_k) = \begin{cases} \langle \pi, \langle E'{\downharpoonleft}_W, \gamma, \rho' \rangle \rangle \text{ if } \pi : E{\downharpoonleft}_W \Rightarrow_{[\text{rm}], e_k} E'{\downharpoonleft}_W \in step(\text{HOLN}_3), \\ \qquad\qquad \text{where } \rho'(i) = \begin{cases} \rho(i) & \text{if } 1 \le i < k, \\ \rho(i+1) & \text{if } k \le i < N \end{cases}. \\ \Phi_{\text{HOLN}}(T, V, e_k) \;\; \text{otherwise.} \end{cases}$$

Obviously, $T' \in Cfg$ and $T \succeq_A T', T \succeq_B T', T \succ_C T'$. Thus, $T \succ T'$. □

If $(E_1, \lambda\overline{x_k}.s \blacktriangleright \lambda\overline{x_k}.X(\overline{y_k}), E_2){\downharpoonleft}_W$ a goal in a HOLN-derivation then, by Lemma 9, we know that $X \notin \mathcal{FV}(E_1, \lambda\overline{x_k}.s)$ and $\overline{y_k}$ is a permuted sequence of $\overline{x_k}$. Thus, we have the following result:

Lemma 14. *Let $\mathcal{R}$ be a confluent LEPRS and $E_1, e, E_2{\downharpoonleft}_W \in Goal(\mathcal{F}, \mathcal{V})$. The equation e is redundant in $(E_1, e, E_2){\downharpoonleft}_W$ if $e = \lambda\overline{x_k}.s \blacktriangleright \lambda\overline{x_k}.X(\overline{y_k})$ and $X \notin \mathcal{FV}(E_2)$.*

5 Conclusions and Future Work

We have presented a new lazy narrowing calculus HOLN for EPRS designed to compute solutions which are normalized with respect to a given set of variables, and then have presented 4 refinements to reduce its nondeterminism in a way which preserves completeness.

The results presented in this paper owe largely to a new formalism in which we treat a goal as a pair consisting of a sequence of equations and a set of variables for which we want to compute normalized answers. This formulation of narrowing has the following advantages:

- it clarifies problems and locates points for optimization during the refutation process of goals,
- it simplifies the soundness and completeness proofs of the calculi,
- it simplifies and systematizes the implementation of the lazy narrowing calculus as a computational model of a higher-order FLP system.

A promising direction of research is to extend HOLN to conditional PRSs. A program specification using conditional PRSs is much more expressive because it allows the user to impose equational conditions under which rewrite steps are allowed. Such an extension is quite straightforward, but it adds nondeterminism in guessing the values of the additional variables in the conditional part of rewrite rules. We expect that this source of nondeterminism can be avoided if we restrict to certain classes of conditional PRSs.

Acknowledgements. The work reported in this paper has been supported in part by the Ministry of Education, Culture, Sports, Science and Technology, Grant-in-Aid for Scientific Research (B) 12480066, 2000-2002. Mircea Marin has been supported by JSPS postdoc fellowship 00096, 2000-2001.

References

1. H.P. Barendregt. *The Lambda Calculus, its Syntax and Semantics*, volume 90. North Holland, second edition, 1984.
2. H.P. Barendregt. *Lambda calculi with types*, volume 2 of *Handbook of Logic in Computer Science*, pages 118–309. Oxford University Press, 1992.
3. N.G. de Bruijn.
4. J.C. Gónzalez-Moreno, M.T. Hortalá Gónzalez, and M. Ródriguez-Artalejo. A Higher-Order Rewriting Logic for Functional Logic Programming. In *Proceedings of International Conference on Logic Programming*, pages 153–167, Leuven, 1997. MIT Press.
5. J.R. Hindley and J.P. Seldin. *Introduction to Combinatorics and λ-Calculus.* Cambridge University Press, 1986.
6. G. Huèt. A Unification Algorithm for Typed λ-Calculus. *Theoretical Computer Science*, 1975.
7. M. Marin, T. Ida, and T. Suzuki. On Reducing the Search Space of Higher-Order Lazy Narrowing. In A. Middeldorp and T. Sato, editors, *FLOPS'99*, volume 1722 of *LNCS*, pages 225–240. Springer-Verlag, 1999.
8. M. Marin, A. Middeldorp, T. Ida, and T. Yanagi. LNCA: A Lazy Narrowing Calculus for Applicative Term Rewriting Systems. Technical Report ISE-TR-99-158, Institute of Information Sciences and Electronics, University of Tsukuba, Tsukuba, Japan, 1999.
9. R. Mayr and T. Nipkow. Higher-order rewrite systems and their confluence. Technical report, Institut für Informatik, TU München, 1994.
10. A. Middeldorp and S. Okui. A deterministic lazy narrowing calculus. *Journal of Symbolic Computation*, 25(6):733–757, 1998.
11. A. Middeldorp, S. Okui, and T. Ida. Lazy narrowing: Strong completeness and eager variable elimination. *Theoretical Computer Science*, 167(1,2):95–130, 1996.
12. D. Miller. A logic programming language with lambda-abstraction, function variables, and simple unification. *Journal of Logic and Computation*, 1:497–536, 1991.
13. K. Nakahara, A. Middeldorp, and T. Ida. A complete narrowing calculus for higher-order functional logic programming. In *Seventh International Conference on Programming Languages: Implementations, Logics and Programs 95 (PLILP'95)*, volume 982 of *LNCS*, 1995.
14. T. Nipkow. Functional unification of higher-order patterns. In *Proceedings of 8th IEEE Symposium on Logic in Computer Science*, pages 64–74, 1993.
15. C. Prehofer. *Solving Higher-Order Equations. From Logic to Programming.* Foundations of Computing. Birkhäuser Boston, 1998.
16. Z. Qian. Linear unification of higher-order patterns. In M.-C. Gaudel and J.-P. Jouannaud, editors, *Proceedings of the Colloquium on Trees in Algebra and Programming*, volume 668 of *LNCS*, pages 391–405, Orsay, France, April 1993.
17. V. van Oostrom. Personal communication.
18. D.A. Wolfram. The clausal theory of types. *Theoretical Computer Science*, 21, 1993.