A Deterministic Lazy Narrowing Calculus

Aart Middeldorp
Institute of Information Sciences and Electronics
University of Tsukuba, Tsukuba 305, Japan
ami@score.is.tsukuba.ac.jp

Satoshi Okui
Faculty of Engineering
Mie University, Tsu 514, Japan
okui@cs.info.mie-u.ac.jp

August 29, 1996

Abstract

In this paper we study the non-determinism between the inference rules of the lazy narrowing calculus LNC [16]. We show that all non-determinism can be removed without losing the important completeness property by restricting the underlying term rewriting systems to left-linear confluent constructor systems and interpreting equality as strict equality.

1 Introduction

Besides being a general method for solving unification problems in equational theories that are presented by confluent term rewriting systems (TRSs for short, [12] contains a comprehensive survey), narrowing (Hullot [10]) is the computation model of many functional logic programming languages (Hanus [6]). Since narrowing is a complicated operation, various calculi consisting of a small number of more elementary inference rules that simulate narrowing have been proposed, e.g. [5, 7, 8, 11, 13, 14, 17]. These calculi are highly non-deterministic: in general all choices of (1) the equation in the current goal, (2) the inference rule to be applied, and (3) the rewrite rule of the TRS (for certain inference rules) have to be considered in order to guarantee the desirable property of completeness. In this paper we address the second kind of non-determinism for the calculus LNC of [16]. This calculus is the specialization of Hölldobler's calculus TRANS [8], which is defined for general equational systems and based on paramodulation, to (confluent) TRSs and narrowing. The main reason for choosing LNC is that its completeness has been established for arbitrary confluent TRSs (Middeldorp, Okui, and Ida [16]).

The lazy narrowing calculus LNC consists of the following five inference rules:

A preliminary version of this paper appeared in the Proceedings of the Fuji International Workshop on Functional and Logic Programming, Susuno, World Scientific, pp. 104–118, 1995. The first author is partially supported by the Advanced Information Technology Program (AITP) of the Information Technology Promotion Agency (IPA).

$[o]$ *outermost narrowing*

$$\frac{G', f(s_1, \ldots, s_n) \simeq t, G''}{G', s_1 \approx l_1, \ldots, s_n \approx l_n, r \approx t, G''}$$

if there exists a fresh variant $f(l_1, \ldots, l_n) \to r$ of a rewrite rule in $\mathcal{R}$,

$[i]$ *imitation*

$$\frac{G', f(s_1, \ldots, s_n) \simeq x, G''}{(G', s_1 \approx x_1, \ldots, s_n \approx x_n, G'')\theta}$$

if $\theta = \{x \mapsto f(x_1, \ldots, x_n)\}$ with $x_1, \ldots, x_n$ fresh variables,

$[d]$ *decomposition*

$$\frac{G', f(s_1, \ldots, s_n) \approx f(t_1, \ldots, t_n), G''}{G', s_1 \approx t_1, \ldots, s_n \approx t_n, G''},$$

$[v]$ *variable elimination*

$$\frac{G', s \simeq x, G''}{(G', G'')\theta}$$

if $x \notin \mathcal{V}\mathrm{ar}(s)$ and $\theta = \{x \mapsto s\}$,

$[t]$ *removal of trivial equations*

$$\frac{G', x \approx x, G''}{G', G''}.$$

In the rules $[o]$, $[i]$, and $[v]$, $s \simeq t$ stands for $s \approx t$ or $t \approx s$. Contrary to usual narrowing, the outermost narrowing rule $[o]$ generates new *parameter-passing* equations $s_1 \approx l_1, \ldots, s_n \approx l_n$ besides the *body* equation $r \approx t$. These parameter-passing equations must eventually be solved, but we don't require that they are solved right away.

This paper is a continuation of [16]. Below we introduce some of the concepts that are used later. Notions not defined in the present paper can be found in [16]. The narrowing calculus NC consists of the following inference rule:

$$\frac{G', e, G''}{(G', e[r]_p, G'')\theta}$$

if there exist a fresh variant $l \to r$ of a rewrite rule in $\mathcal{R} \cup \{x \approx x \to \mathtt{true}\}$, a non-variable position p in e, and a most general unifier θ of $e_{|p}$ and l.

The rewrite rule $x \approx x \to \mathtt{true}$ is used to simulate syntactic unification. It is similar to reflection in other narrowing calculi. In the above situation we write $G', e, G'' \leadsto_\theta (G', e[r]_p, G'')\theta$. This is called an NC-step. A sequence $G_1 \leadsto_{\theta_1} \cdots \leadsto_{\theta_{n-1}} G_n$ of NC-steps is called an NC-derivation and abbreviated to $G_1 \leadsto_\theta^* G_n$ where $\theta = \theta_1 \cdots \theta_{n-1}$. We use the symbol Π (and its derivatives) to denote NC-derivations. For an NC-derivation $\Pi\colon G \leadsto_\theta^* G'$, $\Pi\theta$ denotes the corresponding rewrite sequence $G\theta \to^* G'$. An NC-derivation which ends in $\top$—a generic notation for goals that consist entirely of $\mathtt{true}$'s—is called an NC-refutation.

We extend these notions to the calculus LNC. If G and G' are the upper and lower goal in the inference rule $[\alpha]$ ($\alpha \in \{o, i, d, v, t\}$), we write $G \Rightarrow_{[\alpha]} G'$. This is called an LNC-step. The applied rewrite rule or substitution may be supplied as subscript, that is, we will write things like $G \Rightarrow_{[o], l \to r} G'$ and $G \Rightarrow_{[i], \theta} G'$. A finite LNC-derivation $G_1 \Rightarrow_{\theta_1} \cdots \Rightarrow_{\theta_{n-1}} G_n$ may be abbreviated to $G_1 \Rightarrow_\theta^* G_n$ where $\theta = \theta_1 \cdots \theta_{n-1}$. An LNC-refutation is an LNC-derivation ending in the empty goal $\Box$.

The set of function symbols $\mathcal{F}$ of a TRS $\mathcal{R}$ is partitioned into disjoint sets $\mathcal{F}_\mathcal{D}$ and $\mathcal{F}_\mathcal{C}$ as follows: a function symbol f belongs to $\mathcal{F}_\mathcal{D}$ if there is a rewrite rule $l \to r$ in $\mathcal{R}$ such that $l = f(t_1, \ldots, t_n)$ for some terms $t_1, \ldots, t_n$, otherwise $f \in \mathcal{F}_\mathcal{C}$. Function symbols in $\mathcal{F}_\mathcal{C}$ are called constructors, those in $\mathcal{F}_\mathcal{D}$ defined symbols. A term built from constructors and variables is called a constructor term. A constructor system (CS for short) is a TRS with the property that the arguments $t_1, \ldots, t_n$ of every left-hand side $f(t_1, \ldots, t_n)$ of a rewrite rule are constructor terms. A left-linear TRS without critical pairs is called orthogonal.

The remainder of this paper is organized as follows. In the next section we discuss the non-determinism in LNC and recall some results of [16]. In Section 3 we show under what restrictions the non-determinism between the inference rules of LNC for equations that descend from parameter-passing equations can be removed. In Section 4 we do the same for equations that descend from equations in the initial goal. In Section 5 we present our deterministic lazy narrowing calculus. This calculus is compared with the narrowing calculi of Ida and Nakahara [11] and González-Moreno *et al.* [4]. A comparison of the performance of LNC and our deterministic lazy narrowing calculus on a small number of examples confirms our theoretical result.

2 Non-Determinism

There are three sources of non-determinism in LNC: the choice of the equation, the choice of the inference rule, and the choice of the rewrite rule $l \to r$ (in the case of [o]). In [16] it is shown that all three choices are *don't know* non-deterministic. This means that in general all possible choices have to be considered in order to guarantee completeness, resulting in a huge search space. In particular, LNC lacks *strong* completeness. In other words, completeness is not independent of selection functions. (This contradicts [8, Corollary 7.3.9]. In [16] we showed that LNC is strongly complete whenever *basic* narrowing is complete [10, 15].) Actually things are not that bad. In [16] we showed that for completeness it is sufficient to restrict attention to the selection function $\mathcal{S}_{\text{left}}$ that selects the leftmost equation in every goal. So we may assume that G'—the sequence of equations to the left of the selected equation—in the inference rules of LNC is empty. A formal statement of completeness is given below. Observe that we don't require TRSs to be terminating.

Theorem 2.1 (Middeldorp, Okui, and Ida [16]) *Let $\mathcal{R}$ be a confluent TRS and G a goal. For every normalized solution θ of G there exists an* LNC*-refutation $G \Rightarrow^*_{\theta'} \Box$ respecting $\mathcal{S}_{\text{left}}$ such that $\theta' \leqslant \theta$ $[\mathcal{V}\text{ar}(G)]$.* $\Box$

So the non-determinism of LNC due to the selection of the equation is avoided if we adopt the leftmost section function. The structure of the proof of Theorem 2.1, to which we refer later, is as follows. First of all, every normalized solution of G is subsumed by a substitution produced by a *normal* NC-refutation starting from G. An NC-refutation $\Pi\colon G \rightsquigarrow^*_\theta \top$ is said to be normal if it respects $\mathcal{S}_{\text{left}}$ and satisfies the following property: if narrowing is applied to the left-hand side (right-hand side) of a descendant of an equation $s \approx t$ in G then $\theta_2\restriction_{\mathcal{V}\text{ar}(s\theta_1)}$ $(\theta_2\restriction_{\mathcal{V}\text{ar}(t\theta_1)})$ is normalized. Here θ_1 and θ_2 are (uniquely) defined by writing Π as

$$G = G', s \approx t, G'' \quad \rightsquigarrow^*_{\theta_1} \quad \top, (s \approx t, G'')\theta_1 \quad \rightsquigarrow^*_{\theta_2} \quad \top.$$

The main part of the proof of Theorem 2.1 consists in showing that for every non-empty normal NC-refutation $\Pi\colon G \rightsquigarrow^+_\theta \top$ there exist an LNC-step $\Psi_1\colon G \Rightarrow_{\sigma_1} G_1$ respecting $\mathcal{S}_{\text{left}}$ and a normal NC-refutation $\Pi_1\colon G_1 \rightsquigarrow^*_{\theta_1} \top$ such that Π_1 is smaller than Π in some well-founded order $\gg$ on NC-refutations and $\sigma_1\theta_1 \leqslant \theta$ for a suitable set of variables V:

$$\begin{array}{llcllll} \forall & \text{normal} & \Pi & : & G & \rightsquigarrow^+_\theta & \top \\ \exists & & \Psi_1 & : & \Downarrow_{\sigma_1} & & \qquad \text{such that } \sigma_1\theta_1 \leqslant \theta\ [V] \text{ and } \Pi \gg \Pi_1. \\ \exists & \text{normal} & \Pi_1 & : & G_1 & \rightsquigarrow^*_{\theta_1} & \top \end{array}$$

The inference rule employed in the LNC-step Ψ_1 depends on what happens to the selected (leftmost) equation in the given NC-refutation Π, as shown below.

In the following five lemmata Π denotes a normal NC-refutation $G \rightsquigarrow^+_\theta \top$ with $G = s \approx t, G'$ and V denotes a finite set of variables that includes all variables in the initial goal G of Π.

Lemma 2.2 *Suppose narrowing is applied to a descendant of $s \approx t$ in Π at position 1. If $l \to r$ is the applied rewrite rule in the first such step then there exists a normal* NC*-refutation $\phi_{[o]}(\Pi)\colon s \approx l, r \approx t, G' \rightsquigarrow^*_{\theta_1} \top$ such that $\theta_1 = \theta\ [V]$.* □

Lemma 2.3 *Let $s = f(s_1, \ldots, s_n)$ and $t \in \mathcal{V}$. If* $\mathrm{root}(t\theta) = f$ *then there exists a normal* NC*-refutation $\phi_{[i]}(\Pi)\colon G\sigma_1 \rightsquigarrow^*_{\theta_1} \top$ such that $\sigma_1\theta_1 = \theta\ [V]$ and $\Pi\theta = \phi_{[i]}(\Pi)\theta_1$. Here $\sigma_1 = \{t \mapsto f(x_1, \ldots, x_n)\}$ with $x_1, \ldots, x_n \notin V$.* □

Lemma 2.4 *Let $s = f(s_1, \ldots, s_n)$, $t = f(t_1, \ldots, t_n)$, and suppose that narrowing is never applied to a descendant of $s \approx t$ in Π at position 1 or 2. There exists a normal* NC*-refutation $\phi_{[d]}(\Pi)\colon s_1 \approx t_1, \ldots, s_n \approx t_n, G' \rightsquigarrow^*_{\theta_1} \top$ such that $\theta_1 \leqslant \theta\ [V]$.* □

Lemma 2.5 *Let $t \in \mathcal{V}$, $s \neq t$, and suppose that the first step of Π takes place at the root position. There exists a normal* NC*-refutation $\phi_{[v]}(\Pi)\colon G'\sigma_1 \rightsquigarrow^*_{\theta_1} \top$ with $\sigma_1 = \{t \mapsto s\}$ such that $\sigma_1\theta_1 \leqslant \theta\ [V]$.* □

Lemma 2.6 *Let $t \in \mathcal{V}$, $s = t$, and suppose that the first step of Π takes place at the root position. There exists a normal* NC*-refutation $\phi_{[t]}(\Pi)\colon G' \rightsquigarrow^*_{\theta_1} \top$ such that $\theta_1 \leqslant \theta\ [V]$.* □

The transformations $\phi_{[o]}$ and $\phi_{[d]}$ in Lemmata 2.2 and 2.4 correspond to $\phi_{[o]} \circ \phi_1$ and $\phi_{[d]} \circ \phi_2$ in [16]. The purpose of ϕ_1 and ϕ_2 is to reorder narrowing steps in such a way that subsequent applications of $\phi_{[o]}$ and $\phi_{[d]}$ (in [16]) result in NC-refutations that respect $\mathcal{S}_{\text{left}}$. Since we don't need to know the details of the transformations ϕ_1 and ϕ_2 here, we choose to incorporate them in the definitions of $\phi_{[o]}$ and $\phi_{[d]}$.

According to the next lemma the above transformations decrease the complexity of NC-refutations.

Lemma 2.7 *Let* Π *be a normal* NC*-refutation and* $\alpha \in \{[o],[i],[d],[v],[t]\}$. *We have* $\Pi \gg \phi_\alpha(\Pi)$ *whenever* $\phi_\alpha(\Pi)$ *is defined.* $\square$

Lemma 2.8 explains why we don't need the symmetric versions of Lemmata 2.2, 2.3, and 2.5.

Lemma 2.8 *For every normal* NC*-refutation* $\Pi: s \approx t, G' \rightsquigarrow^*_\theta \top$ *there exists a normal* NC*-refutation* $\phi_{\text{swap}}(\Pi): t \approx s, G' \rightsquigarrow^*_\theta \top$ *with the same complexity.* $\square$

The main lemma for establishing the completeness of LNC is now easily proved.

Lemma 2.9 *For every non-empty normal* NC*-refutation* $\Pi: G \rightsquigarrow^+_\theta \top$ *there exist an* LNC*-step* $\Psi_1: G \Rightarrow_{\sigma_1} G_1$ *respecting* $\mathcal{S}_{\text{left}}$ *and a normal* NC*-refutation* $\Pi_1: G_1 \rightsquigarrow^*_{\theta_1} \top$ *such that* $\sigma_1\theta_1 \leqslant \theta\ [V]$ *and* $\Pi \gg \Pi_1$.

Proof. We distinguish the following cases, depending on what happens to the selected equation $e = \mathcal{S}_{\text{left}}(G)$ in Π. Let $G = e, G'$ and $e = s \approx t$.

(1) Suppose narrowing is never applied to a descendant of $s \approx t$ at position 1 or 2. We distinguish four further cases.
   (a) Suppose $s, t \notin \mathcal{V}$. We may write $s = f(s_1, \ldots, s_n)$ and $t = f(t_1, \ldots, t_n)$. Let $G_1 = s_1 \approx t_1, \ldots, s_n \approx t_n, G'$. We have $\Psi_1: G \Rightarrow_{[d]} G_1$. Lemma 2.4 yields a normal NC-refutation $\Pi_1 = \phi_{[d]}(\Pi): G_1 \rightsquigarrow^*_{\theta_1} \top$ such that $\theta_1 \leqslant \theta\ [V]$. Take $\sigma_1 = \epsilon$.
   (b) Suppose $t \in \mathcal{V}$ and $s = t$. In this case the first step of Π_1 must take place at the root of e. Let $G_1 = G'$. We have $\Psi_1: G \Rightarrow_{[t]} G_1$. Lemma 2.6 yields a normal NC-refutation $\Pi_1 = \phi_{[t]}(\Pi): G_1 \rightsquigarrow^*_{\theta_1} \top$ such that $\theta_1 \leqslant \theta\ [V]$. Take $\sigma_1 = \epsilon$.
   (c) Suppose $t \in \mathcal{V}$ and $s \neq t$. We distinguish two further cases, depending on what happens to e in the first step of Π.
      (i) Suppose narrowing is applied to e at the root position. Let $\sigma_1 = \{t \mapsto s\}$ and $G_1 = G'\sigma_1$. We have $\Psi_1: G \Rightarrow_{[v],\sigma_1} G_1$. Lemma 2.5 yields a normal NC-refutation $\Pi_1 = \phi_{[v]}(\Pi): G_1 \rightsquigarrow^*_{\theta_1} \top$ such that $\sigma_1\theta_1 \leqslant \theta\ [V]$.
      (ii) Suppose narrowing is not applied to e at the root position. This is only possible if $s \notin \mathcal{V}$. Hence we may write $s = f(s_1, \ldots, s_n)$. Let $\sigma_1 = \{t \mapsto f(x_1, \ldots, x_n)\}$, $G_1 = (s_1 \approx x_1, \ldots, s_n \approx x_n, G')\sigma_1$, and $G_2 = G\sigma_1$. Here $x_1, \ldots, x_n$ are fresh variables. We have $\Psi_1: G \Rightarrow_{[i],\sigma_1} G_1$. From Lemma 2.3 we obtain an NC-refutation $\Pi_2 = \phi_{[i]}(\Pi): G_2 \rightsquigarrow^*_{\theta_2} \top$ such that $\sigma_1\theta_2 = \theta\ [V]$. Let $V_2 = V \cup \{x_1, \ldots, x_n\}$. Clearly $\mathcal{V}\text{ar}(G_2) \subseteq V_2$. An application of Lemma 2.4 to Π_2 results in an NC-refutation $\Pi_1 = \phi_{[d]}(\Pi_2): G_1 \rightsquigarrow^*_{\theta_1} \top$ such that $\theta_1 \leqslant \theta_2\ [V_2]$. Using the inclusion $(V - \mathcal{D}(\sigma_1)) \cup \mathcal{I}(\sigma_1 \restriction_V) \subseteq V_2$ we obtain $\sigma_1\theta_1 \leqslant \sigma_1\theta_2 = \theta\ [V]$.
   (d) In the remaining case we have $t \notin \mathcal{V}$ and $s \in \mathcal{V}$. This case reduces to case (1)(c) by an appeal to Lemma 2.8.

(2) Suppose narrowing is applied to a descendant of e at position 1. Let $l = f(l_1, \ldots, l_n) \to r$ be the used rewrite rule the first time this happens. Because Π is normal, s cannot be a variable. Hence we may write $s = f(s_1, \ldots, s_n)$. Let $G_1 = s_1 \approx l_1, \ldots, s_n \approx l_n, r \approx t, G'$ and $G_2 = s \approx l, r \approx t, G'$. We have $\Psi_1: G \Rightarrow_{[o]} G_1$. From Lemma 2.2 we obtain an NC-refutation $\Pi_2 = \phi_{[o]}(\Pi): G_2 \rightsquigarrow^*_{\theta_2} \top$ such that $\theta_2 = \theta\ [V]$. Let $V_2 = V \cup \mathcal{V}\text{ar}(l)$. Clearly $\mathcal{V}\text{ar}(G_2) \subseteq V_2$.

An application of Lemma 2.4 to Π_2 results in an NC-refutation $\Pi_1 = \phi_{[d]}(\Pi_2): G_1 \rightsquigarrow^*_{\theta_1} \top$ such that $\theta_1 \leqslant \theta_2$ $[V_2]$. Using $V \subseteq V_2$ we obtain $\theta_1 \leqslant \theta$ $[V]$. Take $\sigma_1 = \epsilon$.

(3) Suppose narrowing is applied to a descendant of e at position 2. This case reduces to the previous one by an appeal to Lemma 2.8.

In all cases we obtain Π_1 from Π by applying one or more transformations $\phi_{[o]}$, $\phi_{[i]}$, $\phi_{[d]}$, $\phi_{[v]}$, $\phi_{[t]}$ together with an additional application of ϕ_{swap} in case (1)(d) and (3). According to Lemmata 2.7 and 2.8 Π_1 has smaller complexity than Π. □

The case analysis in the above proof is summarized in Table 1. The proof of Theorem 2.1 is completed by a straightforward induction argument.

case	LNC-step	transformation(s)
(1)(a)	$\Rightarrow_{[d]}$	$\phi_{[d]}$
(1)(b)	$\Rightarrow_{[t]}$	$\phi_{[t]}$
(1)(c)(i)	$\Rightarrow_{[v]}$	$\phi_{[v]}$
(1)(c)(ii)	$\Rightarrow_{[i]}$	$\phi_{[d]} \circ \phi_{[i]}$
(1)(d)	$\Rightarrow_{[v]}$ or $\Rightarrow_{[i]}$	$\phi_{[v]} \circ \phi_{\text{swap}}$ or $\phi_{[d]} \circ \phi_{[i]} \circ \phi_{\text{swap}}$
(2)	$\Rightarrow_{[o]}$	$\phi_{[d]} \circ \phi_{[o]}$
(3)	$\Rightarrow_{[o]}$	$\phi_{[d]} \circ \phi_{[o]} \circ \phi_{\text{swap}}$

Table 1: Case analysis in the proof of Lemma 2.9.

In the present paper we address the non-determinism of LNC due to the selection of the inference rule. Part of the work has been done in [16] where the *eager variable elimination problem* is addressed. The non-deterministic application of the various inference rules to selected equations causes LNC to generate many redundant derivations. At several places in the literature it is mentioned that this type of redundancy can be greatly reduced by applying the variable elimination rule $[v]$ prior to other applicable inference rules. In [16] it is shown that a restricted version of the eager variable elimination strategy is complete (with respect to $\mathcal{S}_{\text{left}}$) for orthogonal TRSs. The definition of the strategy relies on a notion of *descendants* for LNC-derivations. The selected equation $f(s_1, \ldots, s_n) \simeq t$ in the outermost narrowing rule $[o]$ has the body equation $r \approx t$ as only one-step descendant. In the imitation rule $[i]$ all equations $s_i\theta \approx x_i$ $(1 \leqslant i \leqslant n)$ are one-step descendants of the selected equation $f(s_1, \ldots, s_n) \simeq x$. The selected equation $f(s_1, \ldots, s_n) \approx f(t_1, \ldots, t_n)$ in the decomposition rule $[d]$ has all equations $s_1 \approx t_1, \ldots, s_n \approx t_n$ as one-step descendants. Finally, the selected equations in $[v]$ and $[t]$ have no one-step descendants. One-step descendants of non-selected equations are defined as expected. Descendants are obtained from one-step descendants by reflexivity and transitivity. Observe that every equation in an LNC-derivation descends from either a parameter-passing equation or an equation in the initial goal. An equation of the form $x \simeq t$, with $x \notin \mathcal{V}\text{ar}(t)$, is called *solved*. An LNC-derivation is called *eager* if the variable elimination rule $[v]$ is applied to all selected solved equations that are descendants of a parameter-passing equation. This is equivalent to saying that the variable elimination rule $[v]$ is applied to all selected solved equations that do not descend from an equation in the initial goal.

Theorem 2.10 (Middeldorp, Okui, and Ida [16]) *Let $\mathcal{R}$ be an orthogonal TRS and G a goal. For every normalized solution θ of G there exists an eager* LNC*-refutation $G \Rightarrow^*_{\theta'} \square$ respecting $\mathcal{S}_{\text{left}}$ such that $\theta' \leqslant \theta$ $[\mathcal{V}\text{ar}(G)]$.* $\square$

Consider the orthogonal TRS

$$\mathcal{R} = \left\{ \begin{array}{rcl} f(g(x)) & \to & a \\ b & \to & g(b) \end{array} \right.$$

and the goal $f(b) \approx a$. Of the infinitely many LNC-refutations that respect the selection function $\mathcal{S}_{\text{left}}$ shown in Figure 1, only the leftmost one is eager.

$$\begin{array}{ccccccc}
f(b) \approx a & & & & & & \\
\Downarrow_{[o]} & & & & & & \\
b \approx g(x), a \approx a & & & & & & \\
\Downarrow_{[o]} & & & & & & \\
g(b) \approx g(x), a \approx a & & & & & & \\
\Downarrow_{[d]} & & & & & & \\
b \approx x, a \approx a & \Rightarrow_{[o]} & g(b) \approx x, a \approx a & \Rightarrow_{[i]} & b \approx x_1, a \approx a & \Rightarrow_{[o]} & \cdots \\
\Downarrow_{[v]} & & \Downarrow_{[v]} & & \Downarrow_{[v]} & & \\
a \approx a & & a \approx a & & a \approx a & & \\
\Downarrow_{[d]} & & \Downarrow_{[d]} & & \Downarrow_{[d]} & & \\
\square & & \square & & \square & &
\end{array}$$

Figure 1: The LNC-refutations starting from $f(b) \approx a$ that respect $\mathcal{S}_{\text{left}}$.

We conclude this section by sketching the proof of Theorem 2.10. The proof consists of two parts. First it is shown that, as a consequence of the standardization theorem of Huet and Lévy [9] for orthogonal TRSs, every normalized solution of G is subsumed by a substitution produced by an *outside-in* normal NC-refutation starting from G. An NC-refutation $\Pi\colon G \rightsquigarrow^*_\theta \top$ is said to be outside-in if for every equation $e \in G$ the rewrite sequence $e\theta \to_{p_1, l_1 \to r_1} \cdots \to_{p_n, l_n \to r_n}$ `true` in $\Pi\theta$ satisfies the following condition: for all $1 \leqslant i < n$, if there exists a j with $i < j \leqslant n$ such that $\epsilon < p_j < p_i$ then $p_i \backslash p_j \in \mathcal{P}\text{os}_{\mathcal{F}}(l_j)$ for the least such j. The NC-refutation Π_1 constructed in the proof of Lemma 2.9 is outside-in whenever Π is outside-in:

$$\begin{array}{llcccc}
\forall & \text{normal outside-in} & \Pi & : & G & \rightsquigarrow^+_\theta & \top \\
\exists & & \Psi_1 & : & \Downarrow_{\sigma_1} & & \\
\exists & \text{normal outside-in} & \Pi_1 & : & G_1 & \rightsquigarrow^*_{\theta_1} & \top
\end{array}$$

In the second part of the proof a property $\mathcal{P}$ of equations in the initial goal of NC-refutations is defined. It is shown that parameter-passing equations introduced in the transformation proof of Lemma 2.9 satisfy this property (Lemma 49 in [16]), because we are dealing with outside-in NC-refutations. The property $\mathcal{P}$ is shown to be preserved by LNC-descendants obtained during the transformation proof (Lemma 51 in [16]). Finally, it is shown that Ψ_1 in Lemma 2.9 consists of a $\Rightarrow_{[v]}$-step whenever the selected (leftmost) equation in Π is solved and satisfies the property $\mathcal{P}$ (Lemma 52 in [16]).

Orthogonality in Theorem 2.10 is assumed only for allowing the use of the standardization theorem of Huet and Lévy. Recently we learned that standardization holds for arbitrary left-linear TRSs (Boudol [2], Suzuki [18]). Hence we can strengthen Theorem 2.10 as follows.

Theorem 2.11 *Let $\mathcal{R}$ be a left-linear confluent TRS and G a goal. For every normalized solution θ of G there exists an eager* LNC*-refutation $G \Rightarrow^*_{\theta'} \square$ respecting $\mathcal{S}_{\text{left}}$ such that $\theta' \leqslant \theta$ $[\mathcal{V}\text{ar}(G)]$.* $\square$

Note that left-linear TRSs are in general not confluent.

3 Descendants of Parameter-Passing Equations

In this section we address the remaining non-determinism between the inference rules of LNC for descendants of parameter-passing equations. Consider the TRS

$$\mathcal{R} = \left\{ \begin{array}{rcl} f(a) & \to & f(b) \\ g(f(b)) & \to & c \end{array} \right.$$

and the goal $g(f(x)) \approx c$. Only the outermost narrowing rule $[o]$ is applicable to this goal, resulting in the new goal $f(x) \approx f(b), c \approx c$. To the parameter-passing equation $f(x) \approx f(b)$ we can either apply the decomposition rule $[d]$ followed by the variable elimination rule $[v]$ or apply $[o]$ followed by $[v]$. In the former case we obtain the solution $\{x \mapsto b\}$ and in the latter the solution $\{x \mapsto a\}$. Since these solutions are incomparable (with respect to subsumption modulo $\mathcal{R}$), we cannot eliminate the non-determinism between the outermost narrowing rule $[o]$ and the decomposition rule $[d]$ while retaining completeness.

Observe that $\mathcal{R}$ is not a CS because the defined symbol f occurs in the argument of the left-hand side $g(f(b))$. In this section we will prove that for left-linear confluent CSs all non-determinism between the inferences rules of LNC can be eliminated for descendants of parameter-passing equations.

Lemma 3.1 *Let $\mathcal{R}$ be a left-linear CS and $G \Rightarrow^* G', s \approx t, G''$ an* LNC*-derivation that respects $\mathcal{S}_{\text{left}}$. If the equation $s \approx t$ descends from a parameter-passing equation then*
(1) *$\mathcal{V}\text{ar}(s) \cap \mathcal{V}\text{ar}(t) = \varnothing$,*
(2) *t is a constructor term.*

Proof. Let Π be the given LNC-derivation. We may write Π as

$$G \Rightarrow^* H \Rightarrow_{[o]} H_1, e', H_2 \Rightarrow^* G', s \approx t, G''$$

where e' is the parameter-passing equation of which $s \approx t$ is a descendant. We use induction on the length n of the subderivation $\Pi' \colon H_1, e', H_2 \Rightarrow^* G', s \approx t, G''$ of Π. In order to make the induction work we prove that (1) $\mathcal{V}\mathrm{ar}(G', s) \cap \mathcal{V}\mathrm{ar}(t) = \varnothing$ and (2) t is a *linear* constructor term. If $n = 0$ then $s \approx t$ is a parameter-passing equation. Let $s' \approx t'$ be the leftmost equation in H and let $f(l_1, \ldots, l_n) \to r$ be the rewrite rule employed in $H \Rightarrow_{[o]} G', s \approx t, G''$. So $s' = f(s_1, \ldots, s_n)$ or $t' = f(s_1, \ldots, s_n)$ for some terms $s_1, \ldots, s_n$. Because Π respects $\mathcal{S}_{\mathrm{left}}$ we have $G' = s_1 \approx l_1, \ldots, s_{i-1} \approx l_{i-1}$ and $s \approx t = s_i \approx l_i$ for some $1 \leqslant i \leqslant n$. Freshness of the variables in the rewrite rule yields $\mathcal{V}\mathrm{ar}(s_1, \ldots, s_i) \cap \mathcal{V}\mathrm{ar}(l_i) = \varnothing$. The linearity of l implies that $\mathcal{V}\mathrm{ar}(l_1, \ldots, l_{i-1}) \cap \mathcal{V}\mathrm{ar}(l_i) = \varnothing$. Hence (1) is satisfied. For (2) we simply note that proper subterms of left-hand sides of rewrite rules in left-linear CSs are linear constructor terms. Suppose $n > 0$. Write Π' as

$$H_1, e', H_2 \Rightarrow^* H_1', s' \approx t', H_2' \Rightarrow G', s \approx t, G''$$

such that $s \approx t$ descends from $s' \approx t'$. According to the induction hypothesis we have (3) $\mathcal{V}\mathrm{ar}(H_1', s') \cap \mathcal{V}\mathrm{ar}(t') = \varnothing$ and (4) t' is a linear constructor term. First we consider the case that $s' \approx t'$ is selected in the last step of Π'. This is equivalent to $H_1' = \square$. The last step of Π' cannot be an application of the inference rules $[v]$ and $[t]$ because then $s' \approx t'$ would have no descendants. So we consider the following three cases.

- $[o]$ Because t' is a constructor term, its root symbol is not defined and hence we must have $s' = f(s_1, \ldots, s_n)$, $G' = s_1 \approx l_1, \ldots, s_n \approx l_n$, and $s \approx t = r \approx t'$ for some rewrite rule $f(l_1, \ldots, l_n) \to r$. Because the variables in the rewrite rule are fresh we obtain (1) from (3). As $t = t'$, (2) is an immediate consequence of (4).
- $[i]$ We have $s' \approx t' = f(s_1, \ldots, s_n) \simeq x$, $G' = s_1\theta \approx x_1, \ldots, s_{i-1}\theta \approx x_{i-1}$, and $s \approx t = s_i\theta \approx x_i$ for fresh variables $x_1, \ldots, x_n$ and substitution $\theta = \{x \mapsto f(x_1, \ldots, x_n)\}$. Property (3) implies $x \notin \mathcal{V}\mathrm{ar}(s_1, \ldots, s_n)$ and thus $s_i\theta = s_i$ for all $1 \leqslant i \leqslant n$. Freshness of the variables $x_1, \ldots, x_n$ implies the desired (1). Term t' is a variable, hence (2) is trivially satisfied.
- $[d]$ In this case we have $s' = f(s_1, \ldots, s_n)$, $t' = f(t_1, \ldots, t_n)$, $G' = s_1 \approx t_1, \ldots, s_{i-1} \approx t_{i-1}$, and $s \approx t = s_i \approx t_i$ for some $1 \leqslant i \leqslant n$. The linearity of t' implies that $\mathcal{V}\mathrm{ar}(t_1, \ldots, t_{i-1}) \cap \mathcal{V}\mathrm{ar}(t_i) = \varnothing$. From (3) we infer that $\mathcal{V}\mathrm{ar}(s_1, \ldots, s_i) \cap \mathcal{V}\mathrm{ar}(t_i) = \varnothing$. Hence (1) holds. Because t_i is a (proper) subterm of t', (2) follows from (4).

Next we consider the case that $s' \approx t'$ is not selected in the last step of Π'. Let θ be the produced substitution in the last step of Π'. We have $s \approx t = s'\theta \approx t'\theta$. If the applied inference rule is $[i]$ or $[v]$ then $\mathcal{D}(\theta) \subseteq \mathcal{V}\mathrm{ar}(H_1')$, otherwise $\theta = \epsilon$ and thus also $\mathcal{D}(\theta) \subseteq \mathcal{V}\mathrm{ar}(H_1')$. From (3) we obtain $t = t'$ and thus (2) follows from (4). We either have $\mathcal{V}\mathrm{ar}(G', s') \subseteq \mathcal{V}\mathrm{ar}(H_1', s')$ (when the applied inference rule is $[d]$, $[v]$, or $[t]$) or $\mathcal{V}\mathrm{ar}(G', s') \subseteq \mathcal{V}\mathrm{ar}(H_1', s') \cup V$ for some set V of fresh variables (when the applied inference rule is $[o]$ or $[i]$). Hence in all cases we obtain (1) from (3). $\square$

One might think that Lemma 3.1(1) holds for arbitrary left-linear TRSs. This is not true. Consider for example the TRS

$$\mathcal{R} = \left\{ \begin{array}{rcl} f(g(x)) & \to & a \\ g(x) & \to & h(x, x) \end{array} \right.$$

and the LNC-derivation

$$\begin{array}{rll} f(h(x,x)) \approx a & \Rightarrow_{[o]} & h(x,x) \approx g(x_1), a \approx a \\ & \Rightarrow_{[o]} & x_1 \approx x_2, h(x_2,x_2) \approx h(x,x), a \approx a \\ & \Rightarrow_{[v]} & h(x_2,x_2) \approx h(x,x), a \approx a \\ & \Rightarrow_{[d]} & x_2 \approx x, x_2 \approx x, a \approx a \\ & \Rightarrow_{[v]} & x \approx x, a \approx a. \end{array}$$

The equation $x \approx x$ descends from the parameter-passing equation $h(x,x) \approx g(x_1)$ generated in the first $\Rightarrow_{[o]}$-step.

The first part of Lemma 3.1 implies in particular that we can forget about the occur-check in the variable elimination rule $[v]$ when dealing with descendants of parameter-passing equations. The second part of the lemma implies that the outermost narrowing rule $[o]$ is only applicable to the left-hand side of descendants of parameter-passing equations. Moreover, if $[o]$ can be applied, then the decomposition rule $[d]$ is not applicable. Combining these observations with Theorem 2.10 yields complete determinism in the choice of inference rule for descendants of parameter-passing equations, provided of course we are dealing with orthogonal CSs. Table 2 shows how the inference rule is completely determined by the root symbols of both sides of the selected descendant $s \approx t$ of a parameter-passing equation.

root(s) \ root(t)	$\mathcal{V}$	$\mathcal{F}_\mathcal{C}$	$\mathcal{F}_\mathcal{D}$
$\mathcal{V}$	$[v]$	$[v]$	$\times$
$\mathcal{F}_\mathcal{C}$	$[v]$	$[d]$	$\times$
$\mathcal{F}_\mathcal{D}$	$[v]$	$[o]$	$\times$

Table 2: Selection of inference rule for descendant $s \approx t$ of parameter-passing equation.

The case $\mathrm{root}(t) \in \mathcal{F}_\mathcal{D}$ is impossible according to Lemma 3.1(2). Observe that the imitation rule $[i]$ is never applied to descendants of parameter-passing equations. This is because if $[i]$ is applicable then, according to Lemma 3.1(1), so is the variable elimination rule $[v]$ and by Theorem 2.10 the latter is given precedence.

Incorporating the above observations into LNC gives rise to the calculus LNC_{pp} whose inference rules are presented below. To distinguish descendants of parameter-passing equations from descendants of initial equations, we use $\triangleright$ rather than $\approx$ to denote the former. The first group of rules are designed for descendants of initial equations. The only difference with the inference rules of LNC specialized to $\mathcal{S}_{\text{left}}$ is that in the outermost narrowing rule we mark the parameter-passing equations using $\triangleright$:

$[o]$ *outermost narrowing*

$$\frac{f(s_1,\ldots,s_n) \simeq t, G}{s_1 \triangleright l_1, \ldots, s_n \triangleright l_n, r \approx t, G}$$

if there exists a fresh variant $f(l_1,\ldots,l_n) \to r$ of a rewrite rule in $\mathcal{R}$,

$[i]$ *imitation*

$$\frac{f(s_1,\ldots,s_n) \simeq x, G}{(s_1 \approx x_1,\ldots,s_n \approx x_n, G)\theta}$$

if $\theta = \{x \mapsto f(x_1,\ldots,x_n)\}$ with $x_1,\ldots,x_n$ fresh variables,

$[d]$ *decomposition*

$$\frac{f(s_1,\ldots,s_n) \approx f(t_1,\ldots,t_n), G}{s_1 \approx t_1,\ldots,s_n \approx t_n, G}$$

$[v]$ *variable elimination*

$$\frac{s \simeq x, G}{G\theta}$$

if $x \notin \mathcal{V}\mathrm{ar}(s)$ and $\theta = \{x \mapsto s\}$,

$[t]$ *removal of trivial equations*

$$\frac{x \approx x, G}{G}$$

The second group of inference rules of LNC_{pp} deals with descendants of parameter-passing equations:

$[o]_\rhd$ *outermost narrowing for parameter-passing equations*

$$\frac{f(s_1,\ldots,s_n) \rhd t, G}{s_1 \rhd l_1,\ldots,s_n \rhd l_n, r \rhd t, G}\ t \notin \mathcal{V}$$

if there exists a fresh variant $f(l_1,\ldots,l_n) \to r$ of a rewrite rule in $\mathcal{R}$,

$[d]_\rhd$ *decomposition for parameter-passing equations*

$$\frac{f(s_1,\ldots,s_n) \rhd f(t_1,\ldots,t_n), G}{s_1 \rhd t_1,\ldots,s_n \rhd t_n, G}\ f \in \mathcal{F}_\mathcal{C}$$

$[v]_\rhd$ *variable elimination for parameter-passing equations*

$$\frac{s \rhd x, G}{G\theta} \quad \text{and} \quad \frac{x \rhd s, G}{G\theta}\ s \notin \mathcal{V}$$

if $\theta = \{x \mapsto s\}$.

Theorem 3.2 *Let $\mathcal{R}$ be a left-linear confluent CS and G a goal. For every normalized solution θ of G there exists an LNC_{pp}-refutation $G \Rightarrow^*_{\theta'} \square$ such that $\theta' \leqslant \theta$ $[\mathcal{V}\mathrm{ar}(G)]$.* $\square$

In the next section we turn our attention to descendants of equations in the initial goal.

4 Descendants of Initial Equations

Whereas the restriction to left-linear confluent CSs is sufficient to remove all non-determinism in the choice of inference rule for descendants of parameter-passing equations, this is not the case for descendants of initial equations. Consider for example the (left-linear confluent) CS

$$\mathcal{R} = \left\{\ f(a) \quad \to \quad f(b)\right.$$

and the goal $f(x) \approx f(b)$. This goal has the two incomparable solutions $\{x \mapsto a\}$ and $\{x \mapsto b\}$. The first solution can only be obtained if we apply the outermost narrowing rule [o]. The second solution requires an application of the decomposition rule [d].

There is also non-determinism in the outermost narrowing rule [o] itself. Consider for example the CS

$$\mathcal{R} = \left\{ \begin{array}{lcl} f(a) & \to & g(a) \\ g(b) & \to & f(b) \end{array} \right.$$

and the goal $f(x) \approx g(x)$. The solution $\{x \mapsto a\}$ can only be obtained if we apply the outermost narrowing rule [o] to the left-hand side of $f(x) \approx g(x)$, but in order to obtain the incomparable solution $\{x \mapsto b\}$ it is essential that we apply [o] to its right-hand side.

In functional logic programming it is customary to consider two expressions to be equal if and only if they reduce to the same ground constructor normal form. This so-called *strict equality* is adopted to model non-termination correctly ([3, 1]). If we interpret $\approx$ as strict equality then the non-determinism in the above examples disappears: neither $\{x \mapsto a\}$ nor $\{x \mapsto b\}$ are (strict) solutions of the goals $f(x) \approx f(b)$ and $f(x) \approx g(x)$.

Definition 4.1 Let $\mathcal{R}$ be a TRS and G a goal. A substitution θ is said to be a *strict solution* of G if for every equation $s \approx t$ in G there exists a ground constructor term u such that $s\theta \to_{\mathcal{R}}^* u$ and $t\theta \to_{\mathcal{R}}^* u$.

In this section we will prove that for confluent TRSs all non-determinism between the inference rules of LNC can be eliminated for descendants of initial equations with strict semantics. We would like to stress that we don't need the restriction to left-linear CSs here. The structure of the proof is similar to that of the second half of the proof of Theorem 2.10: we define a property of initial equations in NC-refutations (Definition 4.2), we show that initial equations of the LNC-refutation obtained in the transformation proof of Theorem 2.1 satisfy this property (Lemma 4.3), we show that the property is preserved by LNC-descendants obtained during the transformation in the proof of Theorem 2.1 (Lemma 4.4), and finally we show that there is no non-determinism between the inference rules for selected equations that have the property (Lemma 4.5).

Definition 4.2 Let $\Pi\colon G \rightsquigarrow_{\theta}^* \top$ be an NC-refutation and $e \in G$. We say that e is *strictly solved* in Π if the sequence starting from $e\theta$ in the corresponding rewrite sequence $\Pi\theta\colon G\theta \to^* \top$ is of the form $e\theta \to^* t \approx t \to \mathtt{true}$ with t a (not necessarily ground) constructor term.

Lemma 4.3 *Let G be a goal and θ a substitution such that $\theta\restriction_{\mathcal{V}\mathrm{ar}(G)}$ is normalized. If θ is a strict solution of G then there exists a normal* NC*-refutation $\Pi\colon G \rightsquigarrow_{\theta'}^* \top$ such that $\theta' \leqslant \theta\ [\mathcal{V}\mathrm{ar}(G)]$ and every equation in G is strictly solved in Π.*

Proof. By assumption there exists a rewrite sequence $G\theta \to^* \top$ with the property that for every $e \in G$ the corresponding subsequence $e\theta \to^* \mathtt{true}$ is of the form $e\theta \to^* t \approx t \to \mathtt{true}$ with $t \in \mathcal{T}(\mathcal{F}_{\mathcal{C}})$. Because steps in different equations of $G\theta$ are independent of each other, we may assume that this rewrite sequence respects $\mathcal{S}_{\mathrm{left}}$. An application of the lifting lemma for NC results in an NC-refutation $\Pi\colon G \rightsquigarrow_{\theta'}^* \top$ respecting $\mathcal{S}_{\mathrm{left}}$ such that $\theta' \leqslant \theta\ [\mathcal{V}\mathrm{ar}(G)]$. Because

$\theta\lceil_{\mathcal{V}\mathrm{ar}(G)}$ is normalized, so is $\theta'\lceil_{\mathcal{V}\mathrm{ar}(G)}$. It follows that Π is a normal NC-refutation. It remains to show that every equation $e \in G$ is strictly solved in Π. Because $G\theta \to^* \top$ is an instance of $\Pi\theta'$, the rewrite sequence starting from $e\theta'$ in $\Pi\theta'$ must be of the form $e\theta' \to^* t' \approx t' \to \mathtt{true}$ with $t' \leqslant t$ for some $t \in \mathcal{T}(\mathcal{F}_\mathcal{C})$. This implies that $t' \in \mathcal{T}(\mathcal{F}_\mathcal{C}, \mathcal{V})$. □

In the following two lemmata, Π_1 and Ψ_1 refer to the NC-refutation and the LNC-step constructed in Lemma 2.9.

Lemma 4.4 *Suppose $\Pi: G \rightsquigarrow^+ \top$ is a normal NC-refutation and let e be an equation in G that is strictly solved in Π. If e' is a descendant of e with respect to Ψ_1 then e' is strictly solved in Π_1.*

Proof. First we consider the case that $e = s \approx t$ is the selected equation in G. Consider the case analysis in the proof of Lemma 2.9. In cases (1)(b) and (1)(c)(i) there is nothing to show as e has no descendants in G_1. In case (1)(a) we have $s = f(s_1, \ldots, s_n)$, $t = f(t_1, \ldots, t_n)$, and $e' = s_i \approx t_i$ for some $1 \leqslant i \leqslant n$. Because e is strictly solved in Π, the equation $e\theta$ is rewritten in $\Pi\theta$ to an equation of the form $f(u_1, \ldots, u_n) \approx f(u_1, \ldots, u_n)$ with $f(u_1, \ldots, u_n) \in \mathcal{T}(\mathcal{F}_\mathcal{C}, \mathcal{V})$. Hence, by construction of $\phi_{[d]}$, in $\Pi_1\theta_1$ the equation $e'\theta_1$ is rewritten to $u_i \approx u_i$. Since subterms of constructor terms are constructor terms, we conclude that e' is strictly solved in Π_1. In case (1)(c)(ii) we have $s = f(s_1, \ldots, s_n)$, $t \in \mathcal{V}$, and $e' = s_i \approx x_i$ for some $1 \leqslant i \leqslant n$. From $\phi_{[i]}(\Pi)\theta_2 = \Pi\theta$ we infer that $e\sigma_1$ is strictly solved in $\Pi_2 = \phi_{[i]}(\Pi)$. By the same reasoning as in case (1)(a)—note that e' is a descendant of $e\sigma_1$ with respect to the LNC-step $G_2 \Rightarrow_{[d]} G_1$—we conclude that e' is strictly solved in $\Pi_1 = \phi_{[d]}(\Pi_2)$. In case (2) the rewrite sequences starting from $e\theta$ in $\Pi\theta$ and $(r \approx t)\theta_2$ in $\phi_{[o]}(\Pi)\theta_2$ can be written as

$$\begin{array}{ccccc} (s \approx t)\theta & \to^* & l\theta' \approx t' & & \\ & & \downarrow & & \\ (r \approx t)\theta_2 & \to^* & r\theta' \approx t' & \to^* & \mathtt{true} \end{array}$$

for some substitution θ' and term t'. Because by assumption $s \approx t$ is strictly solved in Π, the sequence from $r\theta' \approx t'$ to $\mathtt{true}$ has the form $r\theta' \approx t' \to^* u \approx u \to \mathtt{true}$ with $u \in \mathcal{T}(\mathcal{F}_\mathcal{C}, \mathcal{V})$. Hence $r \approx t$ is strictly solved in $\Pi_2 = \phi_{[o]}(\Pi)$. Since the rewrite sequence starting from $(r \approx t)\theta_2$ in $\phi_{[o]}(\Pi)\theta_2$ is an instance* of the rewrite sequence starting from $(r \approx t)\theta_1$ in $\phi_{[d]}(\Pi_2)\theta_1$, the equation $r \approx t$ is also strictly solved in $\Pi_1 = \phi_{[d]}(\Pi_2)$. Observe that $r \approx t$ is the (only) descendant of e with respect to $\Psi_1: G \Rightarrow_{[o]} G_1$. Finally, case (3) reduces to case (2) because strict solvability is trivially preserved by ϕ_{swap}.

Next we consider the case that $e \in G$ is not selected in the first step of Π. The (unique) descendant $e' = e\sigma_1 \in G_1$ of e inherits strict solvability from e as the rewrite sequence starting from $e\theta$ in $\Pi\theta$ is an instance* of the rewrite sequence from $e'\theta_1$ in $\Pi_1\theta_1$. □

It is interesting to note that parameter-passing equations generated in Lemma 2.9 are in

*Modulo an inconsequential reordering of rewrite steps due to the transformations ϕ_1 and ϕ_2 which are part of $\phi_{[o]}$ and $\phi_{[d]}$, cf. the text following Lemma 2.6.

general not strictly solved. Consider for example the TRS

$$\mathcal{R} = \begin{cases} f(x) & \to & a \\ b & \to & b \end{cases}$$

and the normal NC-refutation Π: $f(b) \approx a \rightsquigarrow a \approx a \rightsquigarrow \texttt{true}$. The equation $f(b) \approx a$ is strictly solved in Π because a is a constructor. Since in the first step of Π narrowing is applied at position 1, we obtain

$$\begin{array}{lcccccc} \Pi & : & f(b) \approx a & \rightsquigarrow & a \approx a & \rightsquigarrow & \texttt{true} \\ \Psi_1 & : & \Downarrow_{[o]} & & & & \\ \Pi_1 & : & b \approx x, a \approx a & \rightsquigarrow & \texttt{true}, a \approx a & \rightsquigarrow & \top \end{array}$$

from Lemma 2.9. The parameter-passing equation $b \approx x$ is not strictly solved in Π_1 as b is a defined symbol.

Lemma 4.5 *Let* Π: $G \rightsquigarrow^+ \top$ *be a normal* NC*-refutation and suppose that the selected (leftmost) equation* $e = s \approx t$ *is strictly solved in* Π.

(1) Ψ_1 *consists of an* $\Rightarrow_{[o]}$*-step if and only if* $\mathrm{root}(s) \in \mathcal{F}_\mathcal{D}$ *or* $\mathrm{root}(t) \in \mathcal{F}_\mathcal{D}$. *Moreover, if* $[o]$ *is applied to the right-hand side* t *of* e *then* $\mathrm{root}(s) \notin \mathcal{F}_\mathcal{D}$.

(2) Ψ_1 *consists of a* $\Rightarrow_{[v]}$*-step if and only if* e *is solved and* $s, t \in \mathcal{T}(\mathcal{F}_\mathcal{C}, \mathcal{V})$.

Proof. The following observations follow easily from the fact that e is strictly solved in Π: if $\mathrm{root}(s) \in \mathcal{F}_\mathcal{D}$ then narrowing is applied to a descendant of e in Π at position 1, if $\mathrm{root}(t) \in \mathcal{F}_\mathcal{D}$ then narrowing is applied to a descendant of e in Π at position 2, and if $s \notin \mathcal{T}(\mathcal{F}_\mathcal{C}, \mathcal{V})$ or $t \notin \mathcal{T}(\mathcal{F}_\mathcal{C}, \mathcal{V})$ then narrowing is not applied at the root position in the first step of Π.

(1) According to the proof of Lemma 2.9 Ψ_1 consists of an $\Rightarrow_{[o]}$-step only in cases (2) and (3). In case (2) we have $\mathrm{root}(s) \in \mathcal{F}_\mathcal{D}$ and in case (3) $\mathrm{root}(t) \in \mathcal{F}_\mathcal{D}$. Conversely, if $\mathrm{root}(s) \in \mathcal{F}_\mathcal{D}$ then, according to one of the above observations, narrowing is applied to a descendant of e in Π at position 1. This means that we are in case (2) of the proof of Lemma 2.9, and thus Ψ_1 consists of an $\Rightarrow_{[o]}$-step. The case $\mathrm{root}(t) \in \mathcal{F}_\mathcal{D}$ is similar. Since case (2) precedes case (3) in the proof of Lemma 2.9, $[o]$ is applied to the right-hand side t of e only if case (2) does not apply, so $\mathrm{root}(s) \notin \mathcal{F}_\mathcal{D}$.

(2) According to the proof of Lemma 2.9 Ψ_1 consists of a $\Rightarrow_{[v]}$-step only in cases (1)(c)(i) and (1)(d). In both cases narrowing is applied at the root position in the first step of Π. This is only possible if the equation e is solved and, according to one of the above observations, $s, t \in \mathcal{T}(\mathcal{F}_\mathcal{C}, \mathcal{V})$. Conversely, assume that e is solved and $s, t \in \mathcal{T}(\mathcal{F}_\mathcal{C}, \mathcal{V})$. This is only possible if narrowing is applied at the root position in the first step of Π, hence we are in case (1)(a), (1)(b), (1)(c)(i), or (1)(d) of the proof of Lemma 2.9. In cases (1)(a) and (1)(b) the equation e is not solved. In the other two cases Ψ_1 indeed consists of a $\Rightarrow_{[v]}$-step.

□

Lemmata 4.3–4.5 imply that for completeness (with respect to strict solutions) it suffices to apply a single inference rule of LNC to each goal whose selected leftmost equation descends from an equation in the initial goal. According to Lemma 4.5 the unique inference rule is completely determined by the form of the selected equation. Table 3 shows how the inference rule depends on the selected strictly solved equation $s \approx t$.

root(s) \ root(t)	$\mathcal{V}$	$\mathcal{F}_\mathcal{C}$	$\mathcal{F}_\mathcal{D}$
$\mathcal{V}$	$[v]/[t]^a$	$[v]/[i]^b$	$[o]$
$\mathcal{F}_\mathcal{C}$	$[v]/[i]^c$	$[d]$	$[o]$
$\mathcal{F}_\mathcal{D}$	$[o]$	$[o]$	$[o]^d$

[a] $[t]$ is applied if and only if $s = t$.
[b] $[v]$ is applied if and only if $t \in \mathcal{T}(\mathcal{F}_\mathcal{C}, \mathcal{V})$ and $s \notin \mathcal{V}\text{ar}(t)$.
[c] $[v]$ is applied if and only if $s \in \mathcal{T}(\mathcal{F}_\mathcal{C}, \mathcal{V})$ and $t \notin \mathcal{V}\text{ar}(s)$.
[d] $[o]$ is applied to the left-hand side s.

Table 3: Selection of inference rule for strictly solved equation $s \approx t$.

It is interesting to note that the resulting strategy is almost the opposite of eager variable elimination: conflicts between the variable elimination rule $[v]$ and the outermost narrowing rule $[o]$ are always resolved by giving preference to the latter and often the imitation rule $[i]$ is selected even if $[v]$ is applicable.

Incorporating the above observations into LNC gives rise to the calculus LNC$_\text{i}$ whose inference rules are presented below. We use $\equiv$ to denote descendants of initial equations. The first group of rules are designed for descendants of initial equations:

$[o]_\equiv$ *outermost narrowing for initial equations*

$$\frac{f(s_1,\ldots,s_n) \equiv t, G}{s_1 \approx t_1,\ldots,s_n \approx l_n, r \equiv t, G} \quad\text{and}\quad \frac{t \equiv f(s_1,\ldots,s_n), G}{s_1 \approx l_1,\ldots,s_n \approx l_n, r \equiv t, G}\ \text{root}(t) \notin \mathcal{F}_\mathcal{D}$$

if there exists a fresh variant $f(l_1,\ldots,l_n) \to r$ of a rewrite rule in $\mathcal{R}$,

$[i]_\equiv$ *imitation for initial equations*

$$\frac{f(s_1,\ldots,s_n) \equiv x, G}{(s_1 \equiv x_1,\ldots,s_n \equiv x_n, G)\theta}\ f \in \mathcal{F}_\mathcal{C} \quad\text{and}\quad \frac{x \equiv f(s_1,\ldots,s_n), G}{(s_1 \equiv x_1,\ldots,s_n \equiv x_n, G)\theta}\ f \in \mathcal{F}_\mathcal{C}$$

if $x \in \mathcal{V}\text{ar}(f(s_1,\ldots,s_n))$ or $f(s_1,\ldots,s_n) \notin \mathcal{T}(\mathcal{F}_\mathcal{C}, \mathcal{V})$ and $\theta = \{x \mapsto f(x_1,\ldots,x_n)\}$ with $x_1,\ldots,x_n$ fresh variables,

$[d]_\equiv$ *decomposition for initial equations*

$$\frac{f(s_1,\ldots,s_n) \equiv f(t_1,\ldots,t_n), G}{s_1 \equiv t_1,\ldots,s_n \equiv t_n, G}\ f \in \mathcal{F}_\mathcal{C}$$

$[v]_\equiv$ *variable elimination for initial equations*

$$\frac{s \equiv x, G}{G\theta} \quad\text{and}\quad \frac{x \equiv s, G}{G\theta}\ s \notin \mathcal{V}$$

if $x \notin \mathcal{V}\text{ar}(s)$, $s \in \mathcal{T}(\mathcal{F}_\mathcal{C}, \mathcal{V})$, and $\theta = \{x \mapsto s\}$,

$[t]_\equiv$ *removal of trivial equations for initial equations*

$$\frac{x \equiv x, G}{G}$$

The second group of inference rules of $\textsc{lnc}_\mathrm{i}$ deals with descendants of parameter-passing equations. They are the same as the inference rules of $\textsc{lnc}$ with respect to $\mathcal{S}_\mathrm{left}$:

$[o]$ *outermost narrowing*

$$\frac{f(s_1, \dots, s_n) \simeq t, G}{s_1 \approx l_1, \dots, s_n \approx l_n, r \approx t, G}$$

if there exists a fresh variant $f(l_1, \dots, l_n) \to r$ of a rewrite rule in $\mathcal{R}$,

$[i]$ *imitation*

$$\frac{f(s_1, \dots, s_n) \simeq x, G}{(s_1 \approx x_1, \dots, s_n \approx x_n, G)\theta}$$

if $\theta = \{x \mapsto f(x_1, \dots, x_n)\}$ with $x_1, \dots, x_n$ fresh variables,

$[d]$ *decomposition*

$$\frac{f(s_1, \dots, s_n) \approx f(t_1, \dots, t_n), G}{s_1 \approx t_1, \dots, s_n \approx t_n, G}$$

$[v]$ *variable elimination*

$$\frac{s \simeq x, G}{G\theta}$$

if $x \notin \mathcal{V}\mathrm{ar}(s)$ and $\theta = \{x \mapsto s\}$,

$[t]$ *removal of trivial equations*

$$\frac{x \approx x, G}{G}$$

Theorem 4.6 *Let $\mathcal{R}$ be a confluent TRS and G a goal. For every strict and normalized solution θ of G there exists an $\textsc{lnc}_\mathrm{i}$-refutation $G \Rightarrow^*_{\theta'} \square$ such that $\theta' \leqslant \theta$ $[\mathcal{V}\mathrm{ar}(G)]$.* $\square$

Note that in the above theorem G is assumed to be a sequence $s_1 \equiv t_1, \dots, s_n \equiv t_n$ of initial equations. The calculus $\textsc{lnc}_\mathrm{i}$ can be slightly improved by replacing the selected equation $f(s_1, \dots, s_n) \simeq t$ in the outermost narrowing rule $[o]$ (for parameter-passing equations) by $f(s_1, \dots, s_n) \approx t$, i.e., there is no need to apply outermost narrowing to right-hand sides of descendants of parameter-passing equations. (The reason for this is that the proof of Lemma 49 in [16] reveals that the first part of property $\mathcal{P}$ holds for arbitrary normal $\textsc{nc}$-refutations.)

5 The Final Calculus

The results of the preceding two sections give rise to the *deterministic* lazy narrowing calculus, $\textsc{lnc}_\mathrm{d}$ for short, whose inference rules are presented below. The first group of rules are designed for descendants of initial equations:

$[o]_\equiv$ *outermost narrowing for initial equations*

$$\frac{f(s_1, \dots, s_n) \approx t, G}{s_1 \rhd l_1, \dots, s_n \rhd l_n, r \approx t, G} \quad \text{and} \quad \frac{t \approx f(s_1, \dots, s_n), G}{s_1 \rhd l_1, \dots, s_n \rhd l_n, r \approx t, G} \ \mathrm{root}(t) \notin \mathcal{F}_\mathcal{D}$$

if there exists a fresh variant $f(l_1, \dots, l_n) \to r$ of a rewrite rule in $\mathcal{R}$,

$[i]_{\equiv}$ *imitation for initial equations*

$$\frac{f(s_1,\dots,s_n)\approx x, G}{(s_1\approx x_1,\dots,s_n\approx x_n, G)\theta}\; f\in\mathcal{F}_{\mathcal{C}} \quad\text{and}\quad \frac{x\approx f(s_1,\dots,s_n), G}{(s_1\approx x_1,\dots,s_n\approx x_n, G)\theta}\; f\in\mathcal{F}_{\mathcal{C}}$$

if $x\in\mathcal{V}\mathrm{ar}(f(s_1,\dots,s_n))$ or $f(s_1,\dots,s_n)\notin\mathcal{T}(\mathcal{F}_{\mathcal{C}},\mathcal{V})$ and $\theta=\{x\mapsto f(x_1,\dots,x_n)\}$ with $x_1,\dots,x_n$ fresh variables,

$[d]_{\equiv}$ *decomposition for initial equations*

$$\frac{f(s_1,\dots,s_n)\approx f(t_1,\dots,t_n), G}{s_1\approx t_1,\dots,s_n\approx t_n, G}\; f\in\mathcal{F}_{\mathcal{C}}$$

$[v]_{\equiv}$ *variable elimination for initial equations*

$$\frac{s\approx x, G}{G\theta}\quad\text{and}\quad\frac{x\approx s, G}{G\theta}\; s\notin\mathcal{V}$$

if $x\notin\mathcal{V}\mathrm{ar}(s)$, $s\in\mathcal{T}(\mathcal{F}_{\mathcal{C}},\mathcal{V})$, and $\theta=\{x\mapsto s\}$,

$[t]_{\equiv}$ *removal of trivial equations for initial equations*

$$\frac{x\approx x, G}{G}$$

The second group of inference rules of LNC_d deals with descendants of parameter-passing equations:

$[o]_{\triangleright}$ *outermost narrowing for parameter-passing equations*

$$\frac{f(s_1,\dots,s_n)\triangleright t, G}{s_1\triangleright l_1,\dots,s_n\triangleright l_n, r\triangleright t, G}\; t\notin\mathcal{V}$$

if there exists a fresh variant $f(l_1,\dots,l_n)\to r$ of a rewrite rule in $\mathcal{R}$,

$[d]_{\triangleright}$ *decomposition for parameter-passing equations*

$$\frac{f(s_1,\dots,s_n)\triangleright f(t_1,\dots,t_n), G}{s_1\triangleright t_1,\dots,s_n\triangleright t_n, G}\; f\in\mathcal{F}_{\mathcal{C}}$$

$[v]_{\triangleright}$ *variable elimination for parameter-passing equations*

$$\frac{s\triangleright x, G}{G\theta}\quad\text{and}\quad\frac{x\triangleright s, G}{G\theta}\; s\notin\mathcal{V}$$

if $\theta=\{x\mapsto s\}$.

Observe that there is no non-determinism between the inference rules of LNC_d. This does not imply that every goal admits at most one LNC_d-refutation because the outermost narrowing rules $[o]$ and $[o]_{\triangleright}$ depend on the choice of the rewrite rule $f(l_1,\dots,l_n)\to r$. This latter non-determinism cannot be avoided because goals in general have different incomparable (strict) solutions.

Combining the results of the preceding sections yields the following completeness theorem.

Theorem 5.1 *Let $\mathcal{R}$ be a left-linear confluent CS and G a goal. For every strict and normalized solution θ of G there exists an* LNC$_\mathrm{d}$*-refutation $G \Rightarrow^*_{\theta'} \square$ such that $\theta' \leqslant \theta$ $[\mathcal{V}\mathrm{ar}(G)]$.*

Proof. This follows easily from the results of the preceding sections together with the observation that we may assume that the normal NC-refutation Π constructed in the proof of Lemma 4.3 is outside-in. □

We would like to emphasize that the results in Sections 3 and 4 are independent of each other. So if we are faced with a confluent TRS that is not a left-linear CS, we can still remove all non-determinism due to the selection of the inference rule for descendants of initial equations, provided we adopt strict semantics. Only if both restrictions (left-linear CS and strict semantics) are fulfilled, we can use the fully deterministic calculus LNC$_\mathrm{d}$.

Ida and Nakahara study in [11] a lazy narrowing calculus OINC which is similar to LNC with the leftmost selection function. The essential difference is that OINC lacks the imitation inference rule. Because of this OINC is complete only for orthogonal TRSs and goals that satisfy the restriction that the right-hand side of every equation is a ground normal form. Ida and Nakahara [11] incorporated strict semantics into OINC, resulting in the calculus S-OINC which is very similar to our LNC$_\mathrm{d}$ in that descendants of parameter-passing equations and descendants of initial equations are distinguished. However, S-OINC is not deterministic in the choice of inference rule, neither for descendants of parameter-passing equations nor for descendants of initial equations. We have showed in this paper that non-determinism in the choice of inference rule for descendants of initial equations can be achieved for *arbitrary* confluent TRSs with strict semantics. Moreover, we feel that the imitation rule for descendants of initial equation in S-OINC appears out of the blue as it is not present in the calculus OINC.

González-Moreno *et al.* [4] present the constructor-based lazy narrowing calculus CLNC. Although the underlying theory is different—a rewriting logic for modelling non-deterministic functions by non-confluent systems (CRWL [4]) versus standard equational logic—CLNC and LNC$_\mathrm{d}$ have many features in common. The inference rules of CLNC for both descendants of parameter-passing equations (which are called approximation conditions in [4]) and descendants of initial equations (joinability conditions in [4]) are fully deterministic but, whereas for descendants of parameter-passing equations we always give preference to the variable elimination rule $[v]$, in CLNC conflicts between $[v]$ (OB and IB in [4]), the outermost narrowing rule $[o]$ (NR2) and the imitation rule $[i]$ (IIM) are resolved by giving preference to the latter two in case the variable is *demanded* ([3]). (Giving unrestricted preference to $[o]$ and $[i]$ would result in an incomplete calculus, as can be seen from Figure 1.) However, unlike LNC$_\mathrm{d}$ the calculus CLNC is *not* strongly complete (nor complete with respect to leftmost selection) because the selection of an equation in a goal may have to be postponed before an inference rule can be applied to it. Nevertheless, in [4] it is shown that the choice of equations in goals is don't care non-deterministic. (This latter property is called strong completeness in [4].)

We conclude this section by comparing the performance of LNC with leftmost selection and LNC$_\mathrm{d}$ on a small number of examples. All numbers below are based on an implementation in

Sictus Prolog 2.1#6 running on a Sun Ultra1. Consider the orthogonal CS

$$
\begin{array}{rcllcllcl}
\mathtt{0} + y & \to & y & \mathtt{append}([\,], z) & \to & z & \mathtt{if}(\mathtt{true}, x, y) & \to & x \\
\mathtt{S}(x) + y & \to & \mathtt{S}(x+y) & \mathtt{append}([x|y], z) & \to & [x|\mathtt{append}(y, z)] & \mathtt{if}(\mathtt{false}, x, y) & \to & y \\
\mathtt{0} \leqslant y & \to & \mathtt{true} & \mathtt{qsort}([\,]) & \to & [\,] & & & \\
\mathtt{S}(x) \leqslant \mathtt{0} & \to & \mathtt{false} & \mathtt{qsort}([x|y]) & \to & \mathtt{qsort}'(x, \mathtt{split}(x, y, [\,], [\,])) & & & \\
\mathtt{S}(x) \leqslant \mathtt{S}(y) & \to & x \leqslant y & \mathtt{qsort}'(x, \langle y, z\rangle) & \to & \mathtt{append}(\mathtt{qsort}(y), [x|\mathtt{qsort}(z)]) & & &
\end{array}
$$

$$
\begin{array}{lcl}
\mathtt{split}(x, [\,], y, z) & \to & \langle y, z\rangle \\
\mathtt{split}(x, [y'|z'], y, z) & \to & \mathtt{if}(y' \leqslant x, \mathtt{split}(x, z', [y'|y], z), \mathtt{split}(x, z', y, [y'|z]))
\end{array}
$$

First consider the goals $x + y \approx \mathtt{S}^n(\mathtt{0})$ for $n \geqslant 0$. Table 4 shows for various values of n the size of the (finite) search tree and the runtime in seconds for both calculi. Next consider the goal $x + y \approx z$. This goal has infinitely many incomparable (strict) solutions. The numbers in Table 5 correspond to finite approximations of the infinite search tree. Finally, consider the goal $\mathtt{qsort}([\mathtt{S}(\mathtt{0}),\ \mathtt{0}]) \approx x$ which admits the unique solution $\{x \mapsto [\mathtt{0},\ \mathtt{S}(\mathtt{0})]\}$. It takes 0.61 seconds to compute the (finite) LNC_d search tree for this goal. The total number of nodes is 120 and the depth of the tree is 64. Table 5 shows, for various depths, the performance of LNC and LNC_d.

	LNC		LNC_d	
n	nodes	seconds	nodes	seconds
0	27	0.073	8	0.020
1	197	0.54	16	0.044
2	1225	3.4	24	0.069
3	7401	21	32	0.096
4	44465	130	40	0.12
5	266857	750	48	0.15

Table 4: The goals $x + y \approx \mathtt{S}^n(\mathtt{0})$.

Acknowledgements

We are grateful to Koji Nakagawa for implementing LNC and LNC_d and performing the experiments described above.

References

[1] S. Antoy, R. Echahed, and M. Hanus, *A Needed Narrowing Strategy*, Proceedings of the 21st ACM Symposium on Principles of Programming Languages, Portland, pp. 268–279, 1994.

	LNC		LNC_d		LNC		LNC_d	
depth	nodes	seconds	nodes	seconds	nodes	seconds	nodes	seconds
1	5	0.012	3	0.006	5	0.013	3	0.007
3	23	0.063	7	0.018	12	0.037	5	0.015
5	55	0.16	11	0.030	35	0.12	7	0.023
7	131	0.38	15	0.043	107	0.39	10	0.041
9	307	0.88	19	0.056	339	1.5	13	0.061
11	635	1.9	23	0.068	1083	5.2	15	0.075
13	1499	4.4	27	0.080	2835	14	17	0.087
15	3019	9.0	31	0.093	8035	40	25	0.14
17	6987	21	35	0.11	26515	150	29	0.17
19	14571	43	39	0.12	80275	560	31	0.18
	$x + y \approx z$				$\texttt{qsort([S(0), 0])} \approx x$			

Table 5: The goals $x + y \approx z$ and $\texttt{qsort([S(0), 0])} \approx x$.

[2] G. Boudol, *Computational Semantics of Term Rewriting Systems*, in: Algebraic Methods in Semantics (eds. M. Nivat and J.C. Reynolds), Cambridge University Press, pp. 169–236, 1985.

[3] E. Giovannetti, G. Levi, C. Moiso, and C. Palamidessi, *Kernel-LEAF: A Logic plus Functional Language*, Journal of Computer and System Sciences **42**(2), pp. 139–185, 1991.

[4] J.C. González-Moreno, M.T. Hortalá-González, F.J. López-Fragues, and M. Rodrígue-Artalejo, *A Rewriting Logic for Declarative Programming*, Proceedings of the 6th European Symposium on Programming, Linköping, Lecture Notes in Computer Science **1058**, pp. 156–172, 1996.

[5] M. Hanus, *Lazy Unification with Simplification*, Proceedings of the 5th European Symposium on Programming, Edinburgh, Lecture Notes in Computer Science **788**, pp. 272–286, 1994.

[6] M. Hanus, *The Integration of Functions into Logic Programming: From Theory to Practice*, Journal of Logic Programming **19 & 20**, pp. 583–628, 1994.

[7] S. Hölldobler, *A Unification Algorithm for Confluent Theories*, Proceedings 14th International Colloquium on Automata, Languages and Programming, Lecture Notes in Computer Science **267**, pp. 31–41, 1987.

[8] S. Hölldobler, *Foundations of Equational Logic Programming*, Lecture Notes in Artificial Intelligence **353**, 1989.

[9] G. Huet and J.-J. Lévy, *Computations in Orthogonal Rewriting Systems, I and II*, in: Computational Logic, Essays in Honor of Alan Robinson (eds. J.-L. Lassez and G. Plotkin), The MIT Press, pp. 396–443, 1991.

[10] J.-M. Hullot, *Canonical Forms and Unification*, Proceedings of the 5th Conference on Automated Deduction, Lecture Notes in Computer Science **87**, pp. 318–334, 1980.

[11] T. Ida and K. Nakahara, *Leftmost Outside-In Narrowing Calculi*, report ISE-TR-94-107, University of Tsukuba, 1994. To appear in the Journal of Functional Programming.

[12] J.W. Klop, *Term Rewriting Systems*, in: Handbook of Logic in Computer Science, Vol. II (eds. S. Abramsky, D. Gabbay, and T. Maibaum), Oxford University Press, pp. 1–116, 1992.

[13] A. Martelli, C. Moiso, and G.F. Rossi, *Lazy Unification Algorithms for Canonical Rewrite Systems*, in: Resolution of Equations in Algebraic Structures, Vol. II, Rewriting Techniques (eds. H. Aït-Kaci and M. Nivat), (Academic Press, pp. 245–274, 1989.

[14] A. Martelli, G.F. Rossi, and C. Moiso, *An Algortihm for Unification in Equational Theories*, Proceedings of the 1986 Symposium on Logic Programming, pp. 180–186, 1986.

[15] A. Middeldorp and E. Hamoen, *Completeness Results for Basic Narrowing*, Applicable Algebra in Engineering, Communication and Computing **5**, pp. 213–253, 1994.

[16] A. Middeldorp, S. Okui, and T. Ida, *Lazy Narrowing: Strong Completeness and Eager Variable Elimination*, Theoretical Computer Science **167**, 1996. To appear. Available from `http://www.score.is.tsukuba.ac.jp/~ami/papers/lnc.dvi`. An extended abstract appeared in the Proceedings of the 20th Colloquium on Trees in Algebra and Programming, Aarhus, Lecture Notes in Computer Science **915**, pp. 394–408, 1995.

[17] W. Snyder, *A Proof Theory for General Unification*, Birkhäuser, 1991.

[18] T. Suzuki, *Standardization Theorem Revisited*, Proceedings of the 5th International Conference on Algebraic and Logic Programming, Aachen, Lecture Notes in Computer Science, 1996. To appear.