Electronic Notes in Theoretical Computer Science 58 No. 2 (2001)
URL: http://www.elsevier.nl/locate/entcs/volume58.html 34 pages

Induction for termination with local strategies

Olivier Fissore [a], Isabelle Gnaedig [b] and Hélène Kirchner [c]

[a] *LORIA-CNRS, P 239 F-54506 Vandœuvre-lès-Nancy Cedex,*
e-mail: Olivier.Fissore@loria.fr

[b] *LORIA-INRIA, P 239 F-54506 Vandœuvre-lès-Nancy Cedex,*
e-mail: Isabelle.Gnaedig@loria.fr

[c] *LORIA-INRIA & LORIA-CNRS, P 239 F-54506 Vandœuvre-lès-Nancy Cedex,*
e-mail: Helene.Kirchner@loria.fr

Abstract

In this paper, we propose a method for specifically proving termination of rewriting with particular strategies: local strategies on operators. An inductive proof procedure is proposed, based on an explicit induction on the termination property. Given a term, the proof principle relies on alternatively applying the induction hypothesis on its subterms, by abstracting the subterms with induction variables, and narrowing the obtained terms in one step, according to the strategy. The induction relation, an $\mathcal{F}$-stable ordering having the subterm property, is not given a priori, but its existence is checked along the proof, by testing satisfiability of ordering constraints.

Keywords: Rewriting, termination, local strategy on operators, rule-based languages, induction, narrowing, ordering constraints.

1 Introduction

Termination of rewriting is a crucial problem in automated deduction, for equational logic, as well as in programming, for rule-based languages. As it is undecidable in general, it is ensured in particular contexts with sufficient conditions. A lot of termination proof techniques have been proposed, most of them using noetherian orderings on terms. But they usually tackle the property for the standard rewriting relation and essentially work on free term algebras. In the context of rule-based languages such as ASF+SDF [14], OBJ3 [13], Maude [5], CafeOBJ [10], Stratego [20], or ELAN [3], where programs are sets of rules and executions consist in rewriting ground expressions, it would be useful to have more specific termination proof tools: methods allowing to prove termination under specific reduction strategies, or to prove

©2001 Published by Elsevier Science B. V. Open access under CC BY-NC-ND license.

termination on the ground term algebra, for term rewriting systems (TRSs in short) that are not terminating on the free term one. The proof method we propose here, based on an explicit induction on the termination property, enables us to tackle these problems.

In the context of programming, there are sets of rules, that lead to divergent computations when all derivations are considered, but that terminate for particular strategies. A famous example is the evaluation of a recursive function defined with an `if_then_else_` expression, which can diverge if the first argument is not evaluated first.

Local strategies on operators are used in this context, in particular to force the evaluation of expressions to terminate. This kind of strategy is allowed by languages such that OBJ3, CafeOBJ or Maude, and studied in [7] and [18]. It is defined in the following way: to any operator f is attached an ordered list of integers, giving the positions of the subterms to be evaluated in a given term, whose top operator is f. For example, the TRS

$$\begin{aligned}
&f(i(x)) && \to if_then_else(zero(x), g(x), f(h(x)))\\
&zero(0) && \to true\\
&zero(s(x)) && \to false\\
&if_then_else(true, x, y) && \to x\\
&if_then_else(false, x, y) && \to y\\
&h(0) && \to i(0)\\
&h(x) && \to s(i(x))
\end{aligned}$$

using the conditional expression, does not terminate for the standard rewriting relation, but does with the following strategy: $LS(ite) = [1;0]$, $LS(f) = LS(zero) = LS(h) = [1;0]$ and $LS(g) = LS(i) = [1]$, where if_then_else is denoted ite for short.

As far as we know, specific termination proof tools for rewriting with strategies have only been given for the innermost case [1] and for the context sensitive rewriting [17,16,21,11] on free term algebras, and for the innermost and the outermost cases on ground term ones [12]. Here, we propose a termination proof method for the case of local strategies on operators, following the induction proof principle proposed in [12]. Note that with our approach we handle the leftmost innermost and the innermost strategies: the leftmost innermost strategy is a particular case of local strategy, and as proved in [15], termination of rewriting is equivalent for the leftmost innermost and the innermost strategies. As said above, there are also termination results for a kind of rewriting called context-sensitive rewriting. In this context, rewriting is allowed only at some specified position in the terms, which is different from local strategies, that are more specific: in the second case, not only allowed rewriting positions are specified, but also the order to consider them. Except for particular cases of local strategies, the two kinds of strategy are different.

The main idea of our proof method is to use explicit induction on the termination property in order to prove that any element t of a given set of

terms T terminates i.e. there is no infinite derivation chain starting from t. Our induction principle uses an ordering on ground terms having the subterm property. It is based on the simple idea that if reducing a term t according to a given strategy first requires to normalize a subterm t' of t, we can suppose, by induction hypothesis, that t' terminates for the same strategy. If we replace t' by an induction variable X representing any of its normal forms, it then remains to prove that the term u obtained by replacement of t' by X in t is terminating, to prove that t is terminating. A rewriting step is then performed on u following the different possible values of X: it is computed by narrowing. This process is iterated until obtaining a non narrowable term, or a term the induction hypothesis applies on. Note that the induction ordering is not given a priori but constrained during the proof by setting ordering constraints. Applying the induction hypothesis then lies on testing whether these constraints are satisfiable.

On the previous example, our method consists in proving termination of the constants, and of the terms of the form $f(T), zero(T), ite(T_1, T_2, T_3), h(T)$, $i(T), g(T), s(T)$, for the previously given strategy, whatever the values of the ground terms T, T_1, T_2, T_3. Obviously, $0, true, false$ are in normal form and then terminating. For $i(T)$ (like for $s(T)$ and $h(T)$), using an induction ordering $\succ$ such that $i(T) \succ T$, by induction hypothesis, we can suppose that T is terminating. So is $i(T)$, since i is a constructor (i.e. i is not a top symbol of left-hand side of rule).

By definition of the strategy, normalizing $zero(T)$ first consists in normalizing T, into any of its normal forms $T{\downarrow}$ if it exists, and then $zero(T{\downarrow})$ at the top position. For the same ordering $\succ$ as previously, we have $zero(T) \succ T$. Then, by induction hypothesis, T terminates. Let $T{\downarrow}$ be any of its normal forms (there can be several normal forms if the system is not confluent). The termination of $zero(T)$ is then reduced to the termination of $zero(T{\downarrow})$, which can only reduce into $true$ or $false$.

Normalizing $h(T)$ also first consists in normalizing T. In a similar way as previously, the induction hypothesis can be applied to T. The termination of $h(T)$ is then reduced to the termination of $h(T{\downarrow})$, which can only reduce into $i(0)$, that is in normal form, or into $s(i(T{\downarrow}))$, that is also in normal form.

Normalizing $ite(T_1, T_2, T_3)$ first consists in normalizing T_1. As previously, T_1 can be supposed to be terminating, and the termination of $ite(T_1, T_2, T_3)$ reduced to the termination of $ite(T_1{\downarrow}, T_2, T_3)$. According to the strategy, $ite(T_1{\downarrow}, T_2, T_3)$ is then reduced into T_2 or T_3, that are terminating by induction hypothesis.

We proceed in the same way for studying how $f(T)$ normalizes.

Our goal here is to provide a procedure implementing such a reasoning. In Section 2, the background is presented. Section 3 introduces the basic notions formalizing our induction principle. In Section 4, a rule-based algorithm mechanizing the proof principle is given, its correctness is established and examples are given.

2 The background

We assume that the reader is familiar with the basic definitions and notations of term rewriting given for instance in [6]. $\mathcal{T}(\mathcal{F},\mathcal{X})$ is the set of terms built from a given finite set $\mathcal{F}$ of function symbols having an arity $n \in \mathbb{N}$, and a set $\mathcal{X}$ of variables denoted $x, y \ldots$. $\mathcal{T}(\mathcal{F})$ is the set of ground terms (without variables). The terms composed by a symbol of arity 0 are called constants; $\mathcal{C}$ is the set of constants of $\mathcal{F}$. Positions in a term are represented as sequences of integers; ϵ denote the empty sequence. The top position of a term t is ϵ, and the symbol at the top position of t is written $top(t)$. Let p and p' be two positions. The position p is said to be prefix of p' (and p' suffix of p) if $p' = p\lambda$, where λ is a non empty sequence of integers. Given a term t, $\mathcal{O}(t)$ is the set of positions in t, inductively defined as follows: $\mathcal{O}(t) = \{\epsilon\}$ if $t \in \mathcal{X}$, $\mathcal{O}(t) = \{\epsilon\} \cup \{i.p \mid 1 \leq i \leq n \text{ and } p \in \mathcal{O}(t_i)\}$ if $t = f(t_1, \ldots, t_n)$. This set is partitioned into $\overline{\mathcal{O}}(t) = \{p \in \mathcal{O}(t) \mid t|_p \notin \mathcal{X}\}$ and $\mathcal{O}_{\mathcal{V}}(t) = \{p \in \mathcal{O}(t) \mid t|_p \in \mathcal{X}\}$ where the notation $t|_p$ stands for the subterm of t at position p. If $p \in \mathcal{O}(t)$, then $t[t']_p$ denotes the term obtained from t by replacing the subterm at position p by the term t'.

A substitution is an assignment from $\mathcal{X}$ to $\mathcal{T}(\mathcal{F},\mathcal{X})$, written $\sigma = (x \mapsto t) \ldots (y \mapsto u)$. It uniquely extends to an endomorphism of $\mathcal{T}(\mathcal{F},\mathcal{X})$. We identify a substitution $\sigma = (x \mapsto t) \ldots (y \mapsto u)$ with the finite set of equations $(x = t) \wedge \ldots \wedge (y = u)$. The result of applying σ to a term $t \in \mathcal{T}(\mathcal{F},\mathcal{X})$ is written $\sigma(t)$ or σt. The domain of σ, denoted $Dom(\sigma)$ is the finite subset of $\mathcal{X}$ such that $\sigma x \neq x$. The range of σ, denoted $Ran(\sigma)$, is defined by $Ran(\sigma) = \bigcup_{x \in Dom(\sigma)} Var(\sigma x)$. A ground substitution or instantiation is an assignment from $\mathcal{X}$ to $\mathcal{T}(\mathcal{F})$. Id denotes the identity substitution. The composition of substitutions σ_1 followed by σ_2 is denoted $\sigma_2\sigma_1$. Given two substitutions σ_1 and σ_2, we write $\sigma_1 \leq \sigma_2$ iff $\exists\theta$ such that $\sigma_2 = \theta\sigma_1$. Given a subset $\mathcal{X}_1$ of $\mathcal{X}$, we note $\sigma_{\mathcal{X}_1}$ for the restriction of σ to the variables of $\mathcal{X}_1$, i.e. the substitution such that $Dom(\sigma_{\mathcal{X}_1}) \subseteq \mathcal{X}_1$ and $\forall x \in Dom(\sigma_{\mathcal{X}_1}) : \sigma_{\mathcal{X}_1} x = \sigma x$.

Given a set R of rewrite rules or term rewriting system on $\mathcal{T}(\mathcal{F},\mathcal{X})$, a function symbol in $\mathcal{F}$ is called a constructor if it does not occur in R at the top position of the left-hand side of a rule, and is called a defined function symbol otherwise. The set of constructors of $\mathcal{F}$ for R is denoted by $Cons_R$, the set of defined function symbols of $\mathcal{F}$ for R is denoted by Def_R (R is omitted when there is no ambiguity). The rewriting relation induced by R is called standard rewriting relation and is noted $\rightarrow_R$ ($\rightarrow$ if there is no ambiguity on R). We note $s \rightarrow_{p,l\rightarrow r,\sigma} t$ (or $s \rightarrow^{p,l\rightarrow r,\sigma} t$ where either p or $l \rightarrow r$ or σ may be omitted) if s rewrites into t at position p with the rule $l \rightarrow r$ and the substitution σ. The term $s|_p$ is called a redex, the position p a redex position and the symbol in s at position p is called a redex symbol. The transitive (resp. reflexive transitive) closure of the rewriting relation induced by R is denoted by $\rightarrow_R^+$ (resp. $\rightarrow_R^*$). If it exists, the last term of a finite derivation starting from t is said to be in normal form, and is denoted by $t{\downarrow}$.

An ordering $\succ$ on $\mathcal{T}(\mathcal{F},\mathcal{X})$ is said to be noetherian (or well-founded) iff there is no infinite decreasing chain for this ordering. It is $\mathcal{F}$-stable iff for any pair of terms t,t' of $\mathcal{T}(\mathcal{F},\mathcal{X})$, for any context $f(\ldots\ \ldots)$, $t \succ t'$ implies $f(\ldots t \ldots) \succ f(\ldots t' \ldots)$. It has the subterm property iff for any t of $\mathcal{T}(\mathcal{F},\mathcal{X})$, $f(\ldots t \ldots) \succ t$. Note that if $\succ$ is $\mathcal{F}$-stable and has the subterm property, then it is noetherian. If, in addition, $\succ$ is stable by substitution (for any substitution σ, any pair of terms $t,t' \in \mathcal{T}(\mathcal{F},\mathcal{X})$, $t \succ t'$ implies $\sigma t \succ \sigma t'$), then it is called a simplification ordering. Let t be a term of $\mathcal{T}(\mathcal{F})$; let us recall that t terminates if and only if any rewriting derivation (or derivation chain) starting from t is finite.

3 Induction for termination with local strategies

We now tackle the termination problem for rewriting with local strategies on operators, as expressed in [13] and studied in [7]. A local strategy is defined in the following way.

Definition 3.1 An *LS rewriting strategy* (or LS-strategy) on terms of $\mathcal{T}(\mathcal{F},\mathcal{X})$ (resp. of $\mathcal{T}(\mathcal{F})$) is a function LS from $\mathcal{F}$ to the set of lists of integers $\mathcal{L}(\mathbb{N})$, defining a rewriting strategy as follows.

Given a LS-strategy such that $LS(f) = [p_1,\ldots,p_k]$, $p_i \in [0..arity(f)]$ for all $i \in [1..k]$, for some symbol $f \in \mathcal{F}$, normalizing a term $t = f(t_1,\ldots,t_m) \in \mathcal{T}(\mathcal{F},\mathcal{X})$ (resp $\in \mathcal{T}(\mathcal{F})$) with respect to $LS(f) = [p_1,\ldots,p_k]$, consists in normalizing all subterms of t at positions $p_1,\ldots,p_k$ successively, according to the strategy. If there exists $i \in [1..k]$ such that $p_1,\ldots,p_{i-1} \neq 0$ and $p_i = 0$ (0 is the top position), then

- if the current term t' obtained after normalizing $t|_{p_1},\ldots,t|_{p_{i-1}}$ is reducible at the top position into a term $g(u_1,\ldots,u_n)$, then $g(u_1,\ldots,u_n)$ is normalized with respect to $LS(g)$ and the rest of the strategy $[p_{i+1},\ldots,p_k]$ is ignored,
- if t' is not reducible at the top position, then t' is normalized with respect to $p_{i+1},\ldots,p_k$.

At each rewriting step, the term t is said to LS-rewrite into a term t'. If t does not rewrite for the LS-strategy, it is said to be in LS-normal form (or in normal form if there is no ambiguity). If any LS-rewriting chain starting from t leads to an LS-normal form then t is said to be LS-terminating (or to LS-terminate). If the evaluation strategy of a term t' is the empty list, then t' is in LS-normal form.

In the following, we will use a notion expressing the possible reducible positions of any instantiation of t, with respect to the LS-strategy.

Definition 3.2 A position p of a term $t \in \mathcal{T}(\mathcal{F},\mathcal{X})$ is an LS-position in t if the LS-strategy allows to rewrite t at position p, or if the LS-strategy allows to rewrite any ground instance of t at position p or at a suffix position of p.

The set $LS{-}POS(t)$ of LS-positions of a term t can be computed in the following way.

- $LS{-}POS(f(u_1,\ldots,u_n)) =$
$$\begin{cases} \{\epsilon\} \bigcup_{i\in LS(f)\setminus 0}\{i.p \mid p \in LS{-}POS(u_i)\} & \text{if } 0 \in LS(f), \\ \bigcup_{i\in LS(f)}\{i.p \mid p \in LS{-}POS(u_i)\} & \text{if } 0 \notin LS(f), \end{cases}$$
- $LS{-}POS(x) = \begin{cases} \{\epsilon\} & \text{if } x \in \mathcal{X}, \\ \emptyset & \text{if } x \in \mathcal{N}. \end{cases}$

3.1 Induction for local strategies

For proving that a term t of $\mathcal{T}(\mathcal{F})$ LS-terminates, we proceed by induction on $\mathcal{T}(\mathcal{F})$ with a noetherian ordering $\succ$ (more precisely, an $\mathcal{F}$-stable ordering having the subterm property), assuming that for any t' such that $t \succ t'$, t' LS-terminates. We first prove that a basic set of minimal elements for $\succ$ LS-terminates. As the subterm property for $\succ$ is required, the set of minimal elements is a subset of the set of constants of $\mathcal{F}$.

We then consider the case of any term t of $\mathcal{T}(\mathcal{F})$. For that, we observe the rewriting derivation tree for the LS-strategy starting from a term $t_{ref} = g(x_1,\ldots,x_m)$, for any $g \in \mathcal{F}$, where $x_1,\ldots,x_m$ are induction variables that can be instantiated by any ground term. The LS rewriting relation on ground terms is simulated by the two mechanisms below to follow the derivation tree starting from t_{ref}, and whose current term is t. Let $LS(top(t)) = [p_1,\ldots,p_n]$, and p_k the first element of $[p_1,\ldots,p_n]$ such that $p_k = 0$.

- First, the subterms $t|_{p_1},\ldots,t|_{p_{k-1}}$ of t have to be LS-normalized, by definition of the above LS-strategy. If $t_{ref} \succ t|_{p_1},\ldots,t|_{p_{k-1}}$ we can suppose, by induction hypothesis, that these subterms are LS-terminating. We then replace them in t by *abstraction variables* X_i representing respectively any of their normal forms $t_i{\downarrow}$: these variables will only be instantiated by terms in normal form. Reasoning by induction allows us to only suppose the existence of the $t_i{\downarrow}$ *without explicitly computing them*; this step will be called abstraction step or abstraction of the subterms of t. We also say that t is abstracted into a term v.
- Second, rewriting the resulting term v at position ϵ, following all possible ground instantiations of v. This is computed by a narrowing step on v. Two cases may happen:
  - if v is not narrowable at the top position, the subterms $v|_{p_{k+1}},\ldots,v|_{p_n}$ of v then have to be LS-normalized, and we try to abstract them like above;
  - if v is narrowable at the top position, the narrowing step is computed with all possible rules and all possible substitutions $\sigma_1,\ldots,\sigma_l$ to give terms $w_1,\ldots,w_l$, that have to be considered respectively with the strategies $LS(top(w_1)),\ldots,\ LS(top(w_l))$. So the two mechanisms above are again applied on the terms $w_1,\ldots,w_l$. In addition, instances of v that are not considered by the narrowing have to be reduced at the positions

$p_{k+1}, \ldots, p_n$. So the two mechanisms described above are also applied on v at positions $p_{k+1}, \ldots, p_n$ for the instances of v that are not instances of $\sigma_i v, i \in [1..l]$.

- The process stops on the current terms t having an empty LS-strategy or on current terms the induction hypothesis can be applied on (i.e. such that $t_{ref} \succ t$; in this case, t is supposed to be LS-terminating).

Note that if there does not exist p_k in $\{p_1, \ldots, p_n\}$ such that $p_k = 0$, then only the first point is processed, abstracting every subterm $t|_{p_i}$ of $t, i \in [1..n]$.

3.2 Abstraction

We now give some new definitions to formalize the above mechanisms. Abstraction needs the use of special variables representing LS-normal forms.

Definition 3.3 Let $\mathcal{N}$ be a set of new variables disjoint from $\mathcal{X}$. Symbols of $\mathcal{N}$ are called *NF-variables*. Substitutions and instantiations are extended to $\mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N})$ in the following way. Let $X \in \mathcal{N}$; for any substitution σ (resp. instantiation θ) such that $X \in Dom(\sigma)$, σX (resp. θX) is in normal form.

Note that for abstracting the current term $f(u_1, \ldots, u_m)$, it is not useful to introduce an abstraction variable for the u_j that are ground terms already in normal form, nor for the u_j that are already NF-variables.

Definition 3.4 The term $f(u_1, \ldots, u_m)$ is *abstracted* into $f(U_1, \ldots, U_m)$ at positions $\{i_1, \ldots, i_p\} \subseteq [1..m]$ if :

- $\{i_1, \ldots, i_p\}$ are the positions of $[1..m]$ such that $u_{i_1}, \ldots, u_{i_p}$ are neither ground terms in normal form, nor NF-variables,
- $U_j = X_j$ where X_j is a fresh NF-variable, if $j \in \{i_1, \ldots, i_p\}$, $U_i = u_i$ otherwise.

We will prove LS-termination on $\mathcal{T}(\mathcal{F})$, reasoning on terms with abstraction variables, i.e. on terms of $\mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N})$.

3.3 Constraints

Let us now define the different constraints needed by our proof process. Unlike in classical approaches using induction, the induction ordering is not given a priori. Constraints are set along the proof, following the requirements appearing when induction hypotheses have to be applied. Such ordering constraints are cumulated in a set C and the satisfiability of C is tested any time the induction hypotheses have to be applied.

We now formally define the satisfiability of ordering constraints.

Definition 3.5 *An ordering constraint* $(t > t')$ on terms of $\mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N})$ is *satisfiable* if there exists an ordering $\succ$ and at least one instantiation θ such that $\theta t \succ \theta t'$. We say that $\succ$ and θ satisfy $(t > t')$.

A conjunction C of ordering constraints is satisfiable if there exists an ordering and an instantiation satisfying all conjuncts. The empty conjunction, always satisfied, is denoted by $\top$.

Along our induction process, when abstracting subterms t_i by X_i, we state constraints on NF-variables to express that their instances can only be the normal forms of the corresponding instances of the t_i. They are of the form $t{\downarrow} = X$ where $t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$, and $X \in \mathcal{N}$, or more generally of the form $t{\downarrow} = t'$ where $t, t' \in \mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N})$. Let us call such a constraint an abstraction constraint.

Definition 3.6 An *abstraction constraint* $(t{\downarrow} = t')$ where $t, t' \in \mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N})$ is *satisfiable* if there exists at least one instantiation θ such that $\theta t{\downarrow} = \theta t'$. We say that θ satisfies $(t{\downarrow} = t')$.

A *constraint formula* A is a formula of the form $\bigwedge_i (t_i{\downarrow} = t'_i) \bigwedge_j (\bigvee_{k_j} (x_{k_j} \neq u_{k_j}))$, $x_{k_j} \in \mathcal{X} \cup \mathcal{N}, u_{k_j} \in \mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N})$ and the $(t_i{\downarrow} = t'_i)$ are abstraction constraints. The empty formula is denoted $\top$. A formula A is *satisfiable* if there exists at least one instantiation θ such that $\bigwedge_i (\theta t_i{\downarrow} = \theta t'_i) \bigwedge_j (\bigvee_{k_j} (\theta x_{k_j} \neq \theta u_{k_j}))$. We say that θ satisfies A.

In this paper, we consider constraint problems composed of 2-tuples (A, C) where A is constraint formula and C is a conjunction of ordering constraints.

Definition 3.7 Let A be a constraint formula and C a conjunction of ordering constraints. The constraint problem (A, C) is satisfied by an ordering $\succ$ if A is satisfiable, and for all instantiations θ satisfying A, $\succ$ and θ satisfy C. (A, C) is satisfiable if A is satisfiable and there exists an ordering $\succ$ as above.

Deciding the satisfiability of (A, C) would require to express all instantiations satisfying A. As we will see later, an interesting point of our method is that we do not need to characterize all those instantiations. It is enough to exhibit one of them to prove the satisfiability of A. In such a case, a sufficient condition for an ordering $\succ$ to satisfy (A, C) is that $\succ$ is stable by substitution (the induction ordering is then a simplification ordering) and $t \succ t'$ for any inequality $t > t'$ of C.

3.4 Narrowing

After the abstraction of the term $f(u_1, \ldots, u_m)$ into $f(U_1, \ldots, U_m)$ at positions $\{i_1, \ldots, i_p\}$, where the u_{i_j} are supposed to have a normal form $u_{i_j}{\downarrow}$, and are replaced by abstraction variables X_{i_j}, we test whether the ground instances of $f(U_1, \ldots, U_m)$ are reducible with a case study on the syntactic form of the possible instantiations of the X_{i_j}. This test consists in narrowing $f(U_1, \ldots, U_m)$ at position ϵ with all possible substitutions instantiating the X_i only with irreducible terms, and all possible rewrite rules.

Let us now recall the definition of narrowing.

Definition 3.8 Let $\mathcal{R}$ be a TRS on $\mathcal{T}(\mathcal{F},\mathcal{X})$. A term t is *narrowed* into t', at the non variable position p, using the rewrite rule $l \to r$ of R and the substitution σ, when σ is a most general unifier of $t|_p$ and l, $t' = \sigma(t[r]_p)$. This is denoted $t \leadsto_R^{p,l\to r,\sigma} t'$ where either p, or $l \to r$ or σ may be omitted. It is always assumed that there is no variable in common between the rule and the term, i.e. that $Var(l) \cap Var(t) = \emptyset$.

The requirement of disjoint variables is easily fulfilled by an appropriate renaming of variables in the rules when narrowing is performed. Note that for the most general unifier σ used in the above definition, $Dom(\sigma) \subseteq Var(l) \cup Var(t)$ and we can choose $Ran(\sigma) \cap (Var(l) \cup Var(t)) = \emptyset$, thus introducing in the range of σ only fresh variables. Thus $Var(t) \cap Var(t') = \emptyset$ if in addition, variables of $\mathcal{V}ar(t) - Dom(\sigma)$ are renamed through a substitution denoted σ_{ren}.

As we will see below, in our proof process, we will also have to consider the negation of a substitution.

Definition 3.9 Let σ be a substitution on $\mathcal{T}(\mathcal{F},\mathcal{X}\cup\mathcal{N})$ defined by $\bigwedge_i(x_i = t_i)$ $x_i \in \mathcal{X}\cup\mathcal{N}$, $t_i \in \mathcal{T}(\mathcal{F},\mathcal{X}\cup\mathcal{N})$. The *negation* of σ, denoted $\overline{\sigma}$ is the formula $\bigvee_i(x_i \neq t_i)$.

4 A rule-based algorithm

4.1 The inference rules

Inference rules describing our termination proof mechanism for local strategies work on sets of 4-tuples $T = (\{u\}, [p_1,\ldots,p_m], A, C)$, where:

- $\{u\}$ is a set of terms of $\mathcal{T}(\mathcal{F},\mathcal{X}\cup\mathcal{N})$, containing the current term u whose ground instances have to be proved LS-terminating. This is either a singleton or the empty set.
- $[p_1,\ldots,p_m]$ is the list of positions with respect to whom the current term u has to be evaluated. This is a sublist of $LS(top(u))$.
- A is a constraint formula memorizing the abstractions and narrowing substitutions performed on the current term u. The sub-formulas of the form $u{\downarrow} = X, u \in \mathcal{T}(\mathcal{F},\mathcal{X}\cup\mathcal{N}), X \in \mathcal{N}$ are stated each time a subterm u of the current term is abstracted by a new NF-variable X.
- C is a conjunction of ordering constraints completed by the abstraction steps.

Let us now present the inference rules.

- The rule **Abstract** processes the abstracting step. It applies on $(\{f(u_1,\ldots,u_m)\}, [p_1,\ldots,p_n], A, C)$, when there exists $k \in [2..n], p_k = 0$ and $p_1,\ldots,p_{k-1} \neq 0$. The term $u = f(u_1,\ldots,u_m)$ is abstracted at positions $i_1,\ldots,i_p \in \{p_1,\ldots,p_{k-1}\}$ if there exists an $\mathcal{F}$-stable ordering having the subterm property and such that $(A, C \wedge t_{ref} > u_{i_1},\ldots,u_{i_p})$ is satisfiable. Indeed, by

Table 1
Inference rules for t_{ref} LS-termination

Abstract:
$$\frac{T \cup \{(\{f(u_1,\dots,u_m)\},\ [p_1,\dots,p_n],\ A,\ C)\}}{T \cup \{(\{f(U_1,\dots,U_m)\},\ [0,p_{k+1},\dots,p_n],\ A \bigwedge_i (u_i\downarrow = X_i),\ C \bigwedge_i t_{ref} > u_i)\}}$$

where $f(u_1,\dots,u_m)$ is abstracted by $f(U_1,\dots,U_m)$ at the pos. $i \in \{i_1,\dots,i_p\} \subseteq \{p_1,\dots,p_{k-1}\}$
if $\exists k \in [2..n] : p_1,\dots,p_{k-1} \neq 0, p_k = 0$ and $(A, C \bigwedge_{i\in\{i_1,\dots,i_p\}} t_{ref} > u_i)$ is satisfiable

Abstract-Stop:
$$\frac{T \cup \{(\{f(u_1,\dots,u_m)\},\ [p_1,\dots,p_n],\ A,\ C)\}}{T \cup \{(\emptyset,\ [],\ A,\ C \bigwedge_{i\in\{i_1,\dots,i_p\}} t_{ref} > u_i)\}}$$

where $f(u_1,\dots,u_m)$ can be abstracted by $f(U_1,\dots,U_m)$ at the pos. $i_1,\dots,i_p \in \{p_1,\dots,p_n\}$
if $p_1,\dots,p_n \neq 0$ and $(A, C \bigwedge_{i\in\{i_1,\dots,i_p\}} t_{ref} > u_i)$ is satisfiable

Narrow-Y:
$$\frac{T \cup \{(\{f(u_1,\dots,u_m)\},\ [0,p_1,\dots,p_n],\ A,\ C)\}}{T \cup_{i\in[1..l]} \{(\{w_i\},\ LS(top(w_i)),\ A\wedge\sigma_i,\ \sigma_i C)\} \cup COMPL}$$

if $\exists\sigma$ such that $f(u_1,\dots,u_m) \ _{\epsilon,\sigma}\ w$ and $A\wedge\sigma$ is satisfiable
where $w_i, i \in [1..l]$, are all terms such that $f(u_1,\dots,u_m) \ _{\epsilon,\sigma_i}\ w_i$ and $A\wedge\sigma_i$ is satisfiable,

$$COMPL = \begin{cases} \{(\{f(u_1,\dots,u_m)\},\ [p_1..p_n],\ A\bigwedge_{i=1}^{l}\overline{\sigma_i},\ C)\} & \text{if } (A\bigwedge_{i=1}^{l}\overline{\sigma_i}) \text{ satisfiable} \\ \emptyset & \text{otherwise.} \end{cases}$$

Narrow-N:
$$\frac{T \cup \{(\{f(u_1,\dots,u_m)\},\ [0,p_1,\dots,p_n],\ A,\ C)\}}{T \cup \{(\{f(u_1,\dots,u_m)\},\ [p_1,\dots,p_n],\ A,\ C)\}}$$

if $f(u_1,\dots,u_m)$ is not narrowable at the top position
or $\forall\sigma$ narrowing substitution of $f(u_1,\dots,u_m)$ at the top position , $A\wedge\sigma$ is not satisfiable.

Stop-Ind:
$$\frac{T \cup \{(\{u\},\ [p_1,\dots,p_n],\ A,\ C)\}}{T \cup \{(\emptyset,\ [],\ A,\ C\wedge t_{ref} > u)}$$

if $(A, C\wedge t_{ref} > u)$ is satisfiable

Stop-A:
$$\frac{T \cup \{(\{f(u_1,\dots,u_m)\},\ [p_1,\dots,p_n],\ A,\ C)\}}{T \cup \{(\{\sharp\},\ [],\ A,\ C)\}}$$

if $p_1 \neq 0$ and neither Abstract nor Abstract−Stop applies.

Stop:
$$\frac{T \cup \{(\{f(u_1,\dots,u_m)\},\ [],\ A,\ C)\}}{T \cup \{(\emptyset,\ [],\ A,\ C)\}}$$

induction hypothesis, all ground instances of $u_{i_1},\dots,u_{i_p}$ LS-terminate. So $\{f(u_1,\dots,u_m)\}$ is replaced by $f(U_1,\dots,U_m)$. The list of positions then becomes $[0,p_{k+1},\dots,p_n]$.

- The rule **Abstract−Stop** processes the abstracting step as above, when there is no position 0 in the strategy of the current term. Any ground instance of the term obtained after abstraction is irreducible, by definition of the LS-strategy, which ends the proof on the current derivation chain.

The set containing the current term is then replaced by the empty set.

- The rule **Narrow−Y** processes the narrowing step at position 0 of the current term u. If u is narrowable with a substitution satisfying the current constraint formula A, then u is narrowed in all possible ways in one step, with all possible rewrite rules of the rewrite system R, and all possible substitutions σ_i, into $w_i, i \in [1..l]$. Then $(\{u\}, [0, p_1, \ldots, p_n], A, C)$ is replaced by $\{(\{w_i\}, LS(top(w_i)), A \wedge \sigma_i, \sigma_i C), i \in [1..l]\}$, where σ_i is the most general substitution allowing narrowing of u into terms w_i. Moreover, since in A, we only memorize the abstractions and narrowing substitutions performed on the current term u, and since $\mathcal{V}ar(u)$ is disjoint from the set of variables occuring in the rewrite rules of the TRS, we can restrict σ_i to $\mathcal{V}ar(u)$ in adding σ_i to A. Thus, in the following, we will write σ_i for $\sigma_{i\mathcal{V}ar(u)}$.
  This narrowing step means that $\sigma_1 u, \ldots, \sigma_l u$ are all instances of u that are reducible at the top position. It involves that if $\Phi = \overline{\sigma_1} \wedge \ldots \wedge \overline{\sigma_l}$ is satisfiable, for each substitution μ satisfying Φ, μu is not reducible at the top position. Then, as these μu have to be reduced at positions $[p_1, \ldots, p_n]$, if Φ is satisfiable, to the previous set we must add the set : $(\{u\}, [p_1, \ldots, p_n], A \bigwedge_{i=1}^{l} \overline{\sigma_i}, C)\}$. Note that if $\exists i$ such that σ_i is just a renaming of variables, then $\Phi = \emptyset$.
  Let us also precise that if w_i is a variable $x \in \mathcal{X}$, we cannot conclude anything about termination of ground instances of x. So we force the proof process to stop in setting $LS(x)$ to a particular symbol $\sharp$. However, if $w_i = X \in \mathcal{N}$, $LS(X)$ is set to [], which is coherent with the fact that any ground instance of X is in normal form.
- The rule **Narrow−N** handles the case where u is not narrowable at position 0 or is narrowable with a substitution that does not satisfy the current constraint formula A. Then no narrowing is processed and the current term is evaluated at positions following the top position in the strategy. The list of positions then becomes $[p_1, \ldots, p_n]$.
- We also can test for the current term whether there exists an ordering having the subterm property such that $(A, C \wedge t_{ref} > u)$ is satisfiable. Then, by induction hypothesis, any ground instance of u terminates for the LS-strategy, which ends the proof on the current derivation chain. The **Stop−Ind** rule then replaces the set containing the current term by the empty set.
- The rule **Stop−A** allows to stop the inference process when neither **Abstract** nor **Abstract−Stop** applies, replacing u by the particular symbol $\sharp$.
- The rule **Stop** allows to stop the inference process when the list of positions is empty.

The set of inference rules is given in Table 1.

Once **Abstract** is applied, the evaluation list's first element is 0, so the only rule that applies then is one of {**Narrow−Y**, **Narrow−N**}. When **Narrow−Y** does not apply, **Narrow−N** applies. When **Abstract** does not apply, and the evaluation list's first element is not 0, either **Abstract−Stop**

or **Stop−A** applies, and then no rule applies anymore. The strategy for applying these rules is:

repeat*
((**Abstract/ Abstract−Stop/ Stop−A**);
(**Narrow−Y/ Narrow−N**);
Stop; Stop−Ind)

where ";" expresses the sequential application of the rules, $r_1/\ldots/r_n$ expresses that the rules $r_1, \ldots, r_n$ are mutually exclusive and that one of them will be applied, and repeat* $(r_1; \ldots; r_n)$ stops if none of the r_i applies anymore.

We write $SUCCESS(g, \succ)$ if application of the inference rules on $(\{g(x_1, \ldots, x_m)\}, LS(g), \top, \top)$, whose conditions are satisfied by $\succ$, gives a state of the form $(\emptyset, [], A, C)$ on every branch of the derivation tree.

Theorem 4.1 *Let R be a TRS on $\mathcal{T}(\mathcal{F}, \mathcal{X})$, and $LS : \mathcal{F} \mapsto L(\mathbb{N})$ a LS-strategy such that the constants of $\mathcal{F}$ LS-terminate. If there exists an $\mathcal{F}$-stable ordering $\succ$ having the subterm property, such that for every non constant defined symbol g, $SUCCESS(g, \succ)$, then every term of $\mathcal{T}(\mathcal{F})$ LS-terminates.*

Remark the important point that the ordering $\succ$ has to be the same for all $g(x_1, \ldots, x_m) \in \mathcal{D}ef$. Remark also that the noetherian property of $\succ$ is implied by $\mathcal{F}$-stability and the subterm property.

For the proof of Theorem 4.1, as well as for the next lemmas, proposition and theorems, see the Appendix.

In the proof process, the information that variables are NF-variables can be very important to conclude: if the current term is a NF-variable, its strategy is set to [] and the rule **Stop** applies. This information can be easily deduced when new variables are introduced: the abstracting process directly introduces NF-variables, by definition.

For the narrowing process, the narrowing substitution σ, whose range only contains new variables of $\mathcal{X}$, can be transformed in a new substitution σ_{NF} by replacing some of these variables by NF-variables. Let us consider an equality of the form $X = u$, introduced by the narrowing substitution σ, where X is an NF-variable, and $u \in \mathcal{T}(\mathcal{F}, \mathcal{X})$. As X is an NF-variable, any ground instance of u must be in normal form. So the variables in u that can be replaced by NF-variables are the variables that occur at an LS-position in u.

Let now μ be the substitution $(x_i = X_i, \forall x_i \in Var(u), x_i \; occurs \; at \; an \; LS-position \; in \; u, \; for \; all \; equation \; X = u \; of \; \sigma, \; X \in \mathcal{N}, \; u \in \mathcal{T}(\mathcal{F}, \mathcal{X}))$. Then $\sigma_{NF} = \mu\sigma$.

4.2 Extending the induction principle

When the induction hypothesis cannot be applied on a term u, the inductive reasoning can be completed as follows. It can sometimes be possible to prove termination of any ground instance of u_i (resp. u) by another way. Let $TERMIN(u)$ be a predicate that is true iff any ground instance of u LS-

terminates. In **Abstract**, **Abstract−Stop** and **Stop−Ind**, we can then replace the condition $t > u_i$ for some i (resp. $t > u$) by the alternative predicate $TERMIN(u_i)$ (resp. $TERMIN(u)$). Obviously, in this case, the ordering constraint $t > u_i$ (resp. $t > u$) is not added to C.

As in [12], for establishing that $TERMIN(u)$ is true, in some cases, the notion of usable rules can be used. Given a TRS $\mathcal{R}$ on $\mathcal{T}(\mathcal{F}, \mathcal{X})$ and a term $t \in \mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N})$, we determine the only rewrite rules that are likely to apply to any of its ground instances, for the standard rewriting relation, until its ground normal form is reached, if it exists. Then we try to find a simplification ordering $\succ_N$ so that these rules are oriented. Thus any ground instance αt is bound to terminate for the standard rewriting relation: indeed, if $\alpha t \to t_1 \to t_2 \to \ldots$, then, thanks to the previous hypotheses, $\alpha t \succ_N t_1 \succ_N t_2 \succ_N \ldots$ and, since the ordering $\succ_N$ is noetherian, the rewriting chain cannot be infinite.

More formally, given a TRS $\mathcal{R}$, we call *usable rules* of a term $t \in \mathcal{T} \cup \mathcal{N}$, as in [2], a calculable superset $\mathcal{U}(t)$ of the set of rules of R used in all possible LS-derivations starting from any αt, and defined as follows. When t is a variable of $\mathcal{X}$, then the usable rules of t are $\mathcal{R}$ itself. Likewise, the set of usable rules associated to a NF-variable is empty, since the only possible instances of such a variable are ground terms in normal form. When t is of the form $f(u_1, \ldots, u_n)$, then the usable rules of t are the usable rules of the u_i, for $i \in LS(f), i \neq 0$ and, if $0 \in LS(f)$, all the rules $l \to r$ having the symbol f as top symbol of lhs l, altogether with sets $\mathcal{U}(r)$ of the usable rules of the terms r.

Since in general r contains variables of $\mathcal{X}$, the evaluation of $\mathcal{U}(r)$ and then of $\mathcal{U}(t)$ is likely to result in the whole set of rules $\mathcal{R}$. To compute a smaller set of usable rules, like for the narrowing substitutions, we will replace variables by NF-variables as much as we can. The idea is that the usable rules of some variables in r can be omitted since they are included in the usable rules of the u_i.

More precisely, rewriting αt with $l \to r$ at the top position with the ground substitution σ leads to the term σr. Let x be a variable occuring in the lhs of $l \to r$ at an LS-position p. By definition of rewriting, we have $\sigma x = \alpha t|_p$. Moreover, p is an LS-position in αt, and if x occurs in r at position p', we have $\sigma r|_{p'} = \alpha t|_p$. In addition, we need $\mathcal{U}(x)(\subseteq \mathcal{U}(r))$, only to consider the rules that can apply in the derivations of $\sigma r|_{p'}$. But as $\sigma r|_{p'} = \alpha t|_p$, and the rules that can apply in the the derivations of $\alpha t|_p$ are included in $\mathcal{U}(u_i)$ for some i, we can suppress the computation of the $\mathcal{U}(x)$ in the computation of $\mathcal{U}(r)$.
A formal justification of these facts can be found in the proof of Lemma 4.3.

So, for computing the usable rules of t, we transform each rule $l \to r$ of $\mathcal{R}$ into a rule $NF(l \to r) = l \to r'$ as follows : $r' = \mu r$ where μ is the substitution $(x_i = X_i, x_i \in \mathcal{X}, X_i \in \mathcal{N},\ \forall x_i \in Var(r)\ and\ x_i\ is\ at\ an\ LS{-}position\ in\ l)$. We note $NF(\mathcal{R})$ the set $\{NF(l \to r) | l \to r \in \mathcal{R}\}$.

The formal definition of the usable rules for LS-rewriting is then the following.

Definition 4.2 Let $\mathcal{R}$ be a TRS on a set $\mathcal{F}$ of symbols. Let $Rls(f) = \{l \to r \in \mathcal{R} \mid top(l) = f\}$, and $Rls'(f) = \{l \to r' \in NF(\mathcal{R}) \mid top(l) = f\}$. For any $t \in \mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N})$, the set of usable rules of t, denoted $\mathcal{U}(t)$, is defined by:

- $\mathcal{U}(t) = \mathcal{R}$ if $t \in \mathcal{X}$,
- $\mathcal{U}(t) = \emptyset$ if $t \in \mathcal{N}$,
- $\mathcal{U}(f(u_1, \ldots, u_n)) = \begin{cases} Rls(f) \bigcup_{i \in LS(f), i \neq 0} \mathcal{U}(u_i) \bigcup_{l \to r' \in Rls'(f)} \mathcal{U}(r') & \text{if } 0 \in LS(f) \\ \bigcup_{i \in LS(f), i \neq 0} \mathcal{U}(u_i) & \text{otherwise.} \end{cases}$

Lemma 4.3 *Let $\mathcal{R}$ be a TRS on a set $\mathcal{F}$ of symbols, $\mathcal{R}' = NF(\mathcal{R})$ and for each $l \to r' \in \mathcal{R}'$, no variable of $\mathcal{X}$ occurs in r' at an LS-position. Let $t \in \mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N})$. Whatever αt ground instance of t and $\alpha t \to_{p_1, l_1 \to r_1} t_1 \to_{p_2, l_2 \to r_2} t_2 \to \ldots \to_{p_n, l_n \to r_n} t_n$ LS-rewrite chain starting from αt, then $\forall i \in [1..n] : l_i \to r_i \in \mathcal{U}(t)$.*

We then can give a sufficient criterion for ensuring termination for the standard rewriting relation (and then LS-termination) of any ground instance of a term t.

Proposition 4.4 *Let $\mathcal{R}$ be a TRS on a set $\mathcal{F}$ of symbols, $\mathcal{R}' = NF(\mathcal{R})$ and for each $l \to r' \in \mathcal{R}'$, no variable of $\mathcal{X}$ occurs in r' at an LS-position. Let $t \in \mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N})$. If there exists a simplification ordering $\succ$ such that $\forall l \to r \in \mathcal{U}(t) : l \succ r$, then any ground instance of t is terminating.*

Remark that if there exists a simplification ordering $\succ$ such that $l \succ r$ for any rewrite rule $l \to r$ of a TRS $\mathcal{R}$, then we have $SUCCESS(g, \succ')$, for any defined symbol $g \in \mathcal{D}ef_R$ and any $\mathcal{F}$-stable ordering $\succ'$ having the subterm property. Therefore Theorem 4.1 applies. Indeed, for any $t_{ref} = g(x_1, \ldots, x_m)$, we have $\mathcal{U}(t_{ref}) = \mathcal{R}$, with every rule oriented by $\succ$. Finally, thanks to Proposition 4.4, we have $TERMIN(t_{ref})$, and then **Stop−Ind** applies.
An interesting point of this method is that the ordering $\succ$ that can be used to orient the usable rules is completely independent of the induction ordering $\succ'$.

Let us now illustrate our complete method on the example given in the Introduction.

Example 4.5 Recall the rules are:

$$\begin{array}{ll} f(i(x)) & \to ite(zero(x), g(x), f(h(x))) \\ zero(0) & \to true \\ zero(s(x)) & \to false \\ ite(true, x, y) & \to x \\ ite(false, x, y) & \to y \\ h(0) & \to i(0) \\ h(x) & \to s(i(x)) \end{array}$$

The LS-strategy is the following :

- $LS(ite) = [1; 0]$,
- $LS(f) = LS(zero) = LS(h) = [1; 0]$ and
- $LS(g) = LS(i) = [1]$.

Let us prove the termination of this system on the signature $\mathcal{F} = \{f : 1, zero : 1, ite : 3, h : 1, s : 1, i : 1, g : 1, 0 : 0\}$.

Obviously, the constant 0 LS-terminates. Applying the inference rules on $f(x_1)$, we get :

$f(x_1)$	$[1; 0]$
	$A = \top$
	$C = \top$
Abstract	
$f(X_1)$	$[0]$
	$A = (x_1\downarrow = X_1)$
	$C = (f(x_1) > x_1)$

Abstract applies, since C is satisfiable by any ordering having the subterm property. A is satisfiable with any instantiation θ such that $\theta x_1 = \theta X_1 = 0$.

Narrow−Y	
$ite(zero(X_2), g(X_2), f(h(X_2)))$	$\sigma = (X_1 = i(x_2) \wedge x' = x_2)$
	$\sigma_{NF} = (X_1 = i(X_2) \wedge x' = X_2)$
	$[1; 0]$
	$A = (x_1\downarrow = X_1 \wedge X_1 = i(X_2))$
	$C = (f(x_1) > x_1)$
$f(X_1)$	$[]$
	$A = (x_1\downarrow = X_1) \wedge (X_1 \neq i(X_2))$
	$C = (f(x_1) > x_1)$

Note that x' comes from the renaming of x in the first rule. The renaming of x_2 into X_2 comes from the fact that x_2 occurs in $i(x_2)$ at an LS-position.

Here, the first constraint formula A is satisfiable by any instantiation θ such that $\theta X_2 = 0$ and $\theta x_1 = i(0)$. The second constraint formula is satisfied by any instantiation θ such that $\theta x_1 = \theta X_1 = \theta X_2 = 0$.

Narrow−Y expresses the fact that $\sigma f(X_1)$ is reducible if σ is such that $\sigma X_1 = i(X_2)$, and that the other instances ($\sigma' f(X_1)$ with $\sigma' X_1 \neq i(X_2)$)

cannot be reduced.

Stop

$ite(zero(X_2), g(X_2), f(h(X_2)))$	$[1;0]$
	$A = (x_1{\downarrow} = X_1 \wedge X_1 = i(X_2))$
	$C = (f(x_1) > x_1)$
$\emptyset$	$[]$
	$A = (x_1{\downarrow} = X_1) \wedge (X_1 \neq i(X_2))$
	$C = (f(x_1) > x_1)$

Stop applies, ending the second branch.

Abstract

$ite(X_3, g(X_2), f(h(X_2)))$	$[0]$
	$A = (x_1{\downarrow} = X_1 \wedge X_1 = i(X_2) \wedge zero(X_2){\downarrow} = X_3)$
	$C = (f(x_1) > x_1)$
$\emptyset$	$[]$
	$A = (x_1{\downarrow} = X_1) \wedge (X_1 \neq i(X_2))$
	$C = (f(x_1) > x_1)$

The constraint formula A is satisfiable with any instantiation θ such that $\theta X_1 = i(0)$, $\theta X_2 = 0$, $\theta X_3 = true$ and $\theta x_1 = i(0)$.

Abstract applies here, since $zero(X_2)$ can be abstracted, thanks to Proposition 4.4. Indeed, $\mathcal{U}(zero(X_2)) = \{zero(0) \to true, zero(s(x)) \to false\}$, and both rules can be oriented by a LPO $\succ$ with the precedence $zero \succ_{\mathcal{F}} true$ and $zero \succ_{\mathcal{F}} false$. Then we have $TERMIN(zero(X_2))$.

Narrow−Y

$g(X_4)$	$\sigma = (X_3 = true \wedge X_2 = x_4 \wedge x'' = g(x_4) \wedge y'' = f(h(x_4)))$
	$\sigma_{NF} = (X_3 = true \wedge X_2 = X_4 \wedge x'' = g(X_4) \wedge y'' = f(h(X_4)))$
	$[1]$
	$A = (x_1{\downarrow} = X_1 \wedge X_1 = i(X_2) \wedge zero(X_2){\downarrow} = X_3 \wedge X_3 = true$
	$\wedge\ X_2 = X_4)$
	$C = (f(x_1) > x_1)$

$f(h(X_4))$	$\sigma = (X_3 = false \wedge X_2 = x_4 \wedge x'' = g(x_4) \wedge y'' = f(h(x_4)))$
	$\sigma_{NF} = (X_3 = false \wedge X_2 = X_4 \wedge x'' = g(X_4) \wedge y'' = f(h(X_4)))$
	$[1;0]$
	$A = (x_1{\downarrow} = X_1 \wedge X_1 = i(X_2) \wedge zero(X_2){\downarrow} = X_3$
	$\wedge\ X_3 = false \wedge X_2 = X_4)$
	$C = (f(x_1) > x_1)$
$ite(X_3, g(X_2), f(h(X_2)))$	$[]$
	$A = (x_1{\downarrow} = X_1 \wedge X_1 = i(X_2) \wedge zero(X_2){\downarrow} = X_3)$
	$\wedge\ (X_3 \neq true \wedge X_3 \neq false)$
	$C = (f(x_1) > x_1)$
$\emptyset$	$[]$
	$A = (x_1{\downarrow} = X_1) \wedge (X_1 \neq i(X_2))$
	$C = (f(x_1) > x_1)$

The first constraint formula A is satisfiable by any instantiation θ such that $\theta X_4 = 0$ and $\theta x_1 = i(0)$. The second one is satisfiable by any instantiation θ such that $\theta X_4 = s(0)$ and $\theta x_1 = i(s(0))$. The third one is satisfiable by any instantiation θ such that $\theta X_3 = zero(i(0))$, $\theta X_2 = i(0)$ and $\theta x_1 = i(i(0))$.

The following step ends the third branch, whose strategy evaluation list is empty.

Stop

$g(X_4)$	$[1]$
	$A = (x_1{\downarrow} = X_1 \wedge X_1 = i(X_2) \wedge zero(X_2){\downarrow} = X_3 \wedge X_3 = true \wedge X_2 = X_4)$
	$C = (f(x_1) > x_1)$
$f(h(X_4))$	$[1;0]$
	$A = (x_1{\downarrow} = X_1 \wedge X_1 = i(X_2) \wedge zero(X_2){\downarrow} = X_3 \wedge X_3 = false \wedge X_2 = X_4)$
	$C = (f(x_1) > x_1)$
$\emptyset$	$[]$
	$A = (x_1{\downarrow} = X_1 \wedge X_1 = i(X_2) \wedge zero(X_2){\downarrow} = X_3) \wedge (X_3 \neq true \wedge X_3 \neq false)$
	$C = (f(x_1) > x_1)$
$\emptyset$	$[]$
	$A = (x_1{\downarrow} = X_1) \wedge (X_1 \neq i(X_2))$
	$C = (f(x_1) > x_1)$

Abstract (*twice*)

$g(X_4)$ $[]$

$A = (x_1{\downarrow} = X_1 \wedge X_1 = i(X_2) \wedge zero(X_2){\downarrow} = X_3 \wedge X_3 = true \wedge X_2 = X_4)$

$C = (f(x_1) > x_1)$

$f(X_5)$ $[0]$

$A = (x_1{\downarrow} = X_1 \wedge X_1 = i(X_2) \wedge zero(X_2){\downarrow} = X_3 \wedge X_3 = false \wedge X_2 = X_4 \wedge h(X_4){\downarrow} = X_5)$

$C = (f(x_1) > x_1)$

$\emptyset$ $[]$

$A = (x_1{\downarrow} = X_1 \wedge X_1 = i(X_2) \wedge zero(X_2){\downarrow} = X_3) \wedge (X_3 \neq true \wedge X_3 \neq false)$

$C = (f(x_1) > x_1)$

$\emptyset$ $[]$

$A = (x_1{\downarrow} = X_1) \wedge (X_1 \neq i(X_2))$

$C = (f(x_1) > x_1)$

Abstract trivially applies on $g(X_4)$: since X_4 is an NF-variable, there is no need to abstract it.

The second **Abstract** applies on$f(h(X_4))$, thanks to Proposition 4.4. Indeed, $\mathcal{U}(h(X_4)) = \{h(0) \to i(0), h(x) \to s(i(x))\}$, and both rules can be oriented by the same LPO as previously with the additional precedence $h \succ_{\mathcal{F}} i$ and $h \succ_{\mathcal{F}} s$. Then we have $TERMIN(h(X_4))$.

The first constraint formula has not changed, while the second one is now satisfiable by any instantiation θ such that $\theta X_5 = s(i(s(0)))$, $\theta X_4 = s(0)$ and $\theta x_1 = i(s(0))$.

Narrow−N (*on the second branch*)

$g(X_4)$ $[]$

$A = (x_1{\downarrow} = X_1 \wedge X_1 = i(X_2) \wedge zero(X_2){\downarrow} = X_3 \wedge X_3 = true \wedge X_2 = X_4)$

$C = (f(x_1) > x_1)$

$f(X_5)$ $[]$

$A = (x_1{\downarrow} = X_1 \wedge X_1 = i(X_2) \wedge zero(X_2){\downarrow} = X_3 \wedge X_3 = false \wedge X_2 = X_4 \wedge h(X_4){\downarrow} = X_5)$

$C = (f(x_1) > x_1)$

$\emptyset$ []
$A = (x_1{\downarrow} = X_1 \wedge X_1 = i(X_2) \wedge zero(X_2){\downarrow} = X_3) \wedge (X_3 \neq true \wedge X_3 \neq false)$
$C = (f(x_1) > x_1)$
$\emptyset$ []
$A = (x_1{\downarrow} = X_1) \wedge (X_1 \neq i(X_2))$
$C = (f(x_1) > x_1)$

One could have tried to narrow $f(X_5)$, by using the first rule and the narrowing substitution $\sigma_{NF} = (X_5 = i(X_6) \wedge x''' = X_6)$. But then $A \wedge \sigma_{NF}$ would be $(x_1{\downarrow} = X_1 \wedge X_1 = i(X_2) \wedge zero(X_2){\downarrow} = X_3 \wedge X_3 = false \wedge X_2 = X_4 \wedge h(X_4){\downarrow} = X_5 \wedge X_5 = i(X_6))$. For any θ satisfying $A \wedge \sigma_{NF}$, θ must be such that $\theta h(X_4){\downarrow} = h(\theta X_4{\downarrow}){\downarrow} = i(\theta X_6)$. If $\theta X_4{\downarrow} \neq 0$, then, according to $\mathcal{R}$, $h(\theta X_4{\downarrow}) \to s(i(\theta X_4{\downarrow}))$, where s is a constructor. Then we cannot have $h(\theta X_4{\downarrow}){\downarrow} = i(\theta X_6)$, so θ must be such that $\theta X_4{\downarrow} = 0$. But then $\theta zero(X_4){\downarrow} = true$, which makes $A \wedge \sigma_{NF}$ unsatisfied. Therefore there is no narrowing.

The process is ended by a double application of **Stop**.

Stop (*twice*)
$\emptyset$ []
$A = (x_1{\downarrow} = X_1 \wedge X_1 = i(X_2) \wedge zero(X_2){\downarrow} = X_3 \wedge X_3 = true \wedge X_2 = X_4)$
$C = (f(x_1) > x_1)$
$\emptyset$ []
$A = (x_1{\downarrow} = X_1 \wedge X_1 = i(X_2) \wedge zero(X_2){\downarrow} = X_3 \wedge X_3 = false \wedge X_2 = X_4 \wedge h(X_4){\downarrow} = X_5)$
$C = (f(x_1) > x_1)$
$\emptyset$ []
$A = (x_1{\downarrow} = X_1 \wedge X_1 = i(X_2) \wedge zero(X_2){\downarrow} = X_3) \wedge (X_3 \neq true \wedge X_3 \neq false)$
$C = (f(x_1) > x_1)$
$\emptyset$ []
$A = (x_1{\downarrow} = X_1) \wedge (X_1 \neq i(X_2))$
$C = (f(x_1) > x_1)$

Like for the defined symbols $ite, zero, h$, the inference rules apply successfully through one **Abstract**, **Narrow−Y**, **Abstract** without abstraction, **Narrow−N** and **Stop** application. Therefore $\mathcal{R}$ is LS-terminating.

Let us now give an example that cannot be handled with the context-sensitive approach.

Example 4.6 Let R be the following TRS

$$f(a, g(x)) \to f(a, h(x))$$
$$h(x) \to g(x)$$

with the LS-strategy : $LS(f) = [0; 1; 2]$, $LS(h) = [0]$ and $LS(g) = [1]$. Applying the rules on $f(x_1, x_2)$, we get:

$f(x_1, x_2)$	$[0; 1; 2]$
	$A = \top$
	$C = \top$
Narrow−Y	
$f(a, h(x_3))$	$[0; 1; 2]$
	$\sigma = (x_1 = a \wedge x_2 = g(x_3) \wedge x' = x_3)$
	$A = (x_1 = a \wedge x_2 = g(x_3))$
	$C = \top$
$f(x_1, x_2)$	$[1; 2]$
	$A = (x_1 \neq a \vee x_2 \neq g(x_3))$
	$C = \top$
Abstract−Stop	
$f(a, h(x_3))$	$[0; 1; 2]$
	$A = (x_1 = a \wedge x_2 = g(x_3))$
	$C = \top$
$\emptyset$	$[]$
	$A = (x_1 \neq a \vee x_2 \neq g(x_3))$
	$C = (f(x_1, x_2) > x_1, x_2)$
Narrow−N	
$f(a, h(x_3))$	$[1; 2]$
	$A = (x_1 = a \wedge x_2 = g(x_3))$
	$C = \top$
$\emptyset$	$[]$
	$A = (x_1 \neq a \vee x_2 \neq g(x_3))$
	$C = (f(x_1, x_2) > x_1, x_2)$

Abstract−Stop

$\emptyset$ $[]$

$A = (x_1 = a \wedge x_2 = g(x_3))$

$C = (f(x_1, x_2) > a, h(x_3))$

$\emptyset$ $[]$

$A = (x_1 \neq a \vee x_2 \neq g(x_3))$

$C = (f(x_1, x_2) > x_1, x_2)$

Applying the rules on $h(x_1)$, we get:

$h(x_1)$ $[0]$

$A = \top$ $C = \top$

Narrow−Y

$g(x_2)$ $[1]$

$\sigma = (x_1 = x_2 \wedge x' = x_2)$

$A = (x_1 = x_2)$ $C = \top$

Abstract

$g(X)$ $[]$

$A = (x_1 = x_2 \wedge x_2{\downarrow} = X)$ $C = (h(x_1) > x_2)$

Stop

$\emptyset$ $[]$

$A = (x_1 = x_2 \wedge x_2{\downarrow} = X)$ $C = (h(x_1) > x_2)$

Let us finally give another example, that cannot be be handled with the context-sensitive approach:the TRS $\{f(b) \to c, g(x) \to h(x), h(c) \to g(f(a)), a \to b\}$ with the LS-strategy $LS(f) = [0;1], LS(g) = LS(h) = [1;0], LS(a) = [0]$, and two examples of innermost rewriting: the well-known Toyamas'example $\{f(0,1,x) \to f(x,x,x), g(x,y) \to x, g(x,y) \to y\}$ with the LS-strategy $LS(f) = [1;2;3;0], LS(g) = [1;2;0], LS(0) = LS(1) = [0]$, and $\{f(f(x)) \to f(f(x)), f(a) \to a\}$ with the LS-strategy $LS(f) = [1;0], LS(a) = [0]$, that is innermost terminating on $\mathcal{T}(\mathcal{F})$, but is not on $\mathcal{T}(\mathcal{F}, \mathcal{X})$. The complete development of these examples can be found in the [9].

5 Conclusion

In this paper, we have proposed a method to prove termination of term rewriting with local strategies on operators by explicit induction on the termination property. Our method works on the ground term algebra using as induction relation an $\mathcal{F}$-stable ordering having the subterm property. The general proof principle relies on the simple idea that for establishing termination of a ground term t, it is enough to suppose that terms smaller than t for this ordering are terminating, and that rewriting the context leads to terminating chains. It-

erating this process until obtaining a context which is not reducible anymore establishes the termination of t.

More precisely, the method is applied on terms of the form $g(x_1, \ldots, x_m)$, where g is a defined symbol, and consists in iterating the application of two steps: an abstraction step, replacing immediate subterms by NF-variables, representing any of their normalized instances, and a narrowing step, reducing the resulting term according to the different possible instances of its variables. These two steps are iterated until getting a term for which one can easily say that all ground instances are terminating. The important point to automate our proof principle is the satisfaction of the ordering constraints for **Abstract**, **Abstract−Stop** and **Stop−Ind**.

On many examples as those given in the paper, this is immediate since they are ensured by the subterm property. In many other cases, a LPO is sufficient to satisfy these constraints. Note that such an LPO does not suffice when it is used in the classical way (any left-hand side of rule is greater than any corresponding right-hand side) since we can handle systems that are not terminating for standard rewriting. Testing satisfiability of a constraint formula A remains simple to handle in practice.

We now have a semi-automatic implementation of the inference rules and the strategy for the leftmost innermost case, that can be expressed by a local strategy on operators. It has been implemented in ELAN [4], which is a logical environment for specifying and prototyping deduction systems in a rule based language with strategies. In this case, as any list of reduction positions ends with the top position, the formula A never contains disequations. Sufficient conditions are implemented to detect unsatisfiability of A, by identifying the reducible right-hand sides. The subterm property of the induction ordering to be found is also implemented, allowing the first application of the rule **Abstract** to be completely automatic. Given a TRS, the program interacts with the user and builds the derivation tree resulting from the application of the inference rules according to the strategy we have defined in this paper [8]. Execution examples are available.[1] We also have proposed a variant of the previous implementation, reducing the interaction with the user, by ignoring the satisfiability problem of A. In this case, the obtained proof derivation tree contains the tree we would obtain in using A: states for which A is not satisfiable just correspond to empty sets of ground terms. We thus have in general more computations, but with considerably less user interactions. Moreover, for many examples, the subterm property is the only required property on the induction ordering, so there are no user interactions to test the satisfiability of C and the algorithm is completely automatic.

Our process can also be extended to other strategies. We recently have proposed inference rules for the outermost strategy [12]. Moreover, since our induction principle is based on the rewriting relation itself, the extension to

[1] `http://www.loria.fr/~fissore/Demo/description.html`

equational rewriting as well as to typed rewriting seems to be easy.

Acknowledgments: We would like to thank Salvador Lucas for his helpful comments on this work.

References

[1] T. Arts and J. Giesl. Proving innermost normalisation automatically. In *Proceedings 8th Conference on Rewriting Techniques and Applications, Sitges (Spain)*, volume 1232 of *Lecture Notes in Computer Science*, pages 157–171. Springer-Verlag, 1997.

[2] T. Arts and Giesl. J. Proving innermost normalization automatically. Technical Report 96/39, Technische Hochschule Darmstadt, Germany, 1996.

[3] P. Borovanský, C. Kirchner, H. Kirchner, P.-E. Moreau, and Ch. Ringeissen. An Overview of ELAN. In C. Kirchner and H. Kirchner, editors, *Proc. Second Intl. Workshop on Rewriting Logic and its Applications*, Electronic Notes in Theoretical Computer Science, Pont-à-Mousson (France), September 1998. Elsevier.

[4] Peter Borovanský, Claude Kirchner, Hélène Kirchner, Pierre-Etienne Moreau, and Christophe Ringeissen. An overview of ELAN. In Claude Kirchner and Hélène Kirchner, editors, *Proceedings of the second International Workshop on Rewriting Logic and Applications*, volume 15, `http://www.elsevier.nl/locate/entcs/volume15.html`, Pont-à-Mousson (France), September 1998. Electronic Notes in Theoretical Computer Science. Report LORIA 98-R-316.

[5] M. Clavel, S. Eker, P. Lincoln, and J. Meseguer. Principles of Maude. In J. Meseguer, editor, *Proceedings of the 1st International Workshop on Rewriting Logic and its Applications*, volume 5 of *Electronic Notes in Theoretical Computer Science*, Asilomar, Pacific Grove, CA, USA, September 1996. North Holland.

[6] Nachum Dershowitz and Jean-Pierre Jouannaud. *Handbook of Theoretical Computer Science*, volume B, chapter 6: Rewrite Systems, pages 244–320. Elsevier Science Publishers B. V. (North-Holland), 1990. Also as: Research report 478, LRI.

[7] S. Eker. Term rewriting with operator evaluation strategies. In C Kirchner and H. Kirchner, editors, *Proceedings of the 2nd International Workshop on Rewriting Logic and its Applications*, Pont-à-Mousson, France, September 1998.

[8] O. Fissore. Terminaison par induction. Mémoire de DEA, Université Henri Poincaré – Nancy 1, June 2000.

[9] O. Fissore, I. Gnaedig, and H. Kirchner. Termination of rewriting with local strategies. Technical Report A01-R-177, LORIA, Nancy, France, 2001.

[10] K. Futatsugi and A. Nakagawa. An overview of CAFE specification environment – an algebraic approach for creating, verifying, and maintaining formal specifications over networks. In *Proceedings of the 1st IEEE Int. Conference on Formal Engineering Methods*, 1997.

[11] J. Giesl and Middeldorp A. Transforming Context-Sensitive Rewrite Systems. In *Proceedings of the 10th International Conference on Rewriting Techniques and Applications*, volume 1631 of *Lecture Notes in Computer Science*, pages 271–285, Trento, Italy, 1999. Springer-Verlag.

[12] I. Gnaedig, H. Kirchner, and O. Fissore. Induction for innermost and outermost ground termination. Technical Report A01-R-178, LORIA, Nancy, France, 2001.

[13] J. Goguen, T. Winkler, J. Meseguer, K. Futatsugi, and J.P. Jouannaud. Introducing OBJ3. Technical report, Computer Science Laboratory, SRI International, march 1992.

[14] P. Klint. A meta-environment for generating programming environments. *ACM Transactions on Software Engineering and Methodology*, 2:176–201, 1993.

[15] M.R.K. Krishna Rao. Some characteristics of strong normalization. *Theoretical Computer Science*, 239:141–164, 2000.

[16] S. Lucas. Termination of context-sensitive rewriting by rewriting. In *Proceedings of the 23rd International Colloquium on Automata, Languages and Programming*, volume 1099 of *Lecture Notes in Computer Science*, pages 122–133. Springer-Verlag, 1996.

[17] S. Lucas. Context-sensitive rewriting strategies. Technical Report DSIC-II/7/00, Departamento de Sistemas Informáticos y Computación, Universidad Politécnica de Valencia, Spain, 2000.

[18] Nakamura M. and Ogata K. The evaluation strategy for head normal form with and without on-demand flags. In K. Futatsugi, editor, *Proceedings of the Third International Workshop on Rewriting Logic and its Applications, WRLA'2000*, pages 211–227, Kanazawa City Cultural Halt, Kanazawa, Japan, September 2000. Electronic Notes in Theoretical Computer Science.

[19] A. Middeldorp and E. Hamoen. Completeness results for basic narrowing. *Applicable Algebra in Engineering, Communication and Computation*, 5(3 & 4):213–253, 1994.

[20] E. Visser. Stratego: A Language for Program Transformation based on Rewriting Strategies. System Description for Stratego 0.5. In Aart Middeldorp, editor, *Proceedings of the 12th International Conference on Rewriting Techniques and Applications*. Springer-Verlag, 2001. To appear.

[21] H. Zantema. Termination of context-sensitive rewriting. In *Proceedings of the 8th International Conference on Rewriting Techniques and Applications*, volume 1232 of *Lecture Notes in Computer Science*, pages 172–186. Springer-Verlag, 1997.

Appendix

A The usable rules

To prove Lemma 4.3, we need the next five lemmas. The first one (Lemma A.1) highlights the relationship between LS-positions and usable rules. Lemmas A.2 and A.3 are corollaries of this lemma.
Lemma A.4 expresses the relationship between a rewrite rule and the usable rules of the rhs, and the key Lemma A.5 explains where the symbols of the redex positions come from.

Lemma A.1 *Whatever $t \in \mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N})$ and $p \in LS - POS(t)$, the usable rules of t contain the usable rules of $t|_p$.*

$$\forall t \in \mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N}), \forall p \in LS - POS(t) : \mathcal{U}(t|_p) \subseteq \mathcal{U}(t)$$

Proof.

We proceed by structural induction on t.
If $t \in \mathcal{N}$, then $LS - POS(t) = \emptyset$, and the property is trivial.
If $t \in \mathcal{X} \cup \mathcal{C}$, then $LS - POS(t) = \{\epsilon\}$ and then the property is trivially satisfied.
If $t = f(u_1, \ldots, u_n)$, we then have two cases :

- either $0 \in LS(f)$, and then $LS - POS(t) = \{\epsilon\} \bigcup_{i \in LS(f) \setminus 0} \{i.p \mid p \in LS - POS(u_i)\}$.
  - if $p = \epsilon$, then the property is trivial ;
  - if $p \in \bigcup_{i \in LS(f) \setminus 0} \{i.p \mid p \in LS - POS(u_i)\}$, then $\exists j \in LS(f), j \neq 0$, such that $p \in LS - POS(u_j)$, that is $\exists p' \in LS - POS(u_j)$ such that $p = j.p'$. By induction hypothesis on u_j, strict subterm of t, we have the property : $\forall p \in LS - POS(u_j) : \mathcal{U}(u_j|_p) \subseteq \mathcal{U}(u_j)$. Then, since $p' \in LS - POS(u_j)$, we have $\mathcal{U}(u_j|_{p'}) \subseteq \mathcal{U}(u_j)$, that is $\mathcal{U}(t|_p) \subseteq \mathcal{U}(u_j)$. Moreover, by definition of the usable rules, we have : $\mathcal{U}(t) = Rls(f) \bigcup_{i \in LS(f), i \neq 0} \mathcal{U}(u_i) \bigcup_{l \to r' \in Rls'(f)} \mathcal{U}(r')$, henceforth $\mathcal{U}(u_j) \subseteq \mathcal{U}(t)$. Therefore we get $\mathcal{U}(t|_p) \subseteq \mathcal{U}(t)$.
- or $0 \notin LS(f)$, and then $LS - POS(t) = \bigcup_{i \in LS(f)} \{i.p \mid p \in LS - POS(u_i)\}$. Since $p \in \bigcup_{i \in LS(f)} \{i.p \mid p \in LS - POS(u_i)\}$, then $\exists j \in LS(f)$ such that $p \in LS - POS(u_j)$, that is $\exists p' \in LS - POS(u_j)$ such that $p = j.p'$. By induction hypothesis on u_j, strict subterm of t, we have the property : $\forall p \in LS - POS(u_j) : \mathcal{U}(u_j|_p) \subseteq \mathcal{U}(u_j)$. Then, since $p' \in LS - POS(u_j)$, we have $\mathcal{U}(u_j|_{p'}) \subseteq \mathcal{U}(u_j)$, that is $\mathcal{U}(t|_p) \subseteq \mathcal{U}(u_j)$. Moreover, by definition of the usable rules, we have : $\mathcal{U}(t) = \bigcup_{i \in LS(f)} \mathcal{U}(u_i)$, henceforth $\mathcal{U}(u_j) \subseteq \mathcal{U}(t)$. Therefore we get $\mathcal{U}(t|_p) \subseteq \mathcal{U}(t)$.

□

Lemma A.2 *Let $\mathcal{R}$ be a TRS and $t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$. If a variable occurs in t at an LS-position, then $\mathcal{U}(t) = \mathcal{R}$.*

Proof. This lemma is a corollary of the Lemma A.1. Indeed, if x occurs at an LS-position p in t then, according to Lemma A.1, $\mathcal{U}(t|_p) \subseteq \mathcal{U}(t)$. Since $t|_p = x \in \mathcal{X}$, by definition of the usable rules, we have $\mathcal{U}(t|_p) = \mathcal{R}$, therefore $\mathcal{R} \subseteq \mathcal{U}(t)$. Since we also have $\mathcal{U}(t) \subseteq \mathcal{R}$, we get $\mathcal{U}(t) = \mathcal{R}$. □

Lemma A.3 *Let $\mathcal{R}$ be a TRS on a set $\mathcal{F}$ of symbols and $t \in \mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N})$. Then, every symbol $f \in \mathcal{F}$ occuring in t at a redex position is such that $Rls(f) \subseteq \mathcal{U}(t)$.*

Proof. This lemma is also a corollary of Lemma A.1. Indeed, if f occurs in t at a redex position p, then $p \in LS - POS(t)$. According to Lemma A.1, $\mathcal{U}(t|_p) \subseteq \mathcal{U}(t)$. Since f is a redex symbol, we have $0 \in LS(f)$, and then $\mathcal{U}(t|_p) = Rls(f) \bigcup_{i \in LS(f), i \neq 0} \mathcal{U}(u_i) \bigcup_{l \to r' \in Rls'(f)} \mathcal{U}(r')$. Consequently, we have $Rls(f) \subseteq \mathcal{U}(t)$. □

Lemma A.4 *Let $\mathcal{R}$ be a TRS on a set $\mathcal{F}$ of symbols, $\mathcal{R}' = NF(\mathcal{R})$ and $t \in \mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N})$. For each $l \to r \in \mathcal{R}$, we note $l \to r'$ the rule $NF(l \to r)$. Then $l \to r \in \mathcal{U}(t) \Rightarrow \mathcal{U}(r') \subseteq \mathcal{U}(t)$.*

Proof. According to Lemma A.2, if a variable of $\mathcal{X}$ occurs in t at an LS-position, then $\mathcal{U}(t) = \mathcal{R}$, and then the property is trivially true. We will then suppose in the following that t does not contain any variable of $\mathcal{X}$ at an LS-position.

Let $l \to r \in \mathcal{U}(t)$. By definition of $\mathcal{U}(t)$, among all recursive applications of the definition of $\mathcal{U}$ in $\mathcal{U}(t)$, there is an application $\mathcal{U}(t')$ of $\mathcal{U}$ to some term t' such that $\mathcal{U}(t') = Rls(g) \bigcup_{i \in LS(g), i \neq 0} \mathcal{U}(t'|_i) \bigcup_{l \to r' \in Rls'(g)} \mathcal{U}(r')$, with $\mathcal{U}(t') \subseteq \mathcal{U}(t)$, and $l \to r \in Rls(g)$, with $g = top(l)$.

Then we have $l \to r' \in Rls'(g)$, and then $\mathcal{U}(r') \in \cup_{l \to r' \in Rls'(g)} \mathcal{U}(r')$, and then $l \to r' \in \mathcal{U}(t')$, and therefore $l \to r' \in \mathcal{U}(t)$. □

Lemma A.5 *Let $\mathcal{R}$ be a TRS on a set $\mathcal{F}$ of symbols and $\mathcal{R}' = NF(\mathcal{R})$, so that for each $l \to r' \in \mathcal{R}'$, no variable of $\mathcal{X}$ occurs in r' at an LS-position. Let $t \in \mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N})$ such that no variable of $\mathcal{X}$ occurs in t at an LS-position. Whatever α ground substitution such that $\mathcal{V}ar(t) \subseteq Dom(\alpha)$, and $\alpha t \to_{p_1, l_1 \to r_1} t_1 \to_{p_2, l_2 \to r_2} t_2 \to \ldots \to_{p_n, l_n \to r_n} t_n$ LS-rewrite chain starting from αt, the symbol of $t_k, 1 \leq k \leq n$ at a redex position of t_k is either a redex symbol of t or one of the $r_i, i \in [1..k]$.*

Proof. We proceed by induction on the length of the derivation.
For an empty derivation, the property has to be checked on αt. In this case, let us show that whatever $x \in \mathcal{X} \cup \mathcal{N}$ variable of t, αx does not introduce redex symbol in t. Let x be a variable of t occuring at position p.
For $x \in \mathcal{X}$, by hypothesis, p is not an LS-position of t. Therefore αt cannot reduce at a suffix position of p.
For $x \in \mathcal{N}$, by definition of the NF-variables, αX is normalized. Therefore $\alpha t|_p$ has no redex symbol. Hence, the property is true for αt.

Let us show the property for the first rewriting step $\alpha t \rightarrow_{p_1, l_1 \rightarrow r_1} t_1$. By definition of the rewriting, $\exists \sigma : \sigma l_1 = \alpha t|_{p_1}$ and $t_1 = \alpha t[\sigma r_1]_{p_1}$. Let f be a redex symbol of t_1 :

- either it is a symbol of αt, and then we showed that f occurs in t at an LS-position ;
- or it is a symbol of σr_1, and then :
  - either it belongs to r_1, and the property is satisfied ;
  - or it belongs to σx, for $x \in \mathcal{V}ar(r_1)$. Since $\mathcal{V}ar(r_1) \subseteq \mathcal{V}ar(l_1)$, we have $x \in \mathcal{V}ar(l_1)$.
    - if x does not occur at an LS-position in l_1, then $x \in \mathcal{V}ar(r'_1)$, which is in contradiction with the hypothesis of the current lemma ;
    - if x occurs at an LS-position in l_1, since $\sigma l_1 = \alpha t|_{p_1}$, then σx is a subterm of αt at an LS-position. Then, for a redex symbol f of σx, f is a redex symbol of αt. Then, as shown before, f is a redex symbol of t.

Let us now suppose the property true for any term of the LS-rewrite chain $t_0 = \alpha t \rightarrow_{p_1, l_1 \rightarrow r_1} t_1 \rightarrow \ldots \rightarrow_{p_k, l_k \rightarrow r_k} t_k$, and let us consider $t_k \rightarrow_{p_{k+1}, l_{k+1} \rightarrow r_{k+1}} t_{k+1}$. We then have to show that every defined symbol of t_{k+1} at a redex position comes either from t or from one of the $r_i, i \in [1..k+1]$. By hypothesis, there exists a substitution σ such that $\sigma l_{k+1} = t_k|_{p_{k+1}}$ and $t_{k+1} = t_k[\sigma r_{k+1}]_{p_{k+1}}$. ¿From the first equality, we can deduce that $\forall x \in Dom(\sigma) \cap \mathcal{V}ar(l_{k+1}), \sigma x$ is a subterm of $t_k|_{p_{k+1}}$. Then, by induction hypothesis on the ground term t_k, we can infer that $\forall x \in Dom(\sigma) \cap \mathcal{V}ar(l_{k+1})$, the symbols at redex positions of σx come either from t or from the $r_i, i \in [1..k]$. Henceforth, since $\mathcal{V}ar(r_{k+1}) \subseteq \mathcal{V}ar(l_{k+1})$, the symbols at redex positions of σr_{k+1} come either from t or from the $r_i, i \in [1..k]$, or from r_{k+1}. Then, by induction hypothesis on the ground term t_k, the property is true at the order $k+1$ on $t_{k+1} = t_k[\sigma r_{k+1}]_{p_{k+1}}$. □

We are now able to prove Lemma 4.3.

Lemma 4.3 *Let $\mathcal{R}$ be a TRS on a set $\mathcal{F}$ of symbols, $\mathcal{R}' = NF(\mathcal{R})$ and for each $l \rightarrow r' \in \mathcal{R}'$, no variable of $\mathcal{X}$ occurs in r' at an LS-position. Let $t \in \mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N})$. Whatever αt ground instance of t and $\alpha t \rightarrow_{p_1, l_1 \rightarrow r_1} t_1 \rightarrow_{p_2, l_2 \rightarrow r_2} t_2 \rightarrow \ldots \rightarrow_{p_n, l_n \rightarrow r_n} t_n$ LS-rewrite chain starting from αt, then $\forall i \in [1..n] : l_i \rightarrow r_i \in \mathcal{U}(t)$.*

Proof. If a variable $x \in \mathcal{X}$ occurs in t at an LS-position then, thanks to Lemma A.2, we have $\mathcal{U}(t) = \mathcal{R}$ and the property is trivially true. We then consider in the following that the variables of $\mathcal{X}$ in t do not occur at an LS-position. We proceed by induction on the length of the derivation.

The property is trivially true if αt is in normal form. For any $\alpha t \rightarrow_{p_1, l_1 \rightarrow r_1} t_1$, since no variable of t occurs at an LS-position, p_1 is a non variable position of t. Let f be the symbol at position p_1 in t. Then, thanks to Lemma A.3, $Rls(f) \subseteq \mathcal{U}(t)$. Moreover, since f is the symbol at the redex position, $l_1 \rightarrow r_1 \in Rls(f)$. Therefore we get $l_1 \rightarrow r_1 \in \mathcal{U}(t)$.

Let us now suppose the property is true for any LS-derivation chain starting

from αt whose length is less or equal to k, and consider the chain: $\alpha t \to_{p_1,l_1\to r_1} t_1 \to_{p_2,l_2\to r_2} t_2 \to \ldots \to_{p_k,l_k\to r_k} t_k \to_{p_{k+1},l_{k+1}\to r_{k+1}} t_{k+1}$. By Lemma A.5 with a derivation of length k, we have two cases:

- either the symbol f at position p_{k+1} in t_k is a symbol of t occuring at an LS-position; then, thanks to Lemma A.3 on t, we get $Rls(f) \subseteq \mathcal{U}(t)$; moreover, since f is the symbol at the redex position, $l_{k+1} \to r_{k+1} \in Rls(f)$, henceforth $l_{k+1} \to r_{k+1} \in \mathcal{U}(t)$;
- or the symbol f at position p_{k+1} in t_k is a symbol of a $r'_i, i \in [1..k]$, occuring at an LS-position ; then, thanks to Lemma A.3 on r'_i, we get $Rls(f) \subseteq \mathcal{U}(r'_i)$; moreover, since f is the symbol at the redex position, $l_{k+1} \to r_{k+1} \in Rls(f)$, henceforth $l_{k+1} \to r_{k+1} \in \mathcal{U}(r'_i)$; by induction hypothesis we have $l_i \to r_i \in \mathcal{U}(t)$ and, thanks to Lemma A.4, we have $\mathcal{U}(r'_i) \subseteq \mathcal{U}(t)$. Henceforth $l_{k+1} \to r_{k+1} \in \mathcal{U}(t)$.

□

Proposition 4.4 *Let $\mathcal{R}$ be a TRS on a set $\mathcal{F}$ of symbols, $\mathcal{R}' = NF(\mathcal{R})$ and for each $l \to r' \in \mathcal{R}'$, no variable of $\mathcal{X}$ occurs in r' at an LS-position. Let $t \in \mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N})$. If there exists a simplification ordering $\succ$ such that $\forall l \to r \in \mathcal{U}(t) : l \succ r$, then any ground instance of t is terminating.*

Proof. First, let us prove the following property:

$\forall t, t' \in \mathcal{T}(\mathcal{F}), [(\exists l \to r$ such that $l \succ r$ and a position $p : t \to_{p,l\to r} t') \Rightarrow t \succ t']$.

If $t \to_{p,l\to r} t'$, then there exists a ground substitution σ such that $\sigma l = t|_p$ and $t' = t[\sigma r]_p$. Then comparing t with t' comes to comparing $t[t|_p]_p = t[\sigma l]_p$ with $t[\sigma r]_p$. Since $l \succ r$ and the ordering is stable by substitution, then we have $\sigma l \succ \sigma r$. Then, since the ordering is $\mathcal{F}$-stable, we get $t[\sigma l]_p \succ t[\sigma r]_p$, that is $t \succ t'$.

Let us then suppose $\exists \alpha$ such that there exists an infinite rewrite chain $\alpha t \to t_1 \to t_2 \to \ldots$

According to the hypotheses of the proposition and Lemma 4.3, $\succ$ is such that $\alpha t \succ t_1 \succ t_2 \succ \ldots$, which is in contradiction with the noetherian property of the ordering $\succ$. □

B The lifting lemma

In order to prove Theorem 4.1, we need a lifting lemma for LS-narrowing, lying the following two propositions (the first one is obvious).

Proposition B.1 *Let $t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ and σ a substitution of $\mathcal{T}(\mathcal{F}, \mathcal{X})$. Then $Var(\sigma t) = (Var(t) - Dom(\sigma)) \cup Ran(\sigma_{Var(t)})$.*

Given a set of variables $\mathcal{V}$ and two substitutions σ and θ, we write $\sigma = \theta[V]$ iff $\forall x \in \mathcal{V} : \sigma x = \theta x$.

Proposition B.2 *Suppose we have substitutions σ, μ, ν and sets A, B of variables such that $(B - Dom(\sigma)) \cup Ran(\sigma) \subseteq A$. If $\mu = \nu[A]$ then $\mu\sigma = \nu\sigma[B]$.*

Proof.

Let us consider $(\mu\sigma)_B$, which can be divided as follows :
$(\mu\sigma)_B = (\mu\sigma)_{B \cap Dom(\sigma)} \cup (\mu\sigma)_{B - Dom(\sigma)}$.
For $x \in B \cap Dom(\sigma)$, we have $\mathcal{V}ar(\sigma x) \subseteq Ran(\sigma)$, and then $(\mu\sigma)x = \mu(\sigma x) = \mu_{Ran(\sigma)}(\sigma x) = (\mu_{Ran(\sigma)}\sigma)x$. Therefore $(\mu\sigma)_{B \cap Dom(\sigma)} = (\mu_{Ran(\sigma)}\sigma)_{B \cap Dom(\sigma)}$.
For $x \in B - Dom(\sigma)$, we have $\sigma x = x$, and then $(\mu\sigma)x = \mu(\sigma x) = \mu x$. Therefore we have $(\mu\sigma)_{B - Dom(\sigma)} = \mu_{B - Dom(\sigma)}$. Henceforth we get $(\mu\sigma)_B = (\mu_{Ran(\sigma)}\sigma)_{B \cap Dom(\sigma)} \cup \mu_{B - Dom(\sigma)}$.
By a similar reasoning, we get $(\nu\sigma)_B = (\nu_{Ran(\sigma)}\sigma)_{B \cap Dom(\sigma)} \cup \nu_{B - Dom(\sigma)}$.
By hypothesis, we have $Ran(\sigma) \subseteq A$ and $\mu = \nu[A]$, then we can infer $\mu_{Ran(\sigma)} = \nu_{Ran(\sigma)}$. Likewise, since $B - Dom(\sigma) \subseteq A$, we have $\mu_{BDom(\sigma)} = \nu_{B - Dom(\sigma)}$.
Then we have $(\mu\sigma)_B = (\mu_{Ran(\sigma)}\sigma)_{B \cap Dom(\sigma)} \cup \mu_{B - Dom(\sigma)} = (\nu_{Ran(\sigma)}\sigma)_{B \cap Dom(\sigma)} \cup \nu_{B - Dom(\sigma)} = (\nu\sigma)_B$ with the assumptions used in the second equality. Therefore $(\mu\sigma) = (\nu\sigma)[B]$. □

Lemma B.3 *Let $\mathcal{R}$ be a TRS. Let $s \in \mathcal{T}(\mathcal{F}, \mathcal{X})$, α a ground substitution such that αs is reducible at a position $p \in \overline{\mathcal{O}}(s)$, and $\mathcal{Y} \subseteq \mathcal{X} - \{\mathcal{V}ar(l) \mid l \to r \in \mathcal{R}\}$ a set of variables such that $Var(s) \cup Dom(\alpha) \subseteq \mathcal{Y}$. If $\alpha s \to_{p, l \to r} t'$, then there exist a term $s' \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ and substitutions β, σ such that :*

1. $s \rightsquigarrow_{p, l \to r, \sigma} s'$,
2. $\beta s' = t'$,
3. $\beta\sigma = \alpha[\mathcal{Y}]$.

Proof. The proof is essentially borrowed (hardly adapted) from [19].

If $\alpha s \to_{p, l \to r} t'$, then there exists a substitution τ such that $Dom(\tau) \subseteq \mathcal{V}ar(l)$ and $(\alpha s)|_p = \tau l$. Moreover, since $p \in \overline{\mathcal{O}}(s)$, then $(\alpha s)|_p = \alpha(s|_p)$.
Let $\mu = \alpha \cup \tau$. Then we have : $\mu(s|_p) = \alpha(s|_p) = \tau l = \mu l$, therefore $s|_p$ and l are unifiable. Let us note σ the most general unifier of $s|_p$ and l, and $s' = \sigma(s[r]_p)$. Therefore, by definition : $s \rightsquigarrow_{[p, l \to r, \sigma]} s'$, and then the point 1. of the current lemma holds.

Since $\sigma \leq \mu$, there exists a substitution ρ such that $\rho\sigma = \mu$. Let $\mathcal{Y}_1 = (\mathcal{Y} - Dom(\sigma)) \cup Ran(\sigma)$. We define $\beta = \rho_{\mathcal{Y}_1}$. Clearly $Dom(\beta) \subseteq \mathcal{Y}_1$.
We now want to show that $\mathcal{V}ar(s') \subseteq \mathcal{Y}_1$, by the following reasoning :

- by definition, $s' = \sigma(s[r]_p)$, therefore we have $\mathcal{V}ar(s') = \mathcal{V}ar(\sigma(s[r]_p))$;
- the rule $l \to r$ is such that $\mathcal{V}ar(r) \subseteq \mathcal{V}ar(l)$, therefore we have $\mathcal{V}ar(\sigma(s[r]_p)) \subseteq \mathcal{V}ar(\sigma(s[l]_p))$, and then, thanks to the previous point, $\mathcal{V}ar(s') \subseteq \mathcal{V}ar(\sigma(s[l]_p))$;
- by definition, we have $\sigma(s[l]_p) = \sigma s[\sigma l]_p$ and, since σ unifies l and $s|_p$, we get $\sigma(s[l]_p) = \sigma s[\sigma(s|_p)]_p = \sigma s[s|_p]_p = \sigma(s)$ and, thanks to the previous point :

$\mathcal{V}ar(s') \subseteq \mathcal{V}ar(\sigma(s))$;

- according to Proposition B.1, we have $\mathcal{V}ar(\sigma(s)) = (\mathcal{V}ar(s) - Dom(\sigma)) \cup Ran(\sigma_{\mathcal{V}ar(s)})$; by definition, $\mathcal{V}ar(s) \subseteq \mathcal{Y}$ and $Ran(\sigma_{\mathcal{V}ar(s)}) \subseteq Ran(\sigma)$, therefore $\mathcal{V}ar(\sigma(s)) \subseteq (\mathcal{Y} - Dom(\sigma)) \cup Ran(\sigma)$, that is $\mathcal{V}ar(\sigma(s)) \subseteq \mathcal{Y}_1$. Therefore, with the previous point, we get $Var(s') \subseteq \mathcal{Y}_1$.

¿From $Dom(\beta) \subseteq \mathcal{Y}_1$ and $Var(s') \subseteq \mathcal{Y}_1$, we can infer $Dom(\beta) \cup Var(s') \subseteq \mathcal{Y}_1$.

We are now going to demonstrate the point 2., that is $\beta s' = t'$.
By definition, $\beta = \rho_{\mathcal{Y}_1}$, therefore $\beta = \rho[\mathcal{Y}_1]$. Since $Dom(\beta) \cup Var(s') \subseteq \mathcal{Y}_1$, we get $\beta s' = \rho s'$. By definition, $\rho s' = \rho\sigma(s[r]_p) = \mu(s[r]_p) = \mu s[\mu r]_p$.
By hypothesis, we have $Dom(\tau) \subseteq \mathcal{V}ar(l)$ and $\mathcal{Y} \cap \mathcal{V}ar(l) = \emptyset$, then we have $\mathcal{Y} \cap Dom(\tau) = \emptyset$. Therefore, from $\mu = \alpha \cup \tau$, we get $\mu = \alpha[\mathcal{Y}]$. Since $\mathcal{V}ar(s) \subseteq \mathcal{Y}$, we get $\mu s = \alpha s$.
Likewise, by hypothesis we have $Dom(\alpha) \subseteq \mathcal{Y}$, $\mathcal{V}ar(r) \subseteq \mathcal{V}ar(l)$ and $\mathcal{Y} \cap Dom(\tau) = \emptyset$, then we get $Var(r) \cap Dom(\alpha) = \emptyset$, and then we have $\mu = \tau[Var(r)]$, and therefore $\mu r = \tau r$.
¿From $\mu s = \alpha s$ and $\mu r = \tau r$ we get $\mu s[\mu r]_p = \alpha s[\tau r]_p$. Since, by hypothesis, $\alpha s \to_p t'$, with $\tau l = (\alpha s)|_p$, then $\alpha s[\tau r]_p = t'$. Finally, we get $\beta s' = t'$ (2).

Next we show that $\beta\sigma = \alpha[\mathcal{Y}]$ (point 3. of the current lemma). Reminding that $\mathcal{Y}_1 = (\mathcal{Y} - Dom(\sigma)) \cup Ran(\sigma)$, Proposition B.2 (with the notations A for $\mathcal{Y}_1$, B for $\mathcal{Y}$, μ for β, ν for ρ and σ for σ) yields $\beta\sigma = \rho\sigma[\mathcal{Y}]$. We already noticed that $\mu = \alpha[\mathcal{Y}]$. Linking these two equalities via the equation $\rho\sigma = \mu$ yields $\beta\sigma = \alpha[\mathcal{Y}]$ (3). □

C The correctness theorem

Theorem 4.1 *Let R be a TRS on $\mathcal{T}(\mathcal{F}, \mathcal{X})$, and $LS : \mathcal{F} \mapsto L(\mathbb{N})$ a LS-strategy such that the constants of $\mathcal{F}$ LS-terminate. If there exists an $\mathcal{F}$-stable ordering $\succ$ having the subterm property, such that for every non constant defined symbol g, $SUCCESS(g, \succ)$, then every term of $\mathcal{T}(\mathcal{F})$ LS-terminates.*

The proof of Theorem 4.1 is given for the rules extended with the $TERMIN$ predicate, as introduced in Section 4.2. These rules are given in Table C.1.

Proof.

We prove by induction on $\mathcal{T}(\mathcal{F})$ that any ground instance $\theta f(x_1, \ldots, x_m)$ of any term $f(x_1, \ldots, x_m) \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ LS-terminates. The induction ordering is constrained along the proof. At the beginning, it has at least to be $\mathcal{F}$-stable and to have the subterm property, which ensures its well-foundedness. Such an ordering always exists on $\mathcal{T}(\mathcal{F})$ (for instance the embedding relation). Let us denote it $\succ$.

The minimal elements of $\mathcal{T}(\mathcal{F})$ for $\succ$ LS-terminate since, by hypothesis, the constants of $\mathcal{F}$ LS-terminate, and for any ordering on ground terms having the subterm property, the set of minimal elements is a subset of the set of constants.

Table C.1
Inference rules for t_{ref} LS-termination

Abstract:
$$\frac{T \cup \{(\{f(u_1,\ldots,u_m)\},\ [p_1,\ldots,p_n],\ A,\ C)\}}{T \cup \{(\{f(U_1,\ldots,U_m)\},\ [0,p_{k+1},\ldots,p_n],\ A \bigwedge_i (u_i\downarrow = X_i),\ C \bigwedge_i H(u_i))\}}$$

where $f(u_1,\ldots,u_m)$ is abstracted by $f(U_1,\ldots,U_m)$ at the pos. $i \in \{i_1,\ldots,i_p\} \subseteq \{p_1,\ldots,p_{k-1}\}$
if $\exists k \in [2..n] : p_1,\ldots,p_{k-1} \neq 0, p_k = 0$ and $(A, C \wedge t_{ref} > u_{k_1},\ldots,u_{k_l})$ is satisfiable for $\{k_1,\ldots,k_l\} \subseteq \{i_1,\ldots,i_p\}$ and $TERMIN(u_j)$ for $j \in \{i_1,\ldots,i_p\} \setminus \{k_1,\ldots k_l\}$
where $H(u_i) = true$ if $TERMIN(u_i)$, $\quad t_{ref} > u_i$ otherwise.

Abstract-Stop:
$$\frac{T \cup \{(\{f(u_1,\ldots,u_m)\},\ [p_1,\ldots,p_n],\ A,\ C)\}}{T \cup \{(\emptyset,\ [],\ A,\ C \bigwedge_{i\in\{i_1,\ldots,i_p\}} H(u_i))\}}$$

where $f(u_1,\ldots,u_m)$ can be abstracted by $f(U_1,\ldots,U_m)$ at the pos. $i_1,\ldots,i_p \in \{p_1,\ldots,p_n\}$
if $p_1,\ldots,p_n \neq 0$ and $(A, C \wedge t_{ref} > u_{k_1},\ldots,u_{k_l})$ is satisfiable for $\{k_1,\ldots,k_l\} \subseteq \{i_1,\ldots,i_p\}$ and $TERMIN(u_i)$ for $i \in \{i_1,\ldots,i_p\} \setminus \{k_1,\ldots k_l\}$
where $H(u_i) = true$ if $TERMIN(u_i)$, $\quad t_{ref} > u_i$ otherwise.

Narrow-Y:
$$\frac{T \cup \{(\{f(u_1,\ldots,u_m)\},\ [0,p_1,\ldots,p_n],\ A,\ C)\}}{T \cup_{i\in[1..l]} \{(\{w_i\},\ LS(top(w_i)),\ A \wedge \sigma_i,\ \sigma_i C)\} \cup COMPL}$$

if $\exists \sigma$ such that $f(u_1,\ldots,u_m) \ \ _{\epsilon,\sigma}\ w$ and $A \wedge \sigma$ is satisfiable
where $w_i, i \in [1..l]$, are all terms such that $f(u_1,\ldots,u_m) \ \ _{\epsilon,\sigma_i}\ w_i$ and $A \wedge \sigma_i$ is satisfiable.

$$COMPL = \begin{cases} \{(\{f(u_1,\ldots,u_m)\},\ [p_1..p_n],\ A \bigwedge_{i=1}^{l} \overline{\sigma_i},\ C)\} & \text{if } (A \bigwedge_{i=1}^{l} \overline{\sigma_i}) \text{ satisfiable} \\ \emptyset & \text{otherwise.} \end{cases}$$

Narrow-N:
$$\frac{T \cup \{(\{f(u_1,\ldots,u_m)\},\ [0,p_1,\ldots,p_n],\ A,\ C)\}}{T \cup \{(\{f(u_1,\ldots,u_m)\},\ [p_1,\ldots,p_n],\ A,\ C)\}}$$

if $f(u_1,\ldots,u_m)$ is not narrowable at the top position
or $\forall \sigma$ narrowing substitution of $f(u_1,\ldots,u_m)$ at the top position , $A \wedge \sigma$ is not satisfiable.

Stop-Ind:
$$\frac{T \cup \{(\{u\},\ [p_1,\ldots,p_n],\ A,\ C)\}}{T \cup \{(\emptyset,\ [],\ A,\ C \wedge H(u))\}}$$

if $(A, C \wedge t_{ref} > u)$ is satisfiable or $TERMIN(u)$.
where $H(u) = true$ if $TERMIN(u)$, $\quad t_{ref} > u$ otherwise.

StopA:
$$\frac{T \cup \{(\{f(u_1,\ldots,u_m)\},\ [p_1,\ldots,p_n],\ A,\ C)\}}{T \cup \{(\{\sharp\},\ [],\ A,\ C)\}}$$

if $p_1 \neq 0$ and neither Abstract nor Abstract−Stop applies.

Stop:
$$\frac{T \cup \{(\{f(u_1,\ldots,u_m)\},\ [],\ A,\ C)\}}{T \cup \{(\emptyset,\ [],\ A,\ C)\}}$$

By subterm property of $\succ$, we have $\theta f(x_1,\ldots,x_m) = f(\theta x_1,\ldots,\theta x_m) \succ \theta x_{i_1},\ldots,\theta x_{i_p}$, with $\{i_1,\ldots,i_p\} = \{i_j |\ i_j \in \{p_1,\ldots p_n\}, i_j \neq 0\}$ where $[p_1,\ldots,p_n] = LS(f)$. Then, by induction hypothesis, let us suppose that $\theta x_{i_1},\ldots,\theta x_{i_p}$ LS-terminate. Let $\theta x_{i_1}\downarrow,\ldots,\theta x_{i_p}\downarrow$ be respectively any of their normal forms.

If f is a constructor, then $f(\theta x_1, \ldots, \theta x_m)$ is irreducible at the top position, as well as its reduced forms. Then $f(\theta x_1, \ldots, \theta x_m)\downarrow = f(\theta x_1, \ldots, \theta x_m)[\theta x_{i_1}\downarrow]_{i_1} \ldots [\theta x_{i_p}\downarrow]_{i_p}$. Therefore $f(\theta x_1, \ldots, \theta x_m)$ is terminating for the LS-rewriting relation.

If f is not a constructor, let us denote it g and prove that $g(\theta x_1, \ldots, \theta x_m)$ LS-terminates for any θ satisfying $(A_0 = \top, C_0 = \top)$, if application of the inference rules on $(\{g(x_1, \ldots, x_m)\}, strat(g), \top, \top)$ terminates on states $(\emptyset, [],\ A_p, C_p)$. Let us denote $g(x_1, \ldots, x_m)$ by t_{ref} in the sequel of the proof.

To each step of the procedure characterized by $(\{t\}, l, A, C)$, we associate the set of ground terms $G = \{\alpha t \mid \alpha \text{ and } \succ \text{ satisfy } (A, C)\}$. Inference rules **Abstract**, **Narrow−Y** and **Narrow−N** transform $(\{t\}, l, A, C)$ into $(\{t'\}, l', A', C')$ to which is associated $G' = \{\beta t' \mid \beta \text{ and } \succ \text{ satisfy } (A', C')\}$. We then prove the following result: if for all $\beta t' \in G'$, $\beta t'$ LS-terminates wrt l', then any αt in G LS-terminates wrt l.

- Either **Abstract** is applied, so $\{f(u_1, \ldots, u_m)\}$ becomes $\{f(U_1, \ldots, U_m)\}$ and $l = [p_1, \ldots, p_n]$ becomes $l' = [0, p_{k+1}, \ldots, p_n]$. For each α such that αt is in G, we prove that there exists a β such that $\beta t'$ is in G' and such that LS-termination of $\beta t'$ wrt l' implies LS-termination of αt wrt l.

  According to the condition of **Abstract**, $\succ$ can be chosen to be such that for any α satisfying A, $\alpha t_{ref} \succ \alpha u_{k_1}, \ldots, \alpha u_{k_l}$ for some $k_1, \ldots, k_l \in \{i_1, \ldots, i_p\} \subseteq \{p_1, \ldots, p_{k-1}\}$, where $i_1, \ldots, i_p$ are the positions at which $f(u_1, \ldots, u_m)$ is abstracted. So by induction hypothesis, the αu_{k_j} LS-terminate. Moreover, for $i \in \{i_1, \ldots, i_p\} \backslash \{k_1, \ldots, k_l\}$, we have $TERMIN(u_i)$ and then αu_i LS-terminates. Finally, for $i \in \{p_1, \ldots, p_{k-1}\} \setminus \{i_1, \ldots, i_p\}$, u_i is either a ground term in normal form or an NF-variable, and then αu_i LS-terminates.

  Then let us define $\beta = \alpha \cup \{X_{p_1} = \alpha u_{p_1}\downarrow, \ldots, X_{p_{k-1}} = \alpha u_{p_{k-1}}\downarrow\}$. Clearly β satisfies (A', C'). Moreover, $\beta f(U_1, ..., U_m) = \alpha f(u_1, ..., u_m)[\alpha u_{p_1}\downarrow]_{p_1} ... [\alpha u_{p_k}\downarrow]_{p_k}$.

  Therefore, LS-termination of the $\beta f(U_1, \ldots, U_m)$ wrt $[0, p_{k+1}, \ldots, p_n]$, for all possible normal forms $\alpha u_{p_1}\downarrow, \ldots, \alpha u_{p_{k-1}}\downarrow$ of $\alpha u_{p_1}, \ldots, \alpha u_{p_{k-1}}$, implies LS-termination of $\alpha f(u_1, \ldots, u_m)$ wrt $[p_1, \ldots, p_{k-1}, 0, p_{k+1}, \ldots, p_n]$.

- Or **Narrow−Y** is applied on $\{f(u_1, \ldots, u_m)\}$ with $l = [0, p_1, \ldots, p_n]$. For any α satisfying (A, C),
  - either $\alpha f(u_1, \ldots, u_m)$ is irreducible at the top position, but may be reduced at the positions $p_1, \ldots, p_n$. So termination of the term $\alpha f(u_1, \ldots, u_m)$, where $f(u_1, \ldots, u_m)$ is produced by **Narrow−Y** together with the strategy $[p_1, \ldots, p_n]$ implies termination of $\alpha f(u_1, \ldots, u_m)$ with the strategy $[0, p_1, \ldots, p_n]$.

    Let us now prove that β satisfies $(A \bigwedge_{i=1}^{l} \overline{\sigma_i}, C)$. In this case, $\beta t' = \alpha t$. Thus, since $A \bigwedge_{i=1}^{l} \overline{\sigma_i}$ characterizes the set of instantiations α such that

$\alpha f(u_1, \ldots, u_m)$ is irreducible, α satisfies $A \bigwedge_{i=1}^{l} \overline{\sigma_i}$. Finally, as $\beta t' = \alpha t$ and α satisfies C, β also satisfies C.

- or $\alpha f(u_1, \ldots, u_m)$ is reducible at the top position. In this case, if $\alpha f(u_1, \ldots, u_m) \to_\epsilon t'$, thanks to Lemma B.3, there exists a narrowing step $f(u_1, \ldots, u_m) \leadsto_\sigma v$ and a substitution β such that $t' = \beta v$. Moreover, this narrowing derivation is effectively produced by **Narrow−Y**, which is applied in all possible ways on $\{f(u_1, \ldots, u_m)\}$. Then LS-termination of the βv implies LS-termination of $\alpha f(u_1, \ldots, u_m)$.

  Let us now prove that β satisfies $(A \wedge \sigma, \sigma C)$. On variables of $f(u_1, \ldots, u_m)$, we have $\alpha = \beta\sigma$. In addition, the domain of β, that is the range of σ, can be extended to the variables of $A \wedge \sigma$ and C by setting $\beta x = \alpha x$ for $x \in (\mathcal{V}ar(A) \cup Var(C)) \setminus Var(f(u_1, \ldots, u_m))$. So $\beta\sigma = \alpha$ on $Var(A \wedge \sigma) \cup Var(C)$, and then $\beta(A \wedge \sigma) = \beta A \wedge \beta\sigma = \alpha A \wedge \alpha$. As α satisfies A, αA is true. Moreover, as αA only contains ground terms, $\alpha A \wedge \alpha$ is true. So β satisfies $A \wedge \sigma$. Finally, since $\alpha C = \beta\sigma C$ and α satisfies C, then β satisfies σC.

- Or **Narrow−N** is applied on $f(u_1, \ldots, u_m)$. For any α satisfying (A, C), $\alpha f(u_1, \ldots, u_m)$ is irreducible at the top position. Then, by taking $\beta = \alpha$ (since $A' = A$ and $C' = C$), LS-termination of $\beta f(u_1, \ldots, u_m)$ wrt $[p_1, \ldots, p_n]$ is equivalent to LS-termination of $\alpha f(u_1, \ldots, u_m)$ wrt $[0, p_1, \ldots, p_n]$.

Let us now prove that the ground instances satisfying (A, C) of each term t removed from T during the application of the rules LS-terminate. The first rule removing terms from T is **Stop**. When **Stop** is applied and removes t from T, then the evaluation strategy of t is []. Then, for any ground substitution α satisfying A and C, αt LS-terminates.

The second rule removing terms from T is **Abstract−Stop**. According to the condition of **Abstract−Stop**, $\succ$ can be chosen to be such that for any α satisfying A, $\alpha t_{ref} \succ \alpha u_{k_1}, \ldots, \alpha u_{k_l}$ for some $k_1, \ldots, k_l \in \{i_1, \ldots, i_p\} \subseteq \{p_1, \ldots, p_n\}$, where $i_1, \ldots, i_p$ are the positions $f(u_1, \ldots, u_m)$ is abstracted at. So by induction hypothesis, the αu_{k_j} LS-terminate. Moreover, for $i \in \{i_1, \ldots, i_p\} \setminus \{k_1, \ldots, k_l\}$, we have $TERMIN(u_i)$ and then αu_i LS-terminates. Finally, for $i \in \{p_1, \ldots, p_n\} \setminus \{i_1, \ldots, i_p\}$, u_i is either a ground term in normal form or an NF-variable, and then αu_i LS-terminates. Therefore, by definition of the evaluation strategy of t, αt LS-terminates.

The third rule removing terms from T is **Stop−Ind**. When **Stop−Ind** is applied and removes t from T, then for any ground substitution α satisfying A,

- either the ordering $\succ$ is such that $\alpha t_{ref} \succ \alpha t$ and then, by induction hypothesis, αt LS-terminates,
- or we have $TERMIN(t)$, and then αt LS-terminates too.

As the process is initialized with $\{t_{ref}\}$ and the constraint problem $(A, C) =$

$(\top, \top)$ satisfiable by any ground substitution, we get that $g(\theta x_1, \ldots, \theta x_m)$ is LS-terminating, for any $t_{ref} = g(x_1, \ldots, x_m)$, and any ground instance θ.

Moreover, as the terms $f(x_1, \ldots, x_m)$, where f is a constructor are also LS-terminating for any ground instances $\theta x_1, \ldots, \theta x_m$ and the constants are LS-terminating, then any term of $\mathcal{T}(\mathcal{F})$ is LS-terminating. □