Proving weak termination also provides the right way to terminate - extended version -

Olivier Fissore, Isabelle Gnaedig, Hélène Kirchner

Proving weak termination also provides the right way to terminate – Extended version –

Olivier Fissore, Isabelle Gnaedig, Hélène Kirchner

LORIA-INRIA & LORIA-CNRS
P 239 F-54506 Vandœuvre-lès-Nancy Cedex
Phone: + 33 3 83 58 17 00
Fax: + 33 3 83 27 83 19
e-mail: fissore@loria.fr, gnaedig@loria.fr, Helene.Kirchner@loria.fr

Abstract. From an inductive method for proving weak innermost termination of rule-based programs, we automatically infer, for each successful proof, a finite strategy for data evaluation. The proof principle uses an explicit induction on the termination property, to prove that any input data has at least one finite evaluation. For that, we analyse proof trees built from the rewrite system, schematizing the innermost derivations of any ground term. These proof trees are issued from patterns representing sets of ground terms. They are generated using two mechanisms, namely abstraction, introducing variables that represent ground terms in normal form, and narrowing, schematizing rewriting on ground terms. From the proof trees, we can extract for any given ground term, a rewriting strategy to compute one of its normal form, even if the ground term admits infinite rewriting derivations.

1 Introducing the problem

In the context of programming in general, termination is a key property that warrants, in particular, the existence of a result for every evaluation of a program. For rule-based programs, written in languages like ASF+SDF [14], Maude [3], Cafe-OBJ [8], or ELAN [2], where data evaluation consists in exploring all possible rewriting derivations of an input term, this usual notion of termination corresponds to the strong termination property of rewrite systems. However, strong termination does not always hold since derivations starting from a term may be non terminating. When for any term, there is at least one terminating derivation, the rewrite system is said to have the weak termination property. In the context of programming, this is an interesting situation, since then data evaluation may consist in finding *one* irreducible form. In languages like ELAN [2], strategies are even defined to express that the result of the program evaluation on a data is *one of its possible* finite evaluations, or *the first* one. Weak termination then warrants a result for such strategies.

Analyzing termination allows choosing the good evaluation strategy. Indeed, if the program is strongly terminating, a depth-first evaluation can be used, while if the program is only weakly terminating, a breadth-first algorithm, often much more costly, is necessary in general. In the second case, if there is a way to find terminating branches, the breadth-first technique can be avoided, which yields then a considerable gain for program executions.

In this paper, we tackle the innermost weak termination problem: we focus on the innermost strategy, consisting in rewriting always at the lowest possible positions, and very often used as a built-in mechanism in evaluation of rule-based languages and functional languages.

Obviously, specific methods for proving strong termination of rewriting under strategies give a way to prove weak termination for standard rewriting (i.e. rewriting without any strategy), provided normal forms for rewriting with these strategies are also normal forms for standard rewriting. Let us cite [1] and [9] for the innermost strategy, [7] for the outermost strategy, and [5, 15] for local strategies on operators. Here, we are more specific: we consider directly the weak innermost termination problem, i.e. we prove that among all innermost rewriting derivations starting from any term, there is at least one finite derivation. Unlike the previously cited methods, we can handle term rewriting systems (TRSs in short) that are not strongly innermost terminating. Our method also gives a way to prove

weak termination of standard rewriting. The method we propose is *constructive* in the sense that it is able both to establish that a TRS is weakly innermost terminating, and to give the strategy to follow to obtain one of the finite derivations.

The weak termination property has been proved useful in several situations: weak termination can imply strong termination [11]. J. Goubault-Larrecq proposed a proof of weak termination of typed Lambda-Sigma calculi in [10]. B. Gramlich established conditions on TRSs for the property to be preserved by the union operation on TRSs [12].

As an example, let us consider the following TRS where f and p can be seen as programs working on integers :

$$f(x) \to p(s(x)) \quad (1)$$
$$f(x) \to p(s(s(x))) \quad (2)$$
$$p(s(s(x))) \to p(x) \quad (3)$$
$$p(0) \to 0 \quad (4)$$
$$p(s(0)) \to f(0) \quad (5)$$

Given an integer n, $f(n)$ evaluates either into $p(s(n))$ or into $p(s(s(n)))$. A particularity of p is that, given an integer m, the rewriting derivation starting from $p(m)$ terminates if m is even, and may not terminate if m is odd. Therefore we have at least one evaluation of $f(n)$ that terminates, and one that does not, whatever the integer n.
For instance, considering $n = s(0)$, the following two innermost derivations are possible :

$$f(s(0)) \xrightarrow{(1)} p(s(s(0))) \xrightarrow{(3)} p(0) \xrightarrow{(4)} 0.$$

$$f(s(0)) \xrightarrow{(2)} p(s(s(s(0)))) \xrightarrow{(3)} p(s(0)) \xrightarrow{(5)} f(0) \xrightarrow{(1)} p(s(0)) \longrightarrow \dots$$

We first propose in this paper a method based on an explicit induction on the termination property, to prove that for any element t of a given set of terms T, there is at least one finite innermost rewriting derivation starting from t. For that, we analyze proof trees built from the rewrite system, and schematizing the innermost rewriting derivations of any ground term. These proof trees are issued from patterns $g(x_1, \dots, x_m)$ where g is a defined function symbol, and are built using two mechanisms, namely abstraction, introducing variables that represent ground terms in normal form, and narrowing, schematizing rewriting on ground terms.

When all proof trees have successful branches for all ground instances of the patterns, the weak innermost termination property of the rewrite system is proved. Then from these successful branches, a normalizing strategy can be inferred for any ground term. We show how to extract the relevant information from the proof trees to guide the innermost normalization process.

Proving weak termination of a program and deducing a normalizing strategy can be achieved at *compile-time*. Then, to evaluate a data at *run-time* with no risk of non-termination, it suffices to follow the strategy, which states which rule to apply and at which position in the term at each step of the normalization process. Henceforth, evaluation at run-time is made very efficient, since it always leads to a result, i.e. an irreducible term.

In Section 2, the background is presented. Section 3 introduces the basic concepts of the inductive proof mechanism. In Section 4, our method is formally described with inference rules and a strategy to apply them. Then, In Section 5, a more efficient process is proposed, stopping derivations as soon as a finite normalization has been detected. Finally, in Section 6, a strategy is proposed to reach an innermost normal form from a given term, using information of the proof establishing weak termination.

2 Notations

We assume that the reader is familiar with the basic definitions and notations of term rewriting given for instance in [4]. $\mathcal{T}(\mathcal{F}, \mathcal{X})$ 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 of 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. The empty sequence ϵ denotes the top position. The notation $t|_p$ stands for the subterm of t at position p. If p is a position in 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 conjunction 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 $\mathcal{T}(\mathcal{F})$. 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 there exists a substitution μ such that $\sigma_2 = \mu\sigma_1$.

Given a set $\mathcal{R}$ 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 $\mathcal{R}$ 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 $\mathcal{R}$ is denoted by $\mathcal{C}ons_{\mathcal{R}}$, the set of defined function symbols of $\mathcal{F}$ for $\mathcal{R}$ is denoted by $\mathcal{D}ef_{\mathcal{R}}$ ($\mathcal{R}$ is omitted when there is no ambiguity). The rewriting relation induced by $\mathcal{R}$ is denoted by $\rightarrow_{\mathcal{R}}$ ($\rightarrow$ if there is no ambiguity on $\mathcal{R}$). 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 transitive (resp. reflexive transitive) closure of the rewriting relation induced by $\mathcal{R}$ is denoted by $\rightarrow_R^+$ (resp. $\rightarrow_R^*$). If it exists, the last term of a finite derivation (or chain) 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 iff there is no infinite decreasing derivation 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 $\mathcal{T}(\mathcal{F}, \mathcal{X})$, $f(\ldots t \ldots) \succ t$. Notice that, for $\mathcal{F}$ and $\mathcal{X}$ finite, 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.

3 Induction for weak innermost termination

3.1 Induction

Let t be a term of $\mathcal{T}(\mathcal{F})$; like for standard rewriting, we say that t weakly (resp. strongly) (innermost) terminates if and only if at least one (resp. every) (innermost) rewriting derivation starting from t is finite. Obviously, strong (innermost) termination implies weak (innermost) termination. An innermost rewriting normal form of t is also denoted by $t{\downarrow}$. For proving that every term t of $\mathcal{T}(\mathcal{F})$ weakly innermost terminates, we proceed by induction on $\mathcal{T}(\mathcal{F})$ with a noetherian ordering $\succ$, assuming that for any t' such that $t \succ t'$, t' weakly innermost terminates. We first prove that a basic set of minimal elements for $\succ$ weakly innermost 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 $t = f(t_1, \ldots, t_m)$ of $\mathcal{T}(\mathcal{F})$. For that, the innermost rewriting relation is decomposed into two steps:

- first, the t_i are supposed to be weakly innermost terminating by induction hypothesis, and replaced 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;
- second, rewriting the resulting term $f(X_1, \ldots, X_m)$ at position ϵ: its ground instances $f(t_1{\downarrow}, \ldots, t_m{\downarrow})$ are indeed either irreducible, or only reducible at the top position, into say a term u. In general, several rewriting steps may be applied to $f(t_1{\downarrow}, \ldots, t_m{\downarrow})$. We obviously have to consider all terms u such that $f(t_1{\downarrow}, \ldots, t_m{\downarrow}) \rightarrow^{\epsilon} u$, which corresponds to considering all innermost derivation chains from t.

Then the weak termination problem of t is reduced to the weak termination problem of u. If $t \succ u$, by induction hypothesis, u is supposed to be weakly innermost terminating. Otherwise, one can apply a similar reasoning on $u = g(u_1, \ldots, u_n)$. We then have to assume that $(t \succ t_1, \ldots, t_m) \wedge (t \succ u_1, \ldots, u_n)$. This process is iterated until getting, in a reasonable number of steps, either a term t' such that $t \succ t'$, or such that t' is irreducible. In these cases, we have found a finite innermost derivation starting with $t \rightarrow^+ t'$.

The ordering is more and more constrained along the proof by imposing constraints between terms that must be comparable. This process can fail on $g(u_1, \ldots, u_n)$ if we cannot decide that $(t \succ t_1, \ldots, t_m) \wedge (t \succ u_1, \ldots, u_n)$ is satisfiable. Then, we cannot conclude anything.

Let us now consider the problem of finding an appropriate induction ordering. On the current term $u = g(u_1, \ldots, u_n)$ of a derivation starting from t, we need $t \succ u_1, \ldots, u_n$. But the ordering $\succ$ has to satisfy all inequalities previously stated in the induction proof ($t \succ t_1, \ldots, t_m$ above). So any ordering $\succ$ satisfying the ordering constraints of the form $t > v_1, \ldots, v_m$ accumulated along the proof process, and whose initially required properties are $\mathcal{F}$-stability and the subterm property, holds. It is important to notice that it is not required to exhibit such an ordering; deciding whether it exists is enough. Thus, at a given point of the proof, the induction hypothesis will be applied if the set of ordering constraints accumulated at this point of the proof is proved satisfiable.

We now introduce some concepts in order to formalize and automate this inductive technique.

3.2 Abstraction and ordering constraints

As previously explained, to abstract a term $f(u_1, \ldots, u_m)$, where the u_i are supposed to have a normal form $u_i{\downarrow}$, we replace the u_i by abstraction variables X_i representing respectively any of their normal forms. Let us define these special variables more formally.

Definition 1. *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.*

Given a term u, we define a predicate NF on u to be true if u is either a ground term in normal form, or an NF-variable, and to be false otherwise. Notice that in the current term $f(u_1, \ldots, u_m)$, it is not useful to introduce an abstraction variable for the u_j such that $NF(u_j)$, hence the following definition.

Definition 2. *The term $f(u_1, \ldots, u_m)$ is* abstracted *into $f(U_1, \ldots, U_m)$ if, for $i \in [1..m]$:*

- *$U_i = u_i$ if $NF(u_i)$,*
- *$U_i = X_i$ where X_i is a fresh NF-variable otherwise.*

We will prove weak innermost termination on $\mathcal{T}(\mathcal{F})$, reasoning on terms with abstraction variables, i.e. on terms of $\mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N})$. We now formally define the satisfiability of ordering constraints.

Definition 3. An ordering constraint *is a pair of terms of $\mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N})$ noted $(t > t')$. It is said to be* satisfied *by an ordering $\succ$ and an instantiation θ whose domain contains $\mathcal{V}ar(t) \cup \mathcal{V}ar(t')$, if we have $\theta t \succ \theta t'$. A conjunction C of ordering constraints is satisfied by $\succ$ and θ if $\succ$ and θ satisfy all conjuncts. The empty conjunction, always satisfied, is denoted by $\top$.*

Along our induction process, when abstracting subterms u_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 u_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 $t, t' \in \mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N})$, They are called abstraction constraints, and stored in an abstraction constraint conjunction A. Let us formally define the satisfiability of abstraction constraints.

Definition 4. *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 conjunction A of abstraction constraints is satisfiable if there exists at least one instantiation satisfying all conjuncts. The empty conjunction, always satisfied, is denoted by $\top$.*

In this paper, we consider constraint problems composed of pairs (A, C) where A is a conjunction of abstraction constraints and C is a conjunction of ordering constraints.

Definition 5. *Let A be a conjunction of abstraction constraints 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 such that θ satisfies A, $\succ$ and θ satisfy C. (A, C) is satisfiable if A is satisfiable and there exists an ordering $\succ$ as above.*

By the previous definition, deciding the satisfiability of (A, C) would require to express all instantiations satisfying A, which is undecidable. But an interesting point of our method is that we do not need to characterize all instantiations to prove the satisfiability of A. It is enough to exhibit one of them. Moreover, if there is an ordering $\succ$ satisfying $t \succ t'$ for every ordering constraint $t > t'$ of C, and which is stable by substitution (the induction ordering is then a simplification ordering), then $\theta t \succ \theta t'$ for every instantiation satisfying A.

Then a sufficient condition for (A, C) to be satisfiable is that A is, and there exists a simplification ordering $\succ$ such that $t \succ t'$ for every ordering constraint $t > t'$ of C.

3.3 Narrowing

After abstraction of the term $f(u_1, \ldots, u_m)$, where the u_i are supposed to have a normal form $u_i{\downarrow}$, and are replaced by abstraction variables X_i if we do not have $NF(u_i)$, we test whether a ground instance of the abstracted term $f(U_1, \ldots, U_m)$ is reducible by studying syntactic forms of the possible instantiations of the U_i. This test consists in narrowing $f(U_1, \ldots, U_m)$ at position ϵ with all possible substitutions instantiating the variables of the U_i only with irreducible terms, according to all possible rewrite rules.

Let us recall the definition of narrowing.

Definition 6. *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 $\mathcal{R}$ and the substitution σ, when σ is a most general unifier of $t|_p$ and l, $t' = \sigma(t[r]_p)$. This is denoted $t \leadsto_{\mathcal{R}}^{p, l \to r, \sigma} t'$ where either p, or $l \to r$ or σ may be omitted. It is always supposed 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. Notice 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.

Notice also that in our process, we are interested in the narrowing substitution applied to the current term t, but not in its definition on the variables of the left-hand side of the rule. So the narrowing substitutions we consider are restricted to the variables of the narrowed term t.

Moreover, the variables of $Ran(\sigma)$ are only NF-variables. Indeed, the first abstraction of any pattern $g(x_1, \ldots, x_m)$ gives a term with only NF-variables : $g(X_1, \ldots, X_m)$. Then, if for any narrowing step, the variables of the term to be narrowed are NF-variables $X_1, \ldots, X_k$, then the narrowing substitution σ is of the form $(X_{i_1} = v_{i_1}, \ldots, X_{i_p} = v_{i_p})$. As $X_1, \ldots, X_k$ are NF-variables, all their ground instances must be in normal form, and the same holds for ground instances of the v_i's. So the possible variables of $v_1, \ldots, v_k$ have to be NF-variables. Then all variables of the resulting narrowed term are NF-variables. Finally, the abstraction of a term whose variables are only NF-variables is also a term whose variables are only NF-variables.

Note that if a term $u \in \mathcal{T}(\mathcal{F}, \mathcal{N})$ is not narrowable, then, by definition of the NF-variables, every instance of u is in normal form, hence abstracting u is not needed. Henceforth, the NF predicate can be generalized in the following way: $NF(u)$ is true iff $u \in \mathcal{T}(\mathcal{F}, \mathcal{N})$ is not narrowable. In particular, as previously, $NF(u)$ is true if u is an NF-variable or a ground term in normal form.

4 Inference rules for inductive termination proofs

We are now ready to describe the different steps of our mechanism on a term t, with initial empty constraints conjunctions A, C. It consists in iterating the following steps.

- The first step abstracts the current term $f(u_1, \ldots, u_m)$. The constraints $t > u_{i_1}, \ldots, u_{i_p}$ are set, where $u_{i_1}, \ldots, u_{i_p}$ are the terms $u_i, i \in [1..m]$ such that we do not have $NF(u_i)$. They allow to assume, by induction, the existence of irreducible forms for $u_{i_1}, \ldots, u_{i_p}$. Then, $u_{i_1}, \ldots, u_{i_p}$ are abstracted into NF-variables $X_{i_1}, \ldots, X_{i_p}$, which can only be instantiated by terms in normal form. We call this step the *abstraction* step.
  If the *abstraction* step cannot be applied because $(A, C \wedge\ t > u_{i_1}, \ldots, u_{i_p})$ cannot be proved to be satisfiable, the process stops with a special current term $\sharp$.
- The second step narrows the resulting term $u = f(U_1, \ldots, U_m)$ at position ϵ in all possible ways in one step, with all possible rewrite rules of the rewrite system $\mathcal{R}$, and all possible substitutions, into terms v. This is the *narrowing* step.
  For controlling the narrowing process, well known to diverge, we use the constraints in A. In fact, the possible instances of the X_i are not any instance, but only terms whose ground instances are the normal forms of the ground instances of subterms they abstract. If X_i abstracts a subterm u_i, X_i is constrained by the equality $u_i{\downarrow} = X_i$.
  So for the narrowing step, only the narrowing substitutions are retained that are compatible with A. A substitution σ is called compatible with A if σA is satisfiable. This enables us:
  - to avoid useless branches of the proof tree generated by narrowing. This then reduces the breadth of the proof tree;
  - to stop the process on a branch earlier, if no narrowing applies anymore. In particular, this can avoid the failing cases of the process, appearing after a narrowing step using an ineffective substitution. This then reduces the depth of the proof tree.

  If the current term u is not narrowable, any of its ground instances is in normal form. Then the process stops the current derivation chain of the proof.
- Then we can test for the current term u and the current conjunction of ordering constraints C, whether $(A, C \wedge t > u)$ is satisfiable. In this case, by induction hypothesis, any ground instance of u is weakly innermost terminating, which ends the current derivation chain of the proof.

The previously presented steps are performed by inference rules that transform 3-tuples (T, A, C) where

- T is a set of terms of $\mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N})$, containing the current term whose weak innermost termination has to be proved. This is either a singleton or the empty set.
- A is a conjunction of abstraction constraints. Those of the form $u{\downarrow} = X$ are stated each time a term u is abstracted into a new NF-variable X.
- C is a conjunction of ordering constraints stated by the abstraction steps.

Before giving the corresponding inference rules, let us notice that the inductive reasoning can be completed in the following way. When the induction hypothesis cannot be applied on a term u, it can sometimes be possible to prove weak innermost termination of any ground instance of u by another way. Let $WEAK{-}TERMIN(u)$ be a predicate that is true iff every ground instance of u weakly innermost terminates. In the first and third previous steps of the induction reasoning, we then associate the alternative predicate $WEAK{-}TERMIN(u)$ to the condition $t > u$.

For establishing that $WEAK{-}TERMIN(u)$ is true, in some cases, the notion of usable rules can be used. Given a TRS $\mathcal{R}$ and a term $u \in \mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N})$, 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. Proving weak innermost termination of any ground instance of u then comes down to proving weak innermost termination of its usable rules, which is often much easier than proving weak innermost termination of the whole TRS. In particular, our inductive proof process itself can be applied. Usable rules are even sometimes strongly terminating, which can be ensured with the classical ordering-based termination methods. This approach is fully developed in [9].

Table 1. Inference rules for weak innermost t_{ref}-termination

Abstract: $$\frac{\{f(u_1,\ldots,u_m)\},\ A,\ C}{\{f(U_1,\ldots,U_m)\},\ A \wedge \bigwedge_{i=1}^{m} H_A(u_i),\ C \wedge \bigwedge_{i=1}^{m} H_C(u_i)}$$

where $f(u_1,\ldots,u_m)$ is abstracted into $f(U_1,\ldots,U_m)$

if $(A \wedge \bigwedge_{i=1}^{m} H_A(u_i), C \wedge \bigwedge_{i=1}^{m} H_C(u_i))$ is satisfiable

StopA: $\dfrac{\{f(u_1,\ldots,u_m)\},\ A,\ C}{\{\sharp\},\ A,\ C}$ if Abstract cannot be applied.

Narrow: $\dfrac{\{f(u_1,\ldots,u_m)\},\ A,\ C}{\{v\},\ \sigma A,\ C}$ if $f(u_1,\ldots,u_m) \leadsto_R^{\epsilon,\sigma} v$ and σA satisfiable.

StopN: $\dfrac{\{u\},\ A,\ C}{\emptyset,\ A,\ C}$ if u is not narrowable or (u is narrowable with σ and σA unsatisfiable).

Stop: $$\frac{\{u\},\ A,\ C}{\emptyset,\ A \wedge H_A(u),\ C \wedge H_C(u)}$$ if $(A \wedge H_A(u), C \wedge H_C(u))$ is satisfiable.

---

where $H_A(u) = \begin{cases} true & \text{if } NF(u) \\ u\downarrow = X & \text{otherwise.} \end{cases}$ and $H_C(u) = \begin{cases} true & \text{if } WEAK\text{-}TERMIN(u) \\ t_{ref} > u & \text{otherwise.} \end{cases}$

The termination proof procedure is described by the set of rules given in Table 1. These rules use a reference term $t_{ref} = g(x_1,\ldots,x_m)$, where $x_1,\ldots,x_m \in \mathcal{X}$ and $g \in \mathcal{D}ef$. To prove weak innermost termination of $\mathcal{R}$ on any term $t \in \mathcal{T}(\mathcal{F})$, we proceed as follows: for each defined symbol $g \in \mathcal{D}ef$, we apply the rules using the reference term $t_{ref} = g(x_1,\ldots,x_m)$ on the initial set of terms. These rules must be applied with a specific strategy S on the initial 3-tuples $(\{t_{ref}\}, \top, \top)$.

Before giving the strategy, let us first point out a few properties of these rules: **Stop** does not apply on $(\{t_{ref}\}, \top, \top)$ because its condition is never true in this case. But on a current state $(\{u\}, A, C)$, it is interesting to try this rule first since it allows to reach an irreducible state $(\emptyset, A, C)$. **Abstract** and **StopA** are mutually exclusive, so one of them is always applicable. After an application of **StopA**, the process stops on the current state $(\{\sharp\}, A, C)$ since no more rule applies. **Narrow** is a non-deterministic rule that can produce several results. **Narrow** is applied after **Abstract** with all possible narrowing substitutions and all possible rewrite rules at the position ϵ.

The strategy S used to control the rules is:

(**Abstract**; **StopA**; dk(**Narrow**); **StopN**; **Stop**) *

where ";" denotes the successive application of rules, "dk" the application of a rule in all possible ways and "*" the iterative application of a strategy, until it is not possible anymore. The process stops if no inference rule applies anymore.

There are three cases for the behavior of the termination proof procedure. The strategy applied to the initial state $(\{t_{ref}\}, \top, \top)$ may terminate if the rules do not apply anymore and all states are of the form $(\emptyset, A, C)$. Otherwise, the strategy may not terminate if there is an infinite number of applications of **Abstract** and **Narrow**. It can also happen that all rules fail, and the process stops with a state of the form $(T \neq \emptyset, A, C)$.

Applying S to an initial state $(\{g(x_1,\ldots,x_m)\}, \top, \top)$ builds a derivation tree, also called proof tree, whose nodes are the states produced by the inference rules. Branching is produced by the different narrowing steps performed by the rule **Narrow** for all possible substitutions and all possible rewrite rules. In fact, the states of the proof tree represent sets of ground terms, we are studying the

termination of. More precisely, any state $(\{t\}, A, C)$ represents the set of ground instances αt such that α satisfies A.

A branch of the derivation tree is said to be successful if it is ended by an application of **Stop** or **StopN**. Indeed, the inductive weak termination proof is successful if there is a successful branch corresponding to each possible ground term. With our current set of inference rules, every branch terminating with a state of the form $(\emptyset, A, C)$ is successful.

Then branching produced by **Narrow** can generate different branches with the same narrowing substitution, and with different rewrite rules: all these branches represent the same set of ground instances. So, for proving weak termination of each set of ground instances at this branching point, it will be sufficient to prove weak innermost termination from only one of the states representing these instances.

Let us illustrate this fact by the small example $\{f(a) \to c, f(b) \to d, f(b) \to f(b)\}$, where a, b, c, d are constants.

Applying the inference rules on $f(x)$, we get:

$f(x)$	$A = \top$	$C = \top$		
Abstract				
$f(X)$	$A = (x\downarrow = X)$	$C = (f(x) > x)$		
Narrow				
c	$A = (x\downarrow = a)$	$C = (f(x) > x)$	$\sigma = (X = a)$	$(branch\ 1)$
d	$A = (x\downarrow = b)$	$C = (f(x) > x)$	$\sigma = (X = b)$	$(branch\ 2)$
$f(b)$	$A = (x\downarrow = b)$	$C = (f(x) > x)$	$\sigma = (X = b)$	$(branch\ 3)$

Narrow here produces one state with the substitution $\sigma = (X = a)$ and two states with the same substitution $\sigma = (X = b)$. The first branch represents all ground instances of $f(X)$ satisfying the substitution $\sigma = (X = a)$, i.e. the term $f(a)$. The second and third branches represent all ground instances of $f(X)$ satisfying the substitution $\sigma = (X = b)$, i.e. the term $f(b)$. Then, for proving weak termination of all ground instances of $f(x)$, it will be enough to prove weak innermost termination from the state $(\{c\}, C = (f(x) > x))$, and from the state $(\{d\}, C = (f(x) > x))$. We then have:

StopN$(twice)$		
$\emptyset$	$A = (x\downarrow = a)$	$C = (f(x) > x)$
$\emptyset$	$A = (x\downarrow = b)$	$C = (f(x) > x)$
$f(b)$	$A = (x\downarrow = b)$	$C = (f(x) > x)$

which ends the weak termination proof. The third branch, which gives an infinite succession of **Abstract** and **Narrow** from $(f(b), A = (x\downarrow = b), C = (f(x) > x))$, is useless in the proof.

We write $SUCCESS(g, \succ)$ if the application of the inference rules on $(\{g(x_1, \ldots, x_m)\}, \top, \top)$, whose conditions are satisfied by an ordering $\succ$, gives a derivation tree with at least one successful branch in each set of branches with same (up to a renaming) narrowing substitution σ generated by any application of **Narrow**.

Theorem 1. *Let $\mathcal{R}$ be a TRS on a set $\mathcal{F}$ of symbols, such that the constants of $\mathcal{F}$ weakly innermost terminate. If there exists an $\mathcal{F}$-stable ordering $\succ$ having the subterm property, such that for each non-constant defined symbol g, we have $SUCCESS(g, \succ)$, then any term of $\mathcal{T}(\mathcal{F})$ weakly innermost terminates.*

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

5 Finding a successful branch

The previous theorem gives an abstract sufficient condition for weak termination, but it does not give any operational way to compute the predicate $SUCCESS$. As said previously, a successful branch corresponds to the termination of at least one rewriting chain of a given set of ground instances. By

definition of weak termination, if other branches of the proof tree correspond to the termination proof of other rewriting chains from the same set of ground instances, these branches become useless and we can cut them. This is the case in the previous example for the third branch.

But in general, it is not enough to stop the proof process when a successful branch is found. We have to test, at each branching point, whether the other branches correspond to the same set of ground instances. If they do, the branches are useless and they are cut; otherwise, the proof process has to explore them further. This test needs to backtrack in the proof tree.

The operational way to cut branches as soon as a successful branch has been generated is obviously to develop the proof tree with a breadth-first mechanism. The previous result then can be refined in the following way. To be able to detect useless branches of the proof tree and to cut them, possibly recursively, we need to consider the whole proof tree in our inference process. We now propose a new set of inference rules, applying on the whole proof tree, integrating the backtracking control necessary to this cut principle.

Table 2. Inference rules for weak innermost t_{ref}-termination with the cut process

Termination proving rules:

Abstract: $$\frac{T \sqsupset_l s = (\{f(u_1,\ldots,u_m)\},\ A,\ C,\ \sigma)}{T \sqsupset_l [s \wedge_{child} (\{f(U_1,\ldots,U_m)\},\ A \wedge \bigwedge_{i=1}^{m} H_A(u_i),\ C \wedge \bigwedge_{i=1}^{m} H_C(u_i),\ \sigma)]}$$

where $f(u_1,\ldots,u_m)$ is abstracted into $f(U_1,\ldots,U_m)$

if $(A \wedge H_A(u_1) \wedge \ldots H_A(u_m), C \wedge H_C(u_1) \wedge \ldots H_C(u_m))$ is satisfiable

StopA: $$\frac{T \sqsupset_l (\{f(u_1,\ldots,u_m)\},\ A,\ C,\ \sigma)}{T \sqsupset_l (\{\sharp\},\ A,\ C,\ \sigma)}$$ if Abstract cannot be applied.

Narrow: $$\frac{T \sqsupset_l (\{f(u_1,\ldots,u_m)\},\ A,\ C,\ \mu)}{T \sqsupset_l [(\{f(u_1,\ldots,u_m)\},\ A,\ C,\ \mu) \wedge_{child} (\{v\},\ \sigma A,\ C,\ \sigma)]}$$

if $f(u_1,\ldots,u_m) \leadsto_R^{\epsilon,\sigma} v$ and σA satisfiable .

StopN: $$\frac{T \sqsupset_l (\{u\},\ A,\ C,\ \sigma)}{T \sqsupset_l (\emptyset,\ A,\ C,\ \sigma)}$$

if u is not narrowable or (u is narrowable with σ and σA is unsatisfiable).

Stop: $$\frac{T \sqsupset_l (\{u\},\ A,\ C,\ \sigma)}{T \sqsupset_l (\emptyset,\ A \wedge H_A(u),\ C \wedge H_C(u),\ \sigma)}$$ if $(A \wedge H_A(u), C \wedge H_C(u))$ is satisfiable

Redundancy suppressing rules:

Cut: $$\frac{[T \sqsupset_{p.k} (\emptyset,\ A,\ C,\ \sigma)] \sqsupset_{p.m} (\{u'\},\ A',\ C,\ \mu)}{[T \sqsupset_{p.k} (\emptyset,\ A,\ C,\ \sigma)] \sqsupset_{p.m} (\emptyset,\ A',\ C,\ \mu)}$$ if μ is an instance of σ

Shorten: $$\frac{T \sqsupset_p s = (\{u\},\ A,\ C,\ \sigma)}{T \sqsupset_p (\emptyset,\ A,\ C,\ \sigma)}$$ if every child i of s in T is of the form $(\emptyset,\ A_i,\ C_i,\ \mu_i)$

---

where $H_A(u) = \begin{cases} true & \text{if } NF(u) \\ u\downarrow = X & \text{otherwise.} \end{cases}$ and $H_C(u) = \begin{cases} true & \text{if } WEAK\text{-}TERMIN(u) \\ t_{ref} > u & \text{otherwise.} \end{cases}$

In the first five rules, l is a leaf position in T.

Let us give an idea of how the useless branches are identified and cut. Let the last narrowing substitution (LNS in short) of a state s be the narrowing substitution of the state generated by the

last narrowing step performed before generating the state s. If a final state of the process, generated by **Stop** or **StopN**, has a LNS σ, then by definition of weak termination, it is not necessary to verify that the other branches representing a smaller or equal set of ground instances are successful. The other branches representing a smaller or equal set of ground instances are the branches generated by this narrowing step, with substitutions that are less general or equal to σ.

So, when **Stop** or **StopN** applies on a state s whose LNS is σ, a cut process is applied with a rule **Cut**, transforming each state issued from the same narrowing step as s and whose LNS is an instance of σ, into a new final state.

Moreover, after the cut process has been applied, all branches issued from the state s', on which the concerned narrowing step applies, may have a final state.

This leads to assume weak termination of all ground instances represented by s', and then s' can itself be replaced by a final state. This is processed with a rule named **Shorten**. This process of bottom-up replacing states by final states in the proof tree is iterated, until it is not possible anymore.

The rules **Abstract**, **Narrow**, **Stop**, **StopN** and **StopA** are called termination proving rules, since they are used to build branches that are necessary to establish weak termination of all possible ground terms. By contrast, the rules **Cut** and **Shorten** are called redundancy suppressing rules, since they detect and suppress redundant branches in the proof tree, and for that, suppress branches that have already been proved successful.

Remark that the termination proving rules make the proof tree grow, while the redundancy suppressing rules reduce it.

The states, on which this inference process applies, now have a fourth component: the most general unifier of the last applied narrowing step. To manage the cut process in the proof tree, the structure on which the inference rules applies has to be modified. Our inference rules will not apply on states $(\{u\}, A, C)$ anymore but on the proof tree, whose nodes will be states of the form $s = (\{u\}, A, C, \sigma)$, where σ is the substitution used in the last narrowing step performed to obtain the state s in the proof process.

We will use the following operator on trees:

$$T \sqsupset_p s : \text{ the tree } T \text{ containing the state } s \text{ at position } p,$$

and the operation:

$$T \sqsupset_p (s \wedge_{child} s') : \text{ the state } s' \text{ is added as a child of } s \text{ in the tree } T.$$

The new rules are given in Table 2. Recall that the states of the form $(\emptyset, A, C, \sigma)$ are final states, i.e. leaf states of the proof tree.

The rules are applied with the following strategy $S{-}CUT$ on initial trees of the form $\{(\{g(x_1, \ldots, x_m)\}, \top, \top, \lambda)\}$, where λ denotes the empty substitution:

(**Abstract**; **StopA**; dk(**Narrow**); **StopN**; **Stop**; (**Cut**; **Shorten**)*)*.

The process stops if no inference rule applies anymore. Weak termination is established if each final proof tree is reduced to a state of the form $(\emptyset, \top, \top, \lambda)$.

Considering the behavior of the procedure now, let us notice that if the cut process applies on all branches for which we would have got a state $(T \neq \emptyset, A, C)$ or an infinite number of applications of **Abstract** and **Narrow** with the previous strategy, we now only get successful branches in the proof tree. This can allow to conclude weak innermost termination, whereas with the previous strategy, nothing could be concluded.

We write $SUCCESS{-}CUT(g, \succ)$ if the application on $(\{g(x_1, \ldots, x_m)\}, \top, \top, \lambda)$, with the strategy $S{-}CUT$, of the inference rules of Table 2 whose conditions are satisfied by an ordering $\succ$, gives a proof tree reduced to a state $(\emptyset, \top, \top, \lambda)$.

Theorem 2. *Let $\mathcal{R}$ be a TRS on a set $\mathcal{F}$ of symbols, such that the constants of $\mathcal{F}$ weakly innermost terminate. If there exists an $\mathcal{F}$-stable ordering $\succ$ having the subterm property, such that for each non-constant defined symbol g, we have $SUCCESS{-}CUT(g, \succ)$, then any term of $\mathcal{T}(\mathcal{F})$ weakly innermost terminates.*

We now develop an example, giving the states of the proof trees together with the position they have in these proof trees.

Example 1. Let us consider the following rewrite system $\mathcal{R}$, which is neither terminating, nor innermost terminating, but is weakly innermost terminating.

$$g(g(x)) \to g(x)$$
$$g(g(x)) \to g(g(g(x))).$$

The TRS $\mathcal{R}$ is proved weakly innermost terminating on $\mathcal{T}(\mathcal{F})$ containing at least one constant a (with $\mathcal{F} = \{g : 1, a : 0\}$) as follows.

Obviously, a is weakly innermost terminating.

Applying the rules on $g(x)$, we get:

ϵ	$g(x)$	$A = \top$	$C = \top$	$\sigma = \lambda$
Abstract				
1	$g(X)$	$A = (x\downarrow = X)$	$C = (g(x) > x)$	$\sigma = \lambda$
Narrow				
1.1	$g(X')$	$A = (x\downarrow = g(X'))$	$C = (g(x) > x)$	$\sigma = (X = g(X'))$
1.2	$g(g(g(X'')))$	$A = (x\downarrow = g(X''))$	$C = (g(x) > x)$	$\sigma = (X = g(X''))$
StopN				
1.1	$\emptyset$	$A = (x\downarrow = g(X'))$	$C = (g(x) > x)$	$\sigma = (X = g(X'))$
1.2	$g(g(g(X'')))$	$A = (x\downarrow = g(X''))$	$C = (g(x) > x)$	$\sigma = (X = g(X''))$
Cut				
1.1	$\emptyset$	$A = (x\downarrow = g(X'))$	$C = (g(x) > x)$	$\sigma = (X = g(X'))$
1.2	$\emptyset$	$A = (x\downarrow = g(X''))$	$C = (g(x) > x)$	$\sigma = (X = g(X''))$
Shorten				
1	$\emptyset$	$A = (x\downarrow = X)$	$C = (g(x) > x)$	$\sigma = \lambda$
Shorten				
ϵ	$\emptyset$	$A = \top$	$C = \top$	$\sigma = \lambda$

StopN applies since $g(X')$ can be narrowed only with a substitution σ such that $\sigma X' = g(X'')$, with $\sigma A = (x\downarrow = g(g(X')))$ not satisfiable, since no instance of $g(g(X''))$ can be in normal form. If we did not test the satisfiability of A, **Narrow** would apply infinitely, in alternation with trivial abstracting steps.

Cut applies to state 1.2, since the LNS of this state is the same (up to a variable renaming) as the one of the final state 1.1. Then **Shorten** applies to state 1, since all branches from this state are successful.

Note that the constraint A of the state 1 is satisfiable by any substitution θ such that $\theta x = \theta X = a$. Likewise, the constraint A of the state 1.1 is satisfiable by any substitution θ such that $\theta x = g(a)$ and $\theta X' = a$; the constraint A of the state 1.2 is similar.

Since the only constant of $\mathcal{R}$ is weakly innermost terminating and the proof tree of g is reduced to a state of the form $(\emptyset, \top, \top, \lambda)$, we get that $\mathcal{R}$ is weakly innermost terminating.

6 Finding a good derivation chain

As said previously, establishing weak termination of an undeterministic evaluation process warrants a result if a breadth-first strategy is adopted for this process. But such a strategy is in general very costly, and the ideal would be to have indications about the terminating derivations to compute them directly with a depth-first mechanism.

Our proof process, as it simulates the rewriting mechanism, gives complete information on a terminating rewriting branch. It allows extracting the exact application of rewrite rules that yields a normal form. To rewrite a term, it is enough to follow the rewriting scheme simulated by abstraction and narrowing in the corresponding terminating branch of the proof trees. In fact, the branches to follow in the proof tree are the successful branches, that are branches generated only by the termination proving inference rules.

Remark that now, the states of the form $(\emptyset, A, C)$ do not necessarily belong to successful branches anymore. They can also be generated by **Cut** and **Shorten**. In this case, the previous nodes on the branches represent redundant states, i.e. ground instances already proved to have finite rewriting derivations, but not necessarily on the current branch.

We now formalize the use of the proof trees to compute a normal form for any term.

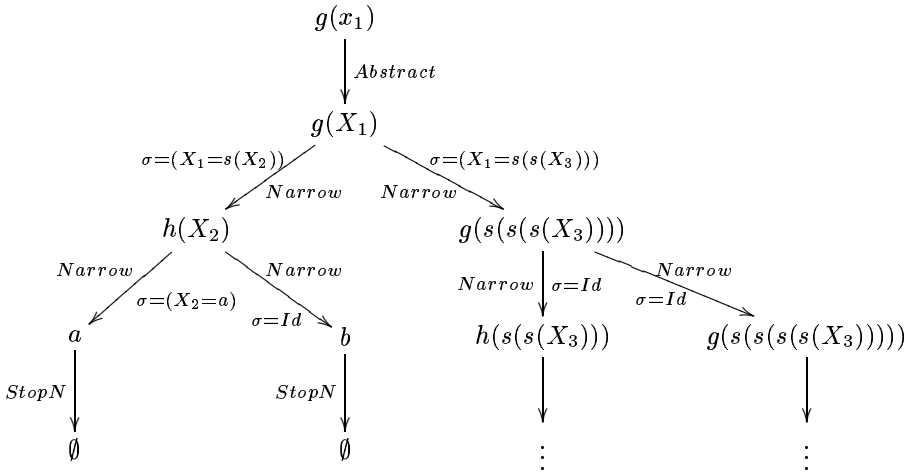
Definition 7. *Let $\mathcal{R}$ be a TRS proved weakly terminating with the strategy $S-CUT$. The strategy tree ST_f associated to $f \in \mathcal{D}ef_R$ is the tree obtained by taking all branches built during the generation of the proof tree of f, by using only the termination proving rules, and ending with a state $(\emptyset, A, C)$.*

The normalizing process following the strategy trees is not necessarily deterministic. Several successful branches representing common ground terms can be generated because of the breadth first proof strategy, before the **Cut** process suppresses the possible redundant ones. This does not matter anyway, since one can choose any successful branch to reach a normal form.

Example 2. Let us consider the following rewrite system $\mathcal{R}$, built on $\mathcal{F} = \{g:1, h:1, s:1, a:0, b:0\}$:

$$\begin{array}{ll} g(s(x)) & \to h(x) \\ g(s(s(x))) & \to g(s(s(s(x)))) \\ h(a) & \to a \\ h(x) & \to b \end{array}$$

and let us develop the proof tree for the symbol g :



The two branches on the left are two successful branches with same depth, and the instances covered by the branch obtained with the substitution $(X_2 = a)$ are obviously also covered by the branch obtained with the substitution Id.

The breadth first strategy, developing the same breadth nodes simultaneously or, for example, from left to right, does not allow to detect that the branch with a is redundant, before the branch with b has been proved successful. So both branches appear in the strategy tree.

Let us remark that the cut process avoids to develop further the other branches, since the substitution $(X_1 = s(X_2))$ used from $g(X_1)$ to obtain successful branches is more general than the other substitution $(X_1 = s(s(X_3)))$.

Definition 8. *Let $\mathcal{R}$ be a TRS proved weakly terminating with the strategy $S-CUT$. Let $ST = \{ST_f | f \in \mathcal{D}ef_R\}$ be the set of strategy trees of $\mathcal{R}$ and $s = f(s_1, \ldots, s_m) \in \mathcal{T}(\mathcal{F})$. Normalizing s with respect to ST into $norm_{ST}(s)$ is defined in the following way:*

- *if $f \in \mathcal{C}$, then $norm_{ST}(f) = f$ if $f \in \mathcal{C}ons_{\mathcal{R}}$, $norm_{ST}(t)$ if $f \longrightarrow t$,*
- *if $f \in \mathcal{C}ons_{\mathcal{R}} \setminus \mathcal{C}$, then $norm_{ST}(f(s_1, \ldots, s_n)) = f(norm_{ST}(s_1), \ldots, norm_{ST}(s_n))$,*

– *if* $f \in \mathcal{D}ef_R \setminus \mathcal{C}$*, then normalizing* s *with respect to* ST *into* $norm_{ST}(s)$ *is performed by following the steps in the strategy tree* ST_f *of* f*, where* $t = g(t_1, \ldots, t_n)$ *is any term of the transformation chain of* t *with respect to* ST *and* $u = g(u_1, \ldots, u_n)$ *is the corresponding term in* ST_f*:*

- *if the step is* **Abstract***, then* $g(t_1, \ldots, t_n) \mapsto g(t'_1, \ldots, t'_n)$*, where* t'_i *is*
  $\begin{cases} t_i\downarrow & \text{if } WEAK\text{-}TERMIN(u_i) \\ norm_{ST}(t_i) & \text{otherwise,} \end{cases}$
- *if the step is* **Narrow** *with* $g(u_1, \ldots, u_n) \rightsquigarrow^{\epsilon, l \to r, \sigma} u'$*, then* $g(t_1, \ldots, t_n) \mapsto t'$*, where* t' *is such that*
  $\begin{cases} g(t_1, \ldots, t_n) \to^{\epsilon, l \to r, \mu} t' = \mu u' & \text{if } \exists \mu, \theta = \mu\sigma[Var(g(u_1, \ldots, u_n)] \text{ and} \\ & \quad g(t_1, \ldots, t_n) = \theta g(u_1, \ldots, u_n) \\ t' = g(t_1, \ldots, t_n) & \text{otherwise,} \end{cases}$
- *if the step is* **Stop***, then* $g(t_1, \ldots, t_n) \mapsto t'$*, where* t' *is*
  $\begin{cases} g(t_1, \ldots, t_n)\downarrow & \text{if } WEAK\text{-}TERMIN(g(u_1, \ldots, u_n)) \\ norm_{ST}(g(t_1, \ldots, t_n)) & \text{otherwise,} \end{cases}$
- *if the step is* **StopN***, stop the rewriting process on the current term* $g(t_1, \ldots, t_n)$*.*

Given a TRS $\mathcal{R}$, the previous definition assumes that if the predicate $WEAK\text{-}TERMIN$ has been used to prove termination of a particular term t during the termination proof of $\mathcal{R}$, one is able to find a normalizing strategy for t. A simple sufficient condition for that is that t is proved strongly terminating, which can be established in most cases, like for $WEAK\text{-}TERMIN$, with the usable rules.

Moreover, as there is no strategy tree for the constants, we have a normalizing strategy for them, only if their rewriting derivations having only constant terms are finite. By the above definition, the derivations from constants containing a non constant term allow to follow a strategy tree.

Under these two assumptions, the following theorem holds.

Theorem 3. *Let* $\mathcal{R}$ *be a TRS proved weakly terminating with the strategy* $S\text{-}CUT$ *and* ST *its set of strategy trees. Then reducing any term* $t \in \mathcal{T}(\mathcal{F})$ *with respect to* ST *leads to an irreducible term, which is an innermost normal form of* t *for* $\mathcal{R}$*.*

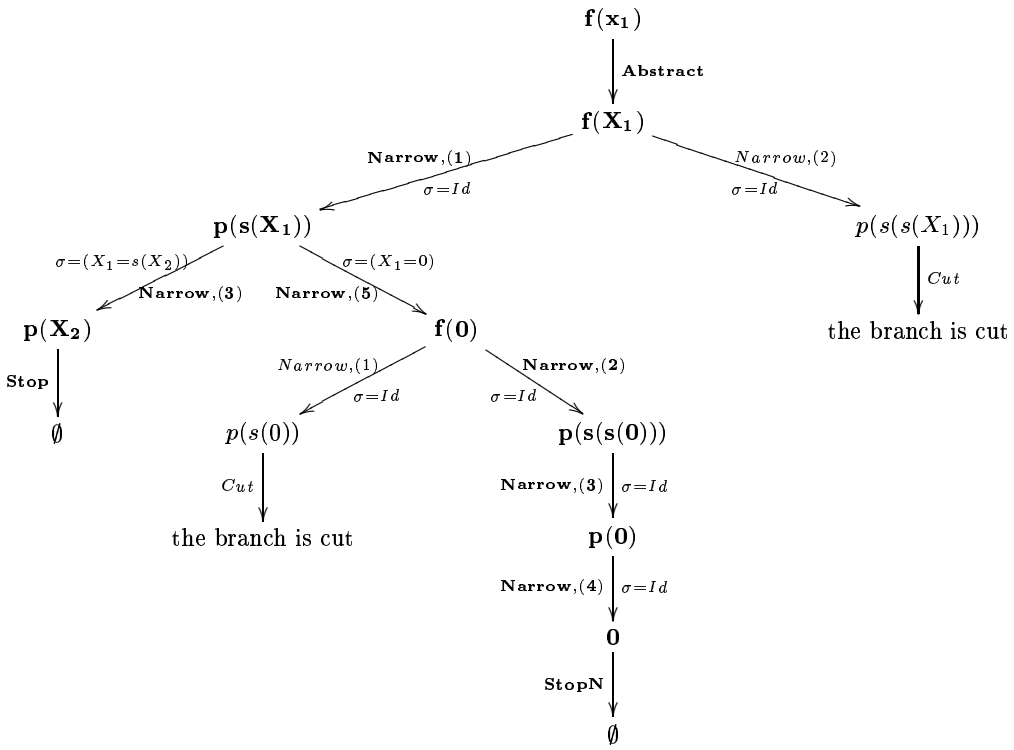
Let us come back to the TRS $\mathcal{R}$ presented in the introduction, built on $\mathcal{F} = \{f : 1, p : 1, s : 1, 0 : 0\}$:

$$\begin{array}{lll} f(x) & \to p(s(x)) & (1) \\ f(x) & \to p(s(s(x))) & (2) \\ p(s(s(x))) & \to p(x) & (3) \\ p(0) & \to 0 & (4) \\ p(s(0)) & \to f(0) & (5) \end{array}$$

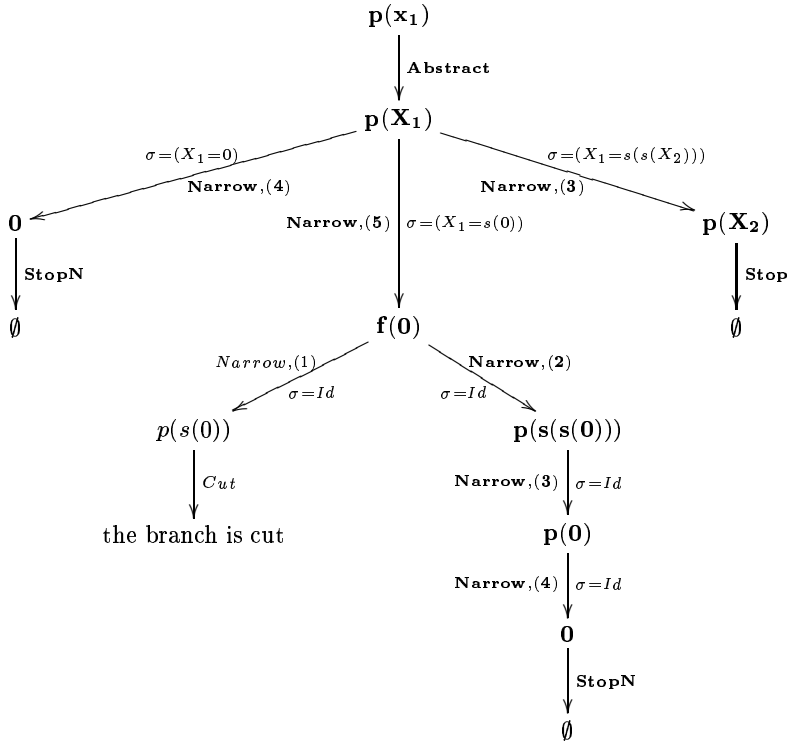
Let us first prove that every ground term t of $\mathcal{T}(\mathcal{F})$ can be innermost normalized with $\mathcal{R}$, and then infer from this proof a strategy allowing normalization of any ground term of $\mathcal{T}(\mathcal{F})$.

Since the only defined symbols of $\mathcal{R}$ are f and p, we have to apply the inference rules to $f(x_1)$ and to $p(x_1)$. The detailed application is fully described in the appendix ; here, we just summarize in two proof trees the information needed to infer a strategy for normalizing any ground term. When **Narrow** applies, we specify the narrowing substitution and, into brackets, the rewrite rule number used for narrowing.

The proof tree obtained by applying the inference rules to $f(x_1)$ is the following.



The proof tree obtained by applying the inference rules to $p(x_1)$ is the following.



The corresponding strategy trees, according to Definition 7, are the bolded subtrees of the proof trees.

We can now infer from these trees a strategy normalizing any ground term t, according to Definition 8. As an example, let us use the strategy to normalize the term $f(s(s(s(0))))$ following the steps of ST_f.

(Step 1 in ST_f : Abstract) The first step is **Abstract**, and then we get $f(s(s(s(0)))) \mapsto f(norm_{ST}(s(s(s(0)))))$. Since s is a constructor, we have :
$norm_{ST}(s(s(s(0)))) = s(norm_{ST}(s(s(0)))) = s(s(norm_{ST}(s(0)))) = s(s(s(norm_{ST}(0))))$.
Since 0 is a constant constructor, we get $norm_{ST}(0) = 0$, and finally $norm_{ST}(s(s(s(0)))) = s(s(s(0)))$.
We are now on state 1 of ST_f, with the current term $f(s(s(s(0))))$ in the derivation.

(Step 2 in ST_f : Narrow) The second step is **Narrow**, with rule (1). The narrowing substitution σ is such that our current term is a ground instance of $\sigma f(X_1)$. So $f(s(s(s(0)))) \longrightarrow^{\epsilon,(1)} p(s(s(s(s(0)))))$.

(Step 3 in ST_f : Narrow) The third step is **Narrow**, with rules (3) and (5). For (5), there is no narrowing substitution σ such that our current term $p(s(s(s(s(0)))))$ is a ground instance of $\sigma p(s(X_1))$. For (3), however, this is the case. So we rewrite our current term in the derivation with (3). We get : $p(s(s(s(s(0))))) \longrightarrow^{\epsilon,(3)} p(s(s(0)))$.

(Step 4 in ST_f : Stop) The current step in the tree is **Stop**, and then we get $norm_{ST}(p(s(s(0))))$. We now have to follow ST_p to evaluate $norm_{ST}(p(s(s(0))))$.

(Step 1 in ST_p : Abstract) Since the first step of ST_p is **Abstract**, we get $p(s(s(0))) \mapsto p(norm_{ST}(s(s(0))))$. Using the same reasoning as previously, we have $norm_{ST}(s(s(0))) = s(s(0))$.

(Step 2 in ST_p : Narrow) The second step is **Narrow** with rules $(3), (4), (5)$. The only rule such that the narrowing substitution σ is such that our current term $p(s(s(0)))$ is a ground instance of $\sigma p(X_i)$ is the rule (3), and then we get : $p(s(s(0))) \longrightarrow^{\epsilon,(3)} p(0)$.

(Step 3 in ST_p : Stop) The current step in ST_p is **Stop**, and then we get $norm_{ST}(p(0))$. Once again, we have to follow ST_p to evaluate $norm_{ST}(p(0))$.

(Step 1 in ST_p : Abstract) The first step is **Abstract**, and then we get $p(0) \mapsto p(norm_{ST}(0))$. Since 0 is a constant constructor, we have $norm_{ST}(0) = 0$.

(Step 2 in ST_p : Narrow) The second step is **Narrow** with rules $(3), (4), (5)$. The only possible narrowing substitution is the one of the rule (4), and then we get : $p(0) \longrightarrow^{\epsilon,(4)} 0$.

(Step 3 in ST_p : StopN) The current step is **StopN**, which ends the normalizing process on 0, which hence is a normal form of $f(s(s(s(0))))$.

For other examples, see the appendix.

7 Conclusion and perspectives

In this paper, we have proposed a method to prove weak innermost termination of term rewriting systems 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 weak innermost termination of a ground term t, it is enough to suppose that subterms of t are smaller than t for this ordering, and that rewriting the context leads to at least one terminating chain. Iterating this process until a non-reducible context is obtained establishes weak innermost termination of t. Up to our knowledge, this is the first method proposed to ensure weak termination of rewriting systems that are not strongly innermost terminating.

From the proof of weak termination of a given TRS, we can extract for any given ground term, a rewriting strategy to compute one of its normal form, even if the ground term admits infinite rewriting

derivations. This leads to the idea of a preprocessor of TRS, at compile time, that builds the strategy trees for each defined symbol of the signature. Then at evaluation time, the execution is guided as described in Section 6 to obtain a normal form of any given term. This can then be very useful in the context of rule-based programming, to choose an efficient strategy for undeterministic and weakly terminating programs.

The important point to automate our proof principle is the satisfaction of the constraints at each step of the proof. On many examples, as those shown in the paper, this is immediate: as the ordering constraints only express the subterm property, they are trivially satisfied by any simplification ordering. Otherwise, we can use automatic ordering constraint solvers. Moreover, the abstraction constraints are only used to control the narrowing process, which converges most of the time without the narrowing substitutions being checked to be compatible with A. So they can be ignored without their compromising correctness of the proof process. Thus, in general, weak termination proof is completely automatic. We are now studying the implementation of our technique in CARIBOO, a toolbox for proving termination under strategies [6].

As in our approach, the rewriting strategy is explicitly handled in the proof principle, the method should be applicable easily to other strategies, especially to outermost strategy, and to local strategies on operators.

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. 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.
3. Manuel Clavel, Francisco Durán, Steven Eker, Patrick Lincoln, Narciso Martí-Oliet, José Meseguer, and José F. Quesada. Maude: Specification and programming in rewriting logic. *Theoretical Computer Science*, 285:187–243, 2002.
4. 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.
5. O. Fissore, I. Gnaedig, and H. Kirchner. Termination of rewriting with local strategies. In M. P. Bonacina and B. Gramlich, editors, *Selected papers of the 4th International Workshop on Strategies in Automated Deduction*, volume 58 of *Electronic Notes in Theoretical Computer Science*. Elsevier Science Publishers, 2001.
6. O. Fissore, I. Gnaedig, and H. Kirchner. CARIBOO : An induction based proof tool for termination with strategies. In *Proceedings of the Fourth International Conference on Principles and Practice of Declarative Programming (PPDP)*, Pittsburgh, USA, October 2002. ACM Press.
7. O. Fissore, I. Gnaedig, and H. Kirchner. Outermost ground termination. In *Proceedings of the Fourth International Workshop on Rewriting Logic and Its Applications*, volume 71 of *Electronic Notes in Theoretical Computer Science*, Pisa, Italy, September 2002. Elsevier Science Publishers.
8. 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.
9. I. Gnaedig, H. Kirchner, and O. Fissore. Induction for innermost and outermost ground termination. Technical Report A01-R-178, LORIA, Nancy, France, 2001.
10. J. Goubault-Larrecq. A proof of weak termination of typed lambda-sigma-calculi. In *Proceedings of the TYPES'96 Workshop*, volume 1512 of *Lecture Notes in Computer Science*, Aussois, France, 1998. Springer-Verlag.
11. Bernhard Gramlich. Relating innermost, weak, uniform and modular termination of term rewriting systems. In Andrei Voronkov, editor, *Proc. 3rd Int. Conf. on Logic Programming and Automated Reasoning (LPAR'92)*, volume 624 of *Lecture Notes in Artificial Intelligence*, pages 285–296, St. Petersburg, Russia, July 1992. Springer-Verlag.
12. Bernhard Gramlich. On termination and confluence properties of disjoint and constructor-sharing conditional rewrite systems. *Theoretical Computer Science*, 165(1):97–131, September 1996.

13. J.-M. Hullot. Canonical forms and unification. In W. Bibel and R. Kowalski, editors, *Proceedings 5th International Conference on Automated Deduction, Les Arcs (France)*, volume 87 of *Lecture Notes in Computer Science*, pages 318–334. Springer-Verlag, July 1980.
14. P. Klint. A meta-environment for generating programming environments. *ACM Transactions on Software Engineering and Methodology*, 2:176–201, 1993.
15. S. Lucas. Termination of OBJ programs with positive local strategies. In M. van der Brand and R. Verma, editors, *Proc. of 2nd International Workshop on Rule-Based Programming, RULE'01*, pages 64–78, Firenze, Italy, September 2001.
16. A. Middeldorp and E. Hamoen. Completeness results for basic narrowing. *Applicable Algebra in Engineering, Communication and Computation*, 5(3 & 4):213–253, 1994.

Appendix

A Proofs

In this section, we provide the proofs of Theorem 1, Theorem 2 and Theorem 3.

A.1 Proof of Theorem 1

In Theorem 1, we use the relation between rewriting and narrowing, expressed in the following proposition [13, 16].

A substitution $\sigma = \bigwedge_i (x_i = t_i)$ is said to be $\mathcal{R}$-normalized iff the t_i are irreducible for $\mathcal{R}$.

Proposition 1. *For any term t_0 and term rewriting system $\mathcal{R}$,*

A. If $t \leadsto_R^{m,l\to r,\sigma} t'$ then $\sigma t \to_R^{m,l\to r} t'$.

B. Let t_0 be a term and ρ be a $\mathcal{R}-$normalized substitution such that $\rho(t_0) \to_R^{m,l\to r} t'$.

Then there exist substitutions σ and μ such that :

1. $t_0 \leadsto_R^{m,l\to r,\sigma} t$,
2. $\mu(t) = t'$,
3. *for any* $x \in Var(t_0)$, $\rho(x) = \mu\sigma(x)$,
4. μ *is normalized.*

We now give the proof of Theorem 1.

Theorem 1. *Let $\mathcal{R}$ be a TRS on a set $\mathcal{F}$ of symbols, such that the constants of $\mathcal{F}$ weakly innermost terminate. If there exists an $\mathcal{F}$-stable ordering $\succ$ having the subterm property, such that for each non-constant defined symbol g, we have $SUCCESS(g, \succ)$, then any term of $\mathcal{T}(\mathcal{F})$ weakly innermost terminates.*

Proof. We prove by induction on $\mathcal{T}(\mathcal{F})$ that any ground instance $\theta f(x_1, \ldots, x_m)$ of any term $f(x_1, \ldots, x_m) \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ weakly innermost 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 noetherianity. 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 $\succ$ weakly innermost terminate since, by hypothesis, the constants of $\mathcal{F}$ weakly innermost 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.

By subterm property of $\succ$, we have $\theta f(x_1, \ldots, x_m) = f(\theta x_1, \ldots, \theta x_m) \succ \theta x_1, \ldots, \theta x_m$. Then, by induction hypothesis, let us suppose that the $\theta x_1, \ldots, \theta x_m$ weakly innermost terminate. Let $\theta x_1{\downarrow}, \ldots, \theta x_m{\downarrow}$ be any of its normal forms. It remains to prove that $f(\theta x_1{\downarrow}, \ldots, \theta x_m{\downarrow})$ weakly innermost terminates.

If f is a constructor, then $f(\theta x_1{\downarrow}, \ldots, \theta x_m{\downarrow})$ is irreducible for the innermost rewriting relation, and therefore, weakly innermost terminating.

If f is not a constructor, let us denote it g and prove that $g(\theta x_1, \ldots, \theta x_m)$ weakly innermost terminates for any θ satisfying $(A_0 = \top, C_0 = \top)$, if application of the inference rules on $(\{g(x_1, \ldots, x_m)\}, \top, \top)$, terminates on a proof tree with at least one successful branch in each set of branches with same (up to a renaming) narrowing substitution σ generated by any application of **Narrow**. 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 a state $s = (\{t\}, A, C)$, we associate the set of ground terms $G = \{\alpha t \mid \alpha \text{ and } \succ \text{ satisfy } (A, C)\}$, that is the set of ground instances represented by s. Inference rules **Abstract** and **Narrow** transform $(\{t\}, A, C)$ into $(\{t'\}, A', C')$ 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 $\beta t' \in G'$, $\beta t'$ weakly innermost terminates, then any αt in G weakly innermost terminates.

– Either **Abstract** is applied, so $\{f(u_1,\ldots,u_m)\}$ becomes $\{f(U_1,\ldots,U_m)\}$. For each α such that $\alpha(t)$ is in G, we prove that there exists a β such that $\beta(t')$ is in G' and such that weak innermost termination of $\beta(t')$ implies weak innermost termination of $\alpha(t)$.
According to the condition of **Abstract**, for some $\{k_1,...,k_l\} \subseteq \{i_1,...i_p\}$, where $\{i_1,...i_p\}$ is the set of positions for which we do not have $NF(u_{ij})$, $(A, C \wedge t_{ref} > u_{k_1},\ldots,u_{k_l})$ is satisfiable, and for $i \in \{i_1,...i_p\} \setminus \{k_1,...,k_l\}$, we have $WEAK{-}TERMIN(u_i)$. Then, for any α satisfying (A,C), $\alpha t_{ref} \succ \alpha u_{k_1},\ldots,\alpha u_{k_l}$. So $\alpha u_{k_1}\ldots,\alpha u_{k_l}$ weakly innermost terminate by induction hypothesis, and for $i \in \{i_1,...i_p\} \setminus \{k_1,...,k_l\}$, αu_i is weakly innermost terminating.
Then let us define $\beta = \alpha \cup \{X_{i_1} = \alpha u_{i_1}{\downarrow},\ldots,X_{i_p} = \alpha u_{i_p}{\downarrow}\}$. Clearly β satisfies (A',C'), and $\beta f(U_1,\ldots,U_m) = f(\alpha u_1{\downarrow},\ldots,\alpha u_m{\downarrow})$. Therefore, weak innermost termination of $\beta f(U_1,\ldots,U_m)$ implies weak innermost termination of $\alpha f(u_1,\ldots,u_m)$.
– Or **Narrow** is applied on $\{f(u_1,\ldots,u_m)\}$. For any α satisfying (A,C), either $\alpha f(u_1,\ldots,u_m)$ is irreducible, and so weakly innermost terminates, or $\alpha f(u_1,\ldots,u_m)$ is innermost reducible at the top position, thanks to the constraints in (A,C) which impose that α is a normalized substitution, and thanks to the fact that $f(u_1,\ldots,u_m)$ is the result of an abstracting step.
If $\alpha f(u_1,\ldots,u_m) \to^{\epsilon}_R t'$, thanks to Proposition 1, there exists a narrowing step $f(u_1,\ldots,u_m) \leadsto^{\sigma}_R v$ and a substitution β such that $t' = \beta v$. Moreover, this narrowing derivation is effectively produced by **Narrow**, which is applied in all possible ways on $\{f(u_1,\ldots,u_m)\}$.
On the other hand, on variables of $f(u_1,\ldots,u_m)$, i.e. variables introduced by the previous application of **Abstract**, we have $\alpha = \beta\sigma$. In addition, the domain of β, that is $Var(v)$, equal to the range of σ, can be extended to the variables of A and C by setting $\beta x = \alpha x$ for $x \in (Var(A) \cup Var(C)) \setminus Var(f(u_1,\ldots,u_m))$. Thus, since α satisfies (A,C), β satisfies $(\sigma A, \sigma C) = (\sigma A, C)$.
Moreover, since **Narrow** is applied with all possible substitutions and all possible rewrite rules, a term $\beta_j v_j, j \in [1..k]$ is produced on every innermost rewriting branch starting from $\alpha f(u_1,\ldots,u_m)$. All branches generated by the same substitution correspond to the same set of ground instances (these branches modelize different rewriting possibilities of the same set of ground instances). Indeed, the state $(\{f(u_1,\ldots,u_m)\}, A, C)$ represents the set of ground terms $\{f(\theta u_1,\ldots,\theta u_m)\}$, where θ is any instantiation satisfying A. If the narrowing step yields two narrowing branches with same most general unifier σ, both branches simulate rewriting possibilities of all ground instances $\{f(\theta u_1,\ldots,\theta u_m)\}$, where now θ is any instantiation satisfying A and σ.
By definition of weak innermost termination, among all branches representing the same set of ground instances it suffices to prove that one of them is successful.
Then, to prove weak innermost termination for every ground instance, it is sufficient, for one branch among the branches generated with the same substitution, to be successful.
Since, by hypothesis, for each set of branches with same substitution, we have a successful one, producing a state with say, a v_{j_l}, then weak innermost termination of all v_{j_l} implies weak innermost termination of $\alpha f(u_1,\ldots,u_m)$.

Let us now prove that the ground instances satisfying (A,C) of each term t removed from the set $\mathcal{T} = \{t\}$ containing the current term of the state during the application of the rules, weakly innermost terminate. The first rule removing terms from T is **Stop**. When **Stop** is applied and removes t from T, either $WEAK{-}TERMIN(t)$ and then αt weakly innermost terminates for any ground substitution α, or $(A, C \wedge t_{ref} > t)$ is satisfiable. Then, for any ground substitution α satisfying A and C, $\alpha t_{ref} \succ \alpha t$. By induction hypothesis, αt weakly innermost terminates.

The second rule removing terms from $\mathcal{T}$ is **StopN**. In that case, either the current term t is not narrowable, or it is narrowable with a substitution that makes the constraint system unsatisfiable. Moreover, any variable of t is a NF-variable. Indeed, if **StopN** is performed after **Abstract**, all variables of t are abstraction variables, i.e. NF-variables. If **StopN** is performed after **Narrow**, all variables of t also are NF-variables, since the range of the narrowing substitution contains only fresh NF-variables. Therefore, for any ground substitution α satisfying A and C, αt is irreducible, so trivially weakly innermost terminates.

Remark that the rule **StopA** never applies on the branches of the proof trees, necessary to establish the termination proof.

As the process is initialized with $\{t_{ref}\}$ and a constraint problem $(A,C) = (\top,\top)$ satisfiable by any ground substitution, we get that $g(\theta x_1,\ldots,\theta x_m)$ is innermost 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 innermost terminating for any ground instances $\theta x_1, \ldots, \theta x_m$ and the constants are innermost terminating, then any term of $\mathcal{T}(\mathcal{F})$ is innermost terminating.

A.2 Proof of Theorem 2

We now give the proof of Theorem 2.

Theorem 2. *Let $\mathcal{R}$ be a TRS on a set $\mathcal{F}$ of symbols, such that the constants of $\mathcal{F}$ weakly innermost terminate. If there exists an $\mathcal{F}$-stable ordering $\succ$ having the subterm property, such that for each non-constant defined symbol g, we have $SUCCESS{-}CUT(g, \succ)$, then any term of $\mathcal{T}(\mathcal{F})$ weakly innermost terminates.*

Proof. We prove by induction on $\mathcal{T}(\mathcal{F})$ that any ground instance $\theta f(x_1, \ldots, x_m)$ of any term $f(x_1, \ldots, x_m) \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ weakly innermost 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 noetherianity. 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 $\succ$ weakly innermost terminate since, by hypothesis, the constants of $\mathcal{F}$ weakly innermost 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.

By subterm property of $\succ$, we have $\theta f(x_1, \ldots, x_m) = f(\theta x_1, \ldots, \theta x_m) \succ \theta x_1, \ldots, \theta x_m$. Then, by induction hypothesis, let us suppose that the $\theta x_1, \ldots, \theta x_m$ weakly innermost terminate. Let $\theta x_1{\downarrow}, \ldots, \theta x_m{\downarrow}$ be any of its normal forms. It remains to prove that $f(\theta x_1{\downarrow}, \ldots, \theta x_m{\downarrow})$ weakly innermost terminates.

If f is a constructor, then $f(\theta x_1{\downarrow}, \ldots, \theta x_m{\downarrow})$ is irreducible for the innermost rewriting relation, and therefore, weakly innermost terminating.

If f is not a constructor, let us denote it g and prove that $g(\theta x_1, \ldots, \theta x_m)$ weakly innermost terminates for any θ satisfying $(A_0 = \top, C_0 = \top)$, if application of the inference rules on $\Lambda \sqsupset_p (\{g(x_1, \ldots, x_m)\}, \top, \top, \lambda)$, terminates on a proof tree reduced to a state $(\emptyset, A_p, C_p, \lambda)$. 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 a state $s = (\{t\}, A, C, \sigma)$, we associate the set of ground terms $G = \{\alpha t \mid \alpha \text{ and } \succ \text{ satisfy } (A, C)\}$, that is the set of ground instances represented by s. Inference rules **Abstract** and **Narrow** transform $(\{t\}, A, C, \sigma)$ into $(\{t'\}, A', C', \mu)$ 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 $\beta t' \in G'$, $\beta t'$ weakly innermost terminates, then any αt in G weakly innermost terminates.

- Either **Abstract** is applied, so $\{f(u_1, \ldots, u_m)\}$ becomes $\{f(U_1, \ldots, U_m)\}$. For each α such that $\alpha(t)$ is in G, we prove that there exists a β such that $\beta(t')$ is in G' and such that weak innermost termination of $\beta(t')$ implies weak innermost termination of $\alpha(t)$.
  According to the condition of **Abstract**, for some $\{k_1, ..., k_l\} \subseteq \{i_1, ...i_p\}$, where $\{i_1, ...i_p\}$ is the set of positions for which we do not have $NF(u_{ij})$, $(A, C \wedge t_{ref} > u_{k_1}, \ldots, u_{k_l})$ is satisfiable, and for $i \in \{i_1, ...i_p\} \setminus \{k_1, ..., k_l\}$, we have $WEAK{-}TERMIN(u_i)$. Then, for any α satisfying (A, C), $\alpha t_{ref} \succ \alpha u_{k_1}, \ldots, \alpha u_{k_l}$. So $\alpha u_{k_1} \ldots, \alpha u_{k_l}$ weakly innermost terminate by induction hypothesis, and for $i \in \{i_1, ...i_p\} \setminus \{k_1, ..., k_l\}$, αu_i is weakly innermost terminating.
  Then let us define $\beta = \alpha \cup \{X_{i_1} = \alpha u_{i_1}{\downarrow}, \ldots, X_{i_p} = \alpha u_{i_p}{\downarrow}\}$. Clearly β satisfies (A', C'), and $\beta f(U_1, \ldots, U_m) = f(\alpha u_1{\downarrow}, \ldots, \alpha u_m{\downarrow})$. Therefore, weak innermost termination of $\beta f(U_1, \ldots, U_m)$ implies weak innermost termination of $\alpha f(u_1, \ldots, u_m)$.
- Or **Narrow** is applied on $\{f(u_1, \ldots, u_m)\}$. For any α satisfying (A, C), either $\alpha f(u_1, \ldots, u_m)$ is irreducible, and so weakly innermost terminates, or $\alpha f(u_1, \ldots, u_m)$ is innermost reducible at the top position, thanks to the constraints in (A, C) which impose that α is a normalized substitution, and thanks to the fact that $f(u_1, \ldots, u_m)$ is the result of an abstracting step.
  If $\alpha f(u_1, \ldots, u_m) \rightarrow^{\epsilon}_R t'$, thanks to Proposition 1, there exists a narrowing step $f(u_1, \ldots, u_m) \rightsquigarrow^{\sigma}_R v$ and a substitution β such that $t' = \beta v$. Moreover, this narrowing derivation is effectively produced by **Narrow**, which is applied in all possible ways on $\{f(u_1, \ldots, u_m)\}$.

On the other hand, on variables of $f(u_1, \ldots, u_m)$, i.e. variables introduced by the previous application of **Abstract**, we have $\alpha = \beta\sigma$. In addition, the domain of β, that is $Var(v)$, equal to the range of σ, can be extended to the variables of A and C by setting $\beta x = \alpha x$ for $x \in (Var(A) \cup Var(C)) \setminus Var(f(u_1, \ldots, u_m)$. Thus, since α satisfies (A, C), β satisfies $(\sigma A, \sigma C) = (\sigma A, C)$.
Moreover, since **Narrow** is applied with all possible substitutions and all possible rewrite rules, a term $\beta_j v_j, j \in [1..k]$ is produced on every innermost rewriting branch starting from $\alpha f(u_1, \ldots, u_m)$.
In proof of Theorem 1, we showed that two branches generated by the narrowing step with the same substitution modelize two reductions of the same set of ground terms.
We can generalize this fact to the case where a substitution σ is more general than another one μ. We then get that the branch generated by the narrowing step with the substitution σ modelizes a reduction of a set of ground terms including the set of ground terms of the other branch, corresponding to μ.
By definition of weak innermost termination, when we have two branches representing two sets of ground instances, such that the first one is included into the second one, it suffices to prove that the second branch is successful.
Therefore, to prove weak innermost termination for every ground instance, it is sufficient, among the branches with comparable substitutions, for one branch with the most general substitution to be successful.
Therefore, as the rule **Cut** selects and keeps one branch with most general substitution, producing, say, a βv_{j_l}, among the branches generated with comparable substitutions, then weak innermost termination of the βv_{j_l} implies weak innermost termination of $\alpha f(u_1, \ldots, u_m)$.

Let us now prove that the ground instances satisfying (A, C) of each term t removed from the set $\mathcal{T} = \{t\}$ containing the current term of the state during the application of the rules, weakly innermost terminate. The first rule removing terms from T is **Stop**. When **Stop** is applied and removes t from T, either $WEAK\text{-}TERMIN(t)$ and then αt weakly innermost terminates for any ground substitution α, or $(A, C \wedge t_{ref} > t)$ is satisfiable. Then, for any ground substitution α satisfying A and C, $\alpha t_{ref} \succ \alpha t$. By induction hypothesis, αt weakly innermost terminates.

The second rule removing terms from $\mathcal{T}$ is **StopN**. In that case, either the current term t is not narrowable, or it is narrowable with a substitution that makes the constraint system unsatisfiable. Moreover, any variable of t is a NF-variable. Indeed, if **StopN** is performed after **Abstract**, all variables of t are abstraction variables, i.e. NF-variables. If **StopN** is performed after **Narrow**, all variables of t also are NF-variables, since the range of the narrowing substitution contains only fresh NF-variables. Therefore, for any ground substitution α satisfying A and C, αt is irreducible, so trivially weakly innermost terminates.

Let us now study the case of the rule **Shorten**. The only rules **Abstract** and **Narrow** generate children of a final state of the current proof tree. As proved above, the weak innermost termination of the ground instances represented by each of the children of a state implies the weak innermost termination of the ground instances represented by that state. The rule **Shorten** also removes a current term s of $\mathcal{T}$. By definition, it applies only if all children of the state s are of the form $(\emptyset, A, C, \mu)$. As proved previously in considering the rules **Stop** and **StopN**, and below in considering the rule **Cut**, a state $(\emptyset, A, C, \mu)$ generated from $s = (\{t\}, A, C, \mu)$ implies that the ground instances represented by s are weakly innermost terminating. So the ground instances represented by s are weakly innermost terminating.

The last rule removing terms from $\mathcal{T}$ is **Cut**. By definition of this rule, $s = (\{t\}, A, C, \mu)$ is replaced by $(\emptyset, A, C, \mu)$ if s is obtained from a state r that initiates a successful branch with a substitution σ more general than u. This successful branch covers the rewriting of a set of ground instances including the ground instances covered by branches obtained with μ. The success of one branch obtained with σ is then sufficient to ensure weak innermost termination of instances covered by branches obtained with μ, which can then be cut.

Remark that the rule **StopA** never applies on the branches of the proof trees, necessary to establish the termination proof.

As the process is initialized with $\{t_{ref}\}$ and a constraint problem $(A, C) = (\top, \top)$ satisfiable by any ground substitution, we get that $g(\theta x_1, \ldots, \theta x_m)$ is innermost 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 innermost terminating for any ground instances $\theta x_1, \ldots, \theta x_m$ and the constants are innermost terminating, then any term of $\mathcal{T}(\mathcal{F})$ is innermost terminating.

A.3 Proof of Theorem 3

We proceed by crossed proof with the following lemma.

Lemma 1. *Let $s = f(s_1, \ldots, s_m) \in \mathcal{T}(\mathcal{F})$ and $t \mapsto t'$ be any step of a transformation chain of s with respect to ST until $norm_{ST}(s)$. If this step is determined relatively to a step $u \hookrightarrow u'$ of the strategy tree ST_f, then t is a ground instance of u.*

Proof. Let $s = f(s_1, \ldots, s_m) \in \mathcal{T}(\mathcal{F})$ be a term and $t \mapsto t'$ any step of a transformation chain of s with respect to ST until $norm_{ST}(s)$. The proof is made by induction on the length l of the transformation chain of s with respect to ST.

Let $l = 1$.

- if $s \in \mathcal{C}$, by Definition 8, there is nothing to prove since the normalization of s with respect to ST does not use any strategy tree.
- if $f \in \mathcal{C}ons_{\mathcal{R}}$, by Definition 8, there is nothing to prove since ST_f is not used for normalizing the whole term s.
- if $g \in \mathcal{D}ef_R \setminus \mathcal{C}$, then by definition of the strategy S, the first step of each proof tree of ST is **Abstract**. As $l = 1$, this should also be the last step of the branch in the proof tree, which is impossible by definition of S.

Let $l = n \geq 1$.

Let $s' = g(s'_1, \ldots, s'_n)$ be the term obtained with a transformation chain of length n from s with respect to ST. Let the property be true for $l = n$, i.e. for any step $t \mapsto t'$ of the transformation chain of s with respect to ST until s', determined relatively to a step $u \hookrightarrow u'$ of the strategy tree ST_f, then t is a ground instance of u.

Let $s' \mapsto s''$ be the n+1-th step of the given transformation chain of s with respect to ST, determined relatively to a step $u' \hookrightarrow u''$ of the strategy tree ST_f. We prove that s'' is a ground instance of u''.

Let $s^0 \mapsto s'$ be the n-th step of the given transformation chain of s with respect to ST, determined relatively to a step $u^0 \hookrightarrow u'$ of the strategy tree ST_f. By hypothesis, s' is a ground instance of u'.

- if $u' \hookrightarrow u''$ is a step **Abstract**, then if u' is of the form $g(u'_1, \ldots, u'_m)$, u'' is of the form $g(U_1, \ldots, U_m)$, where the U_i are NF-variables X_i if we do not have $NF(u'_i)$, and u'_i otherwise.
  By Definition 8, $s' = g(s'_1, \ldots, s'_n) \mapsto g(s''_1, \ldots, s''_n)$, where the s''_i are $s'_i{\downarrow}$ if we have $WEAK\text{-}TERMIN(u'_i)$, and $norm_{ST}(s'_i)$ otherwise. In the first case, we distinguish two subcases :
  - either we have $NF(u'_i)$ and then, since by induction hypothesis s'_i is a ground instance of u'_i and $U_i = u'_i$, we have $s''_i = s'_i{\downarrow} = s'_i$ the same ground instance of U_i ;
  - or we do not have $NF(u'_i)$ and then $s'_i{\downarrow}$ is a ground instance of $U_i = X_i$; let us note that $s'_i{\downarrow}$ exists, since $WEAK\text{-}TERMIN(u'_i)$ ensures that every ground instance of u'_i is has an innermost normal form and, by induction hypothesis, s'_i is a ground instance of u'_i.

  In the second case, by Theorem 3 on each s'_i, we have $norm_{ST}(s'_i)$ in normal form, hence a ground instance of X_i.
  So s'' is a ground instance of u''.
- if $u' \hookrightarrow u''$ is a step **Narrow**, by definition 8, either $s' = g(s'_1, \ldots, s'_n) \rightarrow^{\epsilon, l \to r, \mu} s'' = \mu u''$, where the used branch of the step **Narrow** (always existing since **Narrow** is applied in all possible ways) is such that $g(u'_1, \ldots, u'_n) \rightsquigarrow^{\epsilon, l \to r, \sigma} u''$, and $\theta = \mu\sigma[Var(g(u'_1, \ldots, u'_n)]$ with $g(s'_1, \ldots, s'_n) = \theta g(u'_1, \ldots, u'_n)$. So s'' is a ground instance of u''.
  Or s' is irreducible and $s'' = s'$ and there is no narrowing step in the tree to follow, so there is no term u'' and then, there is nothing to prove.

– if $u' \hookrightarrow u''$ is a step **Stop** or **StopN**, by definition of these two steps, then u'' does not exist, since the state in ST_f after these steps are a state of the form $(\emptyset, C)$. Then there is nothing to prove.

Theorem 3. *Let $\mathcal{R}$ be a TRS proved weakly terminating with the strategy $S-CUT$, and ST its set of strategy trees. Then reducing any term $t \in \mathcal{T}(\mathcal{F})$ with respect to ST leads to an irreducible term, which is an innermost normal form of t for $\mathcal{R}$.*

Proof. Let $s = f(s_1, \ldots, s_m) \in \mathcal{T}(\mathcal{F})$ be a term to be reduced with respect to ST. We first prove if a transformation chain of s with respect to ST gives a term t, t is also an innermost reduced form of s for $\mathcal{R}$. We then show that the last term of any transformation chain of s with respect to ST is irreducible for $\mathcal{R}$. This term eventually exists, since the normalizing process always stops, for ST is a set of successful trees, whose branches only have a finite number of states.

Let us show that if a transformation chain of s with respect to ST gives a term t, t is also an innermost reduced form of s for $\mathcal{R}$, by induction on the length l of the transformation chain of s with respect to ST. For that, we use an induction on $\mathcal{T}(\mathcal{F})$ and the property : $norm_{ST}(s)$ is a normal form of s for $\mathcal{R}$, with an ordering $\succ$ satisfying all constraints generated by S in proving that $\mathcal{R}$ is weakly innermost terminating. We choose for that the ordering used in Theorem 1, for proving termination of $\mathcal{R}$. Such an ordering exists since, by hypothesis, $\mathcal{R}$ is a TRS proved weakly terminating with the strategy S.

Let $l = 1$.

– if $s \in \mathcal{C}$, by Definition 8, $norm_{ST}(s)$ reduces s until getting a constant constructor or a non constant term to be innermost normalized by $norm_{ST}$. In any case, $norm_{ST}(s)$ is an innermost normal form of s.
– if $f \in \mathcal{C}ons_{\mathcal{R}}$, by Definition 8, $norm_{ST}(f(s_1, \ldots, s_m)) = f(norm_{ST}(s_1), \ldots, norm_{ST}(s_n))$. With the second induction, we have $s = f(s_1, \ldots, s_m) \succ s_1, \ldots, s_m$, and then, by induction hypothesis, $norm_{ST}(s_1), \ldots, norm_{ST}(s_n)$ are respective normal forms of $s_1, \ldots, s_m$ for $\mathcal{R}$. Then, by definition of the innermost strategy, $f(norm_{ST}(s_1), \ldots, norm_{ST}(s_n))$ is an innermost reduced form of s for $\mathcal{R}$, and as f is a constructor symbol, is in normal form.
– if $g \in \mathcal{D}ef_R \setminus \mathcal{C}$, then by definition of the strategy S, the first step of each proof tree of ST is **Abstract**. As $l = 1$, this should also be the last step of the branch in the proof tree, which is impossible by definition of S.

Let $l = n \geq 1$.

Let $t = g(t_1, \ldots, t_n)$ be the term obtained with a transformation chain of length n from s with respect to ST. Let the property be true for $l = n$, i.e. t is an innermost reduced form of s with $\mathcal{R}$. We prove it for $l = n + 1$.

– If the current step in the proof tree of f is **Abstract**, by Definition 8, $t = g(t_1, \ldots, t_n) \mapsto g(t'_1, \ldots, t'_n)$, where the t'_i are $t_i{\downarrow}$ if we have $WEAK{-}TERMIN(u_i)$, and $norm_{ST}(t_i)$ otherwise. For $i \in \{1, \ldots, n\}$ such that $WEAK{-}TERMIN(u_i)$, any ground instance of u_i has an innermost normal form. By Lemma 1, t_i is a ground instance of u_i, and $t'_i = t_i{\downarrow}$ is an innermost normal form of t_i.
For $i \in \{1, \ldots, n\}$ such that we do not have $WEAK{-}TERMIN(u_i)$, induction hypothesis of the second induction applies to u_i. By Lemma 1, t_i is a ground instance of u_i, and then, by induction hypothesis, $norm_{ST}(t_i)$ is an innermost normal form of t_i.
Then, by definition of the innermost strategy, $g(norm_{ST}(t_1), \ldots, norm_{ST}(t_n))$ is an innermost reduced form of t for $\mathcal{R}$. As t is an innermost reduced form of s with $\mathcal{R}$, we conclude that $g(norm_{ST}(t_1), \ldots, norm_{ST}(t_m))$ is an innermost reduced form of s with $\mathcal{R}$.

– If the current step in the proof tree of f is **Narrow**, by Definition 8, $t = g(t_1, \ldots, t_n) \rightarrow^{\epsilon, l \rightarrow r, \mu} \mu t'$, where the used branch of the step **Narrow** (always existing since **Narrow** is applied in all possible ways) is such that $g(u_1, \ldots, u_n) \rightsquigarrow^{\epsilon, l \rightarrow r, \sigma} s'$, and $\theta = \mu\sigma[Var(g(u_1, \ldots, u_n)]$ with $g(t_1, \ldots, t_n) = \theta g(u_1, \ldots, u_n)$.

By definition of S, **Narrow** applied to $g(u_1, \ldots, u_n)$ narrows $g(u_1, \ldots, u_n)$ at the top position, and $u_1, \ldots, u_n$ are abstraction NF-variables or ground term in normal form. So $g(t_1, \ldots, t_n) = \theta g(u_1, \ldots, u_n)$ is such that $t_1, \ldots, t_n$ are ground terms in normal form. Therefore, the rewriting step $t = g(t_1, \ldots, t_n) \rightarrow^{\epsilon, l \rightarrow r, \mu} \mu t'$ is an innermost rewriting step of $g(t_1, \ldots, t_n)$. Thus, reasoning as in the previous case, as t is an innermost reduced form of s with $\mathcal{R}$, we conclude that $\mu t'$ is an innermost reduced form of s with $\mathcal{R}$.

If the latter rewriting is not possible (this is the case where μ does not exist), then t is already in normal form. The process stops, and the property is trivially satisfied.

- If the current step in the proof tree of f is **Stop**, then by Definition 8, $t = g(t_1, \ldots, t_m) \mapsto t'$ where $t' = t\downarrow$ if we have $WEAK\text{-}TERMIN(u)$ and $t' = norm_{ST}(t)$ otherwise.
  In the first case, any ground instance of u has an innermost normal form ; by Lemma 1, t is a ground instance of u and then $t' = t\downarrow$, innermost normal form of t, does exist.
  In the second case, induction hypothesis of the second induction applies on u ; by Lemma 1, t is a ground instance of u and then, by induction hypothesis on t, $norm_{ST}(t)$ is an innermost normal form of t for $\mathcal{R}$.
- If the current step in the proof tree of f is **StopN**, by Definition 8, the rewriting process is stopped on $t = g(t_1, \ldots, t_m)$, and then the property is trivially satisfied.

These last two cases, together with the special case in **Narrow**, are the steps where the normalizing process with respect to ST stops, in the case where $l = n > 1$. As shown previously, the first one stops with an irreducible form for $\mathcal{R}$. By definition of **StopN**, the second one stops on a non narrowable term, whose variables are only NF-variables. So t, which is a ground instance of this term by Lemma 1, is irreducible for $\mathcal{R}$. For the third one, t is already in normal form, which ends the proof.

B Examples

In this section, we give extra examples, together with, at the end, the full development of the example given in the introduction of the paper.

Example 3. Let R be the following TRS, which is weakly innermost terminating, but not strongly innermost terminating:

$$\begin{aligned} f(a) &\rightarrow f(b) \\ f(x) &\rightarrow x \\ b &\rightarrow a. \end{aligned}$$

We prove its weak termination on $\mathcal{T}(\mathcal{F})$ with $\mathcal{F} = \{f : 1, a : 0, b : 0\}$ as follows.

The terms a, b are weakly innermost terminating : a is a constructor, and the usable rules on b give the only rule $\{b \rightarrow a\}$, which is orientable with any simplifying ordering with the precedence $b \succ a$.

Applying the rules on $f(x)$, we get:

ϵ	$f(x)$	$A = \top$	$C = \top$	$\sigma = \lambda$
Abstract				
1	$f(X)$	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = \lambda$
Narrow				
1.1	X	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = Id$
1.2	$f(b)$	$A = (x\downarrow = a)$	$C = (f(x) > x)$	$\sigma = (X = a)$
StopN				
1.1	$\emptyset$	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = Id$
1.2	$f(b)$	$A = (x\downarrow = a)$	$C = (f(x) > x)$	$\sigma = (X = a)$

Cut				
1.1	$\emptyset$	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = Id$
1.2	$\emptyset$	$A = (x\downarrow = a)$	$C = (f(x) > x)$	$\sigma = (X = a)$
Shorten				
1	$\emptyset$	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = \lambda$
Shorten				
ϵ	$\emptyset$	$A = \top$	$C = \top$	$\sigma = \lambda$

Cut applies since $\sigma = (X = a)$ is less general the substitution $\sigma = Id$ of the state generated by **StopN**. The abstraction constraints $(x\downarrow = X)$ and $(x\downarrow = a)$ appearing in the proof tree are both satisfiable by any ground substitution θ such that $\theta x = \theta X = a$. The ordering constraints of C are satisfied by any $\mathcal{F}$-stable ordering having the subterm property.

Example 4. Let us consider the following rewrite system $\mathcal{R}$, which is not innermost terminating but is weakly innermost terminating.

$$\begin{array}{ll} f(x) & \to g(x) \\ f(a) & \to f(b) \\ b & \to a \\ g(x) & \to x \\ g(x) & \to f(g(x)) \end{array}$$

The terms a, b are weakly innermost terminating, as in the previous example.

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

ϵ	$g(x_1)$	$A = \top$	$C = \top,$	$\sigma = \lambda$
Abstract				
1	$g(X_1)$	$A = (x_1\downarrow = X_1)$	$C = (g(x_1) > x_1),$	$\sigma = \lambda$
Narrow				
1.1	X_1	$A = (x_1\downarrow = X_1)$	$C = (g(x_1) > x_1),$	$\sigma = Id$
1.2	$f(g(X_1))$	$A = (x_1\downarrow = X_1)$	$C = (g(x_1) > x_1),$	$\sigma = Id$
StopN				
1.1	$\emptyset$	$A = (x_1\downarrow = X_1)$	$C = (g(x_1) > x_1),$	$\sigma = Id$
1.2	$f(g(X_1))$	$A = (x_1\downarrow = X_1)$	$C = (g(x_1) > x_1),$	$\sigma = Id$
Cut				
1.1	$\emptyset$	$A = (x_1\downarrow = X_1)$	$C = (g(x_1) > x_1),$	$\sigma = Id$
1.2	$\emptyset$	$A = (x_1\downarrow = X_1)$	$C = (g(x_1) > x_1),$	$\sigma = Id$
Shorten				
1	$\emptyset$	$A = (x_1\downarrow = X_1)$	$C = (g(x_1) > x_1),$	$\sigma = \lambda$
Shorten				
ϵ	$\emptyset$	$A = \top$	$C = \top,$	$\sigma = \lambda$

The abstraction constraint $(x_1\downarrow = X_1)$ appearing in the proof tree is satisfiable by any ground substitution θ such that $\theta x_1 = \theta X_1 = a$. The ordering constraints of C are satisfied by any $\mathcal{F}$-stable ordering having the subterm property.

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

ϵ	$f(x_1)$	$A = \top$	$C = \top,$	$\sigma = \lambda$
Abstract				
1	$f(X_1)$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1),$	$\sigma = \lambda$
Narrow				
1.1	$g(X_1)$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1),$	$\sigma = Id$
1.2	$f(b)$	$A = (x_1\downarrow = a)$	$C = (f(x_1) > x_1),$	$\sigma = (X_1 = a)$
Abstract (twice)				
1.1.1	$g(X_1)$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1),$	$\sigma = Id$
1.2.1	$f(X_2)$	$A = (x_1\downarrow = a \wedge b\downarrow = X_2)$	$C = (f(x_1) > x_1),$	$\sigma = (X_1 = a)$

Narrow (**twice**)				
1.1.1.1	X_1	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$,	$\sigma = Id$
1.1.1.2	$f(g(X_1))$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$,	$\sigma = Id$
1.2.1.1	$g(X_2)$	$A = (x_1\downarrow = a \wedge b\downarrow = X_2)$	$C = (f(x_1) > x_1)$,	$\sigma = Id$
1.2.1.2	$f(b)$	$A = (x_1\downarrow = a \wedge b\downarrow = a)$	$C = (f(x_1) > x_1)$,	$\sigma = (X_2 = a)$
StopN				
1.1.1.1	$\emptyset$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$,	$\sigma = Id$
1.1.1.2	$f(g(X_1))$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$,	$\sigma = Id$
1.2.1.1	$g(X_2)$	$A = (x_1\downarrow = a \wedge b\downarrow = X_2)$	$C = (f(x_1) > x_1)$,	$\sigma = Id$
1.2.1.2	$f(b)$	$A = (x_1\downarrow = a \wedge b\downarrow = a)$	$C = (f(x_1) > x_1)$,	$\sigma = (X_2 = a)$
Cut				
1.1.1.1	$\emptyset$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$,	$\sigma = Id$
1.1.1.2	$\emptyset$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$,	$\sigma = Id$
1.2.1.1	$g(X_2)$	$A = (x_1\downarrow = a \wedge b\downarrow = X_2)$	$C = (f(x_1) > x_1)$,	$\sigma = Id$
1.2.1.2	$f(b)$	$A = (x_1\downarrow = a \wedge b\downarrow = a)$	$C = (f(x_1) > x_1)$,	$\sigma = (X_2 = a)$
Shorten				
1.1.1	$\emptyset$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$,	$\sigma = Id$
1.2.1.1	$g(X_2)$	$A = (x_1\downarrow = a \wedge b\downarrow = X_2)$	$C = (f(x_1) > x_1)$,	$\sigma = Id$
1.2.1.2	$f(b)$	$A = (x_1\downarrow = a \wedge b\downarrow = a)$	$C = (f(x_1) > x_1)$,	$\sigma = (X_2 = a)$
Shorten				
1.1	$\emptyset$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$,	$\sigma = Id$
1.2.1.1	$g(X_2)$	$A = (x_1\downarrow = a \wedge b\downarrow = X_2)$	$C = (f(x_1) > x_1)$,	$\sigma = Id$
1.2.1.2	$f(b)$	$A = (x_1\downarrow = a \wedge b\downarrow = a)$	$C = (f(x_1) > x_1)$,	$\sigma = (X_2 = a)$
Cut				
1.1	$\emptyset$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$,	$\sigma = Id$
1.2	$\emptyset$	$A = (x_1\downarrow = a)$	$C = (f(x_1) > x_1)$,	$\sigma = (X_1 = a)$
Shorten				
1	$\emptyset$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$,	$\sigma = \lambda$
Shorten				
ϵ	$\emptyset$	$A = \top$	$C = \top$,	$\sigma = \lambda$

Abstract applies on $f(b)$ (state 1.2) since b has been proved to be weakly innermost terminating. The second **Cut** directly applies to the state 1.2.

The abstraction constraint $(x_1\downarrow = X_1)$ is satisfiable by any ground substitution θ such that $\theta x_1 = \theta X_1 = a$, the abstraction constraint $(x_1\downarrow = a \wedge b\downarrow = X_2)$ by any ground substitution θ such that $\theta x_1 = \theta X_2 = a$ and the abstraction constraint $(x_1\downarrow = a \wedge b\downarrow = a)$ by any ground substitution θ such that $\theta x_1 = a$.

The ordering constrains of C are satisfied by any $\mathcal{F}$-stable ordering having the subterm property. Obviously, following Theorem 2, this ordering has to be the same as the previous one, satisfying the constraints related to inference rules applied on $g(x)$.

Example 5. Let us consider the following rewrite system $\mathcal{R}$, which is not innermost terminating but is weakly innermost terminating.

$$\begin{aligned} h(0,x) &\to f(0,x,x) \\ f(0,1,x) &\to h(x,x) \\ g(x,y) &\to x \\ g(x,y) &\to g(g(x,y),y). \end{aligned}$$

The terms $0, 1$ are obviously weakly innermost terminating. Applying the inference rules on $h(x_1, x_2)$, we get:

ϵ	$h(x_1, x_2)$	$A = \top$	$C = \top$	$\sigma = \lambda$
Abstract				
1	$h(X_1, X_2)$	$A = (x_1{\downarrow} = X_1 \wedge x_2{\downarrow} = X_2)$	$C = (h(x_1, x_2) > x_1, x_2)$	$\sigma = \lambda$
Narrow				
1.1	$f(0, X_2, X_2)$	$A = (x_1{\downarrow} = 0 \wedge x_2{\downarrow} = X_2)$	$C = (h(x_1, x_2) > x_1, x_2)$	$\sigma = (X_1 = 0)$
Narrow				
1.1.1	$h(1, 1)$	$A = (x_1{\downarrow} = 0 \wedge x_2{\downarrow} = 1)$	$C = (h(x_1, x_2) > x_1, x_2)$	$\sigma = (X_2 = 1)$
StopN				
1.1.1	$\emptyset$	$A = (x_1{\downarrow} = 0 \wedge x_2{\downarrow} = 1)$	$C = (h(x_1, x_2) > x_1, x_2)$	$\sigma = (X_2 = 1)$
Shorten				
1.1	$\emptyset$	$A = (x_1{\downarrow} = 0 \wedge x_2{\downarrow} = X_2)$	$C = (h(x_1, x_2) > x_1, x_2)$	$\sigma = (X_1 = 0)$
Shorten				
1	$\emptyset$	$A = (x_1{\downarrow} = X_1 \wedge x_2{\downarrow} = X_2)$	$C = (h(x_1, x_2) > x_1, x_2))$	$\sigma = \lambda$
Shorten				
ϵ	$\emptyset$	$A = \top$	$C = \top$	$\sigma = \lambda$

After the first **Narrow**, accordingly to the strategy, **Abstract** applies, but has no effect on the current state, since X_2 is already a NF-variable, and 0 is a ground term in normal form. The abstraction constraint $(x_1{\downarrow} = 0 \wedge x_2{\downarrow} = X_2)$ is satisfiable by any ground substitution θ such that $\theta x_1 = \theta x_2 = \theta X_2 = 0$, and the abstraction constraint $(x_1{\downarrow} = 0 \wedge x_2{\downarrow} = 1)$ is satisfiable by any ground substitution θ such that $\theta x_1 = 0$ and $\theta x_2 = 1$.

Appying the inference rules on $f(x_1, x_2, x_3)$, we get:

ϵ	$f(x_1, x_2, x_3)$	$A = \top$	$C = \top$ $\sigma = \lambda$
Abstract			
1	$f(X_1, X_2, X_3)$	$A = (x_1{\downarrow} = X_1 \wedge x_2{\downarrow} = X_2 \wedge x_3{\downarrow} = X_3)$	$C = (f(x_1, x_2, x_3) > x_1, x_2, x_3)$ $\sigma = \lambda$
Narrow			
1.1	$h(X_3, X_3)$	$A = (x_1{\downarrow} = 0 \wedge x_2{\downarrow} = 1 \wedge x_3{\downarrow} = X_3)$	$C = (f(x_1, x_2, x_3) > x_1, x_2, x_3)$ $\sigma = (X_1 = 0 \wedge X_2 = 1)$
Narrow			
1.1.1	$f(0, 0, 0)$	$A = (x_1{\downarrow} = 0 \wedge x_2{\downarrow} = 1 \wedge x_3{\downarrow} = 0)$	$C = (f(x_1, x_2, x_3) > x_1, x_2, x_3)$ $\sigma = (X_3 = 0)$
StopN			
1.1.1	$\emptyset$	$A = (x_1{\downarrow} = 0 \wedge x_2{\downarrow} = 1 \wedge x_3{\downarrow} = 0)$	$C = (f(x_1, x_2, x_3) > x_1, x_2, x_3)$ $\sigma = (X_3 = 0)$
Shorten			
1.1	$\emptyset$	$A = (x_1{\downarrow} = 0 \wedge x_2{\downarrow} = 1 \wedge x_3{\downarrow} = X_3)$	$C = (f(x_1, x_2, x_3) > x_1, x_2, x_3)$ $\sigma = (X_1 = 0 \wedge X_2 = 1)$
Shorten			
1	$\emptyset$	$A = (x_1{\downarrow} = X_1 \wedge x_2{\downarrow} = X_2 \wedge x_3{\downarrow} = X_3)$	$C = (f(x_1, x_2, x_3) > x_1, x_2, x_3)$ $\sigma = \lambda$
Shorten			
ϵ	$\emptyset$	$A = \top$	$C = \top$ $\sigma = \lambda$

Applying the rules on $g(x_1, x_2)$, we get:

ϵ	$g(x_1, x_2)$	$A = \top$	$C = \top$	$\sigma = \lambda$
Abstract				
1	$g(X_1, X_2)$	$A = (x_1\downarrow = X_1 \wedge x_2\downarrow = X_2)$	$C = (g(x_1, x_2) > x_1, x_2)$	$\sigma = \lambda$
Narrow				
1.1	X_1	$A = (x_1\downarrow = X_1 \wedge x_2\downarrow = X_2)$	$C = (g(x_1, x_2) > x_1, x_2)$	$\sigma = Id$
1.2	$g(g(X_1, X_2), X_2)$	$A = (x_1\downarrow = X_1 \wedge x_2\downarrow = X_2)$	$C = (g(x_1, x_2) > x_1, x_2)$	$\sigma = Id$
StopN				
1.1	$\emptyset$	$A = (x_1\downarrow = X_1 \wedge x_2\downarrow = X_2)$	$C = (g(x_1, x_2) > x_1, x_2)$	$\sigma = Id$
1.2	$g(g(X', Y'), Y')$	$A = (x_1\downarrow = X_1 \wedge x_2\downarrow = X_2)$	$C = (g(x_1, x_2) > x_1, x_2)$	$\sigma = Id$
Cut				
1.1	$\emptyset$	$A = (x_1\downarrow = X_1 \wedge x_2\downarrow = X_2)$	$C = (g(x_1, x_2) > x_1, x_2)$	$\sigma = Id$
1.2	$\emptyset$	$A = (x_1\downarrow = X_1 \wedge x_2\downarrow = X_2)$	$C = (g(x_1, x_2) > x_1, x_2)$	$\sigma = Id$
Shorten				
1	$\emptyset$	$A = (x_1\downarrow = X_1 \wedge x_2\downarrow = X_2)$	$C = (g(x_1, x_2) > x_1, x_2)$	$\sigma = \lambda$
Shorten				
ϵ	$\emptyset$	$A = \top$	$C = \top$	$\sigma = \lambda$

Example 6. Let us consider the following rewrite system $\mathcal{R}$, which is not innermost terminating but is weakly innermost terminating.

$$\begin{aligned} f(x) &\to g(f(x)) \\ f(x) &\to a \\ g(x) &\to f(x). \end{aligned}$$

The term a is obviously weakly innermost terminating. Applying the inference rules on $f(x)$, we get:

ϵ	$f(x)$	$A = \top$	$C = \top$	$\sigma = \lambda$
Abstract				
1	$f(X)$	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = \lambda$
Narrow				
1.1	a	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = Id$
1.2	$g(f(X))$	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = Id$
StopN				
1.1	$\emptyset$	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = Id$
1.2	$g(f(X))$	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = Id$
Cut				
1.1	$\emptyset$	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = Id$
1.2	$\emptyset$	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = Id$
Shorten				
1	$\emptyset$	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = \lambda$
Shorten				
ϵ	$\emptyset$	$A = \top$	$C = \top$	$\sigma = \lambda$

Applying the inference rules on $g(x)$, we get:

ϵ	$g(x)$	$A = \top$	$C = \top$	$\sigma = \lambda$
Abstract				
1	$g(X)$	$A = (x\downarrow = X)$	$C = (g(x) > x)$	$\sigma = \lambda$
Narrow				
1.1	$f(X)$	$A = (x\downarrow = X)$	$C = (g(x) > x)$	$\sigma = Id$
Narrow				
1.1.1	a	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = Id$
1.1.2	$g(f(X))$	$A = (x\downarrow = X)$	$C = (g(x) > x)$	$\sigma = Id$
StopN				
1.1.1	$\emptyset$	$A = (x\downarrow = X)$	$C = (g(x) > x)$	$\sigma = Id$
1.1.2	$g(f(X))$	$A = (x\downarrow = X)$	$C = (g(x) > x)$	$\sigma = Id$
Cut				
1.1.1	$\emptyset$	$A = (x\downarrow = X)$	$C = (g(x) > x)$	$\sigma = Id$
1.1.2	$\emptyset$	$A = (x\downarrow = X)$	$C = (g(x) > x)$	$\sigma = Id$
Shorten				
1.1	$\emptyset$	$A = (x\downarrow = X)$	$C = (g(x) > x)$	$\sigma = Id$
Shorten				
1	$\emptyset$	$A = (x\downarrow = X)$	$C = (g(x) > x)$	$\sigma = \lambda$
Shorten				
ϵ	$\emptyset$	$A = \top$	$C = \top$	$\sigma = \lambda$

Example 7. Let us consider the following rewrite system $\mathcal{R}$, which is not innermost terminating but is weakly innermost terminating.

$$\begin{aligned} f(h(x)) &\to f(g(x)) \\ f(x) &\to i(x) \\ g(x) &\to h(x). \end{aligned}$$

We prove the weak innermost termination of this system on $\mathcal{T}(\mathcal{F}, \mathcal{X})$, where $F = \{f : 1, g : 1, h : 1, i : 1, a : 0\}$.

The term a is obviously weakly innermost terminating. Applying the inference rules on $f(x)$, we get:

ϵ	$f(x)$	$A = \top$	$C = \top$	$\sigma = \lambda$
Abstract				
1	$f(X)$	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = \lambda$
Narrow				
1.1	$i(X)$	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = Id$
1.2	$f(g(X'))$	$A = (x\downarrow = h(X'))$	$C = (f(x) > x)$	$\sigma = (X = h(X'))$
StopN				
1.1.	$\emptyset$	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = Id$
1.2	$f(g(X'))$	$A = (x\downarrow = h(X'))$	$C = (f(x) > x)$	$\sigma = (X = h(X'))$
Cut				
1.1	$\emptyset$	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = Id$
1.2	$\emptyset$	$A = (x\downarrow = h(X'))$	$C = (f(x) > x)$	$\sigma = (X = h(X'))$
Shorten				
1	$\emptyset$	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = \lambda$
Shorten				
ϵ	$\emptyset$	$A = \top$	$C = \top$	$\sigma = \lambda$

On $g(x)$, **Abstract** and **StopN** apply once.

Example 8. Let us consider the following rewrite system $\mathcal{R}$, which is not innermost terminating but is weakly innermost terminating.

$$
\begin{array}{ll}
f(g(x)) & \to f(h(x)) \\
f(h(x)) & \to f(g(x)) \\
f(x) & \to i(x).
\end{array}
$$

We prove the weak innermost termination of this system on $\mathcal{T}(\mathcal{F}, \mathcal{X})$, where $F = \{f : 1, g : 1, h : 1, i : 1, a : 0\}$.

The term a is obviously weakly innermost terminating. Applying the inference rules on $f(x)$, we get:

ϵ	$f(x)$	$A = \top$	$C = \top$	$\sigma = \lambda$
Abstract				
1	$f(X)$	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = \lambda$
Narrow				
1.1	$f(h(X_1))$	$A = (x\downarrow = g(X_1))$	$C = (f(x) > x)$	$\sigma = (X = g(X_1))$
1.2	$f(g(X_2))$	$A = (x\downarrow = h(X_2))$	$C = (f(x) > x)$	$\sigma = (X = h(X_2))$
1.3	$i(X)$	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = Id$
StopN				
1.1	$f(h(X_1))$	$A = (x\downarrow = g(X_1))$	$C = (f(x) > x)$	$\sigma = (X = g(X_1))$
1.2	$f(g(X_2))$	$A = (x\downarrow = h(X_2))$	$C = (f(x) > x)$	$\sigma = (X = h(X_2))$
1.3	$\emptyset$	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = Id$
Cut (twice)				
1.1.	$\emptyset$	$A = (x\downarrow = g(X_1))$	$C = (f(x) > x)$	$\sigma = (X = g(X_1))$
1.2	$\emptyset$	$A = (x\downarrow = h(X_2))$	$C = (f(x) > x)$	$\sigma = (X = h(X_2))$
1.3	$\emptyset$	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = Id$
Shorten				
1	$\emptyset$	$A = (x\downarrow = X)$	$C = (f(x) > x)$	$\sigma = \lambda$
Shorten				
ϵ	$\emptyset$	$A = \top$	$C = \top$	$\sigma = \lambda$

Let us now make precise the full development of the example presented in the introduction of the paper, with detailed applications of the inference rules. For readibility :

- we do write the development of branches that are going to be cut ;
- when there is no subterm to abstract, because every subterm u_i is such that $NF(u_i)$, we do not write the application of **Abstract** ;
- the rules **Stop** and **StopN** always result in one successful state, on which **Shorten** applies. Here, we include the **Shorten** application directly in **Stop** and **StopN**, so that the current state direclty becomes $\emptyset$.

Let us apply the inference rules on $f(x_1)$.

ϵ	$f(x_1)$	$A = \top$	$C = \top$	$\sigma = \lambda$
Abstract				
1	$f(X_1)$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$	$\sigma = \lambda$
Narrow				
1.1	$p(s(X_1))$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$	$\sigma = Id$
1.2	$p(s(s(X_1)))$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$	$\sigma = Id$

In the following, we show that the term of the state 1.1 is transformed into $\emptyset$, which allows to cut the branch 1.2. For readibility, we do not develop the latter branch. Note also that when the **Abstract** rule does not transform the current term, when there is no subterm to abstract, then we just do not write it.

Narrow				
1.1.1	$p(X_2)$	$A = (x_1\downarrow = s(X_2))$	$C = (f(x_1) > x_1)$	$\sigma = (X_1 = s(X_2))$
1.1.2	$f(0)$	$A = (x_1\downarrow = 0)$	$C = (f(x_1) > x_1)$	$\sigma = (X_1 = 0)$
1.2	$p(s(s(X_1)))$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$	$\sigma = Id$
Stop				
1.1.1	$\emptyset$	$A = (x_1\downarrow = s(X_2))$	$C = (f(x_1) > x_1, p(X_2))$	$\sigma = (X_1 = s(X_2))$
1.1.2	$f(0)$	$A = (x_1\downarrow = 0)$	$C = (f(x_1) > x_1)$	$\sigma = (X_1 = 0)$
1.2	$p(s(s(X_1)))$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$	$\sigma = Id$
Narrow				
1.1.1	$\emptyset$	$A = (x_1\downarrow = s(X_2))$	$C = (f(x_1) > x_1, p(X_2))$	$\sigma = (X_1 = s(X_2))$
1.1.2.1	$p(s(0))$	$A = (x_1\downarrow = 0)$	$C = (f(x_1) > x_1)$	$\sigma = Id$
1.1.2.2	$p(s(s(0)))$	$A = (x_1\downarrow = 0)$	$C = (f(x_1) > x_1)$	$\sigma = Id$
1.2	$p(s(s(X_1)))$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$	$\sigma = Id$

Stop applies on $p(X_2)$ with any simplification ordering with the precedence $f \succ_{\mathcal{F}} p$. Indeed, assuming that any term is greater than its normal form, we get from $A = (x_1\downarrow = s(X_2))$ that $x_1 \succeq s(X_2) \succ X_2$, and hence $f(x_1) \succ p(X_2)$ with the latter simplification ordering. Note that in theory, applying **Stop** to the state 1.1.1 should give rise to a single successful state 1.1.1.1, and then **Shorten** would result in our current successful state 1.1.1. Here, and also in the following, we will shorten the application of **Stop** or **StopN** followed by a **Shorten** into one single application.

In the following, we show that the term of the state 1.1.2.2 is transformed into $\emptyset$, which allows to cut the branch 1.1.2.1. For readibility, we do not develop this branch.

Narrow

1.1.1	$\emptyset$	$A = (x_1\downarrow = s(X_2))$	$C = (f(x_1) > x_1, p(X_2))$	$\sigma = (X_1 = s(X_2))$
1.1.2.1	$p(s(0))$	$A = (x_1\downarrow = 0)$	$C = (f(x_1) > x_1)$	$\sigma = Id$
1.1.2.2.1	$p(0)$	$A = (x_1\downarrow = 0)$	$C = (f(x_1) > x_1)$	$\sigma = Id$
1.2	$p(s(s(X_1)))$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$	$\sigma = Id$

Narrow

1.1.1	$\emptyset$	$A = (x_1\downarrow = s(X_2))$	$C = (f(x_1) > x_1, p(X_2))$	$\sigma = (X_1 = s(X_2))$
1.1.2.1	$p(s(0))$	$A = (x_1\downarrow = 0)$	$C = (f(x_1) > x_1)$	$\sigma = Id$
1.1.2.2.1.1	0	$A = (x_1\downarrow = 0)$	$C = (f(x_1) > x_1)$	$\sigma = Id$
1.2	$p(s(s(X_1)))$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$	$\sigma = Id$

StopN

1.1.1	$\emptyset$	$A = (x_1\downarrow = s(X_2))$	$C = (f(x_1) > x_1, p(X_2))$	$\sigma = (X_1 = s(X_2))$
1.1.2.1	$p(s(0))$	$A = (x_1\downarrow = 0)$	$C = (f(x_1) > x_1)$	$\sigma = Id$
1.1.2.2.1.1	$\emptyset$	$A = (x_1\downarrow = 0)$	$C = (f(x_1) > x_1)$	$\sigma = Id$
1.2	$p(s(s(X_1)))$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$	$\sigma = Id$

Shorten

1.1.1	$\emptyset$	$A = (x_1\downarrow = s(X_2))$	$C = (f(x_1) > x_1, p(X_2))$	$\sigma = (X_1 = s(X_2))$
1.1.2.1	$p(s(0))$	$A = (x_1\downarrow = 0)$	$C = (f(x_1) > x_1)$	$\sigma = Id$
1.1.2.2.1	$\emptyset$	$A = (x_1\downarrow = 0)$	$C = (f(x_1) > x_1)$	$\sigma = Id$
1.2	$p(s(s(X_1)))$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$	$\sigma = Id$

Shorten

1.1.1	$\emptyset$	$A = (x_1\downarrow = s(X_2))$	$C = (f(x_1) > x_1, p(X_2))$	$\sigma = (X_1 = s(X_2))$
1.1.2.1	$p(s(0))$	$A = (x_1\downarrow = 0)$	$C = (f(x_1) > x_1)$	$\sigma = Id$
1.1.2.2	$\emptyset$	$A = (x_1\downarrow = 0)$	$C = (f(x_1) > x_1)$	$\sigma = Id$
1.2	$p(s(s(X_1)))$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$	$\sigma = Id$

Cut

1.1.1	$\emptyset$	$A = (x_1\downarrow = s(X_2))$	$C = (f(x_1) > x_1, p(X_2))$	$\sigma = (X_1 = s(X_2))$
1.1.2.1	$\emptyset$	$A = (x_1\downarrow = 0)$	$C = (f(x_1) > x_1)$	$\sigma = Id$
1.1.2.2	$\emptyset$	$A = (x_1\downarrow = 0)$	$C = (f(x_1) > x_1)$	$\sigma = Id$
1.2	$p(s(s(X_1)))$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$	$\sigma = Id$

Shorten

1.1.1	$\emptyset$	$A = (x_1\downarrow = s(X_2))$	$C = (f(x_1) > x_1, p(X_2))$	$\sigma = (X_1 = s(X_2))$
1.1.2	$\emptyset$	$A = (x_1\downarrow = 0)$	$C = (f(x_1) > x_1)$	$\sigma = (X_1 = 0)$
1.2	$p(s(s(X_1)))$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$	$\sigma = Id$

Shorten

1.1	$\emptyset$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$	$\sigma = Id$
1.2	$p(s(s(X_1)))$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$	$\sigma = Id$

Cut

1.1	$\emptyset$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$	$\sigma = Id$
1.2	$\emptyset$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$	$\sigma = Id$

Shorten

1	$\emptyset$	$A = (x_1\downarrow = X_1)$	$C = (f(x_1) > x_1)$	$\sigma = \lambda$

Shorten

ϵ	$\emptyset$	$A = \top$	$C = \top$	$\sigma = \lambda$

Let us now apply the inference rules on $p(x_1)$.

ϵ	$p(x_1)$	$A = \top$	$C = \top$	$\sigma = \lambda$
Abstract				
1	$p(X_1)$	$A = (x_1{\downarrow} = X_1)$	$C = (p(x_1) > x_1)$	$\sigma = \lambda$
Narrow				
1.1	0	$A = (x_1{\downarrow} = 0)$	$C = (p(x_1) > x_1)$	$\sigma = (X_1 = 0)$
1.2	$f(0)$	$A = (x_1{\downarrow} = s(0))$	$C = (p(x_1) > x_1)$	$\sigma = (X_1 = s(0))$
1.3	$p(X_2)$	$A = (x_1{\downarrow} = s(s(X_2)))$	$C = (p(x_1) > x_1)$	$\sigma = (X_1 = s(s(X_2))$
StopN				
1.1	$\emptyset$	$A = (x_1{\downarrow} = 0)$	$C = (p(x_1) > x_1)$	$\sigma = (X_1 = 0)$
1.2	$f(0)$	$A = (x_1{\downarrow} = s(0))$	$C = (p(x_1) > x_1)$	$\sigma = (X_1 = s(0))$
1.3	$p(X_2)$	$A = (x_1{\downarrow} = s(s(X_2)))$	$C = (p(x_1) > x_1)$	$\sigma = (X_1 = s(s(X_2))$
Stop				
1.1	$\emptyset$	$A = (x_1{\downarrow} = 0)$	$C = (p(x_1) > x_1)$	$\sigma = (X_1 = 0)$
1.2	$f(0)$	$A = (x_1{\downarrow} = s(0))$	$C = (p(x_1) > x_1)$	$\sigma = (X_1 = s(0))$
1.3	$\emptyset$	$A = (x_1{\downarrow} = s(s(X_2)))$	$C = (p(x_1) > x_1, p(X_2))$	$\sigma = (X_1 = s(s(X_2))$
Narrow				
1.1	$\emptyset$	$A = (x_1{\downarrow} = 0)$	$C = (p(x_1) > x_1)$	$\sigma = (X_1 = 0)$
1.2.1	$p(s(0))$	$A = (x_1{\downarrow} = s(0))$	$C = (p(x_1) > x_1)$	$\sigma = Id$
1.2.2	$p(s(s(0)))$	$A = (x_1{\downarrow} = s(0))$	$C = (p(x_1) > x_1)$	$\sigma = Id$
1.3	$\emptyset$	$A = (x_1{\downarrow} = s(s(X_2)))$	$C = (p(x_1) > x_1, p(X_2))$	$\sigma = (X_1 = s(s(X_2))$

In the following, we show that the term of the state 1.2.2 is transformed into $\emptyset$, which allows to cut the branch 1.2.1, which we hence do not develop.

Narrow				
1.1	$\emptyset$	$A = (x_1{\downarrow} = 0)$	$C = (p(x_1) > x_1)$	$\sigma = (X_1 = 0)$
1.2.1	$p(s(0))$	$A = (x_1{\downarrow} = s(0))$	$C = (p(x_1) > x_1)$	$\sigma = Id$
1.2.2.1	$p(0)$	$A = (x_1{\downarrow} = s(0))$	$C = (p(x_1) > x_1)$	$\sigma = Id$
1.3	$\emptyset$	$A = (x_1{\downarrow} = s(s(X_2)))$	$C = (p(x_1) > x_1, p(X_2))$	$\sigma = (X_1 = s(s(X_2))$
Narrow				
1.1	$\emptyset$	$A = (x_1{\downarrow} = 0)$	$C = (p(x_1) > x_1)$	$\sigma = (X_1 = 0)$
1.2.1	$p(s(0))$	$A = (x_1{\downarrow} = s(0))$	$C = (p(x_1) > x_1)$	$\sigma = Id$
1.2.2.1.1	0	$A = (x_1{\downarrow} = s(0))$	$C = (p(x_1) > x_1)$	$\sigma = Id$
1.3	$\emptyset$	$A = (x_1{\downarrow} = s(s(X_2)))$	$C = (p(x_1) > x_1, p(X_2))$	$\sigma = (X_1 = s(s(X_2))$
StopN				
1.1	$\emptyset$	$A = (x_1{\downarrow} = 0)$	$C = (p(x_1) > x_1)$	$\sigma = (X_1 = 0)$
1.2.1	$p(s(0))$	$A = (x_1{\downarrow} = s(0))$	$C = (p(x_1) > x_1)$	$\sigma = Id$
1.2.2.1.1	$\emptyset$	$A = (x_1{\downarrow} = s(0))$	$C = (p(x_1) > x_1)$	$\sigma = Id$
1.3	$\emptyset$	$A = (x_1{\downarrow} = s(s(X_2)))$	$C = (p(x_1) > x_1, p(X_2))$	$\sigma = (X_1 = s(s(X_2))$
Shorten				
1.1	$\emptyset$	$A = (x_1{\downarrow} = 0)$	$C = (p(x_1) > x_1)$	$\sigma = (X_1 = 0)$
1.2.1	$p(s(0))$	$A = (x_1{\downarrow} = s(0))$	$C = (p(x_1) > x_1)$	$\sigma = Id$
1.2.2.1	$\emptyset$	$A = (x_1{\downarrow} = s(0))$	$C = (p(x_1) > x_1)$	$\sigma = Id$
1.3	$\emptyset$	$A = (x_1{\downarrow} = s(s(X_2)))$	$C = (p(x_1) > x_1, p(X_2))$	$\sigma = (X_1 = s(s(X_2))$
Shorten				
1.1	$\emptyset$	$A = (x_1{\downarrow} = 0)$	$C = (p(x_1) > x_1)$	$\sigma = (X_1 = 0)$
1.2.1	$p(s(0))$	$A = (x_1{\downarrow} = s(0))$	$C = (p(x_1) > x_1)$	$\sigma = Id$
1.2.2	$\emptyset$	$A = (x_1{\downarrow} = s(0))$	$C = (p(x_1) > x_1)$	$\sigma = Id$
1.3	$\emptyset$	$A = (x_1{\downarrow} = s(s(X_2)))$	$C = (p(x_1) > x_1, p(X_2))$	$\sigma = (X_1 = s(s(X_2))$
Cut				
1.1	$\emptyset$	$A = (x_1{\downarrow} = 0)$	$C = (p(x_1) > x_1)$	$\sigma = (X_1 = 0)$
1.2.1	$\emptyset$	$A = (x_1{\downarrow} = s(0))$	$C = (p(x_1) > x_1)$	$\sigma = Id$
1.2.2	$\emptyset$	$A = (x_1{\downarrow} = s(0))$	$C = (p(x_1) > x_1)$	$\sigma = Id$
1.3	$\emptyset$	$A = (x_1{\downarrow} = s(s(X_2)))$	$C = (p(x_1) > x_1, p(X_2))$	$\sigma = (X_1 = s(s(X_2))$

Shorten

1.1	$\emptyset\ A = (x_1{\downarrow} = 0)$	$C = (p(x_1) > x_1)$	$\sigma = (X_1 = 0)$
1.2	$\emptyset\ A = (x_1{\downarrow} = s(0))$	$C = (p(x_1) > x_1)$	$\sigma = (X_1 = s(0))$
1.3	$\emptyset\ A = (x_1{\downarrow} = s(s(X_2)))$	$C = (p(x_1) > x_1, p(X_2))$	$\sigma = (X_1 = s(s(X_2))$

Shorten

1	$\emptyset\ A = (x_1{\downarrow} = X_1)$	$C = (p(x_1) > x_1)$	$\sigma = \lambda$

Shorten

ϵ	$\emptyset\ A = \top$	$C = \top$	$\sigma = \lambda$