Inductive Proof Search Modulo

Fabrice Nahon, Claude Kirchner, Hélène Kirchner

Inductive Proof Search Modulo

Fabrice Nahon[1], Claude Kirchner[2], Hélène Kirchner[2]

[1] LORIA***
[2] INRIA & LORIA
Nancy, France

Abstract. We present an original narrowing-based proof search method for inductive theorems in equational rewrite theories given by a rewrite system $\mathcal{R}$ and a set E of equalities. It has the specificity to be grounded on deduction modulo and to rely on narrowing to provide both induction variables and instantiation schemas. Whenever the equational rewrite system $(\mathcal{R}, E)$ has good properties of termination, sufficient completeness, and when E is constructor and variable preserving, narrowing at defined-innermost positions leads to consider only unifiers which are constructor substitutions. This is especially interesting for associative and associative-commutative theories for which the general proof search system is refined. The method is shown to be sound and refutationaly complete.
Keywords: Deduction modulo, Noetherian induction, equational rewriting, equational narrowing.

Introduction

Proof by induction is a main reasoning principle and is of prime interest in informatics. Typically in hardware and software verification problems, when dealing with security protocols or safety properties of embedded systems, reasoning on complex data structures with infinite data or states makes a prominent use of induction.

Three main approaches have been developed for mechanizing inductive proofs: (*i*) explicit induction, used in proof assistants like Nqthm-ACL2 [KM96], Coq[BC04], Isabelle[NPW02] or Inka [AHMS99], (*ii*) implicit induction by rewriting used in automated theorem provers like RRL [KZ95] or Spike [BKR92] and that should *not* be confused with the third one, (*iii*) induction by consistency, as clearly emphasized in [Com01, section 1.3] where the interested reader can also find all the relevant references on that last approach. As a bridge between the two first trends, a proof search mechanism for such inductive proofs has been explored in [DKKN03,Dep02,KKN07] relying on the deduction modulo approach [DHK03]. Although already quite expressive, the latter approach is designed for theories expressed as rewrite rules and is thus limited by the fact that axioms like commutativity cannot be oriented as a rule without loosing termination of the underlying rewrite system.

*** UMR 7503 CNRS-INPL-INRIA-Nancy2-UHP

The solution consists then of using equational rewriting (also called rewriting modulo) as pioneered by [PS81] and [JK86] and to extend the proof search method developed in [DKKN03] in order to perform induction in theories containing such non orientable axioms. This extension should also be compared to implicit induction techniques used for induction modulo associativity and commutativity as done in [BBR96] and [Aot06] who generalises [Red90]. The following example is helpful to make this comparison and show how our method is essentially different from the previous ones. Assume that we want to prove

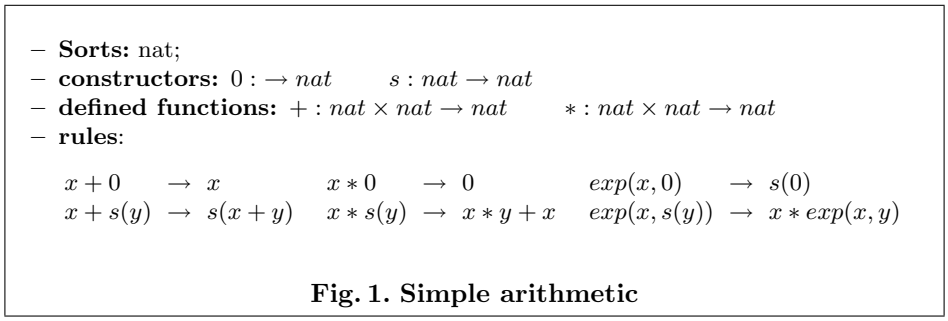

Fig. 1. Simple arithmetic

the proposition $\forall x, y, n\ exp(x * y, n) \approx exp(x, n) * exp(y, n)$, where $+$ and $*$ are also assumed to be associative and commutative (AC). The method developed in [Ber97] is based on *induction schemes*. More precisely, it computes a subset of variables of the goal, the *induction variables*, and a set of terms, the *test set*. The induction variables are replaced by elements of the test set, and such replacements produce new conjectures which are simplified by rewrite rules of the specification and smaller instances of the original conjecture (the induction hypothesis). The proof is completed when all newly generated conjectures are simplified into known or trivial inductive theorems. Algorithms are provided to compute induction variables and test sets. In the example above, the induction variables are x, y, and n, and the test set is $\{0, s(x)\}$. Therefore, a *test instance* is $exp(s(x') * s(y'), s(n')) \approx exp(s(x'), s(n')) * exp(s(y'), s(n'))$. However, this last equality can be reduced by rules of the specification into $s(x') * s(y') * exp(s(x' + y' + x' * y'), n') \approx exp(s(x'), n') * exp(s(y'), n')$, which cannot be simplified by the induction hypothesis, and the proof attempt may fail. One can avoid this difficulty if the set of induction variables is restricted. That is why [Ber97] have defined an heuristic in order to select *good* induction variables relying on observations of the Nqthm-ACL2 system. Using this strategy in the example above, only the variable n is instantiated and the proof search succeeds. However, the method does not remain refutationaly complete under such an heuristic.

In our approach, the induction step is performed by narrowing at *defined-innermost* positions, when the theory is axiomatized by a sufficiently complete and terminating equational rewrite system. More precisely, it suffices to per-

form the narrowing step at only one defined-innermost position. In the situation above, these defined innermost positions are 1.1, 2.1 and 2.2. Now, since $*$ is commutative, the goal remains equivalent by permuting the variables x and y, therefore two possibilities remain: narrowing at the defined-innermost position 1.1 where the symbol $*$ occurs, or 2.1 where the symbol exp occurs. Considering the latter better, since it further creates more reductions, we choose to narrow at the position 2.1. After normalization, we obtain the trivial subgoal $s(0) \approx s(0)$ and $x * y * exp(x * y, n) \approx x * y * exp(x, n) * exp(y, n)$ which can be reduced by the induction hypothesis.

It is important to emphasize that the latter strategy for selecting one defined-innermost position to perform the narrowing step remains refutationaly complete, whenever the specification has *good* properties: more precisely, this is the case when, given a rewrite system $\mathcal{R}$ and a set E of equalities, the rewrite relation $\mathcal{R}, E$ of Peterson and Stickel [PS81,JK86] is terminating and sufficiently complete modulo E, and when E is *constructor preserving*. Furthermore, under those conditions, narrowing at defined-innermost positions leads to consider only unifiers which are constructor substitutions. Hence, serious difficulties, related to the size of complete sets of unifiers, can be avoided. For instance, it becomes possible to perform induction modulo non finitary theories like associativity.

The paper is structured as follows. Section 1 recalls basic results about rewriting and narrowing, and introduces the concepts of *constructor preserving* theories, *defined-innermost* positions and *complete sets of constructor unifiers* that are used in the following. In Section 2, we explain how deduction modulo manages the Noetherian induction principle and we present the proof search system for inductive proofs modulo a general theory E, which is proved sound and refutationaly complete. Section 3 deals with the special case of associative-commutative theories or associative theories. The proof system of Section 2 is instantiated in these cases with more operational proof steps.

1 Basic ingredients

For the main notations and classical results on term rewriting, we refer for instance to [BN98] or [KK99].

We assume given a many sorted signature $(\mathcal{S}, \Sigma)$ (or simply Σ, for short) where $\mathcal{S}$ is a set of *sorts* and Σ is a set of function symbols, each symbol f given with a rank $f : S_1 \times \ldots \times S_n \to S$, where $S_1, \ldots, S_n, S \in \mathcal{S}$ and n is the arity of f. We assume moreover that the signature Σ comes in two parts, $\mathcal{S} = \mathcal{C} \cup \mathcal{D}$, where $\mathcal{C}$ is a set of *constructor symbols*, and $\mathcal{D}$ is a set of *defined symbols*. A constructor term is a term built only with constructor symbols. Let $\mathcal{X}$ be a family of sorted variables. The set of well-sorted terms over Σ (resp. well-sorted constructor terms) with variables in $\mathcal{X}$ will be denoted by $\mathcal{T}(\Sigma, \mathcal{X})$ (resp. $\mathcal{T}(\mathcal{C}, \mathcal{X})$). The subset of $\mathcal{T}(\Sigma, \mathcal{X})$ (resp. $\mathcal{T}(\mathcal{C}, \mathcal{X})$) of variable-free terms, or *ground* terms, is denoted $\mathcal{T}(\Sigma)$ (resp. $\mathcal{T}(\mathcal{C})$). A term $t \in \mathcal{T}(\Sigma, \mathcal{X})$ is identified as usual to a function from its set of *positions* (strings of positive integers) $\mathcal{D}om(t)$ to symbols of Σ and $\mathcal{X}$. We note ε the empty string (root position).

The *subterm* of t at position ω is denoted by $t_{|\omega}$. The result of replacing $t_{|\omega}$ with s at position ω in t is denoted by $t[s]_\omega$. This notation is also used to indicate that s is a subterm of t and, in this case, the position ω may be omitted. $\mathcal{V}ar(t)$ denotes the set of (free) variables of the term t and $|\mathcal{V}ar(t)|$ its cardinality. We define $\overrightarrow{\mathcal{V}ar(t)}$ as the vector of variables assumed linearly ordered by their name. These notations are extended to equalities $t_1 \approx t_2$ seen as terms with top symbol $\approx$ of arity 2, as well as to rewrite rules. A substitution is a finite mapping $\{x_1 \to t_1, \ldots, x_n \to t_n\}$ where $x_1, \ldots, x_n \in \mathcal{X}$ and $t_1, \ldots, t_n \in \mathcal{T}(\Sigma, \mathcal{X})$. We use postfix notation for substitutions application and composition. The domain of a substitution σ is the set $\mathcal{D}om(\sigma) = \{x \in \mathcal{X} \mid x\sigma \neq x\}$, the set of variables introduced by σ is the set $\mathcal{R}an(\sigma) = \bigcup_{x \in \mathcal{D}om(\sigma)} \mathcal{V}ar(x\sigma)$, and the image of σ is the set $Im(\sigma) = \{t \in \mathcal{T}(\Sigma, \mathcal{X}) \mid \exists x \in \mathcal{D}om(\sigma),\ t = x\sigma\}$. A substitution σ is ground whenever $Im(\sigma) \subseteq \mathcal{T}(\Sigma)$, and is constructor whenever $Im(\sigma) \subseteq \mathcal{T}(\mathcal{C}, \mathcal{X})$. Given two terms s and t, a *unifier* of s and t is a substitution σ such that $s\sigma = t\sigma$, and a *most general unifier* of s and t ($mgu(s,t)$ for short) is a unifier σ such that, for any unifier θ of s and t, there exists a substitution μ such that $\theta = \sigma\mu$ on the variables of s and t.
Given a relation $\to$ on $\mathcal{T}(\Sigma, \mathcal{X})$, $\xrightarrow{+}$ and $\xrightarrow{*}$ denote the transitive and the reflexive transitive closure of $\to$ respectively. A normal form of t, denoted $t\downarrow$, is such that $t \xrightarrow{*} t\downarrow$ and $t\downarrow$ cannot be reduced by the relation $\to$. The normalized form $\sigma\downarrow$ of a substitution σ is defined by $x(\sigma\downarrow) = (x\sigma)\downarrow$ for all $x \in \mathcal{D}om(\sigma)$. An equality is an expression of the form $e_1 \approx e_2$, where e_1 and e_2 are two terms of the same sort. Given a set E of equalities, $=_E$ denotes the congruence generated by E. We always understand equalities in a symmetric way, i.e. we make no difference between $e_1 \approx e_2$ and $e_2 \approx e_1$.
Given two terms s and t, an *E-unifier* of s and t is a substitution σ such that $s\sigma =_E t\sigma$, and a *complete set* of E-unifiers of s and t ($CSU_E(s,t)$ for short) is a set of E-unifiers of s and t satisfying: for any E-unifier θ of s and t, there exists a substitution μ such that $\theta =_E \sigma\mu[\mathcal{V}ar(s) \cup \mathcal{V}ar(t)]$, i.e. $\theta(x) =_E \sigma\mu(x)$ for all $x \in \mathcal{V}ar(s) \cup \mathcal{V}ar(t)$.

Definition 1.1. A set E of equalities is *regular* iff for any equality $e_1 \approx e_2 \in E$, $\mathcal{V}ar(e_1) = \mathcal{V}ar(e_2)$. A set E of equalities is *constructor preserving* whenever E is regular, and, for any equality $e_1 \approx e_2 \in E$, $e_1 \in \mathcal{T}(\mathcal{C}, \mathcal{X}) \Rightarrow e_2 \in \mathcal{T}(\mathcal{C}, \mathcal{X})$.

As a consequence of this definition, a set E of equalities is constructor preserving iff two terms cannot be E-equivalent whenever one of them is constructor and the other is not. Typically, if $+ \in \mathcal{D}$ and $0 \in \mathcal{C}$, $0 + x = x$ (as well as all non-constructor headed collapse axioms) is not constructor preserving (since $0 + 0 = 0$) but associativity or commutativity of $+$ are.

1.1 Equational rewriting and narrowing

We recall some basic notions introduced in [JK86]. A rewrite rule is an ordered pair of terms $l \to r$ such that $\mathcal{V}ar(r) \subseteq \mathcal{V}ar(l)$ and l is not a variable. A

conditional rewrite rule $c \Rightarrow l \rightarrow r$ moreover satisfies $\mathcal{V}ar(c) \subseteq \mathcal{V}ar(l)$. A rewrite system $\mathcal{R}$ is a set of rewrite rules. An *equational rewrite system* is given by a set of rewrite rules $\mathcal{R}$ and a set of equalities E. Let $\rightarrow_{\mathcal{R}/E}$ ($\mathcal{R}/E$ for short) be the relation $=_E \circ \underset{\mathcal{R}}{\rightarrow} \circ =_E$ which simulates the relation induced by $\mathcal{R}$ in E-equivalence classes.

Definition 1.2. An equational rewrite system $(\mathcal{R}, E)$ is *terminating modulo* E iff the relation $\mathcal{R}/E$ is Noetherian, i.e. there is no infinite sequence of the form $t_0 =_E t'_0 \underset{\mathcal{R}}{\rightarrow} t_1 \ldots t_n =_E t'_n \underset{\mathcal{R}}{\rightarrow} t_{n+1} \rightarrow \ldots$. It is *ground terminating modulo* E if it is terminating modulo E over the set of ground terms.

Given an equational rewrite system $(\mathcal{R}, E)$, the rewriting modulo E relation $\rightarrow_{\mathcal{R},E}$ ($\mathcal{R}, E$ for short) and the narrowing modulo E relation $\rightsquigarrow_{\mathcal{R},E}$ are defined as follows:

Definition 1.3. Given two terms $s,\ t \in \mathcal{T}(\Sigma, \mathcal{X})$, *$s$ rewrites modulo E to t*, denoted $s \rightarrow_{\mathcal{R},E} t$, whenever there exist a rewrite rule $l \rightarrow r \in \mathcal{R}$, a position $\omega \in \mathcal{D}om(t)$, and a substitution σ, such that $s_{|\omega} =_E l\sigma$ and $t = s[r\sigma]_\omega$. In this case, s is said $\mathcal{R}, E$-reducible. In addition, for a conditional rule $c \Rightarrow l \rightarrow r$, $c\sigma$ must evaluate to true when applying the rule. Also, *s narrows modulo E into t*, denoted $s \rightsquigarrow_{\mathcal{R},E} t$, whenever there exist a rewrite rule $l \rightarrow r \in \mathcal{R}$, a position $\omega \in \mathcal{D}om(t)$, and a substitution σ, such that $s_{|\omega}\sigma =_E l\sigma$ and $t = (s[r]_\omega)\sigma$.

Since $\rightarrow_{\mathcal{R}} \subseteq \rightarrow_{\mathcal{R},E} \subseteq \rightarrow_{\mathcal{R}/E}$, termination of $\mathcal{R}/E$ implies termination of $\rightarrow_{\mathcal{R}}$ and $\rightarrow_{\mathcal{R},E}$. Sufficient completeness is a fundamental property which states that it is always possible to rewrite any ground non-constructor term into a constructor one:

Definition 1.4. A relation $\rightarrow$ is *sufficiently complete modulo* E when, for any $s \in \mathcal{T}(\Sigma)$, there exists $t \in \mathcal{T}(\mathcal{C})$, such that $s \overset{*}{\rightarrow} t$. The equational rewrite system $(\mathcal{R}, E)$ is *sufficiently complete modulo* E if the relation $\rightarrow_{\mathcal{R},E}$ is.

For ground terminating and sufficiently complete modulo E rewrite systems, it is possible to specify particular positions in terms where reductions must apply, and where case analysis by rewriting can usefully be done.

Definition 1.5. For any $t \in \mathcal{T}(\Sigma, \mathcal{X})$, a position ω in t is called *defined-innermost*, and we denote $\omega \in DI(t)$, if $t(\omega) \in \mathcal{D}$ and $t(\omega') \in \mathcal{C} \cup \mathcal{X}$ whenever $\omega < \omega'$.

For instance, considering the Peano's integers defined in the simple arithmetic example of Fig. 1, in $s((0 + 0) + s(0 + s(x)))$, the positions 1.1 and 1.2.1 are defined-innermost but 1 is not.
The following proposition states that defined-innermost positions are ground $\mathcal{R}, E$-reducible under appropriate assumptions:

Proposition 1.1. Assume that $(\mathcal{R}, E)$ is sufficiently complete modulo E and that E is constructor preserving. Then, for any term t, for any ground $\mathcal{R}, E$-normalized substitution α, and for any $\omega \in DI(t)$, $t\alpha$ is $\mathcal{R}, E$-reducible at the position ω.

1.2 Constructor *E*-unifiers

A main difference between previous narrowing or superposition-based approaches and the one proposed in this paper, is that the unification used here to perform narrowing is quite restricted. For instance, when reasoning modulo associativity, instead considering potentialy infinite sets of unifiers, we can safely restrict to finitely many ones.

For a given set E of equalities, *constructor E-unifiers* are a key to tame the proof search system IndNarrowModE presented below. *Complete sets of constructor E-unifiers* are generating sets of constructor unifiers:

Definition 1.6. Let $s,\ t \in \mathcal{T}(\Sigma, \mathcal{X})$, a substitution σ is a constructor E-unifier of s and t if $s\sigma =_E t\sigma$ and $Im(\sigma) \subseteq \mathcal{T}(\mathcal{C}, \mathcal{X})$. Given two terms $s, t \in \mathcal{T}(\Sigma, \mathcal{X})$, $CSUC_E(s,t)$ is a *complete set of constructor E-unifiers* of s and t, if:

Correctness: every σ of $CSUC_E(s,t)$ is a constructor E-unifier of s and t;
Completeness: for any constructor E-unifier of s and t, there exist $\sigma \in CSUC_E(s,t)$ and a substitution μ, such that $\theta =_E \sigma\mu\ [\mathcal{V}ar(s) \cup \mathcal{V}ar(t)]$;
Domain: for any $\sigma \in CSUC_E(s,t)$, $\mathcal{R}an(\sigma) \cap \mathcal{D}om(\sigma) = \emptyset$.

If E is constructor preserving and satisfy syntactic conditions detailled in [Nah07], the subset of all constructor elements of $CSU_E(s,t)$ is a complete set of constructor E-unifiers of s and t. This is in particular the case when considering AC of A theories involving only defined symbols. More precisely, when E is an AC theory involving only defined symbols, if s and t are terms and ω is a defined-innermost position in s, then $CSUC_E(s_{|\omega}, t)$ is $CSUC_F(s_{|\omega}, t)$, where F denotes the subset of commutativity axioms of E. In other words, in this case AC constructor unification reduces to C constructor unification. Similarly if E is an associative theory involving only defined symbols, although $CSU_E(s_{|\omega}, t)$ may be infinite, $CSUC_E(s_{|\omega}, t)$ is $CSUC_\emptyset(s_{|\omega}, t)$ which of course simplifies considerably the induced proof space.

To conclude this section, the following proposition shows that, whenever E is constructor preserving and $(\mathcal{R}, E)$ is sufficiently complete modulo E, the narrowing step at defined-innermost positions is performed with constructor substitutions:

Proposition 1.2. Assume that $(\mathcal{R}, E)$ is sufficiently complete modulo E and that E is constructor preserving. Then, for all $t_1, \ldots, t_n \in \mathcal{T}(\mathcal{C}, \mathcal{X})$, for any $f \in \mathcal{D}$, for any ground $\mathcal{R}, E$-irreducible instantiation α of $f(t_1, \ldots, t_n)$, and for any set V such that $\mathcal{D}om(\alpha) \subseteq V$, there exist a rewrite rule $l \to r \in \mathcal{R}$, a substitution $\sigma \in CSUC_E(f(t_1, \ldots, t_n), l)$ and a substitution μ such that: $\sigma\mu =_E \alpha\ \ [V]$.

Thanks to these settings, we now present an inductive proof search system, relying on a main induction rule that uses narrowing to choose both the induction variables and the instantiation schema.

2 A proof search system for induction modulo

The proof search system IndNarrowModE for inductive proofs introduced in this section is based on (restricted) narrowing and rewriting. The main rule, called **Induce**, performs the induction step. Its intuition is the following: in order to apply the induction hypothesis, one should decrease the size of the goal by rewriting it using a noetherian rewrite system. Whenever the goal does not rewrite, it should be first instantiated to be then rewritten, i.e. it should be narrowed. By expressing this in the sequent calculus modulo, we provide an explicit and constructive bridge between the rewrite-based implicit and explicit approaches of induction.

2.1 The proof search system IndNarrowModE

Let $<$ be a Noetherian order on a set τ, i.e. such that there is no infinite sequence of elements of the form $a_0 > a_1 > \ldots > a_n > \ldots$. The *Noetherian induction principle* states that a proposition P holds for any element x of τ if P holds for all b in τ with $b < x$. Formally, if $\forall x\, x \in \tau \wedge (\forall b\, b \in \tau \wedge b < x \Rightarrow P(b)) \Rightarrow P(x)$, then P holds for all x in τ. Hence, if we write $Noeth(<, \tau)$ to state that $<$ is a Noetherian relation over τ, and $NoethInd(P, <, \tau)$ the proposition above, the Noetherian induction principle is the right-hand side of the following implication:

$$NI: \quad \forall < \forall \tau\, [Noeth(<, \tau) \Rightarrow \forall P\, (NoethInd(P, <, \tau) \Rightarrow \forall x\, P(x))]$$

To emphasize the order condition $b < x$, and since b is universally quantified, we rename b into $\underline{x}$. From now on, we instantiate τ by the set of ground terms $\mathcal{T}(\Sigma)$, P by an equality predicate $\approx$ and $<$ by the proper part of a quasi ordering $\leqslant$ defined on the set of terms $\mathcal{T}(\Sigma, \mathcal{X})$. The induction hypothesis becomes therefore $\forall \underline{x}(\underline{x} < x \Rightarrow s(\underline{x}) \approx t(\underline{x}))$, with x any variable of $s \approx t$. It is also possible to define an induction hypothesis with respect to *all* variables of $s \approx t$. Let $\overrightarrow{x} \in \mathcal{X}^n$ denote the vector of variables of $s \approx t$. In order to compare n-tuples of terms, we use the standard extension on the Cartesian product $\leqslant_n$ of $\leqslant$: $\forall \overrightarrow{u}, \overrightarrow{v} \in \mathcal{T}(\Sigma, \mathcal{X})^n \quad \overrightarrow{u} \leqslant_n \overrightarrow{v} \Leftrightarrow (\forall i\ 1 \leq i \leq n \Rightarrow u_i \leqslant v_i)$. In which case the induction hypothesis becomes:

$$\mathcal{RE}_{ind}(s \approx t, <_n, \mathcal{T}(\Sigma)^n): \quad (\overrightarrow{\underline{x}} \in \mathcal{T}(\Sigma)^n \wedge \overrightarrow{\underline{x}} <_n \overrightarrow{x}) \Rightarrow s(\overrightarrow{\underline{x}}) \approx t(\overrightarrow{\underline{x}})$$

and $\overrightarrow{x}$ is therefore the vector of free variables of $\mathcal{RE}_{ind}(s \approx t, <_n, \mathcal{T}(\Sigma)^n)$.
In order to simplify the notations, and when no confusion can occur, we denote it simply $\mathcal{RE}_{ind}(s \approx t, <)$. The following notation, where σ is any substitution, will also be used:

$$\mathcal{RE}_{ind}(s \approx t, <)\sigma: \quad (\overrightarrow{\underline{x}} \in \mathcal{T}(\Sigma)^n \wedge \overrightarrow{\underline{x}} <_n \overrightarrow{x}\sigma) \Rightarrow s(\overrightarrow{\underline{x}}) \approx t(\overrightarrow{\underline{x}})$$

There is no space here to detail how an inductive proof (that requires at least second-order logic) can be formalized in $HOL_{\lambda\sigma}$[DHK01]. This is in particular detailled in [Dep02] whose main idea relies on deduction modulo [DHK03], where

the computational and deduction parts of a proof, as well as their interactions, are identified as such. In first-order logic, for example based on the sequent calculus, a congruence on *propositions* models computation and often consists of a confluent term rewrite system, rewriting terms to terms and atomic propositions to propositions. For instance, modulo such a congruence $\sim$, the right rule for the conjunction in sequent calculus modulo is written:

$$\frac{\Gamma \vdash_\sim A, \Delta \quad \Gamma \vdash_\sim B, \Delta}{\Gamma \vdash_\sim D, \Delta} \text{ if } D \sim A \wedge B.$$

In order to provide the notational support for expressing our proof search methodology, this is further refined by writing the sequents $\Gamma_1 | \Gamma_2 \vdash_{\mathcal{RE}_1 | \mathcal{RE}_2} Q$, where Γ_1 is the deductive part of the user definitions, $\mathcal{RE}_1$ is their computational part; Γ_2 is the deductive part for other statements, $\mathcal{RE}_2$ is their computational part; Q is an equational goal. The distinction between $\Gamma_1, \mathcal{RE}_1$ and $\Gamma_2, \mathcal{RE}_2$ is needed because only $\mathcal{RE}_1$ will be used for narrowing. For simplicity, we assume that $\mathcal{RE}_1$ contains only unconditional rules or equalities, and we assume from now on, that Γ_1 contains a constructor preserving theory E, such that $(\mathcal{RE}_1, E)$ is terminating and sufficiently complete modulo E. Γ_2 is initialized with the proposition NI defined above, with the theory of equality $Th_\approx$ satisfied by the binary relation $\approx$, and, if the goal and the rules in $\mathcal{RE}_2$ contain n free variables, with the proposition $Noeth(<_n, \mathcal{T}(\Sigma)^n)$, and may contain other lemmas. $\mathcal{RE}_2$ will receive the induction hypotheses provided by application of the proof search rules, so $\mathcal{RE}_2$ may contain conditional equalities.

Example 2.1. Assume that $\mathcal{RE}_1$ contains the rules of simple arithmetic given in Figure 1. $\mathcal{RE}_1$ is terminating and sufficiently complete modulo associativity and commutativity of the $*$ and $+$ operators (denoted $AC(+,*)$). Let $\Gamma_1 = AC(+,*)$, $\Gamma_2 = Th_\approx \cup \{NI,\ Noeth(<_4, \mathcal{T}(\Sigma)^4)\}$, and $Q = (x_1 + x_2 + x_3) * x_4 \approx x_1 * x_4 + x_2 * x_4 + x_3 * x_4$. Then, we can consider the goal $\Gamma_1 | \Gamma_2 \vdash_{\mathcal{RE}_1 | \emptyset} Q$.

The proof search rules are presented in Figure 2.

Sequents are gathered in a multiset structure modeled with the $\bullet$ operator that is an AC operator on sequents with $\diamond$ as neutral element.

The rule **Induce** performs the induction step. It uses narrowing to choose both the induction variable(s) and the instantiation schema. Narrowing is applied only at defined innermost positions $DI(Q')$ of a goal Q' E-equivalent to the current goal Q. Indeed Q' may be Q itself, and this will be the case for the derived inference systems where E is A or AC.

The other rules are doing the following: **Trivial** eliminates a trivial equation, **Rewrite** (1 or 2) rewrites using a rule, an equation, or a smaller instance of a previous goal. **Rewrite** is duplicated because of the $\Gamma_1, \mathcal{RE}_1$ and $\Gamma_2, \mathcal{RE}_2$ distinction.

This inference rule set is generic and prepares to more operational versions tailored for AC and A-theories.

Induce $\Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} Q \rightarrowtail$

$$\bullet_{\substack{l \to r \in \mathcal{RE}_1 \\ \sigma' \in CSUC_E(Q'_{|\omega'}, l)}} \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2\sigma' \cup \{\mathcal{RE}_{ind}(Q,<)\sigma'\}} (Q'[r]_{\omega'})\sigma'$$

if $Q' =_E Q$ and $\omega' \in DI(Q')$

Rewrite$_1$ $\Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} Q \rightarrowtail \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} Q'$

if $Q \rightarrow_{\mathcal{RE}_1/E} Q'$

Rewrite$_2$ $\Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} Q \rightarrowtail \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} Q'$

if $Q \rightarrow_{\mathcal{RE}_2/E} Q'$

Trivial $\Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} t \approx t' \rightarrowtail \diamond$

if $t =_E t'$

Refutation $\Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} Q \rightarrowtail$ Refutation

when no other rules can be applied

Fig. 2. The proof search system IndNarrowModE

2.2 Properties of IndNarrowModE

From now on, let us assume that $(\mathcal{R}, E)$ is terminating and sufficiently complete modulo E, and that E is constructor preserving.

Soundness: Proving soundness amounts showing that for each rule of the proof search system IndNarrowModE of the form $S \rightarrowtail S'$, if S' is derivable in the sequent calculus modulo, then one can also build a proof of S. The main delicate point is to prove this result for the **Induce** rule, as stated in the next theorem.

Theorem 2.1. *If the sequent* $\Gamma_1|\Gamma_2, \overrightarrow{x_{\sigma'}} \in \mathcal{T}(\Sigma)^{n_{\sigma'}} \vdash_{\mathcal{RE}_1|\mathcal{RE}_2\sigma' \cup \{\mathcal{RE}_{ind}(Q)\sigma'\}} (Q'[r]_{\omega'})\sigma'$ *is derivable in the sequent calculus modulo, where:*

1. $Q =_E Q'$ *and* $\omega' \in DI(Q')$*;*
2. $l \to r \in \mathcal{RE}_1$ *and* $\sigma' \in CSUC_E(Q'_{|\omega'}, l)$*;*
3. $\mathcal{RE}_2\sigma'$ *is the rewrite system obtained by the replacement of each free variable* x *of any rewrite rule in* $\mathcal{RE}_2$ *by a corresponding* $x\sigma'$*;*
4. $\overrightarrow{x_{\sigma'}} \in \mathcal{X}^{n_{\sigma'}}$ *is the vector of free variables of* $\mathcal{RE}_2\sigma' \cup \{Q\sigma'\}$*;*

then, one can build a proof in the sequent calculus modulo of $\Gamma, \overrightarrow{x} \in \mathcal{T}(\Sigma)^n \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} Q$ *where* $\overrightarrow{x} \in \mathcal{X}^n$ *denotes the vector of free variables of* $\mathcal{RE}_2 \cup \{Q\}$*.*

Refutational correctness: Proving refutational correctness amounts showing that for each rule of the proof search system IndNarrowModE of the form $S \rightarrowtail S'$, if S is derivable in the sequent calculus modulo, then one can also build a proof

of S'. Again the main delicate point is for the **Induce** rule, and is stated as follows.

Theorem 2.2. *If the sequent* $\Gamma_1|\Gamma_2, \overrightarrow{x} \in \mathcal{T}(\Sigma)^n \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} Q$ *where* $\overrightarrow{x} \in \mathcal{X}^n$ *is the vector of free variables of* $\mathcal{RE}_2 \cup \{Q\}$*, admits a proof in the sequent calculus modulo, then one can build a proof of:*

$$\Gamma_1|\Gamma_2, \overrightarrow{x_{\sigma'}} \in \mathcal{T}(\Sigma)^{n_{\sigma'}} \vdash_{\mathcal{RE}_1|\mathcal{RE}_2\sigma' \cup \{\mathcal{RE}_{ind}(Q,<)\sigma'\}} (Q'[r]_{\omega'})\sigma'$$

where $Q' =_E Q$, $l \to r \in \mathcal{RE}_1$, $\omega' \in DI(Q')$, $\sigma' \in CSUC_E(Q'_{|\omega'}, l)$, *and* $\overrightarrow{x_{\sigma'}} \in \mathcal{X}^{n_{\sigma'}}$ *is the vector of free variables of* $\mathcal{RE}_2\sigma' \cup \{Q\sigma'\}$.

Refutational completeness: Proving refutational completeness is achieved thanks to the **Refutation** rule which applies when no other rule of IndNarrow can be applied.

Theorem 2.3. *If* $\Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} Q \overset{*}{\rightarrowtail}$ *Refutation then the sequent* $\Gamma_1|\Gamma_2, \overrightarrow{x} \in \mathcal{T}(\Sigma)^n \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} Q$ *has no proof in the sequent calculus modulo.*

3 Induction modulo *AC* and *A*

The general IndNarrowModE proof search system is indeed working directly on equivalence classes modulo E, a situation not directly implementable for most theories E. To focus on more operational proof search systems where instead of working with $\to_{R/E}$, we use the operational rewrite relation $\to_{R,E}$, we focus in this section on the case of associative-commutative or associative theories. We introduce two proof search systems IndNarrowModAC and IndNarrowModA as special instances of IndNarrowModE with specific improvements and illustrating examples. Soundness and refutational correctness and completeness of these systems will be consequences of the properties of IndNarrowModE.

3.1 More about flattened terms

In associative and associative-commutative theories, equivalence classes of terms are often represented by flattened terms. We refer for the basic definitions and results about positions and subterms to [Mar93]. Intuitively flattening a term amounts to recursively replace $f(f(s,t),u)$ or $f(s,f(t,u))$ by $f(s,t,u)$ if f is an associative symbol. This is the key point bridging the proof search systems IndNarrowModAC and IndNarrowModA on the one hand, and IndNarrowModE on the other hand.

From now on, we assume that some function symbols in a subset $\mathcal{V}$ of Σ may have an *unbounded arity*. Let $A(\mathcal{V}) = \{f(fxy)z \approx fx(fyz) \mid f \in \mathcal{V}\}$ and $AC(\mathcal{V}) = A(\mathcal{V}) \cup \{fxy \approx fyx \mid f \in \mathcal{V}\}$. In the following, we assume that all symbols in $\mathcal{V}$ are defined symbols, that constructor symbols do *not* have unbounded arities, and that $+$ and $*$ denote symbols with unbounded arity.

[Mar93] defines a transformation which associates to each position in a given term t a position in the flattening $\overline{t}$ of t, also called the *flattening* of this position. However, a position in $\overline{t}$ is not always the flattening of some position in t, and this led us to introduce the following definition:

Definition 3.1. For a flattened term $\overline{s}$, a position $\omega \in \mathcal{D}om(\overline{s})$ is *flattened* if there exist $i, k \in \mathbb{N}$, and a word ω_0, s.t. $\omega = \omega_0.i$ or $\omega = \omega_0.\{i, i+1, \ldots, i+k\}$.

The above flattened positions are precisely the flattening of positions [Nah07]. To define a rewrite relation on the set of flattened terms, the notion of replacement has to be generalized:

Definition 3.2. Given two flattened terms $\overline{s} = f\overline{s_1} \ldots \overline{s_n}$, $\overline{t}$, and a position $\omega \in \overline{s}$, the replacement by $\overline{t}$ in $\overline{s}$ at the position ω is inductively defined by:

- $\overline{s}[\overline{t}]_\varepsilon = \overline{t}$
- If $\omega \in \{1, \ldots, n\}$
  - Case 1: there exist $i,\ k \in \mathbb{N}$, such that $\omega = \{i, i+1, \ldots, i+k\}$. $\overline{s}[\overline{t}]_\omega = \overline{f\overline{s_1} \ldots \overline{s_{i-1}}\ \overline{t}\ \overline{s_{i+k+1}} \ldots \overline{s_n}}$
  - Case 2: otherwise, let $\{i_1, \ldots, i_k\} = \{1, \ldots, n\} - \omega$. $\overline{s}[\overline{t}]_\omega = \overline{f\overline{s_{i_1}} \ldots \overline{s_{i_k}}\ \overline{t}}$.
- $\overline{s}[\overline{t}]_{i.\omega_i} = \overline{f\overline{s_1} \ldots \overline{s_i}[\omega_i \leftarrow \overline{t}] \ldots \overline{s_n}}$.

Now, we introduce a rewrite relation on the set of flattened terms as follows:

Definition 3.3. Given a rewrite system $\mathcal{R}$, we define the relation $\rightarrow_{\overline{\mathcal{R}}}$ on the set of flattened terms by $\overline{s} \rightarrow_{\overline{\mathcal{R}}} \overline{t}$ whenever there exist a rule $c \Rightarrow l \rightarrow r \in \mathcal{R}$, a flattened position $\omega \in \mathcal{D}om(\overline{s})$ and a substitution σ such that:

- $\overline{s}_{|\omega} = \overline{l\sigma}$, $\overline{t} = \overline{s}[\overline{r\sigma}]_\omega$
- and the condition $c\sigma$ is true.

If $\equiv_p$ denotes the classical equivalence induced on the set of flattened terms by permutation of the arguments of symbols in $\mathcal{V}$, we consider the extension $\overline{\mathcal{R}}/\equiv_p$ of $\overline{\mathcal{R}}$ on the set of $\equiv_p$-equivalences. As previously, in order to perform induction by narrowing at *defined-innermost* positions, we must define such positions for flattened terms:

Definition 3.4. For any $s \in \mathcal{T}(\Sigma, \mathcal{X})$, and for any $\omega \in \mathcal{D}om(\overline{s})$, the position ω is called *defined-innermost* whenever there exist $f \in \Sigma$ and terms $s_1, \ldots,\ s_n \in \mathcal{T}(\mathcal{C}, \mathcal{X})$, such that $\overline{s}_{|\omega} = f\overline{s_1} \ldots \overline{s_n}$, and moreover $n = 2$ if $f \in \mathcal{V}$.

Intuitively, the position ω in $\overline{s}$ is defined-innermost when $\overline{s}_{|\omega}$ coincides with its flattened form.

InduceAC $\quad \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} \overline{Q} \rightarrowtail$

$\bullet_{\substack{l \to r \in \mathcal{RE}_1 \\ \sigma \in CSUC_{AC}(\overline{Q}_{|\omega}, l)}} \quad \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2\sigma \cup \{\mathcal{RE}_{ind}(Q,<)\sigma\}} (\overline{Q}[\overline{r}]_\omega)\sigma$

if $\omega \in DI(\overline{Q})$

Rewrite$_1$AC $\quad \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} \overline{Q} \rightarrowtail \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} \overline{Q'}$

if $\overline{Q} \to_{\overline{\mathcal{RE}_1}/\equiv_p} \overline{Q'}$

Rewrite$_2$AC $\quad \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} \overline{Q} \rightarrowtail \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} \overline{Q'}$

if $\overline{Q} \to_{\overline{\mathcal{RE}_2}/\equiv_p} \overline{Q'}$

TrivialAC $\quad \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} \overline{t} \approx \overline{t'} \rightarrowtail \diamond$

if $\overline{t} \equiv_p \overline{t'}$

RefutationAC $\quad \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} \overline{Q} \rightarrowtail$ Refutation

when no other rules can be applied

Fig. 3. The proof search system IndNarrowModAC

InduceA $\quad \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} \overline{Q} \rightarrowtail$

$\bullet_{\substack{l \to r \in \mathcal{RE}_1 \\ \sigma \in CSUC_{A}(\overline{Q}_{|\omega}, l)}} \quad \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2\sigma \cup \{\mathcal{RE}_{ind}(Q,<)\sigma\}} (\overline{Q}[\overline{r}]_\omega)\sigma$

if $\omega \in DI(\overline{Q})$ and ω flattened.

Rewrite$_1$A $\quad \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} \overline{Q} \rightarrowtail \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} \overline{Q'}$

if $\overline{Q} \to_{\overline{\mathcal{RE}_1}} \overline{Q'}$

Rewrite$_2$A $\quad \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} \overline{Q} \rightarrowtail \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} \overline{Q'}$

if $\overline{Q} \to_{\overline{\mathcal{RE}_2}} \overline{Q'}$

TrivialA $\quad \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} \overline{t} \approx \overline{t'} \rightarrowtail \diamond$

if $\overline{t} \neq \overline{t'}$

RefutationA $\quad \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} \overline{Q} \rightarrowtail$ Refutation

when no other rules can be applied

Fig. 4. The proof search system IndNarrowModA

3.2 The proof search systems IndNarrowModAC and IndNarrowModA

The specific proof search systems IndNarrowModAC and IndNarrowModA are respectively given in Figure 3 and Figure 4.

Soundness, refutational correctness and completeness of IndNarrowModAC and IndNarrowModA are consequences of the following proposition that states a correspondence between a deduction on a goal Q using IndNarrowModE and a deduction on the corresponding flattened goal using IndNarrowModAC or IndNarrowModA.

Theorem 3.1. *Let $E = AC(\mathcal{V})$ (resp.$E = A(\mathcal{V})$).*

1. *If $\Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} Q \rightarrowtail_{\mathsf{IndNarrowModE}} \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}'_2} R$, then $\Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} \overline{Q} \rightarrowtail_{\mathsf{IndNarrowModAC}} \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}'_2} \overline{R}$. (resp. $\Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} \overline{Q} \rightarrowtail_{\mathsf{IndNarrowModA}} \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}'_2} \overline{R}$).*

2. *If $\Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} \overline{Q} \rightarrowtail_{\mathsf{IndNarrowModAC}} \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}'_2} \overline{R}$ (resp. $\Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} \overline{Q} \rightarrowtail_{\mathsf{IndNarrowModA}} \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}'_2} \overline{R}$), there exists R' such that $R' =_{AC} R$ (resp. $R' =_A R$), and $\Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}_2} Q \rightarrowtail_{\mathsf{IndNarrowModE}} \Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\mathcal{RE}'_2} R'$*

3.3 Two simple examples

In order to get a better intuition on the way these sets of rules are working, let us look at two examples. In the following, we always refer to the specification and the set of rewrite rules given in Figure 1. The first example of proof uses AC properties of $+$ and $*$ induction modulo AC, and the second one uses the same rules but just associativity of these two symbols.

An $AC-$example: In the context of Example 2.1, let us consider the following sequent: $\Gamma_1|\Gamma_2 \vdash_{\mathcal{RE}_1|\emptyset} Q$ and first apply the rule **InduceAC**. The innermost positions in Q are $1.1.\{1,2\}$, $1.1.\{1,3\}$, $1.1.\{2,3\}$, 2.1, 2.2 and 2.3. Since the goal remains equivalent by permutation of the variables x_1, x_2 and x_3, only two possibilities remain: narrowing at a position where the symbol $+$ occurs, or where the symbol $*$ occurs. Since the last choice creates more reductions than the first one, we arbitrarily choose to narrow at the position 2.1 of the goal. Therefore, we must compute the set $CSUC_{AC}(x_1 * x_4, l)$ for any rewrite rule $l \to r$ of $\mathcal{RE}_1$. This restricts to rules such that $l(\varepsilon) = *$, and we obtain:

l	$CSUC_{AC}(x_1 * x_4, l)$
$x * 0$	$\sigma_1 = \{x_1 \to y_1; x \to y_1; x_4 \to 0\}$ $\sigma_2 = \{x_1 \to 0; x \to y_4; x_4 \to y_4\}$
$x * s(y)$	$\sigma_3 = \{x_1 \to y_1; x \to y_1; y \to y_4; x_4 \to s(y_4)\}$ $\sigma_4 = \{x_1 \to s(y_1); x \to y_4; y \to y_1; x_4 \to y_4\}$

After normalization, this leads us to prove the four sequents:

$\Gamma_1\|\Gamma_2 \vdash_{\mathcal{RE}_1\|\mathcal{RE}_{ind}(Q)\sigma_1} 0 \approx 0$
$\Gamma_1\|\Gamma_2 \vdash_{\mathcal{RE}_1\|\mathcal{RE}_{ind}(Q)\sigma_2} (x_2 + x_3) * y_4 \approx x_2 * y_4 + x_3 * y_4$
$\Gamma_1\|\Gamma_2 \vdash_{\mathcal{RE}_1\|\mathcal{RE}_{ind}(Q)\sigma_3} (y_1 + x_2 + x_3) * y_4 + y_1 + x_2 + x_3 \approx y_1 * y_4 + y_1 + x_2 * y_4 + x_2 + x_3 * y_4 + x_3$
$\Gamma_1\|\Gamma_2 \vdash_{\mathcal{RE}_1\|\mathcal{RE}_{ind}(Q)\sigma_4} (y_1 + x_2 + x_3) * y_4 + y_4 \approx y_1 * y_4 + y_4 + x_2 * y_4 + x_3 * y_4$

Trivial gets rid of the first one. Since $(y_1, x_2, x_3, y_4) <_4 (y_1, x_2, x_3, s(y_4))$, **Rewrite**$_2$ can be applied on the third one, and since $(y_1, x_2, x_3, y_4) <_4 (s(y_1), x_2, x_3, y_4)$, **Rewrite**$_2$ can be applied on the fourth one. Hence we get:

$\Gamma_1 \vert \Gamma_2 \vdash_{\mathcal{RE}_1 \vert \mathcal{RE}_{ind}(Q)\sigma_2} (x_2 + x_3) * y_4 \approx x_2 * y_4 + x_3 * y_4$
$\Gamma_1 \vert \Gamma_2 \vdash_{\mathcal{RE}_1 \vert \mathcal{RE}_{ind}(Q)\sigma_3} y_1 * y_4 + x_2 * y_4 + x_3 * y_4 + y_1 + x_2 + x_3 \approx y_1 * y_4 + y_1 + x_2 * y_4 + x_2 + x_3 * y_4 + x_3$
$\Gamma_1 \vert \Gamma_2 \vdash_{\mathcal{RE}_1 \vert \mathcal{RE}_{ind}(Q)\sigma_4} y_1 * y_4 + x_2 * y_4 + x_3 * y_4 + y_4 \approx y_1 * y_4 + y_4 + x_2 * y_4 + x_3 * y_4$

Trivial gets rid of the two last subgoals. The application of **Induce** to the first one at position 2.1 generates four subgoals. **Trivial** gets rid of the two first ones, the application of **Rewrite**$_2$ to the last ones creates two new subgoals which are trivial and we are done.

An A-example: Assume that $\mathcal{RE}_1$ contains the rules of simple arithmetic given in Figure 1. $\mathcal{RE}_1$ is terminating and sufficiently complete modulo associativity of the $*$ and $+$ operators (denoted $A(+, *)$) Let us prove that distributivity of $*$ over $+$ is an inductive theorem.
Let $\Gamma_1 = A(+, *)$, $\Gamma_2 = Th_{\approx} \cup \{NI,\ Noeth(<_3, \mathcal{T}(\Sigma)^3)\}$, and $Q = x_1 * (x_2 + x_3) \approx x_1 * x_2 + x_1 * x_3$. Let us start from the sequent: $\Gamma_1 \vert \Gamma_2 \vdash_{\mathcal{RE}_1 \vert \emptyset} Q$.
We can apply **InduceA** at the innermost positions 1.2, 2.1 and 2.2 in Q and Theorem 2.1 ensures that each of these choices is correct. Since narrowing at position 2.1 creates less further reductions than the ones at positions 1.2 or 2.2, we choose to narrow at the position 2.2 of the goal. Thus, we need to compute $CSUC_A(x_1 * x_3, l)$ for any rewrite rule $l \to r$ of $\mathcal{RE}_1$. This restricts to rules such that $l(\varepsilon) = *$, and we obtain:

l	$CSUC_A(x_1 * x_3, l)$
$x * 0$	$\sigma_1 = \{x_1 \to y_1; x \to y_1; x_3 \to 0\}$
$x * s(y)$	$\sigma_2 = \{x_1 \to y_1; x \to y_1; y \to y_3; x_3 \to s(y_3)\}$

After normalization, we obtain the subgoals:

$\Gamma_1 \vert \Gamma_2 \vdash_{\mathcal{RE}_1 \vert \mathcal{RE}_{ind}(Q)\sigma_1} y_1 * x_2 \approx y_1 * x_2$
$\Gamma_1 \vert \Gamma_2 \vdash_{\mathcal{RE}_1 \vert \mathcal{RE}_{ind}(Q)\sigma_2} y_1 * (x_2 + y_3) + y_1 \approx y_1 * x_2 + y_1 * y_3 + y_1$

Trivial gets rid of the first subgoal. Since $(y_1, x_2, y_3) <_3 (y_1, x_2, s(y_3))$, **Rewrite**$_2$**A** can be applied on the second one. Hence, we get:

$\Gamma_1 \vert \Gamma_2 \vdash_{\mathcal{RE}_1 \vert \mathcal{RE}_{ind}(Q)\sigma_2} y_1 * x_2 + y_1 * y_3 + y_1 \approx y_1 * x_2 + y_1 * y_3 + y_1$

and **Trivial** gets rid of this last subgoal.

4 Conclusion

We have extended the inductive proof search method based on narrowing to the case where theories contain non-orientable axioms. The main inference rule is based on a restricted application of narrowing at defined-innermost positions and with a restricted notion of equational unifiers based only on constructors. This general approach is proved correct and refutationaly complete. We then applied it to the specific case of rewriting modulo AC or A axioms and show on two examples how the method safely restricts the proof search space. This provides a significant improvement on the current inductive proof search approaches.

An interesting side result of our approach is the introduction of a new kind of E-unifiers that we called constructor E-unifiers. In the case of associative and associative commutative theories E, they have the nice property to considerably reduce the number of unifiers to be considered in a complete set of unifiers that may be huge or even infinite in these theories. A natural and challenging question is to build a unification theory for these specific unifiers.

First motivated by the wish to provide a bridge between explicit and implicit induction, our approach achieves this goal through a specific instance of the sequent calculus modulo [DHK01] that clarifies the respective roles and uses of the noetherian induction principle and of equational rewriting. As a consequence, we plan to have an automated construction of such proofs into the sequent calculus for insertion into proof assistants like lemuridæ which is based on superdeduction [BHK07]. Such proof assistants will therefore rely on an implementation of our inference systems, a task that still remains to be done.

References

AHMS99. S. Autexier, D. Hutter, H. Mantel, and A. Schairer. System description: Inka 5.0 – a logic voyager. In H. Ganzinger, editor, *Proceedings of the 16th International Conference on Automated Deduction (CADE-16)*, volume 1632 of *LNAI*, pages 207–211, Trento, Italy, july 1999. Springer. 1

Aot06. T. Aoto. Dealing with Non-orientable Equations in Rewriting Induction. In F. Pfenning, editor, *Proceedings of the 17th International Conference on Rewriting Techniques and Applications*, volume 4098 of *Lecture Notes in Computer Science*, pages 242–256, Nara (Japan), apr 2006. Springer. 2

BBR96. N. Berregeb, A. Bouhoula, and M. Rusinowitch. Automated verification by induction with associative-commutative operators. In R. Alur and T. A. Henzinger, editors, *CAV*, volume 1102 of *Lecture Notes in Computer Science*, pages 220–231. Springer, 1996. 2

BC04. Y. Bertot and P. Casteran. *Interactive Theorem Proving and Program Development*. Springer-Verlag, 2004. 1

Ber97. N. Berregeb. *Preuves par induction implicite : cas des théories associatives-commutatives et observationnelles*. Thèse de Doctorat d'Université, Université Henri Poincaré – Nancy 1, June 1997. 2

BHK07. P. Brauner, C. Houtmann, and C. Kirchner. Principle of superdeduction. In L. Ong, editor, *Proceedings of LICS*, pages 41–50, jul 2007. 15

BKR92. A. Bouhoula, E. Kounalis, and M. Rusinowitch. Spike: An automatic theorem prover. In *Proceedings of the 1st International Conference on Logic Programming and Automated Reasoning, St. Petersburg (Russia)*, volume 624 of *Lecture Notes in Artificial Intelligence*, pages 460–462. Springer-Verlag, July 1992. 1

BN98. F. Baader and T. Nipkow. *Term Rewriting and all That*. Cambridge University Press, 1998. 3

Com01. H. Comon. Inductionless induction. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume I, chapter 14, pages 914–959. Elsevier Science, 2001. 1

Dep02. E. Deplagne. *Système de preuve modulo récurrence*. Thèse de doctorat, Université Nancy 1, November 2002. 1, 7

DHK01. G. Dowek, T. Hardin, and C. Kirchner. HOL-$\lambda\sigma$ an intentional first-order expression of higher-order logic. *Mathematical Structures in Computer Science*, 11(1):21–45, 2001. 7, 15

DHK03. G. Dowek, T. Hardin, and C. Kirchner. Theorem proving modulo. *Journal of Automated Reasoning*, 31(1):33–72, Nov 2003. 1, 7

DKKN03. E. Deplagne, C. Kirchner, H. Kirchner, and Q.-H. Nguyen. Proof search and proof check for equational and inductive theorems. In F. Baader, editor, *Proceedings of CADE-19*, Miami, Florida, July 2003. Springer-Verlag. 1, 2

JK86. J.-P. Jouannaud and H. Kirchner. Completion of a set of rules modulo a set of equations. *SIAM Journal of Computing*, 15(4):1155–1194, 1986. 2, 3, 4

KK99. C. Kirchner and H. Kirchner. Rewriting, solving, proving. A preliminary version of a book available at `www.loria.fr/~ckirchne/rsp.ps.gz`, 1999. 3

KKN07. C. Kirchner, H. Kirchner, and F. Nahon. Narrowing based inductive proof search: Definition and optimisations. Research report, LORIA, Mars 2007. 1

KM96. M. Kaufmann and J. S. Moore. ACL2: An industrial strength version of nqthm. In *Compass'96: Eleventh Annual Conference on Computer Assurance*, page 23, Gaithersburg, Maryland, 1996. National Institute of Standards and Technology. 1

KZ95. D. Kapur and H. Zhang. An overview of rewrite rule laboratory (RRL). *J. Computer and Mathematics with Applications*, 29(2):91–114, 1995. 1

Mar93. C. Marché. *Réécriture modulo une théorie présentée par un système convergent et décidabilité du problème du mot dans certaines classes de théories équationnelles*. Thèse de Doctorat d'Université, Université de Paris-Sud, Orsay (France), October 1993. 10, 11

Nah07. F. Nahon. *Preuves par induction dans le calcul des séquents modulo*. PhD thesis, UHP, to appear, October 2007. 6, 11

NPW02. T. Nipkow, L. C. Paulson, and M. Wenzel. *Isabelle/HOL — A Proof Assistant for Higher-Order Logic*, volume 2283 of *LNCS*. Springer-Verlag, 2002. 1

PS81. G. Peterson and M. E. Stickel. Complete sets of reductions for some equational theories. *Journal of the ACM*, 28:233–264, 1981. 2, 3

Red90. U. Reddy. Term rewriting induction. In M. E. Stickel, editor, *Proceedings 10th International Conference on Automated Deduction, Kaiserslautern (Germany)*, volume 449 of *Lecture Notes in Computer Science*, pages 162–177. Springer-Verlag, 1990. 2