Modular Instantiation Schemes

Mnacho Echenim, Nicolas Peltier

Modular instantiation schemes

Mnacho Echenim[1,*], Nicolas Peltier[1,*]

University of Grenoble & CNRS, Grenoble

Abstract

Instantiation schemes are proof procedures that test the satisfiability of clause sets by instantiating the variables they contain, and testing the satisfiability of the resulting ground set of clauses. Such schemes have been devised for several theories, including fragments of linear arithmetic or theories of data-structures. In this paper we investigate under what conditions instantiation schemes can be combined to solve satisfiability problems in unions of theories.

Keywords: SMT, Instantiation-based methods, combination problems

1. Introduction

Instantiation schemes are employed to prove the (un)satisfiability of a set of clauses. They are based on Herbrand's theorem, which states that any unsatisfiable set of clauses admits a finite set of ground instances that is also unsatisfiable. Any instantiation scheme can be viewed as a function Θ that maps a set of clauses S to a set of ground instances $\Theta(S)$ of clauses in S; such a scheme is *refutationally complete* if for all sets of clauses S, $\Theta(S)$ is satisfiable exactly when S is. Examples of refutationally complete instantiation schemes include [14, 9, 3, 11]. Some incomplete – but efficient and practically successful – schemes have also been defined, see e.g. [13, 10]. It is clear that an instantiation scheme that is refutationally complete does not always terminate, as $\Theta(S)$ may be infinite; however, terminating and refutationally complete instantiation schemes have been defined for specific theories or classes of theories (see e.g. [1, 4, 7, 6]). Such decision procedures can be very attractive, provided they do not generate too many ground clauses, since it is much easier to test the satisfiability of a ground set of clauses than that of a non ground set (see for instance [2, 5] for surveys of existing approaches in Satisfiability Modulo Theories). Frequently, one has to handle heterogeneous problems, defined on (possibly non disjoint) unions of theories. Thus the idea of combining existing instantiation schemes naturally arises. Most systems rely on the Nelson-Oppen or Shostak

*Corresponding author

[1]Emails: Mnacho.Echenim@imag.fr and Nicolas.Peltier@imag.fr

methods (with many refinments and extensions) to combine decision procedures (see e.g. [15]). In this paper, we use a different and more upstream method by showing how to combine directly the instantiation schemes to produce a unique set of ground instances. Thanks to these results, combining instantiation-based systems now becomes a straightforward task – provided they fulfill the conditions of Section 4 – since the interactions between the theories are directly handled by the decision procedure for ground clauses.

2. Preliminaries

We briefly review usual notions and notations (missing definitions can be found in [8]). Let $\mathfrak{S}$ be a set of *sorts symbols* and let $\mathfrak{F}$ be a set of *function symbols* together with a *ranking function* $\mathrm{rnk} : \mathcal{F} \to \mathfrak{S}^* \times \mathfrak{S}$. For every $f \in \mathcal{F}$, we denote $\mathrm{rnk}(f)$ by $\mathtt{s}_1 \times \cdots \times \mathtt{s}_n \to \mathtt{s}$. If $n = 0$ then f is a *constant symbol*. To every sort $\mathtt{s} \in \mathfrak{S}$ is associated a countably infinite set $\mathcal{X}_\mathtt{s}$ of *variables of sort* $\mathtt{s}$, such that these sets are pairwise disjoint. Let $\mathcal{X} = \bigcup_{s \in \mathfrak{S}} \mathcal{X}_\mathtt{s}$.

For every $\mathtt{s} \in \mathfrak{S}$ and $\mathcal{F} \subseteq \mathfrak{F}$, the *set of $\mathcal{F}$-terms of sort* $\mathtt{s}$ is denoted by $\mathrm{T}_\mathtt{s}(\mathcal{F})$ and built inductively as usual on $\mathcal{X}$ and $\mathcal{F}$. The *set of $\mathcal{F}$-terms* is defined by $\mathrm{T}(\mathcal{F}) = \bigcup_{\mathtt{s} \in \mathfrak{S}} \mathrm{T}_\mathtt{s}(\mathcal{F})$. When no confusion is possible, we may simply mention terms, instead of $\mathcal{F}$-terms. An *$\mathcal{F}$-atom* is an equality $t \simeq s$ between $\mathcal{F}$-terms of the same sort; an *$\mathcal{F}$-literal* is either an $\mathcal{F}$-atom or the negation of an $\mathcal{F}$-atom (written $t \not\simeq s$) and an *$\mathcal{F}$-clause* is a disjunction (or multi-set) of $\mathcal{F}$-literals. The set of $\mathcal{F}$-clauses is denoted by $\mathrm{C}(\mathcal{F})$. The set of variables occurring in an $\mathcal{F}$-expression (term, atom, literal or clause) $\mathcal{E}$ is denoted by $\mathrm{Var}(\mathcal{E})$; $\mathcal{E}$ is *ground* if $\mathrm{Var}(\mathcal{E}) = \emptyset$. An *$\mathcal{F}$-substitution* is a function that maps every variable to an $\mathcal{F}$-term of the same sort. The image of a variable x by a substitution σ is denoted by $x\sigma$. The *domain* of a substitution σ is the set $dom(\sigma) = \{x \in \mathcal{X} \mid x\sigma \neq x\}$, which is assumed to be finite. A substitution is extended to terms, atoms, literals and clauses in the standard way. An $\mathcal{F}$-substitution σ is *ground* if $\forall x \in dom(\sigma)$, $\mathrm{Var}(x\sigma) = \emptyset$. A *ground $\mathcal{F}$-instance* of an $\mathcal{F}$-expression $\mathcal{E}$ is an $\mathcal{F}$-expression of the form $\mathcal{E}\sigma$, where σ is a ground $\mathcal{F}$-substitution of domain $\mathrm{Var}(\mathcal{E})$.

We denote by $\mathfrak{S}_\mathcal{F}$ the set of sort symbols occurring in the rank of a symbol $f \in \mathcal{F}$. An *$\mathcal{F}$-interpretation* is a function I that maps every sort $\mathtt{s} \in \mathfrak{S}_\mathcal{F}$ to a nonempty *domain* $I(\mathtt{s})$ and every function $f \in \mathcal{F}$ of rank $\mathtt{s}_1 \times \cdots \times \mathtt{s}_n \to \mathtt{s}$ to a function $f^I : I(\mathtt{s}_1) \times \cdots \times I(\mathtt{s}_n) \to I(\mathtt{s})$. We assume that the domains of the sorts are pairwise disjoint. A *valuation* for an interpretation I and an expression $\mathcal{E}$ is a function mapping every variable of sort $\mathtt{s}$ occurring in $\mathcal{E}$ to an element of $I(\mathtt{s})$. If I is an interpretation, $\mathcal{E}$ is an expression and σ is a valuation for $\mathcal{E}$ and $I(\mathtt{s})$, then $[\mathcal{E}]^\sigma_I$ denotes the value of $\mathcal{E}$ in I and σ (defined inductively as usual). An $\mathcal{F}$-interpretation I *satisfies* an $\mathcal{F}$-clause C if for every valuation ν for I and C we have $[C]^\nu_I = \mathtt{true}$. A set of $\mathcal{F}$-clauses S is *satisfied* by I if I satisfies every clause in S. If this is the case, I is a *model* of S and we write $I \models S$. A set of clauses S is *satisfiable* if it has a model; two sets of clauses are *equisatisfiable* if one is satisfiable exactly when the other is satisfiable.

Definition 1 A *specification* is a triple $(\mathcal{F}, \mathcal{I}, \mathcal{C})$ where $\mathcal{F}$ is a set of function symbols, $\mathcal{I}$ is a set of interpretations and $\mathcal{C}$ is a class of clause sets. Two specifications $(\mathcal{F}, \mathcal{T}, \mathcal{C})$ and $(\mathcal{F}', \mathcal{T}', \mathcal{C}')$ are *quasi-disjoint* if all the symbols in $\mathcal{F} \cap \mathcal{F}'$ are constants. A clause set $S \in \mathcal{C}$ is $\mathcal{T}$*-satisfiable* if there exists $I \in \mathcal{I}$ such that $I \models S$. S and S' are $\mathcal{T}$*-equisatisfiable* if they are both $\mathcal{T}$-satisfiable or both $\mathcal{T}$-unsatisfiable. $\Diamond$

In many cases, $\mathcal{I}$ is simply the set of all $\mathcal{F}$-interpretations, but our results also apply to domain-specific instantiation schemes such as those for fragments of Presburger arithmetic or of real numbers, for which the class of interpretations must be restricted. Restricting the form of the clause sets in $\mathcal{C}$ is necessary in some cases for defining instantiation schemes that are both terminating and refutationally complete; that is why we do not assume that $\mathcal{C} = \mathrm{C}(\mathcal{F})$. Axioms may be included into $\mathcal{C}$.

For instance the theory of arrays is modeled in our setting as a specification $(\mathcal{F}, \mathcal{I}, \mathcal{C})$ where $\mathcal{F}$ contains the function symbols $\mathit{select} : \mathtt{array} \times \mathtt{ind} \to \mathtt{elem}$ and $\mathit{store} : \mathtt{array} \times \mathtt{ind} \times \mathtt{elem} \to \mathtt{array}$ (together with an infinite set of constant symbols of each sort), $\mathcal{I}$ contains all $\mathcal{F}$-interpretations and $\mathcal{C}$ is the set of clauses of the form $S \cup \{\mathit{select}(\mathit{store}(x, z, v), z) \simeq v, z' \not\simeq z \vee \mathit{select}(\mathit{store}(x, z, v), z') \simeq \mathit{select}(x, z', v)\}$ where S is a finite set of ground clauses. Then the instantiation scheme should simply instantiate the axioms in such a way that $\mathcal{T}$-satisfiability is preserved[2].

The theory of Presburger arithmetic is modeled by a specification $(\mathcal{F}, \mathcal{I}, \mathcal{C})$ where $\mathcal{F}$ contains the function symbols $0 : \mathtt{nat}$,$s : \mathtt{nat} \to \mathtt{nat}$, $+ : \mathtt{nat} \times \mathtt{nat} \to \mathtt{nat} \leq : \mathtt{nat} \times \mathtt{nat} \to \mathtt{bool}$ (together with an infinite set of constant symbols of sort $\mathtt{nat}$), $\mathcal{I}$ is the set of interpretations such that $I(\mathtt{nat}) = \mathbb{N}$ and such that $0, s, +, \leq$ are interpreted as usual, and $\mathcal{C}$ contains every finite set of $\mathcal{F}$-clauses. In this case an instantiation scheme shall instantiate the variables occurring in the considered clause set in such a way that satisfiability is preserved (in Presburger arithmetic).

Let $\mathcal{T} = (\mathcal{F}, \mathcal{I}, \mathcal{C})$ be a specification. An *instantiation procedure for* $\mathcal{T}$ is a function Θ from $\mathcal{C}$ to $\mathcal{C}$ such that for every $S \in \mathcal{C}$, $\Theta(S)$ is a set of ground $\mathcal{F}$-instances of clauses in S. Θ is *complete* for $\mathcal{T}$ if for every $S \in \mathcal{C}$, S and $\Theta(S)$ are $\mathcal{T}$-equisatisfiable. If furthermore $\Theta(S)$ is finite for every $S \in \mathcal{C}$ and if there exists a procedure for checking whether a ground (finite) clause set is satisfiable in $\mathcal{I}$, then the satisfiability problem is decidable for $\mathcal{T}$. Several examples of complete instantiation procedures are available in the literature [14, 9, 3, 11, 12, 1, 4, 7, 6]). Our goal in this paper is to provide a general mechanism for constructing new complete instantiation procedures by combining existing ones.

[2]Alternatively, we could assume that $\mathcal{I}$ is the set of models of the axioms $\{\mathit{select}(\mathit{store}(x, z, v), z) \simeq v, z' \not\simeq z \vee \mathit{select}(\mathit{store}(x, z, v), z') \simeq \mathit{select}(x, z', v)\}$, and that $\mathcal{C}$ contains all finite sets of ground clauses, but then there would be no instantiation required.

3. Combining specifications

The most natural way of combining two specifications $\mathcal{T}$ and $\mathcal{T}'$ is to consider the union of $\mathcal{T}$ and $\mathcal{T}'$:

Definition 2 Let $\mathcal{T} = (\mathcal{F}, \mathcal{I}, \mathcal{C})$ and $\mathcal{T}' = (\mathcal{F}', \mathcal{I}', \mathcal{C}')$ be two specifications. $\mathcal{T} \cup \mathcal{T}'$ denotes the specification $(\mathcal{F} \cup \mathcal{F}', \mathcal{I}'', \mathcal{C}'')$, where $\mathcal{I}''$ denotes the set of $(\mathcal{F} \cup \mathcal{F}')$-interpretations I such that the projection of I onto $\mathcal{F}$ (resp. onto $\mathcal{F}'$) is in $\mathcal{I}$ (resp. in $\mathcal{I}'$), and $\mathcal{C}''$ is the class of clause sets of the form $S \cup S'$ where $S \in \mathcal{C}$ and $S' \in \mathcal{C}'$. $\Diamond$

The next definition shows how to construct an instantiation procedure for $\mathcal{T} \cup \mathcal{T}'$ by combining instantiation procedures for $\mathcal{T}$ and $\mathcal{T}'$ respectively.

Definition 3 Let Θ, Θ' denote two functions of profiles $\mathcal{C} \to \mathcal{C}$ and $\mathcal{C}' \to \mathcal{C}'$ respectively. Then $(\Theta \sqcup \Theta')$ denotes the function of profile $\mathcal{C} \cup \mathcal{C}' \to \mathcal{C} \cup \mathcal{C}'$ defined by: $(\Theta \sqcup \Theta')(S) \stackrel{\text{def}}{=} \Theta(S_{\mathcal{C}}) \cup \Theta'(S_{\mathcal{C}'})$, where $S_{\mathcal{C}} = S \cap \mathcal{C}$ and $S_{\mathcal{C}'} = S \cap \mathcal{C}'$. $\Diamond$

It is simple to find examples showing that $\Theta \sqcup \Theta'$ is not always complete for $\mathcal{T} \cup \mathcal{T}'$, even if Θ and Θ' are complete for $\mathcal{T}$ and $\mathcal{T}'$, respectively. We now exhibit conditions that guarantee completeness. The first condition is on the specifications, more precisely on the cardinality of the domains. Intuitively, the cardinality of a sort domain cannot depend on the considered formula, nor on the considered specification(this condition extends the notion of stably-infinite theories [15] to the case – frequently encountered in practice – where the domain of some sorts can be finite).

Definition 4 For every set D we denote by $card(D)$ the cardinality of D (D may be infinite hence $card(D)$ is an ordinal). A specification $\mathcal{T} = (\mathcal{F}, \mathcal{I}, \mathcal{C})$ is *stable* if there exists a function $card_{\mathcal{T}}$ mapping every sort $\mathtt{s} \in \mathfrak{S}_{\mathcal{F}}$ to an ordinal such that every $\mathcal{T}$-satisfiable clause set $S \in \mathcal{C}$ has a model I such that $\forall \mathtt{s} \in \mathfrak{S}_{\mathcal{F}}, card(I(\mathtt{s})) = card_{\mathcal{T}}(\mathtt{s})$.

Two stable specifications $\mathcal{T} = (\mathcal{F}, \mathcal{I}, \mathcal{C})$ and $\mathcal{T}' = (\mathcal{F}', \mathcal{I}', \mathcal{C}')$ are *structurally equipotent* if $\forall \mathtt{s} \in \mathfrak{S}_{\mathcal{F}} \cap \mathfrak{S}'_{\mathcal{F}}, card_{\mathcal{T}}(\mathtt{s}) = card_{\mathcal{T}'}(\mathtt{s})$. $\Diamond$

This condition is usually satisfied in practice: for instance two stably-infinite theories [15] over a single sort are always structurally equipotent. The second condition is on the instantiation procedures. It states that the addition of new equations and disequations between constant symbols to a given clause set should not affect the corresponding set of instances. This means that the instantiation procedures are unaffected by the equalities and disequalities entailed by the other specification.

Definition 5 Given a theory $\mathcal{T} = (\mathcal{F}, \mathcal{I}, \mathcal{C})$, a function $\Theta : \mathcal{C} \to \mathcal{C}$ is *base-complete for* $\mathcal{T}$ if for all sets of clauses $S \in \mathcal{C}$ and for all sets E only containing equalities or disequalities between constants, the sets $S \cup E$ and $\Theta(S) \cup E$ are $\mathcal{T}$-equisatisfiable. $\Diamond$

Notice that an instantiation procedure that is base-complete for $\mathcal{T}$ is also complete for $\mathcal{T}$ (it suffices to take $E = \emptyset$).

Theorem 6 *Let $\mathcal{T}, \mathcal{T}'$ be two stable, structurally equipotent and quasi-disjoint specifications. If Θ and Θ' are base-complete for $\mathcal{T}$ and $\mathcal{T}$ respectively, then $\Theta \sqcup \Theta'$ is base-complete for $\mathcal{T} \cup \mathcal{T}'$.*

The remainder of this section is devoted to the proof of this result.

Definition 7 A set of clauses S is *a connection* for $\mathcal{F}$ and $\mathcal{F}'$ if for every pair of constant symbols $a, b \in \mathcal{F} \cap \mathcal{F}'$, either $a \simeq b \in S$ or $a \not\simeq b \in S$. $\Diamond$

Lemma 8 *Let $\mathcal{T} = (\mathcal{F}, \mathcal{I}, \mathcal{C})$ and $\mathcal{T}' = (\mathcal{F}', \mathcal{I}', \mathcal{C}')$ be two stable and structurally equipotent specifications. If E is a connection for $\mathcal{F}$ and $\mathcal{F}'$ and if $S \cup E$ and $S' \cup E$ are respectively $\mathcal{T}$-satisfiable and $\mathcal{T}'$-satisfiable, then $S \cup S' \cup E$ is $(\mathcal{T} \cup \mathcal{T}')$-satisfiable.*

Proof Let I and I' be two models of $S \cup E$ and $S' \cup E$ respectively. Let $\mathcal{S} = \mathfrak{S}_F$ and $\mathcal{S}' = \mathfrak{S}'_F$. Since $\mathcal{T}$ and $\mathcal{T}'$ are stable, we may assume that $\forall \mathtt{s} \in \mathcal{S}, \mathit{card}(I(\mathtt{s})) = \mathit{card}_{\mathcal{T}}(\mathtt{s})$ and $\forall \mathtt{s} \in \mathcal{S}', \mathit{card}(I'(s')) = \mathit{card}_{\mathcal{T}'}(s')$. Hence, since $\mathcal{T}$ and $\mathcal{T}'$ are structurally equipotent, $\forall \mathtt{s} \in \mathcal{S} \cap \mathcal{S}', \mathit{card}(I(\mathtt{s})) = \mathit{card}(I'(\mathtt{s}))$. Thus for every $\mathtt{s} \in \mathcal{S} \cap \mathcal{S}'$ there exists a bijection ϕ_s from $I'(\mathtt{s})$ onto $I(\mathtt{s})$. We may assume that for every constant symbol a occurring in $\mathcal{F} \cap \mathcal{F}'$, $\phi_s(I'(a)) = I(a)$. Indeed, if a, b are constant symbols in $\mathcal{F} \cap \mathcal{F}'$ such that $I'(a) = I'(b)$ then necessarily $a \simeq b \in E$ because $I' \models E$ and E is a connection; thus $I(a) = I(b)$, since $I \models E$. This implies that there exists a sort-preserving bijection ϕ from $\bigcup_{s \in \mathcal{S} \cap \mathcal{S}'} I'(\mathtt{s})$ onto $\bigcup_{\mathtt{s} \in \mathcal{S} \cap \mathcal{S}'} I(\mathtt{s})$ such that $\forall a \in \mathcal{F} \cap \mathcal{F}', \phi(I(a)) = I'(a)$. The bijection ϕ is then extended to the elements of $\bigcup_{\mathtt{s} \in (\mathcal{S} \cup \mathcal{S}') \setminus (\mathcal{S} \cap \mathcal{S}')} I(\mathtt{s})$ by $\phi(e) \stackrel{\text{def}}{=} e$, and to truth values by $\phi(\mathtt{true}) \stackrel{\text{def}}{=} \mathtt{true}$ and $\phi(\mathtt{false}) \stackrel{\text{def}}{=} \mathtt{false}$. Let J be the $(\mathcal{F} \cup \mathcal{F}')$-interpretation defined as follows.

- For every sort symbol $\mathtt{s} \in \mathcal{S}$, $J(\mathtt{s}) \stackrel{\text{def}}{=} I(\mathtt{s})$ and for every sort symbol $\mathtt{s} \in \mathcal{S}' \setminus \mathcal{S}$, $J(\mathtt{s}) \stackrel{\text{def}}{=} I'(\mathtt{s})$.
- For all function symbols $f \in \mathcal{F}$ such that $\mathrm{rnk}(f) = \mathtt{s}_1 \times \ldots \times \mathtt{s}_n \to \mathtt{s}$ and for every $(e_1, \ldots, e_n) \in (J(\mathtt{s}_1) \times \ldots J(\mathtt{s}_n))$, $f^J(e_1, \ldots, e_n) \stackrel{\text{def}}{=} f^I(e_1, \ldots, e_n)$. Note that by definition $\mathtt{s}_1, \ldots, \mathtt{s}_n, \mathtt{s} \in \mathcal{S}$, thus for every $i \in [1..n], e_i \in I(\mathtt{s}_i) = J(\mathtt{s}_i)$ and $f^I(e_1, \ldots, e_n) \in I(\mathtt{s}) = J(\mathtt{s})$.
- For all function symbols $f \in \mathcal{F}'$ with $\mathrm{rnk}(f) = \mathtt{s}_1 \times \ldots \times \mathtt{s}_n \to \mathtt{s}$ and for all $(e_1, \ldots, e_n) \in (J(\mathtt{s}_1) \times \ldots J(\mathtt{s}_n))$, $f^J(e_1, \ldots, e_n) \stackrel{\text{def}}{=} \phi(f^{I'}(\phi^{-1}(e_1), \ldots, \phi^{-1}(e_n)))$. Note that by definition $\mathtt{s}_1, \ldots, \mathtt{s}_n, \mathtt{s} \in \mathcal{S}'$, thus, since ϕ is a bijection from $I'(\mathtt{s}_i)$ to $J(\mathtt{s}_i)$, $\phi^{-1}(e_i)$ is well-defined and belongs to $I'(\mathtt{s}_i)$. Furthermore, $f^{I'}(e_1, \ldots, e_n) \in I'(\mathtt{s})$, hence $f^J(e_1, \ldots, e_n) \in \phi(I'(\mathtt{s}))$. If $\mathtt{s} \in \mathcal{S}$ then ϕ is a bijection from $I'(\mathtt{s})$ to $I(\mathtt{s})$ and by the first point above $J(\mathtt{s}) = I(\mathtt{s})$, thus we must have $\phi(I'(\mathtt{s})) = I(\mathtt{s}) = J(\mathtt{s})$. If $s \notin \mathcal{S}$ then ϕ is the identity on $I'(\mathtt{s})$ and by the first point above $J(\mathtt{s}) = I'(\mathtt{s})$. Hence $\phi(I'(\mathtt{s})) = I'(\mathtt{s}) = J(\mathtt{s})$. Thus in both cases $f^J(e_1, \ldots, e_n) \in J(\mathtt{s})$.

This interpretation is well-defined, since $\mathcal{F}$ and $\mathcal{F}'$ are quasi-disjoint and ϕ is such that for all constants a occurring both in $\mathcal{F}$ and in $\mathcal{F}'$, $\phi(I'(a)) = I(a)$. Furthermore, J satisfies the following properties:

Proposition 9 *Let $\mathcal{E}$ be an $\mathcal{F}$-expression, and let σ be a valuation for I and t. Then σ is a valuation for J and $\mathcal{E}$ and $[\mathcal{E}]^\sigma_J = [\mathcal{E}]^\sigma_I$.*

PROOF We prove the result by induction on the size of $\mathcal{E}$. If $\mathcal{E}$ is a variable, then we have by definition $[\mathcal{E}]^\sigma_J = \sigma(x)$ and $[\mathcal{E}]^\sigma_I = \sigma(x)$. If $\mathcal{E}$ is of the form $f(t_1, \ldots, t_n)$ then $[\mathcal{E}]^\sigma_J = f^J([t_1]^\sigma_J, \ldots, [t_n]^\sigma_J)$. By the induction hypothesis, $\forall i \in [1..n], [t_i]^\sigma_J = [t_i]^\sigma_I$. Furthermore, since $f \in \mathcal{F}$, by definition of f^J, $f^J([t_1]^\sigma_I, \ldots, [t_n]^\sigma_I) = f^I([t_1]^\sigma_I, \ldots, [t_n]^\sigma_I) = [\mathcal{E}]^\sigma_I$. If $\mathcal{E}$ is an atom $u \simeq v$ then $[\mathcal{E}]^\sigma_J = \texttt{true}$ iff $[u]^\sigma_J = [v]^\sigma_J$. By the induction hypothesis we have $[u]^\sigma_J = [u]^\sigma_I$ and $[v]^\sigma_J = [v]^\sigma_I$, hence $[\mathcal{E}]^\sigma_J = \texttt{true} \Leftrightarrow ([u]^\sigma_I = [v]^\sigma_I) \Leftrightarrow [\mathcal{E}]^\sigma_I = \texttt{true}$. The case where $\mathcal{E}$ is of the form $u \not\simeq v$ is similar. If $\mathcal{E}$ is a clause then $[\mathcal{E}]^\sigma_J = \texttt{true}$ iff $\exists l \in \mathcal{E}, [l]^\sigma_J = \texttt{true}$ i.e. (by the induction hypothesis) iff $\exists l \in \mathcal{E}, [l]^\sigma_I = \texttt{true}$ i.e. iff $[\mathcal{E}]^\sigma_I = \texttt{true}$. ∎

Proposition 10 *Let $\mathcal{E}$ be an $\mathcal{F}'$-expression. Let σ be a valuation for I' and $\mathcal{E}$. Then $\phi \circ \sigma$ is a valuation for J and $\mathcal{E}$, and $[\mathcal{E}]^{\phi\circ\sigma}_J = \phi([\mathcal{E}]^\sigma_{I'})$.*

PROOF Again, the proof is by induction on the size of $\mathcal{E}$. If $\mathcal{E} \in \mathcal{X}$, then by definition $[\mathcal{E}]^{\phi\circ\sigma}_J = \phi(\sigma(x))$ and $[\mathcal{E}]^\sigma_{I'} = \sigma(x)$. If $\mathcal{E}$ is of the form $f(t_1, \ldots, t_n)$ then $[\mathcal{E}]^{\phi\circ\sigma}_J = f^J([t_1]^{\phi\circ\sigma}_J, \ldots, [t_n]^{\phi\circ\sigma}_J)$. By the induction hypothesis, $\forall i \in [1..n], [t_i]^{\phi\circ\sigma}_J = \phi([t_i]^\sigma_{I'})$. Furthermore, $f \in \mathcal{F}'$ thus by definition of f^J, $f^J(\phi([t_1]^\sigma_{I'}), \ldots, \phi([t_n]^\sigma_{I'})) = \phi(f^{I'}(\phi^{-1}(\phi([t_1]^\sigma_{I'})), \ldots, \phi^{-1}(\phi([t_n]^\sigma_{I'})))) = \phi(f^{I'}([t_1]^\sigma_{I'}, \ldots, [t_n]^\sigma_{I'})) = \phi([\mathcal{E}]^\sigma_{I'})$. If $\mathcal{E}$ is an atom $u \simeq v$ then $[\mathcal{E}]^{\phi\circ\sigma}_J = \texttt{true}$ iff $[u]^{\phi\circ\sigma}_J = [v]^{\phi\circ\sigma}_J$. By the induction hypothesis $[u]^{\phi\circ\sigma}_J = \phi([u]^\sigma_I)$ and $[v]^{\phi\circ\sigma}_J = \phi([v]^\sigma_I)$, hence $[\mathcal{E}]^{\phi\circ\sigma}_J = \texttt{true} \Leftrightarrow \phi([u]^\sigma_I) = \phi([v]^\sigma_I)$ i.e. (since ϕ is a bijection): $[\mathcal{E}]^{\phi\circ\sigma}_J = \texttt{true} \Leftrightarrow [u]^\sigma_I = [v]^\sigma_I \Leftrightarrow [\mathcal{E}]^\sigma_{I'} = \texttt{true}$. The proof is similar for $u \not\simeq v$. If $\mathcal{E}$ is a clause then $[\mathcal{E}]^{\phi\circ\sigma}_J = \texttt{true}$ iff $\exists L \in \mathcal{E}, [L]^{\phi\circ\sigma}_J = \texttt{true}$ i.e. (by the induction hypothesis) iff $\exists L \in \mathcal{E}, \phi([L]^\sigma_{I'}) = \texttt{true}$ i.e. iff $[\mathcal{E}]^\sigma_{I'} = \texttt{true}$. ∎

We now complete the proof of Lemma 8. Let C be a clause in $S \cup S'$ and let σ be a function mapping every variable of sort $\mathtt{s}$ in C to an element of $J(\mathtt{s})$; we prove that $[C]^\sigma_J = \texttt{true}$. Two cases are distinguished. If $C \in S$, then for every variable x of sort $\mathtt{s}$ occurring in C, $\mathtt{s} \in \mathcal{S}$. Thus by definition $J(\mathtt{s}) = I(\mathtt{s})$ hence $\sigma(x) \in I(\mathtt{s})$. By Proposition 9 $[C]^\sigma_J = [C]^\sigma_I$, and since $I \models S$, $[C]^\sigma_I = \texttt{true}$. Thus $[C]^\sigma_J = \texttt{true}$.

If $C \in S'$, then for every variable x of sort $\mathtt{s}$ occurring in C, $\mathtt{s} \in \mathcal{S}'$. If $\mathtt{s} \in \mathcal{S}$, then by definition $J(\mathtt{s}) = I(\mathtt{s}) = \phi(I'(\mathtt{s}))$ (since ϕ is a bijection from $I'(\mathtt{s})$ to $I(\mathtt{s})$); otherwise, $s \notin \mathcal{S}'$ and in this case $J(\mathtt{s}) = I'(\mathtt{s})$, which implies that $\sigma(x) \in I'(\mathtt{s})$. Furthermore, in the latter case, σ is the identity on $I'(\mathtt{s})$. Thus in both cases $\sigma(x) \in \phi(I'(\mathtt{s}))$. By Proposition 10 $[C]^\sigma_J = [C]^{\sigma'}_{I'}$, where $\sigma' = \phi^{-1} \circ \sigma$. Since $I \models S$, $[C]^{\sigma'}_{I'} = \texttt{true}$, thus $[C]^\sigma_J = \texttt{true}$. ∎

PROOF (OF THEOREM 6) $\mathcal{T}_c = \mathcal{T} \cup \mathcal{T}'$ is of the form $(\mathcal{F} \cup \mathcal{F}', \mathcal{I}'', \mathcal{C}'')$ where $\mathcal{I}''$ and $\mathcal{C}''$ are defined as in Definition 2; let $S_c \in \mathcal{C}''$. By definition S_c is of the form $S \cup S'$ where $S = S_c \cap \mathcal{C}$, and $S' = S_c \cap \mathcal{C}'$. Let E_c be a set of equalities and inequalities between constants occurring both in $\mathcal{F}$ and $\mathcal{F}'$; we show that $S_c \cup E_c$ and $(\Theta \sqcup \Theta')(S_c) \cup E_c$ are $\mathcal{T}_c$-equisatisfiable. Assume that $(\Theta \sqcup \Theta')(S_c) \cup E_c$ is $\mathcal{T}_c$-satisfiable, and let I_c be a $\mathcal{T}_c$-model of $(\Theta \sqcup \Theta')(S_c) \cup E_c$. Let F_c be the set of equations and inequations between constant symbols occurring both in $\mathcal{F}$ and $\mathcal{F}'$ that hold in I_c. Note that by definition, $E_c \subseteq F_c$, and by definition of $\sqcup$, $(\Theta \sqcup \Theta')(S_c) = \Theta(S_c \cap \mathcal{C}) \cup \Theta'(S_c \cap C') = \Theta(S) \cup \Theta'(S')$. Since $\Theta(S) \cup F_c \subseteq (\Theta \sqcup \Theta')(S_c) \cup F_c$, necessarily, $I_c \models \Theta(S) \cup F_c$. Hence, since Θ is base-complete for $\mathcal{T}$, $S \cup F_c$ is $\mathcal{T}$-satisfiable. The same reasoning proves that $S' \cup F_c$ is $\mathcal{T}'$-satisfiable. By definition, $\mathcal{F}_c$ is a connection for $\mathcal{F}$ and $\mathcal{F}'$, and Lemma 8 guarantees that $S \cup S' \cup F_c$ is satisfiable. Therefore, $S_c \cup F_c$ and $S_c \cup E_c$ are $\mathcal{T}_c$-satisfiable.

Conversely, assume that $S_c \cup E_c$ is $\mathcal{T}_c$-satisfiable; let I_c be a $\mathcal{T}_c$-model of $S_c \cup E$. Let F_c be the set of equations and inequations between constant symbols occurring both in $\mathcal{F}$ and $\mathcal{F}'$ that hold in I_c; again by definition, $E_c \subseteq F_c$. Since $S \cup F_c \subseteq S_c \cup F_c$, necessarily, $I_c \models S \cup F$; and since Θ is base-complete for $\mathcal{T}$, $\Theta(S) \cup F_c$ is $\mathcal{T}$-satisfiable. Similarly, $\Theta'(S') \cup F_c$ is $\mathcal{T}'$-satisfiable. Thus by Lemma 8, $\Theta(S) \cup \Theta'(S') \cup F_c$ is satisfiable.

4. Base-complete instantiation schemes

Theorem 6 applies only to instantiation procedures that are base-complete. However, most existing instantiation procedures do not satisfy this requirement[3]. In this section we show how to transform any instantiation procedure for a specific theory $\mathcal{T}$ into one that is base-complete for $\mathcal{T}$. We assume that the considered procedure satisfies the following properties:

Definition 11 A function $\theta : 2^{\mathrm{C}(\mathcal{F})} \to 2^{\mathrm{C}(\mathcal{F})}$ is *instantiation-complete* for a specification $\mathcal{T} = (\mathcal{F}, \mathcal{I}, \mathcal{C})$ if the following conditions hold:

1. If $\theta(S)$ is $\mathcal{T}$-satisfiable then S is $\mathcal{T}$-satisfiable.
2. Every clause in $\theta(S)$ is an instance of a clause in S.
3. For every set of clauses S and for every pair of constant symbols a, b we have $\theta(S \cup \{a \simeq b\}) \supseteq \theta(S)$.
4. For every pair of constants a, b: $\theta(S \cup \{a \not\simeq b\}) = \theta(S) \cup \{a \not\simeq b\}$.

These conditions are quite natural: Conditions 1 and 2 are straightforward, Condition 3 always holds if Θ is monotonic, and Condition 4 states that the set of instances should not depend of disequations between constants.

If $\mathcal{F}$ is a set of function symbols, we denote by $\mathrm{E}_{\mathcal{F}}$ the set of equations and disequations of the form $a \simeq b$ where a, b are constant symbols of the same sort

[3] Although it is fulfilled for instance by the instantiation schemes in [7, 6, 11].

in $\mathcal{F}$. If S is a set of $\mathcal{F}$-clauses, $\mathrm{E}_{\mathcal{F}}^{S}$ (resp. $\mathrm{E}_{\mathcal{F}}^{\overline{S}}$) denotes the set of clauses in $\mathrm{E}_{\mathcal{F}}$ that are instances of a clause in S (resp. that are not instances of a clause in S). For every function $\theta : 2^{\mathrm{C}(\mathcal{F})} \to 2^{\mathrm{C}(\mathcal{F})}$ we denote by $\theta^{\diamond}$ the function defined by: $\theta^{\diamond}(S) \stackrel{\text{def}}{=} \theta(S \cup \mathrm{E}_{\mathcal{F}}) \setminus \mathrm{E}_{\mathcal{F}}^{\overline{S}}$. The function $\theta^{\diamond}$ is quite simple to compute, if θ is computable: it suffices to add all the clauses in $\mathrm{E}_{\mathcal{F}}$ to S (i.e. every possible equations between constants) then to run the instantiation procedure θ. Afterwards, the produced clauses that are not instances of a clause in S are discarded.

Theorem 12 *If θ is an instantiation-complete function for $\mathcal{T}$, then $\theta^{\diamond}$ is base-complete for $\mathcal{T}$.*

Proof Let S denote a set of clauses, E denote a set of unit, ground, flat clauses (i.e. of equations and disequations between constants), and let E^{+} (resp. E^{-}) be the set of positive (resp. negative) clauses in E. Assume that $S \cup E$ is $\mathcal{T}$-satisfiable, let I be a $\mathcal{T}$-model of $S \cup E$; we show that I is also a model of $\theta^{\diamond}(S)$. Let $C \in \theta^{\diamond}(S)$, by definition, $C \in \theta(S \cup \mathrm{E}_{\mathcal{F}})$ and $C \notin \mathrm{E}_{\mathcal{F}}^{\overline{S}}$. Since $C \in \theta(S \cup \mathrm{E}_{\mathcal{F}})$, by Condition 2 of Definition 11, C is an instance of a clause $D \in S \cup \mathrm{E}_{\mathcal{F}}$. If C is an instance of a clause D in S, then obviously $I \models D$ thus $I \models C$. Otherwise, C must be an instance of a clause in $\mathrm{E}_{\mathcal{F}}$, hence must occur in $\mathrm{E}_{\mathcal{F}}$ (since $\mathrm{E}_{\mathcal{F}}$ is ground). Since C is not an instance of a clause in S, by definition, $C \notin \mathrm{E}_{\mathcal{F}}^{S}$, hence $C \in \mathrm{E}_{\mathcal{F}}^{\overline{S}}$, which contradicts the assumption above. Thus $I \models \theta^{\diamond}(S)$ and $\theta^{\diamond}(S) \cup E$ is $\mathcal{T}$-satisfiable.

Conversely, assume that $\theta^{\diamond}(S) \cup E$ is $\mathcal{T}$-satisfiable and let I be a $\mathcal{T}$-model of $\theta^{\diamond}(S) \cup E$. Let C be a clause in $\theta(S \cup E)$, we prove that $C \in \theta^{\diamond}(S) \cup E$. This is obvious if $C \in E$, now assume that $C \notin E$. By Condition 4 of Definition 11, $\theta(S \cup E) = \theta(S \cup E^{+} \cup E^{-}) = \theta(S \cup E^{+}) \cup E^{-}$, thus $C \in \theta(S \cup E^{+})$. Since $S \cup E^{+} \subseteq S \cup \mathrm{E}_{\mathcal{F}}$, by Condition 3 of Definition 11, $C \in \theta(S \cup \mathrm{E}_{\mathcal{F}})$. We show that $C \in \theta^{\diamond}(S)$. Assume the contrary. Then by definition of $\theta^{\diamond}$, since $\theta^{\diamond}(S) \stackrel{\text{def}}{=} \theta(S \cup \mathrm{E}_{\mathcal{F}}) \setminus \mathrm{E}_{\mathcal{F}}^{\overline{S}}$, necessarily, $C \in \mathrm{E}_{\mathcal{F}}^{\overline{S}}$, which means that C cannot be an instance of a clause in S. By Condition 2 of Definition 11, since $C \in \theta(S \cup E)$ by hypothesis, C must be an instance of a clause in E, hence must occur in E (since E is ground), which is impossible since we assumed $C \notin E$. Consequently, $\theta(S \cup E) \subseteq \theta^{\diamond}(S) \cup E$, hence $\theta(S \cup E)$ is satisfiable. By Condition 1 of Definition 11, this implies that $S \cup E$ is $\mathcal{T}$-satisfiable. ∎

5. Conclusion

We have provided sufficient conditions that guarantee instantiation schemes for distinct theories can be combined to act as an instantiation scheme for the union of the theories. This result requires the instantiation schemes to be base-complete. Since this property is not always satisfied by the existing procedures, we also showed how existing instantiation schemes can be transformed to satisfy the base-completeness requirement. We are currently investigating how these results can be extended to types of theory combinations other than unions, such as *nested* theory combinations.

[1] A. Abadi, A. Rabinovich, and M. Sagiv. Decidable fragments of many-sorted logic. *Journal of Symbolic Computation*, 45(2):153 – 172, 2010.

[2] C. Barrett, R. Sebastiani, S. Seshia, and C. Tinelli. Satisfiability modulo theories. In A. Biere, M. J. H. Heule, H. van Maaren, and T. Walsh, editors, *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, chapter 26, pages 825–885. IOS Press, 2009.

[3] P. Baumgartner and C. Tinelli. The Model Evolution Calculus. In F. Baader, editor, *CADE-19 – The 19th International Conference on Automated Deduction*, volume 2741 of *LNAI*, pages 350–364. Springer, 2003.

[4] A. R. Bradley, Z. Manna, and H. B. Sipma. What's decidable about arrays? In E. A. Emerson and K. S. Namjoshi, editors, *Proc. VMCAI-7*, volume 3855 of *LNCS*, pages 427–442. Springer, 2006.

[5] L. M. de Moura and N. Bjørner. Satisfiability modulo theories: An appetizer. In M. V. M. Oliveira and J. Woodcock, editors, *SBMF*, volume 5902 of *Lecture Notes in Computer Science*, pages 23–36. Springer, 2009.

[6] M. Echenim and N. Peltier. Instantiation of SMT problems modulo Integers. In *AISC 2010 (10th International Conference on Artificial Intelligence and Symbolic Computation)*, LNCS. Springer, 2010.

[7] M. Echenim and N. Peltier. An instantiation scheme for satisfiability modulo theories. *Journal of Automated Reasoning*, 2010. Accepted, to appear.

[8] M. Fitting. *First-Order Logic and Automated Theorem Proving*. Texts and Monographs in Computer Science. Springer-Verlag, 1990.

[9] H. Ganzinger and K. Korovin. New directions in instantiation-based theorem proving. In *Proc. 18th IEEE Symposium on Logic in Computer Science,(LICS'03)*, pages 55–64. IEEE Computer Society Press, 2003.

[10] Y. Ge, C. W. Barrett, and C. Tinelli. Solving quantified verification conditions using satisfiability modulo theories. *Annals of Mathematics and Artificial Intelligence*, 55(1-2):101–122, 2009.

[11] Y. Ge and L. M. de Moura. Complete instantiation for quantified formulas in satisfiabiliby modulo theories. In A. Bouajjani and O. Maler, editors, *CAV 2009*, volume 5643 of *LNCS*, pages 306–320. Springer, 2009.

[12] R. Loos and V. Weispfenning. Applying linear quantifier elimination. *Comput. J.*, 36(5):450–462, 1993.

[13] L. Moura and N. Bjørner. Efficient E-Matching for SMT Solvers. In *CADE-21: Proceedings of the 21st international conference on Automated Deduction*, pages 183–198, Berlin, Heidelberg, 2007. Springer-Verlag.

[14] D. A. Plaisted and Y. Zhu. Ordered semantic hyperlinking. *Journal of Automated Reasoning*, 25(3):167–217, October 2000.

[15] C. Tinelli and M. Harandi. A new correctness proof of the Nelson-Oppen combination procedure. In *Frontiers of Combining Systems, volume 3 of Applied Logic Series*, pages 103–120. Kluwer Academic Publishers, 1996.