Combining Decision Procedures for Sorted Theories

Cesare Tinelli[1] and Calogero G. Zarba[2]

[1] Department of Computer Science, The University of Iowa, USA
[2] LORIA and INRIA-Lorraine, France

Abstract. The Nelson-Oppen combination method combines decision procedures for theories satisfying certain conditions into a decision procedure for their union. While the method is known to be correct in the setting of unsorted first-order logic, some current implementations of it appear in tools that use a sorted input language. So far, however, there have been no theoretical results on the correctness of the method in a sorted setting, nor is it obvious that the method in fact lifts as is to logics with sorts. To bridge this gap between the existing theoretical results and the current implementations, we extend the Nelson-Oppen method to (order-)sorted logic and prove it correct under conditions similar to the original ones. From a theoretical point of view, the extension is relevant because it provides a rigorous foundation for the application of the method in a sorted setting. From a practical point of view, the extension has the considerable added benefits that in a sorted setting the method's preconditions become easier to satisfy in practice, and the method's nondeterminism is generally reduced.

1 Introduction

The problem of combining decision procedures for logical theories arises in many areas of computer science and artificial intelligence, such as constraint solving, theorem proving, knowledge representation and reasoning. In general, one has two theories T_1 and T_2 over the signatures Σ_1 and Σ_2, for which validity of a certain class of formulae (e.g., universal, existential positive, etc.) is decidable. The question is then whether one can combine the decision procedures for T_1 and for T_2 into a decision procedure for a suitable combination of T_1 and T_2.

The most widely applied and best known method for combining decision procedures is due to Nelson and Oppen [8]. This method is at the heart of the verification systems CVC [9], ARGO-LIB [6] and SIMPLIFY [1], among others.

The Nelson-Oppen method allows one to decide the satisfiability (and hence the validity) of quantifier-free formulae in a combination T of two first-order theories T_1 and T_2, using as black boxes a decision procedure for the satisfiability of quantifier-free formulae in T_1 and a decision procedure for the satisfiability of quantifier-free formulae in T_2. The method is correct whenever the theories T, T_1, and T_2 satisfy the following restrictions: (i) T is logically equivalent to

$T_1 \cup T_2$, (ii) the signatures of T_1 and T_2 are disjoint, and (iii) T_1 and T_2 are both stably infinite.

While the Nelson-Oppen method is defined in the context of unsorted first-order logic (with equality), more recent verification tools that rely on it have a sorted input language. However, strictly speaking, it is not clear how correct these verification tools are, because it is not clear whether the Nelson-Oppen method does in fact lift to a sorted setting. The common consensus among the researchers in the field is that, at least for standard many-sorted logic, "the method should be correct" as is. But to our knowledge there is no formal proof of this conjecture, nor is it obvious that the conjecture holds. In fact, a crucial requirement for the correctness of the method is that the signatures of the component theories share no function or predicate symbols (except equality). Now, in a sorted context, the method is only useful for theories whose signatures Σ_1 and Σ_2 share, if not function/predicate symbols, at least one sort. Otherwise, the only well-sorted $(\Sigma_1 \cup \Sigma_2)$-terms are either Σ_1-terms or Σ_2-terms, with Σ_1-terms sharing no variables with Σ_2-terms, which makes the combination problem trivial. Sharing sorts however essentially amounts to sharing predicate symbols, something that the original Nelson-Oppen method does not allow.

We prove in this paper that the method can indeed be lifted to sorted logics, provided that its applicability conditions are adjusted appropriately. For standard many-sorted logic, the only significant adjustment is to define stable infiniteness with respect to a set of sorts. The added benefit of using a sorted logic then becomes that it is easier to prove that a sorted theory is stably infinite over a certain sort s, than it is to prove that its unsorted version is stably infinite as a whole.[3] Also, one can now combine with no problems theories with sorts admitting only finite interpretations, say, as long as these sorts are not shared.

For *order*-sorted logics, the situation is in general considerably more complicated, requiring substantial additions to the method (see Section 5 for more details). There is however a useful special case in which the many-sorted version of the method works just as well with order-sorted theories: the case in which the shared sorts are pairwise *disconnected* i.e., do not appear in the same connected component of the subsort relation. Because of this we present our correctness results directly for order-sorted logic. Or more accurately, since there exist several, inequivalent order-sorted logics, we present our results for a fairly general version of first-order order-sorted logic based on a well developed and studied equational order-sorted logic by Goguen and Meseguer [5].

We introduce our order-sorted logic in Section 2. Then we present a version of the Nelson-Oppen combination method for this logic in Section 3, and prove it correct in Section 4. The correctness proof is based on a suitable order-sorted version of the model theoretic results used in [12, 15] to prove the correctness of the (unsorted) Nelson-Oppen method. We conclude the paper in Section 5 with some directions for further research. The interested readers can find the complete proofs and more details in [13].

[3] Intuitively, one has to worry only about what the theory says about s, and can ignore what it says about other sorts.

2 An Order-sorted Logic with Decorated Symbols

We will assume some familiarity in the reader with many-sorted and order-sorted algebras and logics with equality (denoted here by $\approx$) as defined for instance in [5]. We will mostly follow the notation used in [5]. The logic we present here is inspired by the order-sorted equational logic proposed by Meseguer in [7] as a successor of the logic in [5]. One main difference will be that our logic uses a vocabulary of *decorated* symbols, that is, function and predicate symbols that carry a sort declaration explicitly in them.

For any set S we denote by S^* the set all words over S, including the empty word ϵ. For the rest of the paper, we fix four countably-infinite sets: a set $\mathcal{F}$ of *function* symbols, a set $\mathcal{P}$ of *predicate* symbols, a set $\mathcal{S}$ of *sort* symbols, and a set $\mathcal{X}$ of *variables* that is disjoint with $\mathcal{F}$, $\mathcal{P}$ and $\mathcal{S}$.

A *decorated function symbol*, written as $f_{w,s}$, is a triple $(f, w, s) \in \mathcal{F} \times \mathcal{S}^* \times \mathcal{S}$. A *decorated constant* is a decorated function symbol of the form $f_{\epsilon,s}$. A *decorated predicate symbol*, written as p_w, is a pair $(p, w) \in \mathcal{P} \times \mathcal{S}^*$. A *decorated variable*, written as x_s, is a pair $(x, s) \in \mathcal{X} \times \mathcal{S}$.

An *order-sorted (decorated) signature* Σ is a tuple $\Sigma = (S, \prec, F, P)$ where $S \subseteq \mathcal{S}$ is a set of sorts, $F \subseteq (\mathcal{F} \times S^* \times S)$ is a set of decorated function symbols, $P \subseteq (\mathcal{P} \times S^*)$ is a set of decorated predicate symbols, and $\prec$ is a binary relation over S. We denote by $\sim$ the symmetric closure of $\prec$, and by $\prec^*$ and $\sim^*$ the reflexive, transitive closure of $\prec$ and $\sim$, respectively. We say that a sort s_1 is a *subsort* of a sort s_2 iff $s_1 \prec^* s_2$, and that s_1 and s_2 are *connected* iff $s_1 \sim^* s_2$. If $w_1, w_2 \in S^*$, we write $w_1 \prec^* w_2$ iff w_1 and w_2 have the same length and each component of w_1 is a subsort of the corresponding component of w_2. (Similarly for $w_1 \sim^* w_2$.) When convenient, we will write Σ^{S} for S, Σ^{F} for F, and Σ^{P} for P. For simplicity, we will consider only signatures with a *finite* set of sorts.

In the following, we fix an order-sorted signature $\Sigma = (S, \prec, F, P)$.

We say that two distinct decorated function symbols $f_{w,s}$ and $f_{w',s'}$ of Σ are *subsort overloaded* (in Σ) if $ws \sim^* w's'$. Otherwise, we say that they are *ad-hoc overloaded*. (Similarly, for predicate symbols.) As we will see, the logic's semantics will allow ad-hoc overloaded symbols to stand for completely unrelated functions/relations, but will require subsort overloaded symbols to stand for functions/relations that agree on the intersection of their domains.

Definition 1 (Order-sorted Terms). *Let $X \subseteq \mathcal{X}$ be a set of variables. For all $s \in S$, the set $\mathcal{T}_s(\Sigma, X)$ of* order-sorted Σ-terms of sort s over X *is the set defined as follows by structural induction:*

- *every decorated variable $x_{s'} \in (X \times S)$ with $s' \prec^* s$ is in $\mathcal{T}_s(\Sigma, X)$;*
- *if $f_{s_1 \cdots s_n, s'} \in F$, $t_i \in \mathcal{T}_{s_i}(\Sigma, X)$ for $i = 1, \ldots, n$, and $s' \prec^* s$, then $f_{s_1 \cdots s_n, s'}(t_1, \ldots, t_n)$ is in $\mathcal{T}_s(\Sigma, X)$.*

We denote by $\mathcal{T}_w(\Sigma, X)$ with $w = s_1 \cdots s_n$ the set $\mathcal{T}_{s_1}(\Sigma, X) \times \cdots \times \mathcal{T}_{s_n}(\Sigma, X)$.

We say that a Σ-term has *nominal sort s* if it is a variable of the form x_s or its top symbol has the form $f_{w,s}$. Note that the nominal sort of a term t is always the least sort of t.

While decorated terms are cumbersome to write in practice, at the theoretical level they dramatically simplify or eliminate a number of problems that vex more standard definitions of sorted logics. For instance, with full decoration of symbols, sort inference is trivial, terms have a least sort, and the union and the intersection of two sorted signatures, crucial operations in combination settings, can be defined in a straightforward way as component-wise union and intersection. Of course we do not advocate that decorated signatures and terms be used in practice. They are just a way to abstract away the usual parsing, sort inference, and signature composition problems that arise when working with sorted languages, but that are not relevant for the essence of our combination results.

Definition 2 (Order-sorted atoms). *A Σ-atom is either an expression of the form $p_w(\mathbf{t})$ where $p_w \in P$ and $\mathbf{t} \in \mathcal{T}_w(\Sigma, X)$, or one of the form $t_1 \approx t_2$ where $(t_1, t_2) \in \mathcal{T}_{s_1 s_2}(\Sigma, X)$ for some $s_1, s_2 \in S$ such that $s_1 \sim^* s_2$.*

Order-sorted (first-order) Σ-formulae are defined on top of Σ-atoms as in the unsorted case, but with the difference that quantifiers bind decorated variables. Following [7], and contrary to [5] which allows only equations between terms of comparable sorts, we allow equations between terms of connected sorts.[4] This makes the logic both more general and more robust—see [7] for a discussion.

A *many-sorted Σ-structure* is a pair $\mathcal{A} = (A, I)$ where $A = \{A_s \mid s \in S\}$ is an S-indexed family of sets, *domains*, and I is a mapping of the decorated symbols of Σ to functions and relations over the carrier sets. Specifically, for each word $w = s_1 \cdots s_n \in S^*$, let A_w denote the set $A_{s_1} \times \cdots \times A_{s_n}$. Then I maps each decorated function symbol $f_{w,s} \in F$ to a (total) function $f^{\mathcal{A}}_{w,s} \in (A_w \to A_s)$, and each decorated predicate symbol $p_w \in P$ to a relation $p^{\mathcal{A}}_w \subseteq A_w$.

Definition 3 (Order-sorted Structure). *An* order-sorted Σ-structure *is a many-sorted (S, F, P)-structure $\mathcal{A} = (A, I)$ such that*

1. *For all $s, s' \in S$ such that $s \prec^* s'$, $A_s \subseteq A_{s'}$.*
2. *For all $f_{w,s}, f_{w',s'} \in F$ such that $ws \sim^* w's'$, the functions $f^{\mathcal{A}}_{w,s}$ and $f^{\mathcal{A}}_{w',s'}$ agree on $A_w \cap A_{w'}$.*[5]
3. *For all $p_w, p_{w'} \in P$ such that $w \sim^* w'$, the restrictions of $p^{\mathcal{A}}_w$ and of $p^{\mathcal{A}}_{w'}$ to $A_w \cap A_{w'}$ coincide.*

This definition of order-sorted structure is modeled after the definition of order-sorted algebra in [7]. As in [7], the given semantics supports subsort overloading of function symbols by requiring that, whenever $ws \sim^* w's'$, the functions denoted by $f_{w,s}$ and $f_{w',s'}$ coincide on the tuples shared by their domains. (Similarly for predicate symbols.)

Satisfiability of *Σ-sentences* (i.e. closed Σ-formulae) in an order-sorted Σ-structure $\mathcal{A}$ is defined similarly to the unsorted case. As usual, we say that $\mathcal{A}$ is a *Σ-model* of a set Φ of Σ-sentences if $\mathcal{A}$ satisfies every sentence in Φ.

[4] So, for instance, we allow an equation between two terms of respective sort s_1 and s_2 if they have a common subsort, even if neither $s_1 \prec^* s_2$ nor $s_2 \prec^* s_1$.

[5] Where $A_w \cap A_{w'}$ denotes the component-wise intersection of the tuples A_w and $A_{w'}$.

Definition 4 (Order-sorted Morphisms). *Let $\mathcal{A}$ and $\mathcal{B}$ be two order-sorted Σ-structures. A* order-sorted Σ-homomorphism $h : \mathcal{A} \to \mathcal{B}$ of $\mathcal{A}$ into $\mathcal{B}$ *is an S-indexed family $\{h_s : A_s \to B_s \mid s \in S\}$ of functions such that:*

1. *for all $f_{w,s} \in F$ with $w = s_1 \cdots s_n$ and all $a_i \in A_{s_i}$ with $i = 1, \ldots, n$, $h_s(f^{\mathcal{A}}_{w,s}(a_1, \ldots, a_n)) = f^{\mathcal{B}}_{w,s}(h_{s_1}(a_1) \ldots, h_{s_n}(a_n))$;*
2. *for all $p_w \in P$ with $w = s_1 \cdots s_n$ and all $a_i \in A_{s_i}$ with $i = 1, \ldots, n$, $(a_1, \ldots, a_n) \in p^{\mathcal{A}}_w \ \Rightarrow\ (h_{s_1}(a_1), \ldots, h_{s_n}(a_n)) \in p^{\mathcal{B}}_w$.*
3. *for all $s, s' \in S$ with $s \sim^* s'$, the functions h_s and $h_{s'}$ agree on $A_s \cap A_{s'}$.*

A Σ-isomorphism h of $\mathcal{A}$ into $\mathcal{B}$ *is an order-sorted Σ-homomorphism $h : \mathcal{A} \to \mathcal{B}$ for which there exists an order-sorted Σ-homomorphism $h' : \mathcal{B} \to \mathcal{A}$ such that $h' \circ h = \{id_s : A_s \to A_s \mid s \in S\}$ and $h \circ h' = \{id_s : B_s \to B_s \mid s \in S\}$.*[6]

We write $\mathcal{A} \cong \mathcal{B}$ if there is an order-sorted Σ-isomorphism from $\mathcal{A}$ onto $\mathcal{B}$. We prove in [13] that $\cong$ is an equivalence relation over Σ-structures. We also prove that isomorphic Σ-structures satisfy exactly the same Σ-formulae.[7] As in the unsorted case, a crucial consequence of these results, which we use later, is that isomorphic order-sorted structures can always be identified.

If $\Sigma_1 = (S_1, \prec_1, F_1, P_1)$ and $\Sigma_2 = (S_2, \prec_2, F_2, P_2)$ are two order-sorted signatures, the *union* and the *intersection* of Σ_1 and Σ_2 are the order-sorted signatures defined as follows:

$$\begin{aligned} \Sigma_1 \cup \Sigma_2 &= (S_1 \cup S_2, \prec_1 \cup \prec_2, F_1 \cup F_2, P_1 \cup P_2) \\ \Sigma_1 \cap \Sigma_2 &= (S_1 \cap S_2, \prec^*_1 \cap \prec^*_2, F_1 \cap F_2, P_1 \cap P_2)\,. \end{aligned}$$

It is easy to see that $\Sigma_1 \cup \Sigma_2$ and $\Sigma_1 \cap \Sigma_2$ are well defined, and thus are indeed order-sorted signatures. We will consider only unions of signatures that are *conservative* in a strong sense with respect to subsort overloading.

Definition 5 (Conservative Union of Signatures). *The order-sorted signature $\Sigma = (S, \prec, F, P)$ is a* conservative union *of an order-sorted signature $\Sigma_1 = (S_1, \prec_1, F_1, P_1)$ and an order-sorted signature $\Sigma_2 = (S_2, \prec_2, F_2, P_2)$ iff $\Sigma = \Sigma_1 \cup \Sigma_2$ and the following hold:*

1. *For all $p_{w'} \in P_i$ and $p_{w''} \in P_j$ with $\{i, j\} \subseteq \{1, 2\}$ and $w' \sim^* w''$, there is a $p_w \in P_i \cap P_j$ such that $w' \prec^*_i w \sim^*_j w''$ or $w'' \prec^*_j w \sim^*_i w'$.*
2. *For all $f_{w',s'} \in F_i$ and $f_{w'',s''} \in F_j$ with $\{i, j\} \subseteq \{1, 2\}$ and $w's' \sim^* w''s''$, there is a $f_{w,s} \in F_i \cap F_j$ such that $w's' \prec^*_i ws \sim^*_j w''s''$ or $w''s'' \prec^*_j ws \sim^*_i w's'$.*

The idea of the definition above is that if two symbols are subsort overloaded in the union signature, that is only because either they were already subsort overloaded in one of the component signatures (case $i = j$ in Conditions 1 and

[6] Where id denotes the identity function.

[7] Note that these two facts are not granted for a sorted logic. For instance, invariance of satisfiability under isomorphism does not hold in general for the logic in [5].

2) or, when the two symbols belong to different component signatures, each was subsort overloaded in its signature with a same *connecting* symbol belonging to the shared signature (case $i \neq j$).

An *order-sorted Σ-theory* is a pair $T = (\Sigma, Ax)$ where Ax is a set of Σ-sentences. A *model* of T is a Σ-structure that models Ax. A set Φ of Σ-sentences is *T-satisfiable (resp. T-unsatisfiable)* if it is satisfied by some (resp. no) model of T. The *combination* of two order-sorted theories $T_1 = (\Sigma_1, Ax_1)$ and $T_2 = (\Sigma_2, Ax_2)$ is defined as $T_1 \cup T_2 = (\Sigma_1 \cup \Sigma_2, Ax_1 \cup Ax_2)$.

In this paper we consider for convenience expansions of order-sorted signatures to sets of new constants. Formally, we will fix a countably-infinite set $\mathcal{C}$ of *free constants*, symbols that do not occur in any of the symbols sets $\mathcal{F}$, $\mathcal{P}$, $\mathcal{S}$ and $\mathcal{X}$ defined earlier. Then, for every order-sorted signature $\Sigma = (S, \prec, F, P)$, we will denote by $\Sigma(\mathcal{C})$ the signature $\Sigma = (S, \prec, F \cup (\mathcal{C} \times \{\epsilon\} \times S), P)$.[8]

The *quantifier-free satisfiability* problem for an order-sorted Σ-theory T is the problem of determining whether a ground $\Sigma(\mathcal{C})$-formula is T-satisfiable.

As we will see, the decidability of the quantifier-free satisfiability problem is modular with respect to the union of order-sorted theories whenever the signatures of theories satisfy certain disjointness conditions and the theories are *stably infinite* with respect to the sorts they share.

Definition 6 (Stably Infinite Theory). *A Σ-theory T is* stably infinite with respect to *S' for some $S' \subseteq \Sigma^{\mathrm{S}}$ if every ground $\Sigma(\mathcal{C})$-formula φ that is T-satisfiable is satisfied by a $\Sigma(\mathcal{C})$-model $\mathcal{A}$ of T such that $|A_s| \geq \aleph_0$ for all $s \in S'$.*

We point out that the logic defined here is a proper extension of conventional many-sorted logic, obtainable from ours by considering only signatures with empty subsort relation $\prec$. All the results presented here then apply for instance to the many-sorted logics used by the verification systems described in [9, 6].

3 The Combination Method

In this section we present a method for combining decision procedures for order-sorted theories whose signatures may share sorts, but no function or predicate symbols. We will further impose the restriction that the union of the two signatures is conservative (cf. Definition 5). The method is closely modeled after the non-deterministic version of the Nelson-Oppen combination method (for unsorted theories) as described in [11] and [15], among others.

For the rest of this section, let $\Sigma_1 = (S_1, \prec_1, F_1, P_1)$ and $\Sigma_2 = (S_2, \prec_2, F_2, P_2)$ be two order-sorted signatures such that

1. $F_1 \cap F_2 = P_1 \cap P_2 = \emptyset$,
2. $\Sigma_1 \cup \Sigma_2$ is a conservative union of Σ_1 and Σ_2,
3. for all distinct $s, s' \in S_1 \cap S_2$, $s \not\prec_1^* s'$ and $s \not\prec_2^* s'$.

[8] All the signature-dependent notions we have introduced so far extend to signatures with free constants in the obvious way.

Condition 1 corresponds to the original restriction in the Nelson-Oppen method that the two theories share no function or predicate symbols. In our case, however, the restriction is on *decorated* symbols. This means, for instance, that we allow one signature to contain a symbol f_{w_1,s_1}, while the other contains a symbol f_{w_2,s_2}, provided that $w_1 s_1 \neq w_2 s_2$. By Condition 2, the two symbols become *ad hoc* overloaded in the union signature, because that condition implies that $w_1 s_1 \not\sim^* w_2 s_2$, where $\sim$ is the symmetric closure of $\prec\, = \,\prec_1 \cup \prec_2$. Note that Condition 2 and 3 are immediately satisfied in the many-sorted case, i.e., when both $\prec_1$ and $\prec_2$ are the empty relation.

The problem. We are interested in the quantifier-free satisfiability problem for a theory $T_1 \cup T_2$ where T_i is a Σ_i-theory, for $i = 1, 2$, and both T_1 and T_2 are stably infinite over $S_0 = S_1 \cap S_2$.

Here are two examples of theories satisfying (or not) the conditions above.

Example 7. Let T_1 be an order sorted version of linear rational arithmetic, with Σ_1 having the sorts Int and Rat, the subsorts $\mathsf{Int} \prec \mathsf{Rat}$, and the expected function and predicate symbols, say $0\colon \mathsf{Int}$, $1\colon \mathsf{Int}$, $+\colon \mathsf{Int} \times \mathsf{Int} \to \mathsf{Int}$, $+\colon \mathsf{Rat} \times \mathsf{Rat} \to \mathsf{Rat}$, $< \,: \mathsf{Int} \times \mathsf{Int}$, and so on.[9] Then let T_2' be the theory of a parametric datatype such as lists, with signature Σ_2' having the "parameter" sort Elem (for the list elements), the list sorts EList, NList (for empty and non-empty lists respectively), and List, the subsorts $\mathsf{EList}, \mathsf{NList} \prec \mathsf{List}$, and the expected function symbols, say, $[\,]\colon \mathsf{EList}$, $\mathsf{hd}\colon \mathsf{NList} \to \mathsf{Elem}$, $\mathsf{tl}\colon \mathsf{List} \to \mathsf{List}$, $\mathsf{cons}\colon \mathsf{Elem} \times \mathsf{List} \to \mathsf{NList}$.

Then consider a renaming T_2 of T_2' in which Elem is renamed as Rat, so that $T_1 \cup T_2$ then becomes a theory of rational lists. Where Σ_2 is the signature of T_2 and $S_0 = \{\mathsf{Rat}\}$, it is easy to see that Σ_1 and Σ_2 satisfy Conditions 1–3 above. The stable infiniteness of T_1 over S_0 is trivial because in all models of T_1 Int is infinite (as the theory entails that all successors of zero are pairwise distinct). As discussed in [13], the stable infiniteness of T_2 over S_0 is not hard to show.

Example 8. Let T_1 be as in Example 7. Then let T_2' be an order-sorted theory of arrays, with signature Σ_2' having the "parameter" sorts Index and Elem (for the array indexes and elements, respectively), the array sort Array, the subsorts $\mathsf{Index} \prec \mathsf{Elem}$, and the usual function symbols $\mathsf{select}\colon \mathsf{Array} \times \mathsf{Index} \to \mathsf{Elem}$ and $\mathsf{store}\colon \mathsf{Array} \times \mathsf{Index} \times \mathsf{Elem} \to \mathsf{Array}$. Then consider a renaming T_2 of T_2' in which Elem is renamed as Rat and Index as Int, so that $T_1 \cup T_2$ then becomes a theory of arrays with integer indeces and rational elements. Where Σ_2 is the signature of T_2, it is immediate that Σ_1 and Σ_2 do not satisfy Condition 3 above because the shared sorts, Int and Rat, are comparable.

While perfectly reasonable in practice, $T_1 \cup T_2$ is a combined theory that the combination method cannot accommodate at the moment (but see Section 5 for possible extensions in this direction).

We remark that a perhaps more natural combination of the two signatures would be the one in which no renamings are applied but Int becomes a subsort

[9] For readability, we use here a more conventional notation for decorated symbols, instead of $0_{\epsilon,\mathsf{Int}}$, $+_{\mathsf{Int\,Int},\mathsf{Int}}$, etc.

of Index and Rat a subsort of Elem. This kind of combination, however, is not achievable by a simple union of signatures and theories, and as such is out the scope of combination methods *a la* Nelson-Oppen.

The method. When the quantifier-free satisfiability problem for T_1 and for T_2 is decidable, we can decide the quantifier-free satisfiability problem for $T_1 \cup T_2$ by means of the combination method described below and consisting of four phases.

To simplify the presentation, and without loss of generality, we restrict ourselves to the $(T_1 \cup T_2)$-satisfiability of conjunctions of literals only.

First phase: Variable abstraction. Let Γ be a conjunction of ground $(\Sigma_1 \cup \Sigma_2)(\mathcal{C})$-literals. In this phase we convert Γ into a conjunction Γ' satisfying the following properties: (a) each literal in Γ' is either a $\Sigma_1(\mathcal{C})$-literal or a $\Sigma_2(\mathcal{C})$-literal, and (b) Γ' is $(T_1 \cup T_2)$-satisfiable if and only if so is Γ.

Properties (a) and (b) can be enforced with the help of new auxiliary constants from $\mathcal{C}$. For instance, in the simplest kind of transformation, Γ can be *purified* by applying to it to completion the following rewriting step, for all terms t of nominal sort $s \in S_0 = S_1 \cap S_2$ occurring in Γ that are not free constants: if t occurs as the argument of an non-equality atom in Γ, or occurs in an atom of the form $t \approx t'$ or $t' \approx t$ where t' is not a free constant, or occurs as a proper subterm of an atom of the form $t_1 \approx t_2$ or $t_2 \approx t_1$, then t is replaced by $c_{\epsilon,s}$ for some fresh $c \in \mathcal{C}$, and the equality $c_{\epsilon,s} \approx t$ is added to Γ. It is easy to see that this transformation satisfies the properties above.[10]

Second phase: Partition. Let Γ' be a conjunction of literals obtained in the variable abstraction phase. We now partition Γ' into two sets of literals Γ_1, Γ_2 such that, for $i = 1, 2$, each literal in Γ_i is a $\Sigma_i(\mathcal{C})$-literal. A literal with an atom of the form $c_{\epsilon,s} \approx c'_{\epsilon,s'}$ with $c, c' \in \mathcal{C}$, which is both a $\Sigma_1(\mathcal{C})$- and a $\Sigma_2(\mathcal{C})$-literal, can go arbitrarily in either Γ_1 or Γ_2.

Third phase: Decomposition. Let $\Gamma_1 \cup \Gamma_2$ be the conjunction of literals obtained in the variable abstraction phase. The only decorated symbols shared by Γ_1 and Γ_2, if any, are decorated free constants of a shared sort—constants of the form $c_{\epsilon,s}$ with $c \in \mathcal{C}$ and $s \in S_0$. For all shared sorts $s \in S_0$, let C_s be the set of constants of sort s shared by Γ_1 and Γ_2. We choose nondeterministically a family $E = \{E_s \subseteq C_s \times C_s \mid s \in S_0\}$ of equivalence relations E_s.

Intuitively, in this phase we guess for each pair of shared constant in C_s, whether they denote the same individual or not. In essence, partitioning the shared free constants into sorted classes and considering identifications only of constants of the same sort is the only difference with respect to the unsorted version of the Nelson-Oppen method, where all pairs of constants are considered for possible identification.

Fourth phase: Check. Given the equivalence relations $E = \{E_s \mid s \in S_0\}$ guessed in the decomposition phase, this phase consists of the following steps:

[10] But see [12], among others, for a more practical kind of abstraction process that minimizes the number of fresh constants introduced.

1. Construct the *arrangement* of $C = \{C_s \mid s \in S_0\}$ induced by E, defined by

$$\begin{aligned} arr(C, E) = \{u \approx v \mid (u, v) \in E_s \text{ and } s \in S_0\} \cup \\ \{u \not\approx v \mid (u, v) \in (C_s^2 \setminus E_s) \text{ and } s \in S_0\} \, . \end{aligned}$$

2. if $\Gamma_1 \cup arr(C, E)$ is T_1-satisfiable and $\Gamma_2 \cup arr(C, E)$ is T_2-satisfiable, output `succeed`; else output `fail`.

In Section 4 we will prove that this combination method is sound and complete in the following sense. If there exists an arrangement $arr(C, E)$ of C for which the check phase outputs `succeed`, then Γ is $(T_1 \cup T_2)$-satisfiable. If instead the check phase outputs `fails` for every possible arrangement $arr(C, E)$ of C, then Γ is $(T_1 \cup T_2)$-unsatisfiable.

4 Correctness of the Method

To prove the combination method correct, we first need a couple of basic model-theoretic results. The first is an order-sorted version of the Downward Löwenheim-Skolem Theorem, whose proof can be found in [13]. The second is an order-sorted version of a general combination result given in [12, 15] for unsorted theories.

Theorem 9 (Order-sorted Löwenheim-Skolem Theorem). *Where Σ is an order-sorted signature, let Φ be a satisfiable set of Σ-formulae, and let $\mathcal{A}$ be a Σ-structure satisfying Φ. Then there exists a Σ-structure $\mathcal{B}$ satisfying Φ such that $|A_s| \geq \aleph_0$ implies $|B_s| = \aleph_0$, for each sort $s \in \Sigma^{\mathrm{S}}$.*

If $\mathcal{A}$ is an order-sorted Σ_1-structure and $\Sigma_0 = \Sigma_1 \cap \Sigma_2$ for some signature Σ_2, we denote by $\mathcal{A}^{\Sigma_0}$ the Σ_0-structure with domains $\{A_s \mid s \in {\Sigma_0}^{\mathrm{S}}\}$ that interprets the function and predicate symbols of Σ_0 exactly as $\mathcal{A}$ does.

Theorem 10 (Order-Sorted Combination Theorem). *Let $\Sigma_A = (S_A, \prec_A, F_A, P_A)$ and $\Sigma_B = (S_B, \prec_B, F_B, P_B)$ are two order-sorted signatures, and let Φ_A and Φ_B be two sets of Σ_A- and Σ_B-sentences, respectively. When $\Sigma_A \cup \Sigma_B$ is a conservative union of Σ_A and Σ_B, $\Phi_A \cup \Phi_B$ is satisfiable iff there is a Σ_A-structure $\mathcal{A}$ satisfying Φ_A and a Σ_B-structure $\mathcal{B}$ satisfying Φ_B such that $\mathcal{A}^{\Sigma_A \cap \Sigma_B} \cong \mathcal{B}^{\Sigma_A \cap \Sigma_B}$.*

Proof. Let $\Sigma_C = \Sigma_A \cap \Sigma_B$, and $\Sigma = \Sigma_A \cup \Sigma_B = (S, \prec, F, P) = (S_A \cup S_B, \prec_A \cup \prec_B, F_A \cup F_B, P_A \cup P_B)$.

Next, assume that $\Phi_A \cup \Phi_B$ is satisfiable, and let $\mathcal{D}$ be a Σ-structure satisfying $\Phi_A \cup \Phi_B$. Then, by letting $\mathcal{A} = \mathcal{D}^{\Sigma_A}$ and $\mathcal{B} = \mathcal{D}^{\Sigma_B}$, we clearly have that $\mathcal{A}$ satisfies Φ_A, $\mathcal{B}$ satisfies Φ_B, and $\mathcal{A}^{\Sigma_C} \cong \mathcal{B}^{\Sigma_C}$.

Vice versa, suppose there exists a Σ_A-structure $\mathcal{A}$ satisfying Φ_A and a Σ_B-structure $\mathcal{B}$ satisfying Φ_B such that $\mathcal{A}^{\Sigma_C} \cong \mathcal{B}^{\Sigma_C}$. Then, as observed in Section 2, we can assume with no loss of generality that $\mathcal{A}^{\Sigma_C} = \mathcal{B}^{\Sigma_C}$. We define a Σ-structure $\mathcal{D}$ by letting for each $s \in S$, $f_{w,s} \in F$, and $p_w \in P$:

$$D_s = \begin{cases} A_s\,, & \text{if } s \in S_A \\ B_s\,, & \text{if } s \in S_B \setminus S_A \end{cases}$$

$$f^{\mathcal{D}}_{w,s} = \begin{cases} f^{\mathcal{A}}_{w,s}\,, & \text{if } f_{w,s} \in F_A \\ f^{\mathcal{B}}_{w,s}\,, & \text{if } f_{w,s} \in F_B \setminus F_A \end{cases} \qquad p^{\mathcal{D}}_w = \begin{cases} p^{\mathcal{A}}_w\,, & \text{if } p_w \in P_A \\ p^{\mathcal{B}}_w\,, & \text{if } p_w \in P_B \setminus P_A \end{cases}$$

Because $\mathcal{A}^{\Sigma_C} = \mathcal{B}^{\Sigma_C}$, it is clear that $\mathcal{D}$ is well defined as a many-sorted Σ-structure. To show that $\mathcal{D}$ is also a well defined order-sorted Σ-structure, we start by showing that in $\mathcal{D}$ the denotation of a sort includes the denotations of its subsorts.

In fact, let $s, s' \in S$ be two distinct sorts such that $s \prec^* s'$. Since $\prec\; = \;\prec_A \cup \prec_B$ (and S is finite), there is a sequence $s = s_0, s_1, \ldots, s_n, s_{n+1} = s'$ such that for all $i = 0, \ldots, n$ either $s_i \prec_A s_{i+1}$ or $s_i \prec_B s_{i+1}$. It is enough to show then that $D_{s_i} \subseteq D_{s_{i+1}}$ for all $i = 0, \ldots, n$. Recall that, since $\mathcal{A}^{\Sigma_C} = \mathcal{B}^{\Sigma_C}$, $D_{s_i} = A_{s_i} = B_{s_i}$ whenever $s_i \in S_A \cap S_B$. Now, if $s_i \prec_A s_{i+1}$ we have by construction of $\mathcal{D}$ and definition of $\mathcal{A}$ that $D_{s_i} = A_{s_i} \subseteq A_{s_{i+1}} = D_{s_{i+1}}$. (Similarly, if instead $s_i \prec_B s_{i+1}$.)

It remains to show that $\mathcal{D}$ respects the subsort overloading of function and predicate symbols.[11] This is true for every two symbols of F_A or of P_A because (i) $\mathcal{D}^{\Sigma_A} = \mathcal{A}$, trivially, and (ii) since $\Sigma = \Sigma_A \cup \Sigma_B$ is a conservative union of Σ_A and Σ_B, if two symbols are subsort overloaded in Σ then they are subsort overloaded in Σ_A. The argument is symmetric for the symbols of F_B and P_B. Finally, $\mathcal{D}$ respects the possible subsort overloading of a symbol of F_A (P_A) and a symbol of F_B (P_B) because again Σ is a conservative union of Σ_A and Σ_B, and $\mathcal{A}$ and $\mathcal{B}$ agree on their shared symbols.

In fact, for illustration, assume that $p_{w'} \in P_A$, $p_{w''} \in P_B \setminus P_A$, and $w' \sim^* w''$. Then, by Definition 5, there is a $p_w \in P_A \cap P_B$ such that $w' \prec^*_A w$ and $w \sim^*_B w''$, say. Let $\mathbf{d} \in D_{w'} \cap D_{w''}$. We show that $\mathbf{d} \in p^{\mathcal{D}}_{w'}$ iff $\mathbf{d} \in p^{\mathcal{D}}_{w''}$. Observing that $D_{w'} = A_{w'} \subseteq A_w = B_w$ and $B_{w''} = D_{w''}$ by construction of $\mathcal{D}$ and definition of $\mathcal{A}$ and $\mathcal{B}$, it is not difficult to see that $\mathbf{d} \in A_{w'} \cap A_w$ and $\mathbf{d} \in B_w \cap B_{w''}$. Then $\mathbf{d} \in p^{\mathcal{D}}_{w'}$ iff $\mathbf{d} \in p^{\mathcal{A}}_{w'}$ (by construction of $\mathcal{D}$) iff $\mathbf{d} \in p^{\mathcal{A}}_w$ (as $w' \sim^*_A w$ and $\mathbf{d} \in A_{w'} \cap A_w$) iff $\mathbf{d} \in p^{\mathcal{B}}_w$ (as $p^{\mathcal{A}}_w = p^{\mathcal{B}}_w$ by $\mathcal{A}^{\Sigma_A \cap \Sigma_B} = \mathcal{B}^{\Sigma_A \cap \Sigma_B}$) iff $\mathbf{d} \in p^{\mathcal{B}}_{w''}$ (as $w \sim^*_B w''$ and $\mathbf{d} \in B_w \cap B_{w''}$) iff $\mathbf{d} \in p^{\mathcal{D}}_{w''}$ (by construction of $\mathcal{D}$). The other cases are proven similarly.

Now, given that $\mathcal{D}$ is well defined, and that $\mathcal{D}^{\Sigma_A} = \mathcal{A}$ and $\mathcal{D}^{\Sigma_B} = \mathcal{B}$ by construction, it is immediate that $\mathcal{D}$ satisfies $\Phi_A \cup \Phi_B$. □

Let us now consider again the order-sorted signatures Σ_1, Σ_2 and the theories T_1, T_2 from Section 3.

[11] That is, $f^{\mathcal{D}}_{w,s}(\mathbf{d}) = f^{\mathcal{D}}_{w',s'}(\mathbf{d})$ for all $f_{w,s}, f_{w',s'} \in F$ with $ws \sim^* w's'$ and $\mathbf{d} \in D_w \cap D_{w'}$, and $\mathbf{d} \in p^{\mathcal{D}}_w$ iff $\mathbf{d} \in p^{\mathcal{D}}_{w'}$ for all $p_w, p_{w'} \in P$ with $w \sim^* w'$ and $\mathbf{d} \in D_w \cap D_{w'}$.

Theorem 11. *For $i = 1, 2$, let Φ_i be a set of $\Sigma_i(\mathcal{C})$-sentences. For each $s \in S_0$ let C_s be the set of decorated free constants $c_{\epsilon,s}$ shared by Φ_1 and Φ_2, with $c \in \mathcal{C}$. Then, $\Phi_1 \cup \Phi_2$ is satisfiable iff there exists a $\Sigma_1(\mathcal{C})$-structure $\mathcal{A}$ satisfying Φ_1 and a $\Sigma_2(\mathcal{C})$-structure $\mathcal{B}$ satisfying Φ_2 such that:*

(i) $|A_s| = |B_s|$*, for all* $s \in S_0$*;*
(ii) $u^{\mathcal{A}} = v^{\mathcal{A}}$ *if and only if* $u^{\mathcal{B}} = v^{\mathcal{B}}$*, for all* $u, v \in C_s$ *and* $s \in S_0$*.*

Proof. Let $\Sigma_0 = \Sigma_1 \cap \Sigma_2 = (S_0, \prec_0, F_0, P_0) = (S_1 \cap S_2, \prec_1^* \cap \prec_2^*, F_1 \cap F_2, P_1 \cap P_2)$. Clearly, if there exists a $(\Sigma_1 \cup \Sigma_2)(\mathcal{C})$-structure $\mathcal{D}$ satisfying $\Phi_1 \cup \Phi_2$, then the only if direction holds by letting $\mathcal{A} = \mathcal{D}^{\Sigma_1(\mathcal{C})}$ and $\mathcal{B} = \mathcal{D}^{\Sigma_2(\mathcal{C})}$.

Concerning the if direction, assume that there exists a $\Sigma_1(\mathcal{C})$-structure $\mathcal{A}$ satisfying Φ_1 and a $\Sigma_2(\mathcal{C})$-structure $\mathcal{B}$ satisfying Φ_2 such that both (i) and (ii) hold. We define a function family $h = \{h_s : {C_s}^{\mathcal{A}} \rightarrow {C_s}^{\mathcal{B}} \mid s \in S_0\}$ by letting $h_s(u^{\mathcal{A}}) = u^{\mathcal{B}}$, for every $u \in {C_s}^{\mathcal{A}}$ and $s \in S_0$. Note that each function h_s is well defined and bijective thanks to property (ii). As a consequence, we have that $|{C_s}^{\mathcal{A}}| = |{C_s}^{\mathcal{B}}|$ for all $s \in S_0$. By property (i) then, we can extend each function h_s to a bijective function $h'_s : A_s \rightarrow B_s$.

Let $\mathcal{C}_0$ be the set of all constants in $\{C_s \mid s \in S_0\}$. Since the signature $\Sigma_0(\mathcal{C}_0)$ has only constant symbols (from $\mathcal{C}_0$) and all of its sorts are pairwise disconnected, it is clear that the family $h' = \{h'_s : A_s \rightarrow B_s \mid s \in S_0\}$ is an order-sorted $\Sigma_0(\mathcal{C}_0)$-isomorphism of $\mathcal{A}^{\Sigma_0(\mathcal{C}_0)}$ into $\mathcal{B}^{\Sigma_0(\mathcal{C}_0)}$. Therefore, by Theorem 10 we obtain the existence of a $(\Sigma_1 \cup \Sigma_2)(\mathcal{C})$-structure $\mathcal{D}$ satisfying $\Phi_1 \cup \Phi_2$. □

Proposition 12 (Correctness). *Let Γ_1 and Γ_2 be conjunctions of ground $\Sigma_1(\mathcal{C})$-literals and ground $\Sigma_2(\mathcal{C})$-literals, respectively, and for all shared sorts $s \in S_0$, let C_s be the set of free constants shared by Γ_1 and Γ_2. The following are equivalent:*

1. *The conjunction $\Gamma_1 \cup \Gamma_2$ is $(T_1 \cup T_2)$-satisfiable.*
2. *There is a family $E = \{E_s \mid s \in S_0\}$ of equivalence relations E_s over C_s such that $\Gamma_i \cup arr(V, E)$ is T_i-satisfiable, for $i = 1, 2$.*

Proof. We only prove that 2 implies 1, as the other direction is straightforward. Assume there exists a family $E = \{E_s \mid s \in S_0\}$ of equivalence relations E_s over C_s such that $\Gamma_i \cup arr(V, E)$ is T_i-satisfiable, for $i = 1, 2$.

Since T_1 is stably infinite with respect to S_0, we can assume that $\Gamma_1 \cup arr(V, E)$ is satisfied by a model $\mathcal{A}$ of T_1 such that A_s is infinite for each $s \in S_0$. By the Order-sorted Löwenheim-Skolem Theorem we can further assume that A_s is *countably* infinite for each $s \in S_0$. Similarly, we can assume that $\Gamma_2 \cup arr(V, E)$ is satisfied by a model $\mathcal{B}$ of T_2 such that B_s is countably infinite for each $s \in S_0$. But then we obtain a model $\mathcal{A}$ of $T_1 \cup \Gamma_1$ and a model $\mathcal{B}$ of $T_2 \cup \Gamma_2$ such that (i) $|A_s| = |B_s|$, for each $s \in S_0$, and (ii) $u^{\mathcal{A}} = v^{\mathcal{A}}$ iff $u^{\mathcal{B}} = v^{\mathcal{B}}$, for each $u, v \in C_s$ and $s \in S_0$. By Theorem 11 it follows that $(T_1 \cup \Gamma_1) \cup (T_2 \cup \Gamma_2)$ is satisfiable, which is equivalent to saying that $\Gamma_1 \cup \Gamma_2$ is $(T_1 \cup T_2)$-satisfiable. □

Combining Proposition 12 with the observation that the nondeterminism of the decomposition phase of the sorted Nelson-Oppen method is finitary, we obtain the following modular decidability result for order-sorted theories T_1 and T_2 defined as in Section 3.

Theorem 13 (Modular Decidability). *If the quantifier-free satisfiability problems of T_1 and of T_2 are decidable, then the quantifier-free satisfiability problem of $T_1 \cup T_2$ is also decidable.*

5 Conclusions and Further Research

We addressed the problem of modularly combining order-sorted first-order theories and their decision procedures. For that, we first defined a fairly general version of order-sorted logic. Then we presented and proved correct a method for combining decision procedures for two order-sorted theories that have no function or predicate symbols in common and are stably infinite with respect to a set of shared, disconnected sorts.

The method is a direct lifting to the given order-sorted logic of the Nelson-Oppen method for combining theories in (unsorted) first-order logic. The main difference with the unsorted version is that the introduction of sorts helps reduce the nondeterminism of the decomposition phase—because the guessing of equalities between shared constant is limited to constants with the same nominal sort—and allows one to limit the stable infiniteness requirement to just the shared sorts.

We used the assumption that the shared sorts are disconnected in order to obtain a method that is as close as possible to the Nelson-Oppen method for unsorted logic. When the shared sorts are connected, the combination problem becomes considerably more complex model-theoretically, and consequently so does any corresponding combination method. More in detail, consider the case of two theories T_1 and T_2 sharing two sorts s_1, s_2, with $s_1 \prec^* s_2$ (in both theories), and assume that u is a shared free constant of nominal sort s_2. Then, in a combination method for T_1 and T_2, the component decision procedures also need to share the information on whether u "is in s_1" or not—that is, whether u could be interpreted as an element of the set denoted by s_1 or not. Thinking of the problem in terms of Venn diagrams for the denotations of the sorts, a combination procedure also has to guess the portion of the diagram to which u belongs, and generate a *sort membership* constraint to that extent. Such constraints are easily expressible in our logic—to say that u is [not] in s_1, one simply writes $[\neg](\exists x_{s_1}\, x_{s_1} \approx u)$—but involve quantifiers.

Clearly, membership constraints increase the complexity of the combination procedure because there is much more to guess. Furthermore, since some of added constraints are (inherently) non-quantifier-free, they add to the complexity of the component decision procedures as well. Finally, the requirements that the two component theories be stably infinite over their shared sorts is not enough anymore—at least if one wants to use an extension of the proofs given here.[12]

Another limitation of the current method is that it does not apply to component theories that share function or predicate symbols. The problem of extending the Nelson-Oppen method to theories with symbols in common has recently received much attention [3, 4, 10, 12, 14]. Concurrently with the work presented

[12] See [13] for a discussion on this last point.

here, the specific approach of [3, 4] has been adapted in [2], with comparable results, to many-sorted logic (with no subsorts). An important direction for future research then would be to see how those results, which allow shared symbols but no subsorts, can be combined with the ones presented here, which allow subsorts but no shared function or predicate symbols.

Acknowledgments. We would like to thank José Meseguer for his insightful comments and suggestions on the order-sorted logic presented here.

The second author was supported in part by the project GECCOO in the context of the French national research program *ACI Sécurité Informatique*.

References

1. D. Detlefs, G. Nelson, and J. B. Saxe. Simplify: A theorem prover for program checking. Technical Report HPL-2–3-148, HP Laboratories, Palo Alto, CA, 2003.
2. V. Ganesh, S. Berezin, C. Tinelli, and D. Dill. Combination results for many sorted theories with overlapping signatures. Technical report, Department of Computer Science, Stanford University, 2004.
3. S. Ghilardi. Quantifier elimination and provers integration. In I. Dahn and L. Vigneron, editors, *First Order Theorem Proving*, volume 86.1 of *Electronic Notes in Theoretical Computer Science*. Elsevier, 2003.
4. S. Ghilardi. Model theoretic methods in combined constraint satisfiability. *Journal of Automated Reasoning*, 2004. (To appear).
5. J. A. Goguen and J. Meseguer. Order-sorted algebra I: Equational deduction for multiple inheritance, overloading, exceptions and partial operations. *Theoretical Computer Science*, 105(2):217–173, 1992.
6. F. Maric and P. Janičić. ARGO-LIB: A generic platform for decision procedures. In *International Joint Conference on Automated Reasoning*, Lecture Notes in Computer Science. Springer, 2004.
7. J. Meseguer. Membership algebra as a logical framework for equational specification. In *Recent Trends in Algebraic Development Techniques*, volume 1376 of *Lecture Notes in Computer Science*, pages 18–61. Springer, 1998.
8. G. Nelson and D. C. Oppen. Simplification by cooperating decision procedures. *ACM Transactions on Programming Languages and Systems*, 1(2):245–257, 1979.
9. A. Stump, C. W. Barrett, and D. L. Dill. CVC: A cooperating validity checker. In E. Brinksma and K. G. Larsen, editors, *Computer Aided Verification*, volume 2404 of *Lecture Notes in Computer Science*, pages 500–504, 2002.
10. C. Tinelli. Cooperation of background reasoners in theory reasoning by residue sharing. *Journal of Automated Reasoning*, 30(1):1–31, 2003.
11. C. Tinelli and M. T. Harandi. A new correctness proof of the Nelson-Oppen combination procedure. In F. Baader and K. U. Schulz, editors, *Frontiers of Combining Systems*, volume 3 of *Applied Logic Series*, pages 103–120. Kluwer, 1996.
12. C. Tinelli and C. Ringeissen. Unions of non-disjoint theories and combinations of satisfiability procedures. *Theoretical Computer Science*, 290(1):291–353, 2003.
13. C. Tinelli and C. G. Zarba. Combining decision procedures for sorted theories. Technical Report 04-01, The University of Iowa, 2004.
14. C. G. Zarba. C-tableaux. Technical Report RR-5229, INRIA, 2004.
15. C. G. Zarba. *The Combination Problem in Automated Reasoning*. PhD thesis, Stanford University, 2004.