Combining Decision Procedures for Theories in Sorted Logics

Cesare Tinelli
tinelli@cs.uiowa.edu

Calogero G. Zarba[1]
zarba@theory.stanford.edu

Department of Computer Science
The University of Iowa

Report No. 04-01

February 2004

[1]Address: Department of Computer Science, Stanford University, Gates Building, Stanford, CA 94305 – USA.

Combining Decision Procedures for Theories in Sorted Logics

Cesare Tinelli
Department of Computer Science
The University of Iowa
14 MacLean Hall, Iowa City, IA 52242 – USA
tinelli@cs.uiowa.edu

Calogero G. Zarba
Department of Computer Science
Stanford University
Gates Building, Stanford, CA 94305 – USA
zarba@theory.stanford.edu

February 2004

Revised: May 9, 2004

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 it is 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 standpoint, the extension is relevant because it provides a rigorous foundation for the application of the method in a sorted setting. From a practical standpoint, the extension has the considerable added benefits that in a sorted setting the method's preconditions become easier to satify in practice, and the method's nondeterminism is generally reduced.

Keywords: Combination of decision procedures, order-sorted logics, the Nelson-Oppen method.

Contents

1 Introduction

The problem of combining decision procedures for logical theories arises in many areas of logic in computer science, such as constraint solving, automated deduction, term rewriting, modal logics, and description logics. 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 [NO79]. This method is at the heart of the verification systems CVC [SBD02, BB04], ARGO-LIB [MJ04] EVES [CKM+91], SDVS [LFMM92], and SIMPLIFY [DNS03], 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:

- T is logically equivalent to $T_1 \cup T_2$;
- the signatures of T_1 and T_2 are disjoint;
- T_1 and T_2 are both stably infinite.[1]

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, such as CVC or ARGO-LIB, have a sorted input language. The choice of a sorted language is most natural for verification applications, which deal with properties of basic data types, such as integers, reals, lists, arrays, binary strings, and so on. 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 as is 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 it is 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.[2] 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. In fact, if two sorted signatures Σ_1 and Σ_2 do not even share sorts, 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

[1] A theory T is stably infinite if every quantifier-free formula satisfiable in a model of T is satisfiable in an infinite model of T.

[2] Except for the equality symbol, which however is a logical symbol in first-order logic with equality.

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 to the order-sorted setting. 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 methods (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* in each component signature, that is, 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 [GM92].

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 [TR03, Zar04] to prove the correctness of the (unsorted) Nelson-Oppen method. We conclude the paper in Section 5 with some directions for further research.

2 The Logic

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 [GM92]. We will mostly follow the notation used in [GM92]. The logic we present here is inspired by the order-sorted equational logic proposed by Meseguer in Section 11 of [Mes98] as a successor of the logic in [GM92]. To simplify the presentation of our logic we introduce it in two steps, by first defining a basic many-sorted logic (with unordered sorts), and then extending that logic to accommodate subsorts and, consequently, the subsort overloading of function and predicate symbols.

For reasons we explain later, our approach will differ from more conventional ones in that our logic uses a vocabulary of *decorated* symbols, that is, function and predicate symbols that carry a sort declaration explicitly in them.

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

2.1 A Many-sorted Logic with Decorated Symbols

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 a countably-infinite set $\mathcal{F}$ of *function* symbols, a countably-infinite set $\mathcal{P}$ of *predicate* symbols, and a countably-infinite set $\mathcal{S}$ of *sort* symbols.[4] We further fix a countably-infinite 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 $(x, s) \in \mathcal{X} \times \mathcal{S}$.

A *many-sorted (decorated) signature* Σ is a tuple $\Sigma = (S, F, P)$ 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, and $P \subseteq (\mathcal{P} \times S^*)$ is a set of decorated predicate symbols. When convenient, we will write Σ^{S} for S, Σ^{F} for F, and Σ^{P} for P. For simplicity, in this paper, we will consider only signatures with a finite set of sorts.

A *many-sorted (decorated)* Σ*-term of sort* s *(over a set* X *of decorated variables)* is a well sorted term built in the usual way, but out of decorated symbols from F and decorated variables from $\mathcal{X} \times S$. More precisely, the set of $\mathcal{T}_s(\Sigma, X)$ *of many-sorted* Σ*-terms of sort* s *over the variables* $X \subseteq \mathcal{X}$ is defined as follows by structural induction:

- every decorated variable $x_s \in (X \times S)$ is in $\mathcal{T}_s(\Sigma, X)$;
- if $t_1, \ldots, t_n$ are in $\mathcal{T}_{s_1}(\Sigma, X), \ldots, \mathcal{T}_{s_n}(\Sigma, X)$, respectively, for some $s_1, \ldots, s_n \in \Sigma^{\mathrm{S}}$, and $f_{s_1 \cdots s_n, s} \in \Sigma^{\mathrm{F}}$, then the word $f_{s_1 \cdots s_n, s} t_1 \cdots t_n$, written $f_{s_1 \cdots s_n, s}(t_1, \ldots, t_n)$ for clarity, is in $\mathcal{T}_s(\Sigma, X)$.

We denote by $\mathcal{T}_s(\Sigma, X)$ the set of such terms. Many-sorted atomic Σ-formulae are also defined as expected: a Σ*-atom* (over a set X of decorated variables) is either an expression of the form $p_{s_1 \cdots s_n}(t_1, \ldots, t_n)$ where $p_{s_1 \cdots s_n} \in \Sigma^{\mathrm{P}}$ and $t_i \in \mathcal{T}_{s_i}(\Sigma, X)$ for $i = 1, \ldots, n$, or one of the form $t_1 \approx t_2$ where $t_1, t_2 \in \mathcal{T}_s(\Sigma, X)$ for some $s \in \Sigma^{\mathrm{S}}$. Many-sorted (first-order) formulae and sentences are defined as usual, but with the difference that quantifiers bind decorated variables. For simplicity, and without loss of generality, we will consider only formulae built with the $\neg$ and $\wedge$ connectives and the $\exists$ quantifier symbol.

While decorated terms/predicates 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, *ad hoc* overloading in the usual sense is not a problem as there is no need of such a notion with decorated terms. We still get the same flexibility as usual overloading however because we can use, say, the same symbol f in a decorated constant $f_{\epsilon,s}$, in another constant $f_{\epsilon,s'}$, in a decorated function symbol f_{s_1,s_2}, or even in a decorate predicate symbol f_w.

Furthermore, and more importantly, with full decoration of symbols sort inference is trivial, terms have a unique sort, and unions and intersections of sorted signatures, crucial operations in combination settings, can be defined in a straightforward way. In fact, if

[4] For our purposes, $\mathcal{F}$, $\mathcal{P}$ or $\mathcal{S}$ need not be pairwise disjoint—they could even coincide.

$\Sigma_1 = (S_1, F_1, P_1)$ and $\Sigma_2 = (S_2, F_2, P_2)$ are two sorted signatures in the sense above, then we simply define $\Sigma_1 \cup \Sigma_2$ as $(S_1 \cup S_2, F_1 \cup F_2, P_1 \cup P_2)$, without having to worry about unintended overloading of symbols, for instance. Similarly, we simply define $\Sigma_1 \cap \Sigma_2$ as $(S_1 \cap S_2, F_1 \cap F_2, P_1 \cap P_2)$. The intersection of signatures is not as trivial to define in more usual definitions of many-sorted logic, where signatures consist of undecorated symbols and mappings of these symbols to strings of sorts. There the mappings themselves have to be "intersected" appropriately. In contrast to other approaches, defining the union and intersection operators as above always yield well-defined sorted signatures, as one can easily verify.

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 in computer science practice when working with sorted languages, but that are not relevant for the essence of our combination results. In a sense, the situation here is analogous to the difference between abstract syntax and concrete syntax in the programming languages literature. A sorted logic with decorated symbols can be seen as the abstract version of a more conventional sorted logic, after all parsing issues have been resolved and abstracted away.

For every many-sorted signature $\Sigma = (S, F, P)$, 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 predicate 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 $A_{s_1} \times \cdots \times A_{s_n}$.[5] 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 $p^{\mathcal{A}}_w \subseteq A_w$. The support for *ad hoc* overloading of function symbols comes from the fact that the interpretations $f^{\mathcal{A}}_{w,s}$ and $f^{\mathcal{A}}_{w',s'}$, say, need not coincide if ws and $w's'$ are different. (Similarly for predicate symbols interpretations.)

Definition 1 (Many-sorted Morphisms). Let $\Sigma = (S, F, P)$ be a many-sorted signature, and let $\mathcal{A}$ and $\mathcal{B}$ be two many-sorted Σ-structures. A *many-sorted Σ-homomorphism $h : \mathcal{A} \to \mathcal{B}$ of $\mathcal{A}$ into $\mathcal{B}$* is a 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 $i = 1, \ldots, n$,

$$(a_1, \ldots, a_n) \in p^{\mathcal{A}}_w \Longrightarrow (h_{s_1}(a_1), \ldots, h_{s_n}(a_n)) \in p^{\mathcal{B}}_w. \qquad \square$$

[5] With A_ϵ denoting an arbitrary singleton set.

2.2 An Order-sorted Logic with Decorated Symbols

An *order-sorted (decorated) signature* Σ is a tuple $\Sigma = (S, \prec, F, P)$ where (S, F, P) is a many-sorted decorated signature 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 $s_1 \prec^* s_2$. We say that two sorts 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 $w_1 \sim^* w_2$.)

For greater flexibility we do not insist, as in other approaches, that $\prec$ be a partial ordering. The relation $\prec$ is just the specification of the ordering $\prec^*$, the relation we really use. Even $\prec^*$ is not really a partial-ordering, but just a quasi-ordering. As we will see, semantically, (proper) cycles in $\prec^*$ are never a problem. They just mean that all the sorts participating in a cycle denote the same set. In the sorted logics in the literature cycles are essentially a syntactical problem, as they complicate sort inference for terms. In our case, however, thanks to the use of decorated symbols, sort inference is always trivial.

We say that two signatures $\Sigma_1 = (S_1, \prec_1, F_1, P_1)$ and $\Sigma_2 = (S_2, \prec_2, F_2, P_2)$ are *equivalent*, and write $\Sigma_1 \equiv \Sigma_2$, iff $(S_1, F_1, P_1) = (S_2, F_2, P_2)$ and $\prec_1^* = \prec_2^*$.[6] Two equivalent signatures differ only in the way they *present* their subsort relation, and nothing else. Since for all purposes of this paper equivalent signatures are interchangeable, we will tacitly identify them when needed.

Definition 2 (Compositions of signatures). 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* of Σ_1 and Σ_2 is the order-sorted signature

$$\Sigma_1 \cup \Sigma_2 = (S_1 \cup S_2, \prec_1 \cup \prec_2, F_1 \cup F_2, P_1 \cup P_2);$$

the *intersection* of Σ_1 and Σ_2 is the order-sorted signature

$$\Sigma_1 \cap \Sigma_2 = (S_1 \cap S_2, \prec_1^* \cap \prec_2^*, F_1 \cap F_2, P_1 \cap P_2)\,. \qquad \square$$

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.

For the rest of the section, we fix an order-sorted signature $\Sigma = (S, \prec, F, P)$ for convenience, and use it throughout.

Definition 3 (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 $\mathcal{T}_s(\Sigma, X)$.

[6] It is immediate that $\equiv$ is in fact an equivalence relation over signatures.

We denote by $\mathcal{T}(\Sigma, X)$ the set $\bigcup_{s\in S} \mathcal{T}_s(\Sigma, X)$ and 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. More precisely, we have the following.

Lemma 4. *Let $X \subseteq \mathcal{X}$ be a set of variables. For all Σ-terms t over X of nominal sort s' and all $s \in S$, $t \in \mathcal{T}_s(\Sigma, X)$ iff $s' \prec^* s$.* □

Order-sorted (first-order) Σ-formulae with free-variables X are defined as in the many-sorted case, with the difference that equational atoms are all and only those formulas of the form $t \approx t'$ where the nominal sorts of t and t' are connected.

Definition 5 (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 $A_w \cap A_{w'}$.[7]
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 [Mes98], where the subsort relation denotes set inclusion. As in [Mes98], 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.)

We will 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 are *ad-hoc overloaded* (in Σ). Similarly, we will say that two distinct decorated predicate symbols p_w and $p_{w'}$ of Σ are *subsort overloaded* (in Σ) if $w \sim^* w'$ and *ad-hoc overloaded* otherwise. As in previous works, the semantic of the logic will allow two ad-hoc overloaded function symbols to stand for completely different functions, whereas it will require *subsort overloaded* function symbols to stand for functions that agree on the intersection of their domains.

We point out that every many-sorted structure of signature (S, F, P) can be seen as an order-sorted structure of signature $\Sigma = (S, \prec, F, P)$ with $\prec$ empty, that is, as an order-sorted structure in which the subsort relation $\prec^*$ and the "connected" relation $\sim^*$ both coincide with the identity relation. A similar relation exists between many-sorted homomorphisms and order-sorted homomorphisms, defined later.

An order-sorted signature Σ_0 is a *subsignature* of Σ iff $\Sigma_0 \cap \Sigma \equiv \Sigma_0$. Let $\mathcal{A}$ be an order-sorted Σ-structure. If $\mathcal{A}$ is an order-sorted Σ-structure, the *reduct* of $\mathcal{A}$ to Σ_0, denoted

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

by $\mathcal{A}^{\Sigma_0}$, is the order-sorted 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.

For our combination purposes we will consider only combinations of signatures that are *conservative* in a strong sense with respect to subsort overloading. The idea is that if two symbols are subsort overloaded in the union signature, then that is only because either they were already subsort overloaded in one of the component signatures 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. More precisely, we will consider only combined signature from the class defined below.

Definition 6 (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 all of the following hold:

1. $\Sigma = \Sigma_1 \cup \Sigma_2$;
2. 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 $w' \prec_i^* w$ and $w \sim_j^* w''$ or $w'' \prec_j^* w$ and $w \sim_i^* w'$.
3. 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 $w's' \prec_i^* ws$ and $ws \sim_j^* w''s''$ or $w''s'' \prec_j^* ws$ and $ws \sim_i^* w's'$.

In Condition 2 above we consider at the same time the case in which two overloaded symbols belong to the same component signature ($i = j$) and the case in which they do not ($i \neq j$). In the first case, w might as well be w' or w''. (Similarly for Condition 3.)

In essence, conservative unions have no new subsort overloadings with respect to their component signatures. Examples of conservative unions are easy to generate. It might be more instructive then to consider some counter-examples instead.

Example 7. Let

$$\Sigma_1 = (S_1, \prec_1, F_1, P_1) = (\{s_1, s_2\}, \{\}, \{\}, \{p_{s_1}, p_{s_2}\}) \text{ and}$$
$$\Sigma_2 = (S_2, \prec_2, F_2, P_2) = (\{s_1, s_2\}, \{(s_1, s_2)\}, \{\}, \{\}) .$$

Then

$$\Sigma = \Sigma_1 \cup \Sigma_2 = (S, \prec, F, P) = (\{s_1, s_2\}, \{(s_1, s_2)\}, \{\}, \{p_{s_1}, p_{s_2}\})$$

is not a conservative union of Σ_1 and Σ_2 because $s_1 \sim^* s_2$ and, while p_{s_1} and p_{s_2} are both in P_1, it is not the case that $s_1 \sim_1^* s_2$. □

Example 8. Let

$$\Sigma_1 = (S_1, \prec_1, F_1, P_1) = (\{s_1, s_2\}, \{(s_1, s_2)\}, \{\}, \{p_{s_1}\}) \text{ and}$$
$$\Sigma_2 = (S_2, \prec_2, F_2, P_2) = (\{s_1, s_2\}, \{(s_1, s_2)\}, \{\}, \{p_{s_2}\}) .$$

Then

$$\Sigma = \Sigma_1 \cup \Sigma_2 = (S, \prec, F, P) = (\{s_1, s_2\}, \{(s_1, s_2)\}, \{\}, \{p_{s_1}, p_{s_2}\})$$

is not a conservative union of Σ_1 and Σ_2 because $s_1 \sim^* s_2$ but neither p_{s_1} nor p_{s_2} is in $P_1 \cap P_2$. □

Example 9. Let

$$\Sigma_1 = (S_1, \prec_1, F_1, P_1) = (\{s_1, s\}, \{(s_1, s)\}, \{\}, \{p_{s_1}\}) \text{ and}$$
$$\Sigma_2 = (S_2, \prec_2, F_2, P_2) = (\{s, s_2\}, \{(s, s_2)\}, \{\}, \{p_{s_2}\}).$$

Then

$$\Sigma = \Sigma_1 \cup \Sigma_2 = (S, \prec, F, P) = (\{s_1, s, s_2\}, \{(s_1, s), (s, s_3)\}, \{\}, \{p_{s_1}, p_{s_2}\})$$

is not a conservative union of Σ_1 and Σ_2 because $p_{s_1} \in P_1$, $p_{s_2} \in P_2$, $s_1 \sim^* s_2$ but $P_1 \cap P_2$ is empty. □

Definition 10 (Order-sorted Valuation). Let $\mathcal{A}$ be an order-sorted Σ-structure. Let $X \subseteq (\mathcal{X} \times S)$ be a set of decorated variables, and for all $s \in S$ let X_s be the set of all elements of X of the form x_s. A *valuation α of X in $\mathcal{A}$* is a family $\alpha = \{\alpha_s : X_s \to A_s \mid s \in S\}$.

For any $a \in A_s$, $x \in \mathcal{X}$ and $s \in S$, we denote by $\alpha[x_s \mapsto a]$ the valuation of $X \cup \{x_s\}$ in $\mathcal{A}$ that maps x_s to a and is otherwise identical to α. □

Where $\mathcal{A}$ is an order-sorted Σ-structure, $X \subseteq (\mathcal{X} \times S)$ a set of decorated variables, and α a valuation of X in $\mathcal{A}$, we call the pair $(\mathcal{A}, \alpha)$ an *order-sorted Σ-interpretation over X*. A Σ-interpretation $(\mathcal{A}, \alpha)$ over X induces a mapping $(\_)^{\mathcal{A},\alpha}$ of terms in $\mathcal{T}(\Sigma, X)$ into elements of $\mathcal{A}$, inductively defined as follows:

- $x_s^{\mathcal{A},\alpha} = \alpha_s(x_s)$ for all $x_s \in X$.
- $f_{s_1 \cdots s_n, s}(t_1, \ldots, t_n)^{\mathcal{A},\alpha} = f^{\mathcal{A}}_{s_1 \cdots s_n, s}(t_1^{\mathcal{A},\alpha}, \ldots, t_n^{\mathcal{A},\alpha})$ for all $f_{s_1 \cdots s_n, s} \in F$ and $t_i \in \mathcal{T}_{s_i}(\Sigma, X)$ with $i = 1, \ldots, n$.

It is easy to see that this definition is well defined, and in particular that if $f_{w,s}, f_{w',s'} \in F$, with $ws \sim^* w's'$, and $\mathbf{t}$ is a tuple of terms in both $\mathcal{T}_w(\Sigma, X)$ and $\mathcal{T}_{w'}(\Sigma, X)$, then $f_{w,s}(\mathbf{t})^{\mathcal{A},\alpha} = f_{w',s'}(\mathbf{t})^{\mathcal{A},\alpha}$. For ground (i.e. variable-free) terms t, the valuation α is irrelevant in determining the value $t^{\mathcal{A},\alpha}$. So we will abuse the notation and write just $t^{\mathcal{A}}$.

Satisfiability of formulae in an order-sorted Σ-interpretation is defined exactly as in the unsorted case.

Definition 11 (Satisfiability). Let $\mathcal{A}$ be an order-sorted Σ-structure, $X \subseteq (\mathcal{X} \times S)$ a set of decorated variables, α a valuation of X in $\mathcal{A}$, and φ a Σ-formula with free variables from X. The interpretation $(\mathcal{A}, \alpha)$ *satisfies* φ iff $(\mathcal{A}, \alpha)$ and φ are in the relation $\models$ inductively defined as follows:

- $(\mathcal{A}, \alpha) \models t_1 \approx t_2$ iff $t_1^{\mathcal{A},\alpha} = t_2^{\mathcal{A},\alpha}$;
- $(\mathcal{A}, \alpha) \models p_w(t_1, \ldots, t_n)$ iff $(t_1^{\mathcal{A},\alpha}, \ldots, t_n^{\mathcal{A},\alpha}) \in p_w^{\mathcal{A}}$;
- $(\mathcal{A}, \alpha) \models \neg\psi$ iff $(\mathcal{A}, \alpha) \not\models \psi$;
- $(\mathcal{A}, \alpha) \models \varphi_1 \wedge \varphi_2$ iff $(\mathcal{A}, \alpha) \models \varphi_1$ and $(\mathcal{A}, \alpha) \models \varphi_2$;
- $(\mathcal{A}, \alpha) \models \exists x_s\ \psi$ iff $(\mathcal{A}, \alpha[x_s \mapsto a]) \models \varphi$ for some $a \in A_s$. □

As usual, we say that a Σ-structure $\mathcal{A}$ *satisfies, or is a model of*, a Σ-sentence (i.e. a closed Σ-formula) φ iff any (or equivalently, every) Σ-interpretation $(\mathcal{A}, \alpha)$ satisfies φ. Similarly, we say that $\mathcal{A}$ is a model of a set Γ of Σ-sentences if $\mathcal{A}$ satisfies every sentence in Γ.

Definition 12 (Order-sorted Morphisms). Let $\mathcal{A}$ and $\mathcal{B}$ be two order-sorted Σ-structures. An *order-sorted Σ-homomorphism $h : \mathcal{A} \to \mathcal{B}$ of $\mathcal{A}$ into $\mathcal{B}$* is a many-sorted (S, F, P)-homomorphism such that

(*) for all $s, s' \in S$ with $s \sim^* s'$, the component maps 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 $h' \circ h$ is the identify function on A and $h \circ h'$ is the identity function on B.

It is a simple exercise to show that if h is a Σ-isomorphism of $\mathcal{A}$ into $\mathcal{B}$, then h_s is a bijection of A_s into B_s for all $s \in S$; moreover, for all $p_w \in P$ with $w = s_1 \cdots s_n$ and all $(a_1, \ldots, a_n) \in A_w$, $(a_1, \ldots, a_n) \in p_w^{\mathcal{A}}$ iff $(h_{s_1}(a_1), \ldots, h_{s_n}(a_n)) \in p_w^{\mathcal{B}}$.

[to do: note Venn diagrams and on on why isomorphisms are defined this way]

We point out that, although we will not need it here, the semantics of our logic can be also equipped with a notion of *embedding* analogous to the one used in unsorted first-order logic. It is enough to define a notion of substructure, similarly to the unsorted case, and then define an embedding from a structure $\mathcal{A}$ to a structure $\mathcal{B}$ as an order-sorted Σ-isomorphism of $\mathcal{A}$ into a substructure of $\mathcal{B}$.

Lemma 13. *For all order-sorted Σ-structures $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$ the following hold:*

1. *The family $\{id_s \mid s \in S\}$, where id_s is the identity function of A_s, is an order-sorted Σ-isomorphism of $\mathcal{A}$ into itself.*
2. *The family $h^{-1} = \{h_s^{-1} : B_s \to A_s \mid s \in S\}$ obtained by inverting component-wise an order-sorted Σ-isomorphism $h : \mathcal{A} \to \mathcal{B}$ is an order-sorted Σ-isomorphism of $\mathcal{B}$ into $\mathcal{A}$.*
3. *The component-wise composition $g \circ h = \{g_s \circ h_s : A_s \to C_s \mid s \in S\}$ of two order-sorted Σ-isomorphisms $h : \mathcal{A} \to \mathcal{B}$ and $g : \mathcal{B} \to \mathcal{C}$ is an order-sorted Σ-isomorphism of $\mathcal{A}$ into $\mathcal{C}$.* □

PROOF. Simple exercise. ∎

We write $\mathcal{A} \cong \mathcal{B}$ if there is an order-sorted Σ-isomorphism from $\mathcal{A}$ onto $\mathcal{B}$. By the lemma above it is immediate that $\cong$ is an equivalence relation over Σ-structures. Another crucial property of order-sorted Σ-isomorphisms is that the satisfiability relation is invariant under them.

Proposition 14. *Let $\mathcal{A}$ and $\mathcal{B}$ be two Σ-structures and assume that there is a Σ-isomorphism $h : \mathcal{A} \to \mathcal{B}$. Then, for all valuations α and Σ-formulae φ,*

$$(\mathcal{A}, \alpha) \models \varphi \quad \textit{iff} \quad (\mathcal{B}, h \circ \alpha) \models \varphi\,. \qquad \square$$

PROOF. Let $\beta = h \circ \alpha$. We first prove the claim, by structural induction on formulae, under the assumption that φ is quantifier-free.

($\varphi = p_w(t_1, \ldots, t_n)$) Let $w = s_1 \cdots s_n$. By induction on terms one can show for all $i = 1, \ldots, n$ that $t_i^{\mathcal{A},\alpha} \in A_{s_i}$ and $h_{s_i}(t_i^{\mathcal{A},\alpha}) = t_i^{\mathcal{B},\beta}$. As observed earlier, we then have that $(t_1^{\mathcal{A},\alpha}, \ldots, t_n^{\mathcal{A},\alpha}) \in p_w^{\mathcal{A}}$ iff $(t_1^{\mathcal{B},\beta}, \ldots, t_n^{\mathcal{B},\beta}) \in p_w^{\mathcal{B}}$. The claim then derives by definition of satisfiability of atomic formulae.

($\varphi = t_1 \approx t_2$) By well-sortedness there are $s_1, s_2 \in S$ such that $s_1 \sim^* s_2$ and $t_i \in \mathcal{T}_{s_i}(\Sigma, X)$ for $i = 1, 2$. Again, we have that $h_{s_i}(t_i^{\mathcal{A},\alpha}) = t_i^{\mathcal{B},\beta}$ for $i = 1, 2$. Assume that $(\mathcal{A}, \alpha) \models t_1 \approx t_2$, which means that $t_1^{\mathcal{A},\alpha} = t_2^{\mathcal{A},\alpha}$. Since s_1 and s_2 are connected, we have that $h_{s_1}(t_1^{\mathcal{A},\alpha}) = h_{s_2}(t_2^{\mathcal{A},\alpha})$. It follows that

$$t_1^{\mathcal{B},\beta} = h_{s_1}(t_1^{\mathcal{A},\alpha}) = h_{s_2}(t_2^{\mathcal{A},\alpha}) = t_2^{\mathcal{B},\beta}$$

and so $(\mathcal{B}, \beta) \models t_1 \approx t_2$. Now assume that $(\mathcal{B}, \beta) \models t_1 \approx t_2$, i.e., $t_1^{\mathcal{B},\beta} = t_2^{\mathcal{B},\beta}$. By Lemma 13(2), h^{-1} is a Σ-isomorphism of $\mathcal{B}$ into $\mathcal{A}$. As before, since s_1 and s_2 are connected, we then have that $h_{s_1}^{-1}(t_1^{\mathcal{B},\beta}) = h_{s_2}^{-1}(t_2^{\mathcal{B},\beta})$. It follows that

$$t_1^{\mathcal{A},\alpha} = h_{s_1}^{-1}(h_{s_1}(t_1^{\mathcal{A},\alpha})) = h_{s_1}^{-1}(t_1^{\mathcal{B},\beta}) = h_{s_2}^{-1}(t_2^{\mathcal{B},\beta}) = h_{s_2}^{-1}(h_{s_2}(t_2^{\mathcal{A},\alpha})) = t_2^{\mathcal{A},\alpha}$$

and so $(\mathcal{A}, \alpha) \models t_1 \approx t_2$.

($\varphi = \neg\psi$), ($\varphi = \varphi_1 \wedge \varphi_2$) Immediately by induction hypothesis and the definition of $\models$.

Now we show, again by structural induction, that the claim holds for arbitrary formulas. The base case in which φ is quantifier-free was already proven above. The cases in which φ has the form $\neg\psi$ or $\varphi_1 \wedge \varphi_2$ are again elementary. We show then only the case in which $\varphi = \exists x_s\, \psi$.

Assume that $(\mathcal{A}, \alpha) \models \varphi$. Then there in an $a \in A_s$ such that $(\mathcal{A}, \alpha[x_s \mapsto a]) \models \psi$. By induction hypothesis this implies that $(\mathcal{B}, h \circ (\alpha[x_s \mapsto a])) \models \psi$, or, equivalently, that $(\mathcal{B}, \beta[x_s \mapsto h_s(a)]) \models \psi$. It follows by definition of $\models$ that $(\mathcal{B}, \beta) \models \exists x_s\, \psi$. Now assume that $(\mathcal{B}, \beta) \models \varphi$. Then we can prove that $(\mathcal{A}, \alpha) \models \varphi$ in a similar way by using the isomorphism $h^{-1} : \mathcal{B} \to \mathcal{A}$. ∎

Like the equational logic proposed by Meseguer in [Mes98], with regards to the proposition above our order-sorted logic contrasts with the one given in [GM92], where invariance under isomorphism is achieved only by requiring the signature in question to be *locally-filtered*, i.e., such that each connected component of its $\prec^*$ relation contains a largest sort. In our case, no such restriction is necessary. This makes the logic more robust and also better suited to combination settings given that the property of being locally filtered is not modular with respect to the union of signatures.

As in the unsorted case, a crucial consequence of Proposition 14—which we will use later—is that for all purposes isomorphic order-sorted structures are indistinguishable. Hence, they can always be identified.

An *order-sorted* Σ*-theory* is a pair $T = (\Sigma, Ax)$ where Ax is a set of Σ*-sentences*, that is, closed Σ-formulae.

Definition 15. Let T be a Σ theory, φ a Σ-formula over the variables X, and Γ a set of Σ-formulae over X. The formula is T*-entailed* by Γ, written $\Gamma \models_T \varphi$, iff every Σ-interpretation over X that satisfies $T \cup \Gamma$ satisfies φ as well. The formula φ is T*-valid*, written $\models_T \varphi$, iff $\emptyset \models_T \varphi$. The set Γ is T*-satisfiable (resp.* T*-unsatisfiable)* if it is satisfied by some (resp. no) Σ-interpretation satisfying T. □

Definition 16 (Combinations of Σ-theories). 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 $\Sigma = (S, \prec, F \cup (\mathcal{C} \times \{\epsilon\} \times S), P)$. All the signature-dependent notions we have introduced so far extend to signatures with free constants in the obvious way.

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 their share sorts.

Definition 17 (Stably Infinite Theory). Let Σ be an order-sorted signature, and let $S \subseteq \Sigma^{\mathrm{S}}$. A Σ-theory T is *stably infinite with respect to* S if for 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 $s \in S$. □

This definition extends to the sorted case the definition of stable infiniteness used with the (unsorted) Nelson-Oppen method, according to which a theory T is stably infinite if every T-satisfiable ground formula is satisfied by an infinite model of T.

We point out that the logic defined in this subsection is a proper extension of conventional many-sorted logic, defined in the previous subsection. All the results presented in this paper then apply for instance to the many-sorted logics used by the verification systems described in [SBD02, MJ04].

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 6).

The method is closely modeled after the non-deterministic version of the Nelson-Oppen combination method (for unsorted theories) as described in [TH96] and [Zar04], 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 $s, s' \in S_1 \cap S_2$, $s \not\sim_1^* s'$ and $s \not\sim_2^* s'$.

By Condition 3 above, the restrictions of $\sim_1^*$ and of $\sim_2^*$ to $S_0 \times S_0$ coincide (with the identity relation). We will denote this common relation by $\sim_0^*$.

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, f becomes *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 reflexive, symmetric and transitive closure of $\prec = \prec_1 \cup \prec_2$. To see that, assume that instead $w_1 s_1 \sim^* w_2 s_2$. Then, by Definition 6, both signatures must contain a symbol $f_{w',s'}$ (for some $w_1 s_1$ such that $w_1 s_1 \sim_1^* w' s'$ and $w' s' \sim_2^* w_2 s_2$). But that contradicts Condition 1. 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.

We are interested in the quantifier-free satisfiability problem for a theory $T_1 \cup T_2$ where

- T_1 is a Σ_1-theory,
- T_2 is a Σ_2-theory,

- both T_1 and T_2 are stably infinite over S_0.

Here is an example of two theories satisfying the conditions above.

Example 18. 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 $0\colon \mathsf{Int}$, $1\colon \mathsf{Int}$, $+\colon \mathsf{Int} \times \mathsf{Int} \to \mathsf{Int}$, $+\colon \mathsf{Int} \times \mathsf{Rat} \to \mathsf{Rat}$, $< : \mathsf{Int} \times \mathsf{Int}$, and so on.[8] 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). The stable infiniteness of T_2 over S_0 is not difficult to show. One possible proof uses the fact that T is *convex* over the sort Rat[9] and the general facts that i) every ground formula satisfiable in T_2 is satisfiable in $T = T_2 \cup \{\exists x_s, y_s\ x \not\approx y\}$ and ii) all convex theories like T are stably infinite. The third point, can be shown, as in the unsorted case [BDS02], by means of a compactness argument.[10] □

In contrast to the previous one, here is an example of two theories not satisfying the conditions above.

Example 19. Let T_1 be as in Example 18. 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 indexes 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).

Finally, 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 of Index and Rat a subsort of Elem. This kind of combination, however, is not achievable by a simple

[8] For convenience, we are using here a more conventional notation here for decorated symbols, instead of the less readable in this case $0_{\epsilon,\mathsf{Int}}$, $1_{\epsilon,\mathsf{Int}}$, $+_{\mathsf{Int}\,\mathsf{Int},\mathsf{Int}}$, etc.

[9] That is, whenever a quantifier-free formula φ T_2-entails a disjunction of equations between variables of sort Rat, it T_2-entails one of the equations.

[10] This of course presupposes that our logic is compact, but that too is not very difficult to show using standard model-theoretic techniques.

union of signatures and theories, and as such is out the scope of combination methods *a la* Nelson-Oppen. □

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: **Variable abstraction**, **Partition**, **Decomposition**, and **Check**.

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;

(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 decorated 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 decorated 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.[11]

Second phase: Partition.

Let Γ' be a conjunction of literals obtained in the variable abstraction phase. In the second phase we 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.

We call $\Gamma_1 \cup \Gamma_2$ a conjunction of literals in *separate* form.

Third phase: Decomposition.

Let $\Gamma_1 \cup \Gamma_2$ be the conjunction of literals in separate form obtained in the variable abstraction phase. Note that 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 $s \in S_0$.

[11] But see [TR03], among others, for a more practical kind of abstraction process that minimizes the number of fresh constants introduced.

For all shared sorts $s \in S_0$, let C_s be the set of decorated constants of sort s shared by Γ_1 and Γ_2. In this phase 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 the two constants 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 of this version of the Nelson-Oppen method with respect to the unsorted version, where all pairs of constants are considered for possible identification.

We point out that the nondeterministic choice in the Decomposition Phase above is finitary because it always involves a finite number of shared constants. Correspondingly, every possible arrangement is finite.

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

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

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

Step 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 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 our the combination method correct, we first need to prove a few of basic results used in the correctness proof. The first result is an order-sorted version of a general combination result given in [TR03, Zar04] for unsorted theories. The second result is an order-sorted version of the classic Downward Löwenheim-Skolem Theorem for first-order logic, which in turn is proved by using an order-sorted version of the Hintikka Lemma on the existence of term-generated models for Hintikka sets.

4.1 The Order-sorted Combination Theorem

Theorem 20 (Order-Sorted Combination Theorem). *Let $\Sigma_A = (S_A, \prec_A, F_A, P_A)$ and $\Sigma_B = (S_B, \prec_B, F_B, P_B)$ be two order-sorted signatures, and let Φ_A and Φ_B be two sets of Σ_A- and Σ_B-sentences, respectively. Whenever $\Sigma_A \cup \Sigma_B$ is a conservative union of Σ_A and Σ_B, the set $\Phi_A \cup \Phi_B$ is satisfiable if and only if there exists a Σ_A-structure $\mathcal{A}$ satisfying Φ_A and a Σ_B-structure $\mathcal{B}$ satisfying Φ_B such that*

$$\mathcal{A}^{\Sigma_A \cap \Sigma_B} \cong \mathcal{B}^{\Sigma_A \cap \Sigma_B}\,. \qquad \square$$

Proof. Let

$$\begin{aligned}\Sigma_C = \Sigma_A \cap \Sigma_B &= (S_C \prec_C, F_C, P_C)\\ &= (S_A \cap S_B, \prec_A^* \cap \prec_B^*, F_A \cap F_B, P_A \cap P_B)\,,\end{aligned}$$

and

$$\begin{aligned}\Sigma = \Sigma_A \cup \Sigma_B &= (S, \prec, F, P)\\ &= (S_A \cup S_B, \prec_A \cup \prec_B, F_A \cup F_B, P_A \cup P_B)\,.\end{aligned}$$

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;
- $\mathcal{A}^{\Sigma_C} \cong \mathcal{B}^{\Sigma_C}$.

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, by Proposition 14, we can assume with no loss of generality that $\mathcal{A}^{\Sigma_C} = \mathcal{B}^{\Sigma_C}$.[12] We define a Σ-structure $\mathcal{D}$ by letting for each $s \in S$, $f_{w,s} \in F$, and $p_w \in P$:

- for the domains:
$$D_s = \begin{cases} A_s\,, & \text{if } s \in S_A \\ B_s\,, & \text{if } s \in S_B \setminus S_A \end{cases}$$
- for each $f_{w,s} \in F$:
$$f^{\mathcal{D}}_{w,s} = \begin{cases} f^{\mathcal{A}}_{w,s}\,, & \text{if } f_{w,s} \in F_A \\ f^{\mathcal{B}}_{w,s}\,, & \text{if } f_{w,s} \in F_B \setminus F_A \end{cases}$$

[12] Observe that is assumption implies that $A_s = B_s$ for all shared sorts s.

- for each $p_w \in P$:

$$p_w^{\mathcal{D}} = \begin{cases} p_w^{\mathcal{A}}, & \text{if } p_w \in P_A \\ p_w^{\mathcal{B}}, & \text{if } p_w \in P_B \setminus P_A \end{cases}$$

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 $D_s = A_{s_i} \subseteq A_{s_{i+1}} = D_{s_{i+1}}$. Similarly, if instead $s_i \prec_B s_{i+1}$, we have that $D_s = B_{s_i} \subseteq B_{s_{i+1}} = D_{s_{i+1}}$.

It remains to show that $\mathcal{D}$ respects the subsort overloading of function and predicate symbols.[13] 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 6, 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 $\mathbf{d} \in p_{w'}^{\mathcal{D}}$ iff $\mathbf{d} \in p_{w''}^{\mathcal{D}}$.

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

$$\begin{aligned} \mathbf{d} \in p_{w'}^{\mathcal{D}} \quad & \text{iff} \quad \mathbf{d} \in p_{w'}^{\mathcal{A}} && \text{(by construction of } \mathcal{D}) \\ & \text{iff} \quad \mathbf{d} \in p_{w}^{\mathcal{A}} && \text{(as } w' \sim_A^* w \text{ and } \mathbf{d} \in A_{w'} \cap A_w) \\ & \text{iff} \quad \mathbf{d} \in p_{w}^{\mathcal{B}} && \text{(as } p_w^{\mathcal{A}} = p_w^{\mathcal{B}} \text{ by } \mathcal{A}^{\Sigma_A \cap \Sigma_B} = \mathcal{B}^{\Sigma_A \cap \Sigma_B}) \\ & \text{iff} \quad \mathbf{d} \in p_{w''}^{\mathcal{B}} && \text{(as } w \sim_B^* w'' \text{ and } \mathbf{d} \in B_w \cap B_{w''}) \\ & \text{iff} \quad \mathbf{d} \in p_{w''}^{\mathcal{D}} && \text{(by construction of } \mathcal{D}) \end{aligned}$$

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 $\Phi_A \cup \Phi_B$. ■

4.2 The Order-sorted Hintikka Lemma

In the following, we write $\varphi(x_s)$ to indicate that the decorated variable x_s may occur free in φ. We write $\varphi(t)$ to denote the formula obtained from $\varphi(x_s)$ by replacing one or more

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

occurrences of x_s in φ by t. Also, we denote by $\mathcal{T}_s(\Sigma)$ the set $\mathcal{T}_s(\Sigma, \emptyset)$ of ground Σ-terms of sort s, and by $\mathcal{T}_w(\Sigma)$ where $w = s_1 \dots s_n$ the set $\mathcal{T}_{s_1}(\Sigma) \times \cdots \times \mathcal{T}_{s_n}(\Sigma)$.

Definition 21. Let $\Sigma = (S, \prec, F, P)$ be an order-sorted signature. A *Hintikka set* H with respect to Σ is a set of order-sorted Σ-sentences such that:

1. for each atomic formula φ, it is not the case that both φ and $\neg\varphi$ belong to H;
2. if $\neg\neg\varphi \in H$, then $\varphi \in H$;
3. if $\varphi_1 \wedge \varphi_2 \in H$, then both $\varphi_1, \varphi_2 \in H$;
4. if $\neg(\varphi_1 \wedge \varphi_2) \in H$, then $\neg\varphi_1 \in H$ or $\neg\varphi_2 \in H$;
5. if $\neg\exists x_s\, \varphi(x_s) \in H$, then $\neg\varphi(t) \in H$, for all $t \in \mathcal{T}_s(\Sigma)$;
6. if $\exists x_s\, \varphi(x_s) \in H$, then $\varphi(t) \in H$, for some $t \in \mathcal{T}_s(\Sigma)$;
7. for every t in $\mathcal{T}_s(\Sigma)$ and $s \in S$, $t \approx t \in H$;
8. if $t_1 \approx t_2 \in H$, then $t_2 \approx t_1 \in H$;
9. if $t_1 \approx t_2, t_2 \approx t_3 \in H$, then $t_1 \approx t_3 \in H$;
10. for every $f_{w,s}, f_{w',s'} \in F$, $\mathbf{t} \in \mathcal{T}_w(\Sigma)$, and $\mathbf{t}' \in \mathcal{T}_{w'}(\Sigma)$ with $ws \sim^* w's'$, if $\mathbf{t} \approx \mathbf{t}' \subseteq H$,[14] then $f_{w,s}(\mathbf{t}) \approx f_{w',s'}(\mathbf{t}') \in H$;
11. for every $p_w, p_{w'} \in P$, $\mathbf{t} \in \mathcal{T}_w(\Sigma)$, and $\mathbf{t}' \in \mathcal{T}_{w'}(\Sigma)$ with $w \sim^* w'$, if $\mathbf{t} \approx \mathbf{t}' \subseteq H$ and $p_w(\mathbf{t}) \in H$, then $p_{w'}(\mathbf{t}') \in H$.

As in unsorted logic, all Hintikka sets are satisfiable.

Lemma 22 (Order-sorted Hintikka Lemma). *Let Σ be an order-sorted signature. Then any set H of Σ-formulae that is a Hintikka set wrt. Σ is satisfied by an order-sorted Σ-structure $\mathcal{A}$ with A_s countable for every $s \in \Sigma^{\mathrm{S}}$.* □

Proof. Let us define a relation $\equiv$ over $\mathcal{T}(\Sigma) = \bigcup_{s \in \Sigma^{\mathrm{S}}} \mathcal{T}_s(\Sigma)$ by letting

$$t_1 \equiv t_2 \qquad \text{iff} \qquad t_1 \approx t_2 \in H\,.$$

Note that by Properties 7, 8, 9, and 10 of a Hintikka set, $\equiv$ is a congruence relation on $\mathcal{T}(\Sigma)$. If $t \in \mathcal{T}(\Sigma)$, let us denote with $[t]$ the equivalence class of t with respect to $\equiv$.

We then define a Σ-structure $\mathcal{A}$ by letting:

- for each sort $s \in \Sigma^{\mathrm{S}}$:
$$A_s = \{[t] \mid t \in \mathcal{T}_s(\Sigma)\}\,.$$

[14] Where $\mathbf{t} \approx \mathbf{t}'$ denotes the set of equations between the corresponding components of $\mathbf{t}$ and $\mathbf{t}'$.

- for each function symbol $f_{w,s} \in \Sigma^{\mathrm{F}}$ with $w = s_1 \dots s_n$ and each $([t_1], \dots, [t_n]) \in A_w$:[15]

$$f^{\mathcal{A}}_{w,s}([t_1], \dots, [t_n]) = [f_{w,s}(t_1, \dots, t_n)]\,,$$

- for each predicate symbol $p_w \in \Sigma^{\mathrm{P}}$ and each $([t_1], \dots, [t_n]) \in A_w$:

$$([t_1], \dots, [t_n]) \in p^{\mathcal{A}}_w \quad \text{iff} \quad p_w(t_1, \dots, t_n) \text{ is in } H\,.$$

Using the fact that $\equiv$ is a congruence relation and H satisfies Property 1 of Hintikka sets, it is easy to show that $\mathcal{A}$ is well-defined as a many-sorted structure and $t^{\mathcal{A}} = [t]$, for each ground term $t \in \mathcal{T}(\Sigma)$.

To see that $\mathcal{A}$ is also well-defined as an order-sorted structure, first note that $A_{s_1} \subseteq A_{s_2}$ for all $s_1, s_2 \in \Sigma^{\mathrm{S}}$ with $s_1 \prec^* s_2$. In fact, let $[t] \in A_{s_1}$. As explained earlier, we can assume with no loss of generality that $t \in \mathcal{T}_{s_1}(\Sigma)$. Since $\mathcal{T}_{s_1}(\Sigma) \subseteq \mathcal{T}_{s_2}(\Sigma)$, we can then conclude that $[t] \in A_{s_2}$. Finally, $\mathcal{A}$ respects the overloading condition on function symbols thanks to Property 10 of a Hintikka set, and respects the overloading condition on predicate symbol thanks to Property 11. In fact, let $f_{w,s}, f_{w',s'} \in F$ with $ws \sim^* w's'$ and let $([t_1], \dots, [t_n]) \in A_w \cap A_{w'}$. Then we can assume that $(t_1, \dots, t_n) \in \mathcal{T}_w(\Sigma)$, say, and moreover there is a tuple $(t'_1, \dots, t'_n) \in \mathcal{T}_{w'}(\Sigma)$ such that $[t_i] = [t'_i]$ for all $i = 1, \dots, n$. By definition of $\equiv$ it must then be that $t_i \approx t'_i \in H$ for all $i = 1, \dots, n$. But then $f_{w,s}(t_1, \dots, t_n) \approx f_{w',s'}(t'_1, \dots, t'_n) \in H$ by Property 10 and so $[f_{w,s}(t_1, \dots, t_n)] = [f_{w',s'}(t'_1, \dots, t'_n)]$. It follows that $f^{\mathcal{A}}_{w,s}$ agrees with $f^{\mathcal{A}}_{w',s'}$ over $A_w \cap A_{w'}$. The proof for predicate symbols is analogous.

We claim that $\mathcal{A}$ satisfies H. This can be easily verified by proving by structural induction on Σ-formulae that every formula $\varphi \in H$ is satisfied $\mathcal{A}$. Finally note that A_s is countable for every $s \in \Sigma^{\mathrm{S}}$ because we consider only countable sets of function symbols, and so each $\mathcal{T}_{s_i}(\Sigma)$ is countable. ■

4.3 The Order-sorted Löwenheim-Skolem Theorem

Theorem 23 (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 $\mathcal{A}$ 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 $s \in \Sigma^{\mathrm{S}}$.* □

Proof. The idea of the proof is to extend Φ to an appropriate Hintikka set H with respect to an extended signature $\Sigma(\mathcal{C}')$, where $\mathcal{C}' \subseteq \mathcal{C}$ is a set of fresh constant symbols constructed as follows.

Let S be the set of sorts in Σ^{S} such that $|A_s| \geq \aleph_0$. For each sort $s \in S$, let $C_s \subseteq \mathcal{C} \times \{s\}$ be a countably infinite set of fresh decorated constants of sort s. In addition, for each sort

[15] Note that for all $i = 1, \dots, n$, A_{s_i} contains by construction at least one term from $\mathcal{T}_{s_i}(\Sigma)$. Therefore, we can assume that each given t_i is in fact in $\mathcal{T}_{s_i}(\Sigma)$. This assures that the term $f_{w,s}(t_1, \dots, t_n)$ is well sorted. A similar observation applies later to predicate symbols.

$s \in \Sigma^{\mathrm{S}}$, let $D_s \subseteq \mathcal{C} \times \{s\}$ be another countably infinite set of fresh constant symbols of sort s. Then let

$$C = \bigcup_{s \in S} C_s \cup \bigcup_{s \in \Sigma^{\mathrm{S}}} D_s \,.$$

Let Σ_C be the signature obtained from Σ by adding C to Σ^{F}, and let H_0 be the following set of Σ_C-sentences:

$$H_0 = \Phi \cup \{u \not\approx v \mid u, v \in C_s \text{ are distinct and } s \in S\} \,.$$

By construction, H_0 is satisfiable. A structure $\mathcal{A}_0$ satisfying H_0 can be obtained by appropriately expanding $\mathcal{A}$ to interpret the constant symbols in C_s, for all $s \in S$.

We now describe an infinite process that will allow us to construct an increasing sequence of sets of sentences $H_0, H_1, H_2, \ldots$, and another sequence of structures $\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2$ such that $\mathcal{A}_i \models H_i$, for each i. Letting

$$H = \bigcup_i H_i \,,$$

we will prove later that H is a Hintikka set with respect to the signature Σ_C.

Given H_i and $\mathcal{A}_i$, we construct H_{i+1} by applying one of the rules in Figure 1 to H_i. Then, we define $\mathcal{A}_{i+1}$ to be a structure that is either identical to $\mathcal{A}_i$, or appropriately expands $\mathcal{A}_i$. More precisely, if H_{i+1} is derived by a rule other than the quantifier rules, $\mathcal{A}_{i+1} = \mathcal{A}_i$. If H_{i+1} is derived by the first quantifier rule, then $\mathcal{A}_{i+1}$ is an expansion of $\mathcal{A}_i$ that interprets arbitrarily any constants in the term t that are not already in the signature of $\mathcal{A}_i$.[16] If H_{i+1} is derived by the second quantifier rule, then $\mathcal{A}_{i+1}$ expands $\mathcal{A}_i$ by interpreting the new constant $u = c_{\epsilon,s}$ as the element a of A_s such that $(\mathcal{A}_i, \{x_s \mapsto a\}) \models \varphi(x_s)$.

It is crucial that this process be done *fairly*, in the sense that no applicable instance of a rule is delayed infinitely often. We refer the reader to [Fit96], for instance, for details on how to ensure fairness.[17] Fairness ensures that H is saturated with respect to the rules in Figure 1, i.e., every consequence of H by one of the rules is already in H.

We claim that the obtained set H is a Hintikka set. Clearly, Properties 2–11 of a Hintikka set follow by saturation of H. To prove Property 1, suppose, ad absurdum, that H contains a pair of contradictory atoms $\varphi, \neg\varphi$. Then there is an $i \geq 0$ such that H_i contains both φ and $\neg\varphi$. But this contradicts the fact that A_i is a model of H_i by construction.

Since H is a Hintikka set, Lemma 22 tells us that H is satisfiable in a structure $\mathcal{B}$ all of whose domains are countable. Since H contains all the literals in

$$\{u \not\approx v \mid u, v \in C_s \text{ are distinct and } s \in S\} \,,$$

[16] This is possible because there is no overloading among the free constants in C.

[17] Strictly speaking, the fairness arguments in [Fit96] apply only to a finite initial set, whereas our set H_0 can be infinite. However, those arguments can be extended to an infinite initial set by a standard diagonalization construction.

Propositional rules

$$\frac{\neg\neg\varphi}{\varphi} \qquad \frac{\varphi_1 \wedge \varphi_2}{\varphi_1 \quad \varphi_2} \qquad \frac{\neg(\varphi_1 \wedge \varphi_2)}{\neg\varphi_1} \qquad \frac{\neg(\varphi_1 \wedge \varphi_1)}{\neg\varphi_2}$$

The third rule applies to the set H_i provided that $\mathcal{A}_i$ satisfies $\neg\varphi_1$. The fourth rule applies to H_i provided that $\mathcal{A}_i$ satisfies $\neg\varphi_2$.

Quantifier rules

$$\frac{\neg\exists x_s\, \varphi(x_s)}{\neg\varphi(t)} \qquad \frac{\exists x_s\, \varphi(x_s)}{\varphi(u)}$$

In these rules t is any ground term in $\mathcal{T}_s(\Sigma_C)$ and $u \in D_s$ is a *fresh* constant—that is, not appearing in the particular set H_i the rule is applied to.

Equality rules

$$\frac{}{t_1 \approx t_1} \qquad \frac{t_1 \approx t_2}{t_2 \approx t_1} \qquad \frac{t_1 \approx t_2 \quad t_2 \approx t_3}{t_1 \approx t_3}$$

In these rules t_1, t_2, t_3 are any ground Σ_C-terms.

Congruence and overloading rules

$$\frac{\mathbf{t} \approx \mathbf{t}'}{f_{w,s}(\mathbf{t}) \approx f_{w',s'}(\mathbf{t}')} \qquad \frac{\mathbf{t} \approx \mathbf{t}' \quad p_w(\mathbf{t})}{p_{w'}(\mathbf{t}')}$$

In both rules, $\mathbf{t} \in \mathcal{T}_w(\Sigma_C)$ and $\mathbf{t}' \in \mathcal{T}_{w'}(\Sigma_C)$. The first rule applies if $f_{w,s}, f_{w',s'}$ are in Σ_C and $ws \sim^* w's'$. The second rule applies if $p_w, p_{w'}$ are in Σ_C and $w \sim^* w'$.

Figure 1: Constructing a Hintikka set.

we also have that $|B_s|$ is countably infinite for each $s \in S$. Finally, note that $\mathcal{B}$ satisfies Φ because $\Phi \subseteq H$ by construction. ■

4.4 Correctness and Decidability Results

We are now ready to proof that the combination method is correct. We will therefore consider again the order-sorted signatures Σ_1, Σ_2 and the theories T_1 and T_2 defined in Section 3.

Theorem 24. *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 if and only if 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 decorated constants* $u, v \in C_s$ *and* $s \in S_0$. □

PROOF. Let $\Sigma_0 = \Sigma_1 \cap \Sigma_2 = (S_0, \prec_0, F_0, P_0) = (S_1 \cap S_2, \prec_1^* \cap \prec_2^*, F_1 \cap F_2, P_1 \cap P_2)$. Observe that because of the assumptions, F_0 and P_0 are both empty, and $\prec_0^*$ coincides with the identity relation.

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. With the goal of applying the Many-Sorted Combination Theorem we define a function family $h = \{h_s : {C_s}^{\mathcal{A}} \to {C_s}^{\mathcal{B}} \mid s \in S_0\}$ 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 $h'_s : A_s \to B_s$.

Let $\mathcal{C}_0$ be the set of all constant symbols in $\{C_s \mid s \in S_0\}$. Observing that the only symbols in the signature $\Sigma_0(\mathcal{C}_0)$ are constant symbols (from $\mathcal{C}_0$) and all of its sorts are pairwise disconnected, it is clear that the function family $h' = \{h'_s : A_s \to B_s \mid s \in S_0\}$ 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 20 we obtain the existence of a $(\Sigma_1 \cup \Sigma_2)(\mathcal{C})$-structure $\mathcal{D}$ satisfying $\Phi_1 \cup \Phi_2$. ■

Proposition 25 (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 decorated free constants shared by* Γ_1 *and* Γ_2*. Then, 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. (1 ⇒ 2) Let $\mathcal{D}$ be a $(T_1 \cup T_2)$-structure satisfying $\Gamma_1 \cup \Gamma_2$. We define a family $E = \{E_s \mid s \in S_0\}$ of equivalence relations E_s over C_s by letting $(u, v) \in E_\sigma$ if and only if $u^{\mathcal{D}} = v^{\mathcal{D}}$, for every $u, v \in C_s$. By construction, $\mathcal{D}$ satisfies both T_i and $\Gamma_i \cup arr(V, E)$, for $i = 1, 2$.

(2 ⇒ 1) 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:

- $|A_s| = |B_s|$, for each $s \in S_0$;
- $u^{\mathcal{A}} = v^{\mathcal{A}}$ iff $u^{\mathcal{B}} = v^{\mathcal{B}}$, for each $u, v \in C_s$ and $s \in S_0$.

By Theorem 24 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 25 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 26 (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 that uses decorated symbols to abstract away, for simplicity, the usual parsing and sort inference problems that arise in order-sorted logics. 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 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 add to 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, with connected shared sorts, 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. In fact, even with just two connected shared sorts s_1 and s_2, the analogous of Theorem 24 requires on the structures $\mathcal{A}$ and $\mathcal{B}$ not only that $|A_s| = |B_s|$ for every shared sort s, but also that $|A_{s_1} \cap A_{s_2}| = |B_{s_1} \cap B_{s_2}|$, $|A_{s_1} \setminus A_{s_2}| = |B_{s_1} \setminus B_{s_2}|$, and $|A_{s_2} \setminus A_{s_1}| = |B_{s_2} \setminus B_{s_1}|$. Note that these are cardinality requirements on the regions of the Venn diagram determined by A_{s_1}, A_{s_2} and B_{s_1}, B_{s_2}. They are a necessary condition for the family of mappings h in the proof of Theorem 24 to be extensible to an isomorphism between the reducts of $\mathcal{A}$ and $\mathcal{B}$ to the shared signature. With $n > 2$ pairwise connected sorts, the requirements explode along with the explosion of the Venn regions. It is not clear at the moment if there are local conditions on the component theories (along the lines of stable infiniteness, for instance) that are sufficient for these extended requirements.

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 [TR03, Tin03, Zar02, Zar04, Ghi03, Ghi04]. Concurrently with the work presented here, the specific approach of [Ghi03, Ghi04] has been adapted in [GBTD04], 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.

References

[BB04] Clark Barrett and Sergey Berezin. CVC Lite: A new implementation of the cooperating validity checker. In *Proceedings of the 16th International Conference on Computer Aided Verification (CAV)*, 2004. (To appear.).

[BDS02] Clark W. Barrett, David L. Dill, and Aaron Stump. A generalization of Shostak's method for combining decision procedures. In A. Armando, editor, *Proceedings of the 4th International Workshop on Frontiers of Combining Systems, FroCoS'2002 (Santa Margherita Ligure, Italy)*, volume 2309 of *Lecture Notes in Computer Science*, pages 132–147, apr 2002.

[CKM+91] Dan Craigen, Sentot Kromodimoeljo, Irwin Meisels, Bill Pase, and Mark Saaltink. EVES: An overview. In Soren Prehen and Hans Toetenel, editors, *Formal Software Development Methods*, volume 552 of *LNCS*, pages 389–405. Springer, 1991.

[DNS03] David Detlefs, Greg Nelson, and James B. Saxe. Simplify: A theorem prover for program checking. Technical Report HPL-2–3-148, HP Laboratories, Palo Alto, CA, 2003.

[Fit96] Melvin C. Fitting. *First-Order Logic and Automated Theorem Proving*. Graduate Texts in Computer Science. Springer, 2nd edition, 1996.

[GBTD04] Vijay Ganesh, Sergey Berezin, Cesare Tinelli, and David Dill. Combination results for many sorted theories with overlapping signatures. (Submitted), 2004.

[Ghi03] Silvio Ghilardi. Quantifier elimination and provers integration. In Ingo Dahn and Laurent Vigneron, editors, *First Order Theorem Proving*, volume 86.1 of *Electronic Notes in Theoretical Computer Science*. Elsevier, 2003.

[Ghi04] Silvio Ghilardi. Model theoretic methods in combined constraint satisfiability. *Journal of Automated Reasoning*, 2004. (To appear).

[GM92] Joseph A. Goguen and José Meseguer. Order-sorted algebra I: Equational deduction for multiple inheritance, overloading, exceptions and partial operations. *Theoretical Computer Science*, 105(2):217–173, 1992.

[LFMM92] Beth Levy, Ivan Filippenko, Leo Marcus, and Telis Menas. Using the state delta verification system (SDVS) for hardware verification. In Tom F. Melham, V. Stavridou, and Raymond T. Boute, editors, *Theorem Prover in Circuit Design: Theory, Practice and Experience*, pages 337–360. Elsevier Science, 1992.

[Mes98] José Meseguer. Membership algebra as a logical framework for equational specification. In *In Proceedings of the 12th International Workshop on Recent Trends in Algebraic Development Techniques (WADT'97)*, volume 1376 of *Lecture Notes in Computer Science*, pages 18–61. Springer-Verlag, 1998.

[MJ04] Filip Maric and Predrag Janičić. ARGO-LIB: A generic platform for decision procedures. In *Proceedings of the 2nd International Joint Conference on Automated Reasoning, IJCAR'04, (Cork, Ireland)*, Lecture Notes in Artificial Intelligence. Springer, 2004. (To appear).

[NO79] Greg Nelson and Derek C. Oppen. Simplification by cooperating decision procedures. *ACM Transactions on Programming Languages and Systems*, 1(2):245–257, 1979.

[SBD02] Aaron Stump, Clark W. Barrett, and David L. Dill. CVC: A cooperating validity checker. In Ed Brinksma and Kim Guldstrand Larsen, editors, *Computer Aided Verification*, volume 2404 of *Lecture Notes in Computer Science*, pages 500–504, 2002.

[TH96] Cesare Tinelli and Mehdi T. Harandi. A new correctness proof of the Nelson-Oppen combination procedure. In Franz Baader and Klaus U. Schulz, editors, *Frontiers of Combining Systems*, volume 3 of *Applied Logic Series*, pages 103–120. Kluwer, 1996.

[Tin03] Cesare Tinelli. Cooperation of background reasoners in theory reasoning by residue sharing. *Journal of Automated Reasoning*, 30(1):1–31, January 2003.

[TR03] Cesare Tinelli and Christophe Ringeissen. Unions of non-disjoint theories and combinations of satisfiability procedures. *Theoretical Computer Science*, 290(1):291–353, 2003.

[Zar02] Calogero G. Zarba. A tableau calculus for combining non-disjoint theories. In Uwe Egly and Christian G. Fermüller, editors, *Automated Reasoning with Analytical Tableaux and Related Methods*, volume 2381 of *Lecture Notes in Artifical Intelligence*, pages 315–329. Springer, 2002.

[Zar04] Calogero G. Zarba. *The Combination Problem in Automated Reasoning*. PhD thesis, Stanford University, 2004.