Sharing is Caring: Combination of Theories*

Dejan Jovanović and Clark Barrett

New York University

Abstract. One of the main shortcomings of the traditional methods for combining theories is the complexity of guessing the arrangement of the variables shared by the individual theories. This paper presents a reformulation of the Nelson-Oppen method that takes into account explicit equality propagation and can ignore pairs of shared variables that the theories do not care about. We show the correctness of the new approach and present care functions for the theory of uninterpreted functions and the theory of arrays. The effectiveness of the new method is illustrated by experimental results demonstrating a dramatic performance improvement on benchmarks combining arrays and bit-vectors.

1 Introduction

The seminal paper of Nelson and Oppen [15] introduced a general framework for combining quantifier-free first-order theories in a modular fashion. Using the Nelson-Oppen framework, decision procedures for two individual theories can be used as black boxes to create a decision procedure for the combined theory. Although the Nelson-Oppen combination method as originally formulated requires stably-infinite theories, it can be extended to handle non-stably-infinite theories using an approach based on polite theories [12, 13, 18].

The core idea driving the method (and ensuring its correctness) is the exchange of equalities and disequalities over the interface variables between the theories involved in the combination. Interface variables are the problem variables that are shared by both theories (or an extended set of variables in the polite combination framework), and both theories must agree on an arrangement over these variables. Most modern satisfiability modulo theories (SMT) solvers perform the search for such an arrangement by first using aggresive theory propagation to determine as much of the arrangement as possible and then relying on an efficient SAT solver to guess the rest of the arrangement, backtracking and learning lemmas as necessary [1, 3, 6].

In some cases, if the theories that are being combined have additional properties, such as convexity and/or complete and efficient equality propagation, there are more efficient ways of obtaining a suitable arrangement. But, in general, since the number of shared variables can be substantial, guessing an arrangement over the shared variables can have an exponential impact on the running time [16]. Trying to minimize the burden of non-deterministic guessing is thus

* This work was funded in part by SRC contract 2008-TJ-1850.

of the utmost importance for a practical and efficient combination mechanism. For example, a recent model-based theory combination approach [7], in which the solver keeps a model for each theory, takes the optimistic stance of eagerly propagating all equalities that hold in the model (whether or not they are truly implied), obtaining impressive performance improvements.

In this paper we tackle the problem of minimizing the amount of non-deterministic guessing by equipping the theories with an *equality propagator* and a *care function*. The role of the theory-specific equality propagator is, given a context, to propagate entailed equalities and disequalities over the interface variables. The care function, on the other hand, provides information about which variable pairs among the interface variables are important for maintaining the satisfiability of a given formula. With the information provided by these two functions we can, in many cases, drastically reduce the search space for finding a suitable arrangement. We present a reformulation of the Nelson-Oppen method that uses these two functions to decide a combination of two theories. The method can easily be adapted to the combination method for polite theories, where reducing the number of shared variables is even more important (the polite theory combination method requires extending the set of interface variables significantly).

2 Preliminaries

We start with a brief overview of the syntax and semantics of many-sorted first-order logic. For a more detailed exposition, we refer the reader to [11, 21].

A *signature* Σ is a triple (S, F, P) where S is a set of *sorts*, F is a set of *function symbols*, and P is a set of *predicate symbols*. For a signature $\Sigma = (S, F, P)$, we write $\Sigma^{\mathbb{S}}$ for the set S of sorts, $\Sigma^{\mathbb{F}}$ for the set F of function symbols, and $\Sigma^{\mathbb{P}}$ for the set P of predicates. Each predicate and function symbol is associated with an *arity*, a tuple constructed from the sorts in S. Functions whose arity is a single sort are called *constants*. We write $\Sigma_1 \cup \Sigma_2 = (S_1 \cup S_2, F_1 \cup F_2, P_1 \cup P_2)$ for the union[1] of signatures $\Sigma_1 = (S_1, F_1, P_1)$ and $\Sigma_2 = (S_2, F_2, P_2)$. Additionally, we write $\Sigma_1 \subseteq \Sigma_2$ if $S_1 \subseteq S_2$, $F_1 \subseteq F_2$, $P_1 \subseteq P_2$, and the symbols of Σ_1 have the same arity as those in Σ_2. We assume the standard notions of a Σ-*term*, Σ-*literal*, and Σ-*formula*. In the following, we assume that all formulas are quantifier-free, if not explicitly stated otherwise. A literal is called *flat* if it is of the form $x = y$, $x \neq y$, $x = f(y_1, \ldots, y_n)$, $p(y_1, \ldots, y_n)$, or $\neg p(y_1, \ldots, y_n)$, where $x, y, y_1, \ldots, y_n$ are variables, f is a function symbol, and p is a predicate symbol. If ϕ is a term or a formula, we will denote by $vars_\sigma(\phi)$ the set of variables of sort σ that occur (free) in ϕ. We overload this function in the usual way, $vars_S(\phi)$ denoting variables in ϕ of the sorts in S, and $vars(\phi)$ denoting all variables in ϕ. We also sometimes refer to a set Φ of formulas as if it were a single formula, in

[1] In this paper, we always assume that function and predicate symbols from different theories do not overlap, so that the union operation is well-defined. On the other hand, two different theories are allowed to have non-disjoint sets of sorts.

which case the intended meaning is the conjunction $\bigwedge \Phi$ of the formulas in the set.

Let Σ be a signature, and let X be a set of variables whose sorts are in $\Sigma^{\mathbb{S}}$. A *Σ-interpretation* $\mathcal{A}$ over X is a map that interprets each sort $\sigma \in \Sigma^{\mathbb{S}}$ as a non-empty domain A_σ, each variable $x \in X$ of sort σ as an element $x^{\mathcal{A}} \in A_\sigma$, each function symbol $f \in \Sigma^{\mathbb{F}}$ of arity $\sigma_1 \times \cdots \times \sigma_n \times \tau$ as a function $f^{\mathcal{A}} : A_{\sigma_1} \times \cdots \times A_{\sigma_n} \to A_\tau$, and each predicate symbol $p \in \Sigma^{\mathbb{P}}$ of arity $\sigma_1 \times \cdots \times \sigma_n$ as a subset $p^{\mathcal{A}}$ of $A_{\sigma_1} \times \cdots \times A_{\sigma_n}$. A *$\Sigma$-structure* is a Σ-interpretation over an empty set of variables. As usual, the interpretations of terms and formulas in an interpretation $\mathcal{A}$ are defined inductively over their structure. For a term t, we denote with $t^{\mathcal{A}}$ the evaluation of t under the interpretation $\mathcal{A}$. Likewise, for a formula ϕ, we denote with $\phi^{\mathcal{A}}$ the truth-value (true or false) of ϕ under interpretation $\mathcal{A}$. A Σ-formula ϕ is *satisfiable* iff it evaluates to true in some Σ-interpretation over (at least) *vars*(ϕ). Let $\mathcal{A}$ be an Ω-interpretation over some set V of variables. For a signature $\Sigma \subseteq \Omega$, and a set of variables $U \subseteq V$, we denote with $\mathcal{A}^{\Sigma,U}$ the interpretation obtained from $\mathcal{A}$ by restricting it to interpret only the symbols in Σ and the variables in U.

We will use the definition of theories as classes of structures, rather than sets of sentences. We define a theory formally as follows (see e.g. [20] and Definition 2 in [18]).

Definition 1 (Theory). *Given a set of Σ-sentences* $\mathbf{Ax}$ *a Σ-*theory $T_{\mathbf{Ax}}$ *is a pair $(\Sigma, \mathbf{A})$ where Σ is a signature and $\mathbf{A}$ is the class of Σ-structures that satisfy* $\mathbf{Ax}$.

Given a theory $T = (\Sigma, \mathbf{A})$, a *T-interpretation* is a Σ-interpretation $\mathcal{A}$ such that $\mathcal{A}^{\Sigma,\emptyset} \in \mathbf{A}$. A Σ-formula ϕ is T-satisfiable iff it is satisfiable in some T-interpretation $\mathcal{A}$. This is denoted as $\mathcal{A} \vDash_T \phi$, or just $\mathcal{A} \vDash \phi$ if the theory is clear from the context.

As theories in our formalism are represented by classes of structures, a combination of two theories is represented by those structures that can interpret both theories (Definition 3 in [18]).

Definition 2 (Combination). *Let $T_1 = (\Sigma_1, \mathbf{A}_1)$ and $T_2 = (\Sigma_2, \mathbf{A}_2)$ be two theories. The* combination *of T_1 and T_2 is the theory $T_1 \oplus T_2 = (\Sigma, \mathbf{A})$ where $\Sigma = \Sigma_1 \cup \Sigma_2$ and $\mathbf{A} = \{\Sigma$-structures $\mathcal{A} \mid \mathcal{A}^{\Sigma_1,\emptyset} \in \mathbf{A_1}$ and $\mathcal{A}^{\Sigma_2,\emptyset} \in \mathbf{A_2}\}$.*

The set of Σ-structures resulting from the combination of two theories is indeed a theory in the sense of Definition 1. If $\mathbf{Ax}_1$ is the set of sentences defining theory T_1, and $\mathbf{Ax}_2$ is the set of sentences defining theory T_2, then $\mathbf{A}$ is the set of Σ-structures that satisfy the set $\mathbf{Ax} = \mathbf{Ax}_1 \cup \mathbf{Ax}_2$ (see Proposition 4 in [18]).

Given decision procedures for the satisfiability of formulas in theories T_1 and T_2, we are interested in constructing a decision procedure for satisfiability in $T_1 \oplus T_2$ using these procedures as black boxes. The Nelson-Oppen combination method [15, 20, 21] gives a general mechanism for doing this. Given a formula ϕ over the combined signature $\Sigma_1 \cup \Sigma_2$, the first step is to *purify* ϕ by constructing an equisatisfiable set of formulas $\phi_1 \cup \phi_2$ such that each ϕ_i consists of only Σ_i-formulas. This can easily be done by finding a pure (i.e. Σ_i- for some i) subterm t, replacing it with a new variable v, adding the equation $v = t$, and then repeating this process until all formulas are pure. The next step is to force the decision procedures for the individual theories to agree on whether variables appearing in both ϕ_1 and ϕ_2 (called *shared* or *interface* variables) are equal. This is done by introducing an *arrangement* over the shared variables [18, 20].

Here we will use a more general definition of an arrangement that allows us to restrict the pairs of variables that we are interested in. We do so by introducing the notion of a *care graph*. Given a set of variables V, we will call any graph $\mathbf{G} = \langle V, E\rangle$ a care graph, where $E \subseteq V \times V$ is the set of care graph edges. If an edge $(x, y) \in E$ is present in the care graph, it means that we are interested in the relationship between the variables x and y.

Definition 3 (Arrangement). *Given a care graph $\mathbf{G} = \langle V, E\rangle$ where sorts of variables in V range over a set of sorts S, with $V_\sigma = vars_\sigma(V)$, we call $\delta_\mathbf{G}$ an* arrangement over $\mathbf{G}$ *if there exists a family of equivalence relations*

$$\mathcal{E} = \{\ E_\sigma \subseteq V_\sigma \times V_\sigma \ \mid\ \sigma \in S\ \}\ ,$$

such that the equivalence relations restricted to E induce $\delta_\mathbf{G}$, i.e. $\delta_\mathbf{G} = \bigcup_{\sigma\in S} \delta_\sigma$, where each δ_σ is an individual arrangement of V_σ (restricted to E):

$$\delta_\sigma = \{\ x = y \ \mid\ (x, y) \in E_\sigma \cap E\ \} \cup \{\ x \neq y \ \mid\ (x, y) \in \overline{E}_\sigma \cap E\ \}\ ,$$

where $\overline{E}_\sigma$ denotes the complement of E_σ (i.e. $V_\sigma \times V_\sigma \setminus E_\sigma$). If the care graph $\mathbf{G}$ is a complete graph over V, we will denote the arrangement simply as δ_V.

The Nelson-Oppen combination theorem states that ϕ is satisfiable in $T_1 \oplus T_2$ iff there exists an arrangement δ_V of the shared variables $V = vars(\phi_1) \cap vars(\phi_2)$ such that $\phi_i \cup \delta_V$ is satisfiable in T_i. However, as mentioned earlier, some restrictions on the theories are necessary in order for the method to be complete. Sufficient conditions for completeness are: the two signatures have no function or predicate symbols in common; and the two theories are *stably-infinite* over (at least) the set of common sorts $\Sigma_1^{\mathbb{S}} \cap \Sigma_2^{\mathbb{S}}$. Stable-infiniteness was originally introduced in a single-sorted setting [16]. In the many-sorted setting stable-infiniteness is defined with respect to a subset of the signature sorts (see Definition 6 from [21]).

3 New Combination Method

In this section we present a new method for combining two signature-disjoint theories. The method is based on Nelson-Oppen, but it makes equality propagation explicit and also includes a *care function* for each theory, enabling a more efficient mechanism for determining equalities and dis-equalities among the shared variables. Another notable difference from the original method is that we depart from viewing the combination problem as symmetric. Instead, as in the method for combining polite theories [12, 13, 18], one of the theories is designated to take the lead in selecting which variable pairs are going to be part of the final arrangement.

We first define the equality propagator and the care function, and then proceed to presenting and proving the combination method.

Definition 4 (Equality Propagator). *For a Σ-theory T we call a function $\mathfrak{P}_T^{=}[\![\cdot]\!]$ an* equality propagator *for T if, for every set V of variables, it maps every set ϕ of flat Σ-literals into a set of equalities and dis-equalities between variables:*

$$\mathfrak{P}_T^{=}[\![V]\!](\phi) = \{x_1 = y_1, \ldots, x_m = y_m\} \cup \{z_1 \neq w_1, \ldots, z_n \neq w_n\} \ ,$$

where $vars(\mathfrak{P}_T^{=}[\![V]\!](\phi)) \subseteq V$ and

1. *for each equality $x_i = y_i \in \mathfrak{P}_T^{=}[\![V]\!](\phi)$ it holds that $\phi \vDash_T x_i = y_i$;*
2. *for each dis-equality $z_i \neq w_i \in \mathfrak{P}_T^{=}[\![V]\!](\phi)$ it holds that $\phi \vDash_T z_i \neq w_i$;*
3. *$\mathfrak{P}_T^{=}[\![V]\!]$ is monotone, i.e. $\phi \subseteq \psi \implies \mathfrak{P}_T^{=}[\![V]\!](\phi) \subseteq \mathfrak{P}_T^{=}[\![V]\!](\psi)$; and*
4. *$\mathfrak{P}_T^{=}[\![V]\!]$ contains at least those equalities and dis-equalities, over variables in V, that appear in ϕ.*

An equality propagator, given a set of theory literals, returns a set of entailed equalities and dis-equalities between the variables in V. It does not need to be complete (i.e. it does not need to return *all* entailed equalities and dis-equalities), but the more complete it is, the more helpful it is in reducing the arrangement search space.

When combining two theories, the combined theory can provide more equality propagation than just the union of the individual propagators. The following construction defines an equality propagator that reuses the individual propagators in order to obtain a propagator for the combined theory. This is achieved by allowing the propagators to incrementally exchange literals until a fix-point is reached.

Definition 5 (Combined Propagator). *Let T_1 and T_2 be two theories over the signatures Σ_1 and Σ_2, equipped with equality propagators $\mathfrak{P}_{T_1}^{=}[\![\cdot]\!]$ and $\mathfrak{P}_{T_2}^{=}[\![\cdot]\!]$, respectively. Let $T = T_1 \oplus T_2$ and $\Sigma = \Sigma_1 \cup \Sigma_2$. Let V be a set of variables and ϕ a set of flat Σ-literals partitioned into a set ϕ_1 of Σ_1-literals and a set ϕ_2 of Σ_2-literals. We define the combined propagator $\mathfrak{P}_T^{=}[\![\cdot]\!]$ for the theory T as*

$$\mathfrak{P}_T^{=}[\![V]\!](\phi) = (\mathfrak{P}_{T_1}^{=} \oplus \mathfrak{P}_{T_2}^{=})[\![V]\!](\phi) = \psi_1^* \cup \psi_2^* \ ,$$

where $\langle \psi_1^*, \psi_2^* \rangle$ is the least fix-point of the following operator $\mathcal{F}$

$$\mathcal{F}\langle \psi_1, \psi_2 \rangle = \left\langle \mathfrak{P}_{T_1}^{=}[\![V]\!](\phi_1 \cup \psi_2), \mathfrak{P}_{T_2}^{=}[\![V]\!](\phi_2 \cup \psi_1) \right\rangle \ .$$

The fix-point exists as the propagators are monotone and the set V is finite. Moreover, the value of the fix-point is easily computable by iteration from $\langle \emptyset, \emptyset \rangle$. Also, it's clear from the definition that the combined propagator is at least as strong as the individual propagators, i.e. $\mathfrak{P}_{T_1}^{=}[\![V]\!](\phi_1) \subseteq \mathfrak{P}_T^{=}[\![V]\!](\phi_1) \subseteq \mathfrak{P}_T^{=}[\![V]\!](\phi)$, $\mathfrak{P}_{T_2}^{=}[\![V]\!](\phi_2) \subseteq \mathfrak{P}_T^{=}[\![V]\!](\phi_2) \subseteq \mathfrak{P}_T^{=}[\![V]\!](\phi)$.

Definition 6 (Care Function). *For a Σ-theory T we call a function $\mathfrak{C}[\![\cdot]\!]$ a* care function *for T with respect to a T-equality propagator $\mathfrak{P}_T^{=}[\![\cdot]\!]$ when for every set V of variables and every set ϕ of flat Σ-literals*

1. *$\mathfrak{C}[\![V]\!]$ maps ϕ to a care graph $\mathbf{G} = \langle V, E \rangle$;*
2. *if $x = y$ or $x \neq y$ are in $\mathfrak{P}_T^{=}[\![V]\!](\phi)$ then $(x, y) \notin E$;*
3. *if $\mathbf{G} = \langle V, \emptyset \rangle$ and ϕ is T-satisfiable then, for any arrangement δ_V such that $\mathfrak{P}_T^{=}[\![V]\!](\phi) \subseteq \delta_V$, it holds that $\phi \cup \delta_V$ is also T-satisfiable.*

For any Σ-theory T and a set of variables V, the *trivial care function* $\mathfrak{C}_0[\![\cdot]\!]$ is the one that maps a set of variables to a complete graph over the pairs of variables that are not yet decided. i.e.

$$\mathfrak{C}_0[\![V]\!](\phi) = \langle V, \{(x, y) \in V \times V \mid x = y, x \neq y \notin \mathfrak{P}_T^{=}[\![V]\!](\phi)\} \rangle \ .$$

Notice that $\mathfrak{C}_0[\![\cdot]\!]$ trivially satisfies the conditions of Definition 6 with respect to any equality propagator. To see this, the only case to consider is when the care graph returned has no edges and ϕ is satisfiable. But in this case, if $\mathfrak{P}_T^{=}[\![V]\!](\phi) \subseteq \delta_V$, then we must have $\mathfrak{P}_T^{=}[\![V]\!](\phi) = \delta_V$, and so clearly $\phi \cup \delta_V$ is satisfiable.

3.1 Combination Method

Let T_i be a Σ_i-theory, for $i = 1, 2$ and let $S = \Sigma_1^{\mathbb{S}} \cap \Sigma_2^{\mathbb{S}}$. Further, assume that each T_i is stably-infinite with respect to S_i, decidable, and equipped with a theory propagator $\mathfrak{P}_{T_i}^{=}[\![\cdot]\!]$. Additionally, let T_2 be equipped with a care function $\mathfrak{C}_{T_2}[\![\cdot]\!]$ operating with respect to $\mathfrak{P}_{T_2}^{=}[\![\cdot]\!]$. We are interested in deciding the combination theory $T = T_1 \oplus T_2$ over the signature $\Sigma = \Sigma_1 \cup \Sigma_2$. We denote the combined theory propagator with $\mathfrak{P}_T^{=}[\![\cdot]\!]$.The combination method takes as input a set ϕ of Σ-literals and consists of the following steps:

Purify: The output of the purification phase is two new sets of literals, ϕ_1 and ϕ_2 such that $\phi_1 \cup \phi_2$ is equisatisfiable (in T) with ϕ and each literal in ϕ_i is a flat Σ_i-literal, for $i = 1, 2$.

Arrange: Let $V = \mathit{vars}(\phi_1) \cap \mathit{vars}(\phi_2)$ be the set of all variables shared by ϕ_1 and ϕ_2. Let the care graph $\mathbf{G}_2$ be a fix-point of the following operator

$$\mathcal{G}\langle \mathbf{G} \rangle = \mathbf{G} \cup \mathfrak{C}_{T_2}[\![V]\!](\phi_2 \cup \mathfrak{P}_T^{=}[\![V]\!](\phi_1 \cup \phi_2 \cup \delta_{\mathbf{G}})) \ , \tag{1}$$

where we choose the arrangement $\delta_{\mathbf{G}}$ non-deterministically.

Check: Check the following formulas for satisfiability in T_1 and T_2 respectively

$$\phi_1 \cup \mathfrak{P}_T^{=}[\![V]\!](\phi_1 \cup \phi_2 \cup \delta_{\mathbf{G}_2}) \ , \qquad \phi_2 \cup \mathfrak{P}_T^{=}[\![V]\!](\phi_1 \cup \phi_2 \cup \delta_{\mathbf{G}_2}) \ .$$

If both are satisfiable, output satisfiable, otherwise output unsatisfiable.

Notice that above, since the graph is finite, and the the operator $\mathcal{G}$ is increasing, the fix-point always exists. Moreover, it is in our interest to choose the minimal such fix-point, which we can obtain by doing a fix-point iteration starting from

$\mathbf{G}_0 = \langle V, \emptyset \rangle$. Another important fact is that for any fix-point $\mathbf{G}$, with respect to the $\delta_{\mathbf{G}}$ we have chosen, of the operator $\mathcal{G}$ above, we must have that the care function from (1) returns an empty graph. This follows from the fact that the propagator must return all the equalities and dis-equalities from $\delta_{\mathbf{G}}$, by definition, and the care function then must ignore them, also by definition.

Example 1. Consider the case of combining two theories T_1 and T_2 equipped with trivial care functions and propagators $\mathfrak{P}^{=}_{T_i}[\![V]\!]$ that simply return those input literals that are either equalities or dis-equalities over variables in V. Assume that ϕ_1 and ϕ_2 are the output of the purification phase, and let V be the set of variables shared by ϕ_1 and ϕ_2. Since $\mathfrak{C}_{T_2}[\![\cdot]\!]$ is a trivial care function, we will choose a arrangement $\delta_{\mathbf{G}_2}$ over the set V of shared variables that completes the set of equalities and dis-equalities over V. Since equality propagators simply keep the input equalities and dis-equalities over V, and all other relationships between variables in V are determined by $\delta_{\mathbf{G}_2}$, the combined propagator will return a complete arrangement δ_V and we will then check $\phi_1 \cup \delta_V$ and $\phi_2 \cup \delta_V$ for satisfiability. This shows that our method can effectively simulate the standard Nelson-Oppen combination method. We now show the correctness of the method.

Theorem 1. *Let T_i be a Σ_i-theory, stably-infinite with respect to the set of sorts S_i, and equipped with equality propagators $\mathfrak{P}^{=}_{T_i}[\![\cdot]\!]$, for $i = 1, 2$. Additionally, let T_2 be equipped with a care function $\mathfrak{C}_{T_i}[\![\cdot]\!]$ operating with respect to $\mathfrak{P}^{=}_{T_2}[\![\cdot]\!]$. Let $\Sigma = \Sigma_1 \cup \Sigma_2$, $T = T_1 \oplus T_2$ and let ϕ be a set of flat Σ-literals, which can be partitioned into a set ϕ_1 of Σ_1-literals and a set ϕ_2 of Σ_2-literals, with $V = vars(\phi_1) \cap vars(\phi_2)$. If $\Sigma_1^{\mathbb{S}} \cap \Sigma_2^{\mathbb{S}} = S_1 \cap S_2$, then following are equivalent*

1. *ϕ is T-satisfiable;*
2. *there exists some care-graph $\mathbf{G}_2$, and a corresponding arrangement $\delta_{\mathbf{G}_2}$, that are fix-point solutions of (1), such that the following sets are T_1- and T_2-satisfiable respectively*

$$\phi_1 \cup \mathfrak{P}^{=}_{T}[\![V]\!](\phi_1 \cup \phi_2 \cup \delta_{\mathbf{G}_2}) \ , \qquad \phi_2 \cup \mathfrak{P}^{=}_{T}[\![V]\!](\phi_1 \cup \phi_2 \cup \delta_{\mathbf{G}_2}) \ .$$

Moreover, T is stably-infinite with respect to $S_1 \cup S_2$.

Proof. (1) $\Rightarrow$ (2) : Suppose $\phi = \phi_1 \cup \phi_2$ is T-satisfiable in a T-interpretation $\mathcal{A}$. Let δ_V be the full arrangement over V satisfied by $\mathcal{A}$. Since δ_V trivially is a fix-point of (1), $\mathcal{A}$ satsifies δ_V, and the propagator only adds formulas that are entailed, it is clear that $\mathcal{A}$ satisfies both sets of formulas, which proves one direction.

(2) $\Leftarrow$ (1) : Assume that there is a T_1-interpretation $\mathcal{A}_1$ and a T_2-interpretation $\mathcal{A}_2$ (and assume wlog that both interpret all the variables in V) such that $\mathcal{A}_1 \vDash_{T_1} \phi_1 \cup \mathfrak{P}^{=}_{T}[\![V]\!](\phi_1 \cup \phi_2 \cup \delta_{\mathbf{G}_2})$ and $\mathcal{A}_2 \vDash_{T_2} \phi_2 \cup \mathfrak{P}^{=}_{T}[\![V]\!](\phi_1 \cup \phi_2 \cup \delta_{\mathbf{G}_2})$. Let δ_V be the arrangement over the complete graph on V satisfied by $\mathcal{A}_1$, so

$$\delta_{\mathbf{G}_2} \subseteq \mathfrak{P}^{=}_{T_2}[\![V]\!](\phi_2 \cup \delta_{\mathbf{G}_2}) \subseteq \mathfrak{P}^{=}_{T}[\![V]\!](\phi_1 \cup \phi_2 \cup \delta_{\mathbf{G}_2}) \subseteq \delta_V \ .$$

Because $\mathbf{G}_2$ is a fix-point, we know that $\mathfrak{C}_{T_2}[\![V]\!](\phi_2 \cup \mathfrak{P}^{=}_{T}[\![V]\!](\phi_1 \cup \phi_2 \cup \delta_{\mathbf{G}})) = \langle V, \emptyset \rangle$. We then know, by property 3 of the care function, that there is a T_2-interpretation $\mathcal{B}_2$ such that $\mathcal{B}_2 \vDash_{T_2} \phi_2 \cup \delta_V$. Since δ_V is a complete arrangement over all the shared variables and we also have that $\mathcal{A}_1 \vDash_{T_1} \phi_1 \cup \delta_V$, we can now appeal to the correctness of the standard Nelson-Oppen combination method to obtain a T-interpretation $\mathcal{C}$ that satisfies $\phi_1 \cup \phi_2 = \phi$. The proof that the combined theory is stably-infinite can be found in [12]. $\square$

3.2 Extension to Polite Combination

The method described in Section 3 relies on the correctness argument for the standard Nelson-Oppen method, meaning that the theories involved should be stably-infinite for completeness. A more general combination method based on the notion of *polite* theories (and not requiring that both theories be stably-infinite) was introduced in [18] and clarified in [12, 13]. Here, we assume familiarity with the concepts appearing in those papers, and show how they can be integrated into the combination method of this paper.

Assume that the theory T_2 is polite with respect to the set of sorts S_2 such that $\Sigma_1 \cap \Sigma_2 \subseteq S_2$, and is equipped with a witness function $\mathit{witness}_2$. We modify the combination method of Section 3.1 as follows:

1. In the **Arrange** and **Check** phases, instead of using ϕ_2, we use the formula produced by the witness function, i.e. $\phi'_2 = \mathit{witness}_2(\phi_2)$.
2. We define $V = \mathit{vars}_S(\phi'_2)$ instead of $V = \mathit{vars}(\phi_1) \cap \mathit{vars}(\phi_2)$.

Theorem 2. *Let T_i be a Σ_i-theory polite with respect to the set of sorts S_i, and equipped with equality propagators $\mathfrak{P}^{=}_{T_i}[\![\cdot]\!]$, for $i = 1, 2$. Additionally, let T_2 be equipped with a care function $\mathfrak{C}_{T_2}[\![\cdot]\!]$ operating with respect to $\mathfrak{P}^{=}_{T_2}[\![\cdot]\!]$. Let $\Sigma = \Sigma_1 \cup \Sigma_2$, $T = T_1 \oplus T_2$ and let ϕ be a set of flat Σ-literals, which can be partitioned into a set ϕ_1 of Σ_1-literals and a set ϕ_2 of Σ_2-literals. Let $\phi'_2 = \mathit{witness}_{T_2}(\phi_2)$ and $V = \mathit{vars}_S(\phi'_2)$. If $S \subseteq S_1 \cap S_2$, then following are equivalent*

1. *ϕ is T-satisfiable;*
2. *there exists a care-graph $\mathbf{G}_2$ and arrangement $\delta_{\mathbf{G}_2}$, fix-point solutions of (1), such that the following sets are T_1- and T_2-satisfiable respectively*

$$\phi_1 \cup \mathfrak{P}^{=}_{T}[\![V]\!](\phi_1 \cup \phi'_2 \cup \delta_{\mathbf{G}_2}) \ , \qquad \phi'_2 \cup \mathfrak{P}^{=}_{T}[\![V]\!](\phi_1 \cup \phi'_2 \cup \delta_{\mathbf{G}_2}) \ .$$

Moreover, T is polite with respect to $S_1 \cup (S_2 \setminus \Sigma_1)$. [2]

4 Theory of Uninterpreted Functions

The theory of uninterpreted functions over a signature $\Sigma_{\mathtt{euf}}$ is the theory $T_{\mathtt{euf}} = (\Sigma_{\mathtt{euf}}, \mathbf{A})$, where $\mathbf{A}$ is simply the class of all $\Sigma_{\mathtt{euf}}$-structures. Conjunctions of

[2] The remaining proofs are relegated to the technical report [14] due to space constraints.

literals in this theory can be decided in polynomial time by congruence closure algorithms (e.g. [19]). We make use of insights from these algorithms in defining both the equality propagator and the care function. For simplicity, we assume $\Sigma_{\mathtt{euf}}$ contains no predicate symbols, but the extension to the case with predicate symbols is straightforward.

Equality Propagator. Let ϕ be a set of flat literals, let V be a set of variables, and let $\sim_c$ be the smallest congruence relation[3] over terms in ϕ containing $\{(x,t) \mid x = t \in \phi\}$. We define a dis-equality relation $\neq_c$ as the smallest relation satisfying

$$x \sim_c x' \wedge y \sim_c y' \wedge x' \neq y' \in \phi \Longrightarrow x \neq_c y \enspace .$$

Now, we define the equality propagator as

$$\mathfrak{P}^{=}_{\mathtt{euf}}[\![V]\!](\phi) = \{x = y \mid x, y \in V, x \sim_c y\} \cup \{x \neq y \mid x, y \in V, x \neq_c y\}.$$

It is easy to see that $\mathfrak{P}^{=}_{\mathtt{euf}}[\![\cdot]\!]$ is indeed an equality propagator. Moreover, it can easily be implemented as part of a decision procedure based on congruence closure.

Example 2. Given the set $\phi = \{\ x = z,\ y = f(a),\ z \neq f(a)\ \}$, the equality propagator would return $\mathfrak{P}^{=}_{\mathtt{euf}}[\![\{\ x, y\ \}]\!](\phi) = \{\ x = x,\ y = y,\ x \neq y,\ y \neq x\ \}$.

Care Function. The definition of the care function is based on the fact that during congruence closure, we only care about equalities between pairs of variables that occur as arguments in the same position of the same function symbol. Again, let V be a set of variables and let ϕ be a set of flat literals, such that ϕ only contains function symbols from $F = \{f_1, f_2, \ldots, f_n\}$.

For a set of formulas ϕ, let $\mathcal{E}(\phi)$ denote the smallest equivalence relation over the terms occurring in ϕ containing $\{(x,t) \mid x = t \in \phi\}$. For an equivalence relation E, let E^* denote the *congruence closure* of E (i.e. the smallest congruence relation containing E). In order to make our care-function more precise, we will first approximate the implications that possible equalities over variables in V could trigger. We do so by taking all possible equalities over V, i.e. let $\delta^{=}_{V}$ be the full arrangement over the shared variables where all variables of the same sort are equal. Now, to see what these equalities could imply, we let $E^{=}_{\phi} = \mathcal{E}(\phi \cup \delta^{=}_{V})^*$.

For each function symbol $f \in F$ of arity $\sigma_1 \times \sigma_2 \times \cdots \times \sigma_k \mapsto \sigma$, let E_f be a set containing pairs of variables that could trigger an application of congruence because of two terms that are applications of f. More precisely, let $E_f \subseteq V \times V$ be a maximal set of pairs $(x,y) \in V \times V$, that are not already decided by the propagator ($x \nsim_c y$ and $\neg x \neq_c y$), such that for each $(x,y) \in E_f$ we have:

1. there are x_i and y_i such that $x \sim_c x_i$ and $y \sim_c y_i$;

[3] In this context, a congruence relation is an equivalence relation that also satisfies the congruence property: if $f(x_1,\ldots,x_n)$ and $f(y_1,\ldots,y_n)$ are terms in ϕ, and if for each $1 \leq i \leq n$, $x_i \sim_c y_i$, then $f(x_1,\ldots,x_n) \sim_c f(y_1,\ldots,y_n)$.

2. there are terms $f(x_1, \ldots, x_i, \ldots, x_k) \not\sim_c f(y_1, \ldots, y_i, \ldots, y_k)$ in ϕ;
3. for $1 \leq j \leq k$, variables x_j and y_j could become equal, $(x_j, y_j) \in E^{=}_{\phi}$;
4. for $1 \leq j \leq k$, variables x_j and y_j are not known to be disequal, $\neg(x_j \neq_c y_j)$.

Now, we let $E = \bigcup_{f \in F} E_f$, and define the care function mapping ϕ to the care graph $\mathbf{G}$ as $\mathfrak{C}_{\mathsf{euf}}[\![V]\!](\phi) = \mathbf{G} = \langle V, E \rangle$.

Example 3. Consider the following sets of literals

$$\phi_1 = \{f(x_1) \neq f(y_1), y_1 = x_2\} \quad ,$$
$$\phi_2 = \{z_1 = f(x_1), z_2 = f(y_1), g(z_1, x_2) \neq g(z_2, y_2)\} \quad ,$$
$$\phi_3 = \{y_1 = f(x_1), y_2 = f(x_2), z_1 = g(x_1), z_2 = g(x_2), h(y_1) \neq h(z_1)\} \quad .$$

and corresponding sets of shared variables $V_1 = \{x_1, x_2\}$, $V_2 = \{x_1, x_2, y_1, y_2\}$, $V_3 = \{x_1, x_2, y_2, z_2\}$. The care function above would return the care graphs $\mathbf{G}_1 = \langle V, \{(x_1, x_2)\}\rangle$, $\mathbf{G}_2 = \langle V, \{(x_1, y_1), (x_2, y_2)\}\rangle$, and $\mathbf{G}_2 = \langle V, \{(x_1, x_2)\}\rangle$.

Note that the the care function for ϕ_3 does not return the pair (y_2, z_2), which is important in case x_1 and x_2 become equal. This is remedied in the procedure itself, by computing the fix-point, which, in case we choose $x_1 = x_2$, will add the pair (y_2, z_2) to the care graph in the second step.

Theorem 3. *Let T_{euf} be the theory of uninterpreted functions with equality over the signature Σ_{euf}. $\mathfrak{C}_{\mathsf{euf}}[\![\cdot]\!]$ is a care function for T_{euf} with respect to the equality propagator $\mathfrak{P}^{=}_{\mathsf{euf}}[\![\cdot]\!]$.*

5 Theory of Arrays

The extensional theory of arrays T_{arr} operates over the signature Σ_{arr} that contains the sorts $\{\mathsf{array}, \mathsf{index}, \mathsf{elem}\}$ and function symbols

$$\mathsf{read} : \mathsf{array} \times \mathsf{index} \mapsto \mathsf{elem} \quad , \qquad \mathsf{write} : \mathsf{array} \times \mathsf{index} \times \mathsf{elem} \mapsto \mathsf{array} \quad ,$$

where read represents reading from an array at a given index, and write represents writing a given value to an array at an index. The semantics of the theory are given by the three axioms:

1. $\forall\, a{:}\mathsf{array}.\ \forall\, i{:}\mathsf{index}.\ \forall\, v{:}\mathsf{elem}.\ \ \mathsf{read}(\mathsf{write}(a, i, v), i) = v$,
2. $\forall\, a{:}\mathsf{array}.\ \forall\, i, j{:}\mathsf{index}.\ \forall\, v{:}\mathsf{elem}.\ \ i \neq j \rightarrow \mathsf{read}(\mathsf{write}(a, i, v), j) = \mathsf{read}(a, j)$,
3. $\forall\, a, b{:}\mathsf{array}.\ \ (\forall\, i{:}\mathsf{index}.\ \mathsf{read}(a, i) = \mathsf{read}(b, i)) \rightarrow a = b$.

The flat literals of the theory are of the form $x = \mathsf{read}(a, i)$, $a = \mathsf{write}(b, i, x)$, $i = j$, $i \neq j$, $x = y$, $x \neq y$, $a = b$, $a \neq b$, where here and below we use the convention that x, y, v are variables of sort elem, i, j are variables of sort index, a, b, and c are variables of sort array, and w, z are variables of any sort. For a set ϕ of flat T_{arr}-literals, we also define $\alpha(\phi)$ to be the subset of ϕ that does not contain literals of the form $a = \mathsf{write}(b, i, v)$.

Decision Procedure. Before presenting the equality propagator and care function, it will be helpful to present a simple rule-based decision procedure for $T_{\mathtt{arr}}$ based on [9].[4] Given a set Γ of flat $T_{\mathtt{arr}}$-literals, we define $\approx_a^\Gamma$ as $\mathcal{E}\left(\alpha(\Gamma)\right)^*$ and the corresponding disequality relation $\neq_a^\Gamma$ as the smallest relation satisfying:

$$w \approx_a^\Gamma w' \wedge z \approx_a^\Gamma z' \wedge w \neq z \in \Gamma \implies w' \neq_a^\Gamma z' \ .$$

Additionally, let $\Gamma[l_1, \ldots, l_n]$ denote that literals $l_i, 1 \leq i \leq n$ appear in Γ, and for every pair (a, b) of variables in $\mathit{vars}_{\mathsf{array}}(\Gamma)$, let $k_{a,b}$ be a distinguished fresh variable of sort index. Let $\mathcal{D}_{\mathtt{arr}}$ be the following set of inference rules.

$$\mathsf{RIntro1} \quad \frac{\Gamma[a = \mathsf{write}(b, i, v)]}{\Gamma, v = \mathsf{read}(a, i)} \quad \text{if} \quad v \not\approx_a^\Gamma \mathsf{read}(a, i)$$

$$\mathsf{RIntro2} \quad \frac{\Gamma[a = \mathsf{write}(b, i, v), x = \mathsf{read}(c, j)]}{\Gamma, i = j \qquad \Gamma, \mathsf{read}(a, j) = \mathsf{read}(b, j)} \quad \text{if} \quad \begin{array}{l} a \approx_a^\Gamma c \text{ or } b \approx_a^\Gamma c, \\ i \not\approx_a^\Gamma j, \\ \mathsf{read}(a, j) \not\approx_a^\Gamma \mathsf{read}(b, j) \end{array}$$

$$\mathsf{ArrDiseq} \quad \frac{\Gamma[a \neq b]}{\Gamma, \mathsf{read}(a, k_{a,b}) \neq \mathsf{read}(b, k_{a,b})} \quad \text{if} \quad \neg(\mathsf{read}(a, k_{a,b}) \neq_a^\Gamma \mathsf{read}(b, k_{a,b}))$$

Note that non-flat literals appear in the conclusions of rules $\mathsf{RIntro2}$ and $\mathsf{ArrDiseq}$. We use this as a shorthand for the flattened version of these literals. For example, $\mathsf{read}(a, j) = \mathsf{read}(b, j)$ is shorthand for $x = \mathsf{read}(a, j) \wedge y = \mathsf{read}(b, j) \wedge x = y$, where x and y are fresh variables (there are other possible flattenings, especially if one or more of the terms appears already in Γ, but any of them will do). We say that a set Γ of literals is $\mathcal{D}_{\mathtt{arr}}$*-saturated* if no rules from $\mathcal{D}_{\mathtt{arr}}$ can be applied.

Theorem 4. *The inference rules of $\mathcal{D}_{\mathtt{arr}}$ are sound and terminating.*

Theorem 5. *Let Γ be a $\mathcal{D}_{\mathtt{arr}}$-saturated set of flat $T_{\mathtt{arr}}$-literals. Then Γ is $T_{\mathtt{arr}}$-satisfiable iff $\alpha(\Gamma)$ is $T_{\mathtt{euf}}$-satisfiable.*

Equality Propagator. Let ϕ be a set of flat literals and V a set of variables. Consider the following modified versions of $\mathsf{RIntro2}$ that are enabled only if one of the branches can be ruled out as unsatisfiable:

$$\mathsf{RIntro2a} \quad \frac{\Gamma[a = \mathsf{write}(b, i, v), x = \mathsf{read}(c, j)]}{\Gamma, i = j} \quad \text{if} \quad \begin{array}{l} a \approx_a^\Gamma c \text{ or } b \approx_a^\Gamma c, \\ i \not\approx_a^\Gamma j, \\ \mathsf{read}(a, j) \neq_a^\Gamma \mathsf{read}(b, j) \end{array}$$

$$\mathsf{RIntro2b} \quad \frac{\Gamma[a = \mathsf{write}(b, i, v), x = \mathsf{read}(c, j)]}{\Gamma, \mathsf{read}(a, j) = \mathsf{read}(b, j)} \quad \text{if} \quad \begin{array}{l} a \approx_a^\Gamma c \text{ or } b \approx_a^\Gamma c, \\ i \neq_a^\Gamma j, \\ \mathsf{read}(a, j) \not\approx_a^\Gamma \mathsf{read}(b, j) \end{array}$$

Let $\mathcal{D}'_{\mathtt{arr}}$ be obtained from $\mathcal{D}_{\mathtt{arr}}$ by replacing $\mathsf{RIntro2}$ with the above rules. Since these rules mimic $\mathsf{RIntro2}$ when they are enabled, but are enabled less often, it is clear that $\mathcal{D}'_{\mathtt{arr}}$ remains sound and terminating. Let Γ' be the result of applying

[4] The main difference is that in our procedure, we exclude literals containing write from the $T_{\mathtt{euf}}$-satisfiability check as they are not needed and this allows us to have a simpler care function.

$\mathcal{D'}_{\texttt{arr}}$ until no more apply (we say that Γ' is $\mathcal{D'}_{\texttt{arr}}$-saturated). We define the equality propagator as:

$$\mathfrak{P}^{=}_{\texttt{arr}}[\![V]\!](\phi) = \{w = z \mid w, z \in V, w \approx^{\Gamma'}_{a} z\} \cup \{w \neq z \mid w, z \in V, w \neq^{\Gamma'}_{a} z\}.$$

It is easy to see that $\mathfrak{P}^{=}_{\texttt{arr}}[\![\cdot]\!]$ satisfies the requirements for a propagator. Though not necessary for the care function we present here, a more powerful propagator can be obtained by additionally performing congruence closure over write terms.

Care Function. Let ϕ be a set of flat literals and V a set of variables. First, since a simple propagator cannot compute all equalities between array variables, we will ensure that the relationships between all pairs of array variables in V have been determined. To do so we define the set E^{ϕ}_{a} of pairs of array variables in V that are not yet known equal or dis-equal

$$E^{\phi}_{a} = \{(a, b) \in V \times V \mid a \not\approx^{\phi}_{a} b \wedge \neg(a \neq^{\phi}_{a} b)\} \ .$$

Next, since the inference rules can introduce new read terms, we compute the smallest set R^{ϕ} with possible such terms, i.e

- if $x = \mathsf{read}(a, i) \in \phi$ or $a = \mathsf{write}(b, i, v) \in \phi$, then $\mathsf{read}(a, i) \in \mathsf{R}^{\phi}$,
- if $a = \mathsf{write}(b, i, v) \in \phi$, $\mathsf{read}(c, j) \in \mathsf{R}^{\phi}$, $i \not\approx^{\phi}_{a} j$, and $a \approx^{\phi}_{a} c \vee b \approx^{\phi}_{a} c$, then both $\mathsf{read}(a, j) \in \mathsf{R}^{\phi}$ and $\mathsf{read}(b, j) \in \mathsf{R}^{\phi}$,
- if $a \neq b \in \phi$, then both $\mathsf{read}(a, k_{a,b}) \in \mathsf{R}^{\phi}$ and $\mathsf{read}(b, k_{a,b}) \in \mathsf{R}^{\phi}$.

Crucial in the introduction of the above read terms, is the set of index variables whose equality could affect the application of the $\mathsf{RIntro2}$ rule. We capture these variables by defining the set E^{ϕ}_{i} as the set of all pairs (i, j) such that:

- $i \not\approx^{\phi}_{a} j$ and $\neg(i \neq^{\phi}_{a} j)$
- $\exists\, a, b, c, v.\ a = \mathsf{write}(b, i, v) \in \phi, \mathsf{read}(c, j) \in \mathsf{R}^{\phi}$, and $a \approx^{\phi}_{a} c \vee b \approx^{\phi}_{a} c$.

Finally, we claim that with the variables in E^{ϕ}_{a} and E^{ϕ}_{i} decided, we can essentially use the same care function as for $T_{\texttt{euf}}$, treating read as uninterpreted. We therefore define the third set E^{ϕ}_{r} to be the set of all pairs $(i, j) \in V \times V$ of undecided indices, $i \not\approx^{\phi}_{a} j$ and $\neg(i \neq^{\phi}_{a} j)$, such that there are a, b, i', j' with $a \approx^{\phi}_{a} b$, $i \approx^{\phi}_{a} i'$, $j \approx^{\phi}_{a} j'$, $\mathsf{read}(a, i') \in \mathsf{R}^{\phi}$, $\mathsf{read}(b, j') \in \mathsf{R}^{\phi}$, $\mathsf{read}(a, i') \not\approx^{\phi}_{a} \mathsf{read}(b, j')$.

With the definitions above, we can define the care graph as $\mathfrak{C}_{\texttt{arr}}[\![V]\!](\phi) = \mathbf{G} = \langle V, E\rangle$, where the set of edges is defined as

$$E = \begin{cases} E^{\phi}_{a} & \text{if } E^{\phi}_{a} \neq \emptyset, \\ E^{\phi}_{i} & \text{if } E^{\phi}_{i} \neq \emptyset, \text{ and} \\ E^{\phi}_{r} & \text{otherwise.} \end{cases}$$

Note that as defined, E^{ϕ}_{i} may include pairs of index variables, one or more of which are not in V. Unfortunately, the care function fails if E^{ϕ}_{i} is not a subset of $V \times V$. We can ensure that it is either by expanding the set V until it includes all variables in E^{ϕ}_{i} or doing additional case-splitting up front on pairs in E^{ϕ}_{i}, adding formulas to ϕ, until $E^{\phi}_{i} \subseteq V \times V$.

Theorem 6. *Let T_{arr} be the theory of arrays. $\mathfrak{C}_{\mathsf{arr}}[\![\cdot]\!]$ is a care function for T_{arr} with respect to the equality propagator $\mathfrak{P}^{=}_{\mathsf{arr}}[\![\cdot]\!]$ for all sets ϕ of literals and V of variables such that $E_i^{\phi} \subseteq V \times V$.*

Example 4. Consider the following constraints involving arrays and bit-vectors of size m, where $\times_m$ denotes unsigned bit-vector multiplication:

$$\bigwedge_{k=1}^{n} (\mathsf{read}(a_k, i_k) = \mathsf{read}(a_{k+1}, i_{k+1}) \wedge i_k = x_k \times_m x_{k+1}) \quad . \tag{2}$$

Assume that only the index variables are shared, i.e. $V = \{i_1, \ldots, i_{n+1}\}$. In this case, both E_a^{ϕ} and E_i^{ϕ} will be empty and the only read terms in R^{ϕ} will be those appearing in the formula. Since none of these are reading from equivalent arrays, the empty care graph is a fix-point for our care function, and we do not need to guess an arrangement.

Note that in the case when V contains array variables, the care graph requires us to split on all pairs of these variables (i.e. the care function is trivial over these variables). Fortunately, in practice it appears that index and element variables are typically shared, and only rarely are array variables shared.

6 Experimental Evaluation

We implemented the new method in the Cvc3 solver [2], and in the discussion below, we denote the new implementation as Cvc3+C. We focused our attention on the combination of the theory of arrays and the theory of fixed-size bit-vectors (QF_AUFBV). This seemed like a good place to start because there are many benchmarks which generate a non-trivial number of shared variables, and additional splits on shared bit-vector variables can be quite expensive. This allowed us to truly examine the merits of the new combination method. In order to evaluate our method against the current state-of-the-art, we compared to Boolector [4], Yices [10], Cvc3, and MathSAT [5], the top solvers in the QF_AUFBV category from the 2009 SMT-COMP competition (in order). Additionally, we included the Z3 solver [8] so as to compare to the model-based theory combination method [7]. All tests were conducted on a dedicated Intel Pentium E2220 2.4 GHz processor with 4GB of memory. Individual runs were limited to 15 minutes.

We crafted a set of new benchmarks based on Example 4 from Section 5, taking $n = 10, \ldots, 100$, with increments of 10, and $m = 32, \ldots, 128$, with increments of 32. We also included a selection of problems from the QF_AUFBV division of the SMT-LIB library. Since most of the benchmarks in the library come from model-checking of software and use a flat memory model, they mostly operate over a single array representing the heap. Our method is essentially equivalent to the standard Nelson-Oppen approach for such benchmarks, so we selected only the benchmarks that involved constraints over at least two arrays. We anticipate that such problems will become increasingly important as static-analysis tools become more precise and are able to infer separation of the heap (in the style

Table 1. Experimental results.

	Boolector		Yices		MathSAT		Z3		Cvc3		Cvc3+C	
`crafted` (40)	2100.13	40	6253.32	34	468.73	30	112.88	40	388.29	9	**14.22**	**40**
`matrix` (11)	1208.16	10	683.84	6	474.89	4	927.12	11	831.29	11	**45.08**	**11**
`unconstr` (10)	**3.00**	**10**		0	706.02	3	54.60	2	185.00	5	340.27	8
`copy` (19)	11.76	19	**1.39**	**19**	1103.13	19	4.79	19	432.72	17	44.75	19
`sort` (6)	691.06	6	557.23	4	82.21	4	248.94	3	**44.89**	**6**	**44.87**	**6**
`delete` (29)	**3407.68**	**18**	1170.93	10	2626.20	14	1504.46	10	1766.91	17	1302.32	17
`member` (24)	2807.78	24	185.54	24	217.35	24	**112.23**	**24**	355.41	24	320.80	24
	10229.57	**127**	8852.25	97	5678.53	98	2965.02	109	4004.51	89	2112.31	125

of Burstall, e.g. [17]). All the benchmarks and the Cvc3 binaries used in the experiments are available from the authors' website.[5]

The combined results of our experiments are presented in Table 1, with columns reporting the total time (in seconds) that a solver used on the problem instances it solved (not including time spent on problem instances it was unable to solve), and the number of solved instances. Compared to Cvc3, the new implementation Cvc3+C performs uniformly better. On the first four classes of problems, Cvc3+C greatly outperforms Cvc3. On the last three classes of problems, the difference is less significant. After examining the benchmarks, we concluded that the multitude of arrays in these examples is artificial – the many array variables are just used for temporary storage of sequential updates on the same starting array – so there is not a great capacity for improvement using the care function that we described. A scatter-plot comparison of Cvc3 vs Cvc3+C is shown in Figure 1(a). Because the only difference between the two implementations is the inclusion of the method described in this paper, this graph best illustrates the performance impact this optimization can have.

When compared to the other solvers, we find that whereas Cvc3 is not particularly competitive, Cvc3+C is very competitive and in fact, for several sets of benchmarks, performs better than all of the others. This again emphasizes the strength of our results and suggests that combination methods can be of great importance for performance and scalability of modern solvers. Overall, on this set of benchmarks, Boolector solves the most (solving 2 more than Cvc3+C). However, Cvc3+C is significantly faster on the benchmarks it solves. Figure 1(b) shows a scatter-plot comparison of Cvc3+C against Boolector.

7 Conclusion

We presented a reformulation of the classic Nelson-Oppen method for combining theories. The most notable novel feature of the new method is the ability to leverage the structure of the individual problems in order to reduce the complexity of finding a common arrangement over the interface variables. We do this by defining theory-specific care functions that determine the variable pairs

[5] http://cs.nyu.edu/∼dejan/sharing-is-caring/

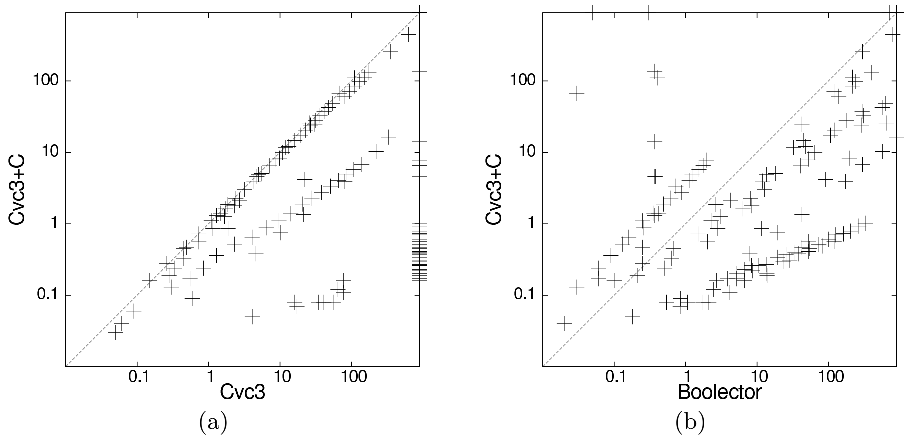

Fig. 1. Comparison of Cvc3, Cvc3+C and Boolector. Both axes use a logarithmic scale and each point represents the time needed to solve an individual problem.

that are relevant in a specific problem. We proved the method correct, and presented care functions for the theories of uninterpreted functions and arrays. We draw intuition for the care functions and correctness proofs directly from the decision procedures for specific theories, leaving room for new care functions backed by better decision algorithms. Another benefit of the presented method is that it is orthogonal to the previous research on combinations of theories. For example, it would be easy to combine our method with a model-based combination approach–instead of propagating all equalities between shared variables implied by the model, one could restrict propagation to only the equalities that correspond to edges in the care graph, gaining advantages from both methods.

We also presented an experimental evaluation of the method, comparing the new method to a standard Nelson-Oppen implementation and several state-of-the art solvers. Compared to the other solvers on a selected set of benchmarks, the new method performs competitively, and shows a robust performance increase over the standard Nelson-Oppen implementation.

References

1. C. Barrett, R. Nieuwenhuis, A. Oliveras, and C. Tinelli. Splitting on demand in SAT Modulo Theories. In *Logic for Programming, Artificial Intelligence, and Reasoning*, volume 4246 of *LNCS*, pages 512–526. Springer, 2006.
2. C. Barrett and C. Tinelli. CVC3. In *Computer Aided Verification*, volume 4590 of *LNCS*, pages 298–302. Springer-Verlag, 2007.
3. M. Bozzano, R. Bruttomesso, A. Cimatti, T. Junttila, S. Ranise, P. van Rossumd, and R. Sebastiani. Efficient theory combination via Boolean search. *Information and Computation*, 204(10):1493–1525, 2006.

4. R. Brummayer and A. Biere. Boolector: An efficient SMT solver for bit-vectors and arrays. In *Tools and Algorithms for the Construction and Analysis of Systems*, volume 5505 of *LNCS*, pages 174–177. Springer, 2009.
5. R. Bruttomesso, A. Cimatti, A. Franzén, A. Griggio, and R. Sebastiani. The MathSAT 4 SMT solver. In *Computer Aided Verification*, volume 5123 of *LNCS*, pages 299–303. Springer, 2008.
6. R. Bruttomesso, A. Cimatti, A. Franzen, A. Griggio, and R. Sebastiani. Delayed theory combination vs. Nelson-Oppen for satisfiability modulo theories: A comparative analysis. *Annals of Mathematics and Artificial Intelligence*, 55(1):63–99, 2009.
7. L. de Moura and N. Bjørner. Model-based Theory Combination. In *5th International Workshop on Satisfiability Modulo Theories*, volume 198 of *Electronic Notes in Theoretical Computer Science*, pages 37–49. Elsevier, 2008.
8. L. de Moura and N. Bjørner. Z3: An Efficient SMT Solver. In *Tools and Algorithms for the Construction and Analysis of Systems*, volume 4963 of *LNCS*, page 337. Springer, 2008.
9. L. de Moura and N. Bjørner. Generalized, efficient array decision procedures. In *Formal Methods in Computer-Aided Design, 2009*, pages 45–52. IEEE, November 2009.
10. B. Dutertre and L. de Moura. The YICES SMT Solver. *Tool paper at http://yices. csl. sri. com/tool-paper. pdf*, 2006.
11. H. B. Enderton. *A mathematical introduction to logic.* Academic press New York, 1972.
12. D. Jovanović and C. Barrett. Polite theories revisited. Technical Report TR2010-922, Department of Computer Science, New York University, Jan. 2010.
13. D. Jovanović and C. Barrett. Polite theories revisited. In *Logic for Programming, Artificial Intelligence, and Reasoning*, volume 6397 of *LNCS*, pages 402–416. Springer Berlin / Heidelberg, 2010.
14. D. Jovanović and C. Barrett. Sharing is Caring: Combination of Theories. Technical Report TR2011-940, New York University, 2011.
15. G. Nelson and D. C. Oppen. Simplification by cooperating decision procedures. *ACM Transactions on Programming Languages and Systems*, 1(2):245–257, Oct. 1979.
16. D. C. Oppen. Complexity, convexity and combinations of theories. *Theoretical Computer Science*, 12(3):291–302, 1980.
17. Z. Rakamarić and A. J. Hu. A Scalable Memory Model for Low-Level Code. In *Verification, Model Checking, and Abstract Interpretation*, LNCS, page 304. Springer-Verlag, 2009.
18. S. Ranise, C. Ringeissen, and C. G. Zarba. Combining Data Structures with Nonstably Infinite Theories Using Many-Sorted Logic. In *Frontiers of Combining Systems*, volume 3717 of *LNCS*, pages 48–64. Springer, 2005.
19. R. E. Shostak. An algorithm for reasoning about equality. In *5th international joint conference on Artificial intelligence*, pages 526–527. Morgan Kaufmann Publishers Inc., 1977.
20. C. Tinelli and M. T. Harandi. A new correctness proof of the Nelson–Oppen combination procedure. In *Frontiers of Combining Systems*, Applied Logic, pages 103–120. Kluwer Academic Publishers, Mar. 1996.
21. C. Tinelli and C. Zarba. Combining decision procedures for sorted theories. In *Logic in Artificial Intelligence*, volume 3229 of *LNAI*, pages 641–653. Springer, 2004.