Sharing is Caring

Dejan Jovanović
New York University
dejan@cs.nyu.edu

Clark Barrett
New York University
barrett@cs.nyu.edu

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 theories 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 [12] introduced a general framework for combining 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 [15, 11].

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 extensive use of theory propagation and then using 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 [13]. 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[1]. Trying to minimize the burden of non-deterministic guessing is thus of the utmost importance for

[1] If the two theories can decided in time $O(\mathcal{T}_1(n))$ and $O(\mathcal{T}_2(n))$, the combination can be decided in $O(2^{n^2} \times (\mathcal{T}_1(n) + \mathcal{T}_2(n)))$.

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 improvements over the current state-of-the art.

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 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).

The paper is organized as follows. In Section 2 we introduce the background and some necessary notation. We then proceed to present the new combination method and prove its correctness in Section 3. The care functions for the theories of uninterpreted functions and arrays are presented in Section 4 and Section 5 respectively. We present experimental results in Section 6, and conclude in Section 7.

Due to space limitations, proofs of some of the results in this paper are omitted. The full paper with proofs is available from the authors as a technical report.[2]

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 [10, 19].

Syntax. 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[3] 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 $x = y$, $x \neq y$, $x = f(y_1,\ldots,y_n)$,

[2] All proofs are included in an appendix for the benefit of SMT reviewers.

[3] 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.

$p(y_1,\ldots,y_n)$, 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 which case the intended meaning is the conjunction $\bigwedge\Phi$ of the formulas in the set.

Semantics. 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,[4] 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 Σ-*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. A Σ-formula ϕ is *satisfiable* iff it evaluates to `true` in some Σ-interpretation over $vars(\phi)$.

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.

Theories. 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. [18] and Definition 2 in [15]).

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 $\mathcal{A} \models_T \phi$, or just $\mathcal{A} \models \phi$ 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 [15]).

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* $\mathcal{A}^{\Sigma_2,\emptyset} \in \mathbf{A_2}\}$*.*

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 as black boxes the known procedures for T_1 and T_2. The Nelson-Oppen combination method [12, 18, 19] 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

[4] In the rest of the paper we will use the calligraphic letters $\mathcal{A}, \mathcal{B}, \ldots$ to denote interpretations, and the corresponding subscripted Roman letters $A_\sigma, B_\sigma, \ldots$ to denote the domains of the interpretations.

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 [15, 18].

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. Given two care graphs $\mathbf{G}_1 = \langle V_1, E_1 \rangle$ and $\mathbf{G}_2 = \langle V_2, E_2 \rangle$, we denote the union graph as $\mathbf{G}_1 \cup \mathbf{G}_2 = \langle V_1 \cup V_2, E_1 \cup E_2 \rangle$

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 Nelson-Oppen 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 [13]. In the many-sorted setting stable-infiniteness is defined with respect to a subset of the signature sorts (see [19]).

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, making way for a more efficient mechanism of 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 [15, 11], 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.

3.1 Equality Propagation

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:*

$$\mathfrak{P}^{=}_{T}[\![V]\!](\phi) = \{x_1 = y_1, x_2 = y_2, \ldots, x_m = y_m\} \cup \{z_1 \neq w_1, z_2 \neq w_2, \ldots, z_n \neq w_n\} \ ,$$

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 $\phi \vDash_T x_i = y_i$; and*
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. $\phi \subseteq \psi \implies \mathfrak{P}^{=}_{T}[\![V]\!](\phi) \subseteq \mathfrak{P}^{=}_{T}[\![V]\!](\psi)$.*

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 it will be useful to have it propagate at least a basic set of equalities.

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 fixpoint 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 $\Sigma = \Sigma_1 \cup \Sigma_2$.*

Assume we are given a set of variables V and a set ϕ 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

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

where $\langle \psi_1^*, \psi_2^* \rangle$ is the least fixpoint of the following operator

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

The fixpoint exists as the propagators are monotone and the set V is finite. Moreover, the value of the propagator 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. $\mathfrak{P}^{=}_{T_1}[\![V]\!](\phi_1) \subseteq \mathfrak{P}^{=}_{T}[\![V]\!](\phi_1)$ and $\mathfrak{P}^{=}_{T_2}[\![V]\!](\phi_2) \subseteq \mathfrak{P}^{=}_{T}[\![V]\!](\phi_2)$.

3.2 Care Functions

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 $\phi \cup \delta_{\mathbf{G}}$ is T-satisfiable for an arrangement $\delta_{\mathbf{G}}$, then for any arrangement δ_V such that $\delta_V \supseteq \mathfrak{P}_T^{=}[\![V]\!](\phi \cup \delta_{\mathbf{G}})$, it holds that $\phi \cup \delta_V$ is also T-satisfiable.*

Definition 7 (Trivial Care Function)**.** *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 (i.e. $\mathfrak{C}_0[\![V]\!](\phi) = \langle V, V \times V \rangle$).*

Notice that $\mathfrak{C}_0[\![\cdot]\!]$ trivially satisfies the conditions of Definition 6 with respect to any equality propagator.

Definition 8 (Combined Care Function)**.** *Let T_1 and T_2 be two theories over the signatures Σ_1 and Σ_2, equipped with care functions $\mathfrak{C}_{T_1}[\![\cdot]\!]$ and $\mathfrak{C}_{T_2}[\![\cdot]\!]$ with respect to the equality propagators $\mathfrak{P}_{T_1}^{=}[\![\cdot]\!]$ and $\mathfrak{P}_{T_2}^{=}[\![\cdot]\!]$, respectively. Let $T = T_1 \oplus T_2$ and $\Sigma = \Sigma_1 \cup \Sigma_2$.*

Assume we are given a set of variables V and a set ϕ 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 care function $\mathfrak{C}_T[\![\cdot]\!]$ for the theory T as $\mathfrak{C}_T[\![V]\!](\phi) = \mathfrak{C}_{T_1}[\![V]\!](\phi_1) \cup \mathfrak{C}_{T_2}[\![V]\!](\phi_2)$.

Lemma 1. *The combined care function $\mathfrak{C}_T[\![\cdot]\!]$ is indeed a care function for theory T with respect to the combined propagator $\mathfrak{P}_T^{=}[\![\cdot]\!]$.*

3.3 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 and decidable, and that each T_i is equipped with a care function $\mathfrak{C}_{T_i}[\![\cdot]\!]$ operating with respect to an equality propagator $\mathfrak{P}_{T_i}^{=}[\![\cdot]\!]$. We designate $\mathfrak{C}_{T_2}[\![\cdot]\!]$ as the *active* care function. We are interested in deciding the combination theory $T = T_1 \oplus T_2$ over the signature $\Sigma = \Sigma_1 \cup \Sigma_2$. 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 $i = 1,2$. This step is identical to the first step in the standard Nelson-Oppen combination method.

Arrange: Let $V = vars(\phi_1) \cap vars(\phi_2)$ be the set of all variables shared by ϕ_1 and ϕ_2. We now non-deterministically choose a variable arrangement $\delta_{\mathbf{G}_2}$ over the care graph produced by the active care function: $\mathbf{G}_2 = \mathfrak{C}_{T_2}[\![V]\!](\phi_2)$.

Check: We 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 formulas are satisfiable we output satisfiable, otherwise we output unsatisfiable.

Example 1. *Consider the case of combining two theories T_1 and T_2 equipped with trivial care functions. 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 full arrangement δ_V over the set V of shared variables. Since equality propagators only propagate equalities

and dis-equalities over V, and all relationships between variables in V are already included in δ_V, the combined propagator will return 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 a care function $\mathfrak{C}_{T_i}[\![\cdot]\!]$ operating with respect to an equality propagator $\mathfrak{P}^{=}_{T_i}[\![\cdot]\!]$, for $i = 1,2$. 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 an arrangement $\delta_{\mathbf{G}_2}$ over the care graph $\mathbf{G}_2 = \mathfrak{C}_{T_2}[\![V]\!](\phi_2)$ 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 $S_1 \cup S_2$, and has combined care function $\mathfrak{C}_T[\![\cdot]\!]$ that operates with respect to the combined theory propagator $\mathfrak{P}^{=}_{T}[\![\cdot]\!]$.

Proof. $(1) \Rightarrow (2)$: Suppose $\phi = \phi_1 \cup \phi_2$ is T-satisfiable in a T-interpretation $\mathcal{A}$. Let $\delta_{\mathbf{G}_2}$ be the arrangement over $\mathbf{G}_2$ satisfied by $\mathcal{A}$. Since the combined 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

$$\mathcal{A}_1 \models_{T_1} \phi_1 \cup \mathfrak{P}^{=}_{T}[\![V]\!](\phi_1 \cup \phi_2 \cup \delta_{\mathbf{G}_2}) \ , \quad \mathcal{A}_2 \models_{T_2} \phi_2 \cup \mathfrak{P}^{=}_{T}[\![V]\!](\phi_1 \cup \phi_2 \cup \delta_{\mathbf{G}_2}) \ .$$

Let δ_V be the arrangement over the complete care graph satisfied by $\mathcal{A}_1$, which means that $\delta_V \supseteq \mathfrak{P}^{=}_{T}[\![V]\!](\phi_1 \wedge \phi_2 \wedge \delta_{\mathbf{G}_2}) \supseteq \mathfrak{P}^{=}_{T_2}[\![V]\!](\phi_2 \wedge \delta_{\mathbf{G}_2})$. By construction of δ_V and the property of the care function, we know there is a T_2-interpretation $\mathcal{B}_2$ such that $\mathcal{B}_2 \models_{T_2} \phi_2 \wedge \delta_V$. Since δ_V is a complete arrangement over all the shared variables and we also have that $\mathcal{A}_1 \models_{T_1} \phi_1 \wedge \delta_V$, we can now appeal to the correctness 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 [11], and the fact that the combined care function works was shown in Lemma 1. □

The extension of the method to the case of polite theories is straightforward and is presented in Appendix B.

4 Theory of Uninterpreted Functions

The theory of uninterpreted function 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 literals in this theory can be decided in polynomial time by congruence closure algorithms (e.g. [16]). 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 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

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

Now, we define the equality propagator as

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

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

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

Care Function. We now define the care function for the theory of uninterpreted functions. The definition 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 each function symbol $f \in F$ of arity $\sigma_1 \times \sigma_2 \times \cdots \times \sigma_k \mapsto \sigma$, let $E_f = \cup_{i=1}^{k} E_f^i$ with E_f^i being the set of all pairs of variables from V that occur as the i-th argument of function f in two terms occurring in ϕ that are not already equivalent according to $\sim_c$, i.e.:

$$E_f^i = \{ (x_i, y_i) \in V \times V \mid f(x_1, \ldots, x_i, \ldots, x_k) \nsim_c f(y_1, \ldots, y_i, \ldots, y_k) \} \ .$$

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

Example 3. *Given the set of literals* $\phi = \{ x_f = f(x), y_f = f(y) \}$ *with* $V = \{x, x_f, y, y_f\}$*, the care function would return* $\mathfrak{C}_{\mathtt{euf}}[\![V]\!](\phi) = \mathbf{G} = \langle V, \{(x,y)\} \rangle$*. The reason why the pair* (x_f, y_f) *does not need to be an edge in the care graph is as follows. If* $\delta_{\mathbf{G}}$ *includes* $x = y$*, then the propagator will discover that* $x_f = y_f$*. On the other hand, if* $\delta_{\mathbf{G}}$ *includes* $x \neq y$*, we can accept any relationship between* x_f *and* y_f.

Theorem 2. *Let* $T_{\mathtt{euf}}$ *be the theory of uninterpreted functions with equality over the signature* $\Sigma_{\mathtt{euf}}$*.* $\mathfrak{C}_{\mathtt{euf}}[\![\cdot]\!]$ *is a care function for* $T_{\mathtt{euf}}$ *with respect to the equality propagator* $\mathfrak{P}^{=}_{\mathtt{euf}}[\![\cdot]\!]$.

5 Theory of Arrays

The extensional theory of arrays $T_{\mathtt{arr}}$ operates over the signature $\Sigma_{\mathtt{arr}}$ that contains the sorts $\{\mathsf{array}, \mathsf{index}, \mathsf{elem}\}$ and function symbols $\mathsf{read} : \mathsf{array} \times \mathsf{index} \mapsto \mathsf{elem}$ and $\mathsf{write} : \mathsf{array} \times \mathsf{index} \times \mathsf{elem} \mapsto \mathsf{array}$, where read represents reading from an array at a given index, and write represents writing a given value to an array at an index. Semantics of the theory are well known, and we refer the reader to [17] for details. The flat literals of the theory are of the form $x = \mathsf{read}(a,i)$, $a = \mathsf{write}(b,i,x)$, $x = y$, $x \neq y$, $a = b$, where x, y

are variables of sort elem, i is an index variable of sort index, and a and b are variables of sort array. Note that we do not allow dis-equalities $a \neq b$ between array variables. This is not an issue as these can always be rewritten as $\mathsf{read}(a,k) \neq \mathsf{read}(b,k)$ for a fresh index variable k.

Equality Propagator. Let ϕ be a set of flat literals and V a set of variables. We define $\approx_a$ (and corresponding dis-equality closure $\neq_a$) to be the smallest equivalence relation over the terms in ϕ containing $\{\ (x,t)\ \mid\ x = t \in \phi\ \}$, that additionally satisfies the following closure rules.

1. Congruence closure rules:

$$i \approx_a j \wedge x \approx_a \mathsf{read}(a,i) \implies x \approx_a \mathsf{read}(a,j)\ , \tag{CCR1}$$
$$a \approx_a b \wedge x \approx_a \mathsf{read}(a,i) \implies x \approx_a \mathsf{read}(b,i)\ , \tag{CCR2}$$
$$i \approx_a j \wedge a \approx_a \mathsf{write}(b,i,x) \implies a \approx_a \mathsf{write}(b,j,x)\ , \tag{CCW1}$$
$$x \approx_a y \wedge a \approx_a \mathsf{write}(b,i,x) \implies a \approx_a \mathsf{write}(b,i,y)\ , \tag{CCW2}$$
$$b \approx_a c \wedge a \approx_a \mathsf{write}(b,i,x) \implies a \approx_a \mathsf{write}(c,i,x)\ . \tag{CCW3}$$

2. Array closure rules:

$$a \approx_a \mathsf{write}(b,j,y) \implies y \approx_a \mathsf{read}(a,j)\ , \tag{WR}$$
$$x \approx_a \mathsf{read}(a,i) \wedge a \approx_a \mathsf{write}(b,j,y) \wedge i \neq_a j \implies x \approx_a \mathsf{read}(b,i)\ , \tag{RW1}$$
$$x \approx_a \mathsf{read}(b,i) \wedge a \approx_a \mathsf{write}(b,j,y) \wedge i \neq_a j \implies x \approx_a \mathsf{read}(a,i)\ . \tag{RW2}$$

3. Dis-equality closure rule:

$$x \approx_a x' \wedge y \approx_a y' \wedge x' \neq y' \in \phi \implies x' \neq_a y'\ .$$

In all the above rules, we require that if the hypotheses are satisfied, then the conclusions must also hold in the equivalence relation, even if this means adding new terms to the domain of the relation. Since all the terms are flat, and we have a finite number of variables, the resulting relation will be finite. We now define the equality propagator as

$$\mathfrak{P}^{=}_{\mathtt{arr}}[\![V]\!](\phi) = \ \{\ x = y\ \mid\ x,y \in V \wedge x \approx_a y\ \} \cup \{\ x \neq y\ \mid\ x,y \in V \wedge x \neq_a y\ \}\ .$$

It is not hard to see that each of the closure rules is sound with respect to $T_{\mathtt{arr}}$ and that $\mathfrak{P}^{=}_{\mathtt{arr}}[\![\cdot]\!]$ is monotone, and thus fulfills the requirements for a propagator.

Care Function. Given a set ϕ of flat literals, we define a parameterized family of equivalence relations $\sim_I$ over the variables of sort array in ϕ, where the parameter I can be any set of variables of sort index from ϕ. For each I,J, let $\sim_I$ be the smallest equivalence relation that satisfies

$$a \approx_a b \implies a \sim_{\emptyset} b\ ,$$
$$a \approx_a \mathsf{write}(b,i,x) \implies a \sim_{\{\,i\,\}} b\ ,$$
$$a \sim_I b \wedge b \sim_J c \implies a \sim_{I \cup J} c\ .$$

Relation $\sim_I$ collects all the array variables that differ at not more than finitely many indices.

The care function depends on whether the set V contains any variables of the array sort or not, and whether all the index variables from ϕ are in V.

First we consider the case when V does *not* contain any array variables, but *contains all* variables of sort index from ϕ. Let E be the set of all pairs of variables from V that occur as the index argument of a read and might (if set equal) cause two terms to become equal, i.e.:

$$\begin{aligned} E \;=\; & \{\, (i,j) \in V \times V \;\mid\; a \sim_I b \wedge \mathsf{read}(a,i) \not\approx_a \mathsf{read}(b,j) \,\} \;\cup \\ & \{\, (i,j) \in V \times V \;\mid\; a \sim_I b \wedge i \in I \wedge x \approx_a \mathsf{read}(a,j) \,\} \;. \end{aligned}$$

Now, we define the care function as $\mathfrak{C}_{\mathtt{arr}}[\![V]\!](\phi) = \mathbf{G} = \langle V, E \rangle$.

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

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

Considering that all the read applications are over arrays that are not related by $\sim_I$, our care graph will in fact be empty and we do not need to guess an arrangement.

In the case when V contains array variables, or V does not contain all of the variables of sort index, there is no clear technique for reducing the care graph. Therefore we settle for using the trivial care function in this case, i.e. $\mathfrak{C}_{\mathtt{arr}}[\![V]\!](\phi) = \langle V, V \times V \rangle$. Fortunately, this appears to match well with how the theory of arrays is used in practice: index and element variables are typically shared, and only rarely are array variables shared.

Theorem 3. *Let $T_{\mathtt{arr}}$ be the theory of arrays. $\mathfrak{C}_{\mathtt{arr}}[\![\cdot]\!]$ is a care function for $T_{\mathtt{arr}}$ with respect to the equality propagator $\mathfrak{P}^{=}_{\mathtt{arr}}[\![\cdot]\!]$.*

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 [9], Cvc3, and MathSAT [5], the top solvers in the QF_AUFBV category from the 2009 SMT-COMP competition (in order).[5] Additionaly, 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.

[5] http://www.smtcomp.org/2009/

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

Benchmarks. 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.[6] 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 of Burstall, e.g. [14]). All the benchmarks and the Cvc3 binaries used in the experiments are available from the authors' website.[7]

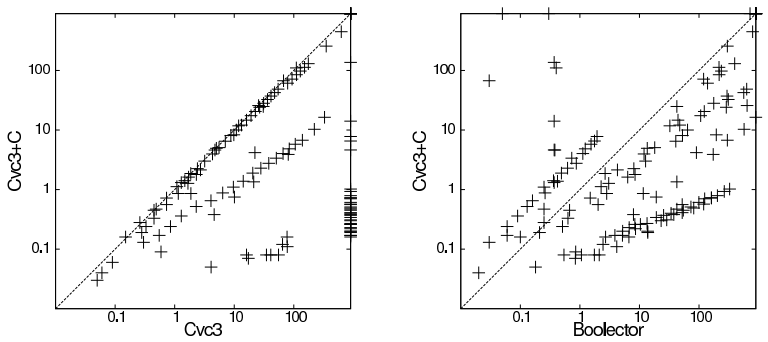
Results. The combined results of our experiments are presented in Table 1, with columns reporting the total time (in seconds) that a solver used on the problem instances it solved (not including time spent on problem instances it was unable to solve), and the number of solved instances. Compared to Cvc3, 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. Z3 appears to be the most robust of the other solvers on the benchmarks where we saw a dramatic improvement by Cvc3. We assume this is due to their model-based combination mechanism which has some similar advantages.

[6] http://combination.cs.uiowa.edu/smtlib/

[7] http://cs.nyu.edu/~dejan/smt2010/

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.



(a) (b)

Figure 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.

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 that are important for a specific combination problem. We proved the method correct, and presented care functions for the theories of uninterpreted functions and arrays.

The new method is asymmetric as only one of the theories takes charge of creating the arrangement graph over the interface variables. Since many theories we combine in practice are parametrized by other theories, the non-symmetric approach has an intuitive appeal. We draw intuition for constructing the care functions and the proofs of correctness 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 can be seen as orthogonal to the 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] Clark Barrett, Robert Nieuwenhuis, Albert Oliveras, and Cesare Tinelli. Splitting on demand in SAT Modulo Theories. In *Logic for Programming, Artificial Intelligence, and Reasoning: Proceedings of the 13th International Conference, LPAR 2006, Phnom Penh, Cambodia, November 13-17, 2006*, volume 4246 of *Lecture Notes in Computer Science*, pages 512–526. Springer, 2006.

[2] Clark Barrett and Cesare Tinelli. CVC3. In *Computer Aided Verification: Proceedings of the 19th International Conference, CAV 2007, Berlin, Germany, July 3-7, 2007*, volume 4590 of *Lecture Notes in Computer Science*, pages 298–302. Springer-Verlag, 2007.

[3] Marco Bozzano, Roberto Bruttomesso, Alessandro Cimatti, Tommi Junttila, Silvio Ranise, Peter van Rossumd, and Roberto Sebastiani. Efficient theory combination via Boolean search. *Information and Computation*, 204(10):1493–1525, 2006.

[4] Robert Brummayer and Armin Biere. Boolector: An efficient SMT solver for bit-vectors and arrays. In *Tools and Algorithms for the Construction and Analysis of Systems: Proceedings of the 15th International Conference, TACAS 2009, York, UK, March 22-29, 2009*, volume 5505 of *Lecture Notes in Computer Science*, pages 174–177. Springer, 2009.

[5] Roberto Bruttomesso, Alessandro Cimatti, Anders Franzén, Alberto Griggio, and Roberto Sebastiani. The MathSAT 4 SMT solver. In *Computer Aided Verification: Proceedings of the 20th International Conference, CAV 2008 Princeton, NJ, USA, July 7-14, 2008*, volume 5123 of *Lecture Notes in Computer Science*, pages 299–303. Springer, 2008.

[6] Roberto Bruttomesso, Alessandro Cimatti, Anders Franzen, Alberto Griggio, and Roberto 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] Leonardo de Moura and Nikolaj Bjørner. Model-based Theory Combination. In Sava Krstić and Albert Oliveras, editors, *Proceedings of the 5th International Workshop on Satisfiability Modulo Theories (SMT 2007), Berlin, Germany, July 2007*, volume 198 of *Electronic Notes in Theoretical Computer Science*, pages 37–49. Elsevier, 2008.

[8] Leonardo de Moura and Nikolaj Bjørner. Z3: An Efficient SMT Solver. In *Tools and Algorithms for the Construction and Analysis of Systems: Proceedings of the 14th International Conference, TACAS 2008, Budapest, Hungary, March 29-April 6, 2008*, volume 4963 of *Lecture Notes in Computer Science*, page 337. Springer, 2008.

[9] Bruno Dutertre and Leonardo de Moura. The YICES SMT Solver. *Tool paper at* `http://yices.csl.sri.com/tool-paper.pdf`, 2006.

[10] Herbert B. Enderton. *A mathematical introduction to logic*. Academic press New York, 1972.

[11] Dejan Jovanović and Clark Barrett. Polite theories revisited. Technical Report TR2010-922, Department of Computer Science, New York University, January 2010.

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

[13] Derek C. Oppen. Complexity, convexity and combinations of theories. *Theoretical Computer Science*, 12(3):291–302, 1980.

[14] Zvonimir Rakamarić and Alan J. Hu. A Scalable Memory Model for Low-Level Code. In *Verification, Model Checking, and Abstract Interpretation: Proceedings of the 10th International Conference, VMCAI 2009, Savannah, GA, USA, January 18-20, 2009*, Lecture Notes in Computer Science, page 304. Springer-Verlag, 2008.

[15] Silvio Ranise, Christophe Ringeissen, and Calogero G. Zarba. Combining Data Structures with Nonstably Infinite Theories Using Many-Sorted Logic. In Bernhard Gramlich, editor, *Frontiers of Combining Systems, 5th International Workshop, FroCoS 2005, Vienna, Austria, September 19-21, 2005, Proceedings*, volume 3717 of *Lecture Notes in Computer Science*, pages 48–64. Springer, 2005.

[16] Robert E. Shostak. An algorithm for reasoning about equality. In *Proceedings of the 5th international joint conference on Artificial intelligence-Volume 1*, pages 526–527. Morgan Kaufmann Publishers Inc., 1977.

[17] Aaron Stump, David L. Dill, Clark W. Barrett, and Jeremy Levitt. A decision procedure for an extensional theory of arrays. In *Proceedings of the* 16^{th} *IEEE Symposium on Logic in Computer Science (LICS '01)*, pages 29–37. IEEE Computer Society, June 2001. Boston, Massachusetts.

[18] Cesare Tinelli and Mehdi T. Harandi. A new correctness proof of the Nelson–Oppen combination procedure. In Franz Baader and Klaus Ulrich Schulz, editors, *Frontiers of Combining Systems: Proceedings of the 1st International Workshop (Munich, Germany)*, Applied Logic, pages 103–120. Kluwer Academic Publishers, March 1996.

[19] Cesare Tinelli and Calogero Zarba. Combining decision procedures for sorted theories. In José Alferes and João Leite, editors, *Proceedings of the 9th European Conference on Logic in Artificial Intelligence (JELIA'04), Lisbon, Portugal*, volume 3229 of *Lecture Notes in Artificial Intelligence*, pages 641–653. Springer, 2004.

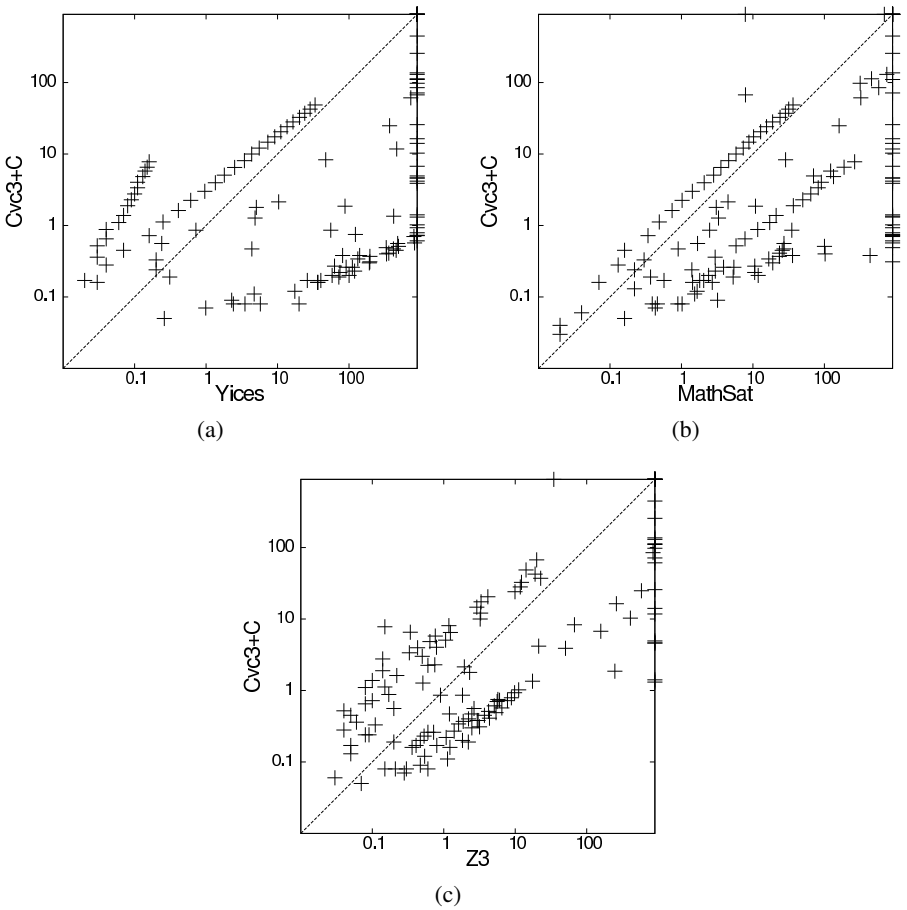


Figure 2: Additional scatter-plot comparisons. Both axes are in logarithmic scale and each point represents the times needed to solve an individual problem.

A Proofs

A.1 Proof of Correctness for the Combined Care Function

Lemma 1. *The combined care function* $\mathfrak{C}_T[\![\cdot]\!]$ *is indeed a care function for theory T with respect to the combined propagator* $\mathfrak{P}^{=}_T[\![\cdot]\!]$.

Proof. Assume we are given a set V of variables and a set ϕ of flat Σ-literals. Note that it is always possible to partition such a ϕ into a set ϕ_1 of Σ_1-literals and a set ϕ_2 of Σ_2-literals. Let $\mathbf{G}_1 = \mathfrak{C}_{T_1}[\![V]\!](\phi_1)$, $\mathbf{G}_2 = \mathfrak{C}_{T_2}[\![V]\!](\phi_2)$, and $\mathbf{G} = \mathfrak{C}_T[\![V]\!](\phi) = \mathbf{G}_1 \cup \mathbf{G}_2$. Then, let $\delta_{\mathbf{G}}$ be an arrangement such that $\phi \cup \delta_{\mathbf{G}}$ is satisfiable in a T-interpretation $\mathcal{A}$ (wlog assume that $\mathcal{A}$ interprets all variables in V). Let $\delta_{\mathbf{G}_1}$ and $\delta_{\mathbf{G}_2}$ be the restrictions of the arrangement $\delta_{\mathbf{G}}$ to $\mathbf{G}_1$ and $\mathbf{G}_2$, respectively (i.e. the largest arrangements over $\mathbf{G}_1$ and $\mathbf{G}_2$ respectively that are subsets of $\delta_{\mathbf{G}}$). Now, for $i = 1,2$, since $\phi_i \cup \delta_{\mathbf{G}_i} \subseteq \phi \cup \delta_{\mathbf{G}}$, and since $\mathcal{A} \models_T \phi \cup \delta_{\mathbf{G}}$, we know that $\phi_i \cup \delta_{\mathbf{G}_i}$ is T_i-satisfiable.

Let δ_V be an arrangement such that $\delta_V \supseteq \mathfrak{P}^{=}_{T}[\![V]\!](\phi \cup \delta_{\mathbf{G}})$. Then by, property 3 of equality propagators, we can conclude that $\delta_V \supseteq \mathfrak{P}^{=}_{T_1}[\![V]\!](\phi_1 \cup \delta_{\mathbf{G}_1})$ and $\delta_V \supseteq \mathfrak{P}^{=}_{T_2}[\![V]\!](\phi_2 \cup \delta_{\mathbf{G}_2})$. Finally, using the fact that $\mathfrak{C}_{T_1}[\![\cdot]\!]$ and $\mathfrak{C}_{T_2}[\![\cdot]\!]$ are both care functions, we have that $\phi_1 \cup \delta_V$ and $\phi_2 \cup \delta_V$ are T_1- and T_2-satisfiable, respectively. It follows (by the correctness of the standard Nelson-Oppen procedure) that $\phi \cup \delta_V$ is T-satisfiable. □

A.2 Proof of Correctness for $\mathfrak{C}_{\mathtt{euf}}[\![\cdot]\!]$

We next show that the above function does indeed define a care function in terms of Definition 6. First, we introduce some notation and a well-known proposition that will be helpful in the proof. For a set of formulas ϕ, let $\mathcal{E}(\phi)$ denote the smallest equivalence relation over the terms occurring in ϕ containing $\{ (x,y) \mid x = y \in \phi \}$. For an equivalence relation E, let E^* denote the congruence closure of E.

Proposition 1. *A set ϕ of flat literals is tEUF-satisfiable iff for each dis-equality $x \neq y \in \phi$, $(x,y) \notin \mathcal{E}(\phi)^*$.*

Theorem 2. *Let $T_{\mathtt{euf}}$ be the theory of uninterpreted functions with equality over the signature $\Sigma_{\mathtt{euf}}$. $\mathfrak{C}_{\mathtt{euf}}[\![\cdot]\!]$ is a care function for $T_{\mathtt{euf}}$ with respect to the equality propagator $\mathfrak{P}^{=}_{\mathtt{euf}}[\![\cdot]\!]$.*

Proof. Let ϕ be a set of flat literals, V a set of variables, let $\mathbf{G} = \mathfrak{C}_{\mathtt{euf}}[\![V]\!](\phi)$, and assume that for some arrangement $\delta_{\mathbf{G}}$ over $\mathbf{G}$, $\phi \cup \delta_{\mathbf{G}}$ is satisfiable. Then, suppose we are given an arrangement δ_V (induced by equivalence relation E_V) with $\delta_V \supseteq \mathfrak{P}^{=}_{\mathtt{euf}}[\![V]\!](\phi \wedge \delta_{\mathbf{G}}) \supseteq \delta_{\mathbf{G}}$. We must show that $\phi \wedge \delta_V$ is satisfiable.

Let $E_1 = \mathcal{E}(\phi \cup \delta_{\mathbf{G}})^*$. We know by proposition 1 above, that for each $x \neq y \in \phi \cup \delta_{\mathbf{G}}$, $(x,y) \notin E_1$. Now, let $E_2 = \mathcal{E}(\phi \cup \delta_V)^*$. It suffices to show that if $x \neq y \in \phi \cup \delta_V$, then $(x,y) \notin E_2$.

We start by showing that $E_2 = E_1 \cup E_V$. To see this, note first that because $\delta_V \supseteq \delta_{\mathbf{G}}$, we can write $E_2 = \mathcal{E}(\phi \cup \delta_V)^* = \mathcal{E}(\phi \cup \delta_G \cup \delta_V)^*$. Now, by basic properties of equivalence and congruence closures, this is equivalent to $(\mathcal{E}(\phi \cup \delta_G)^* \cup \mathcal{E}(\delta_V))^* = (E_1 \cup E_V)^*$. To see that $(E_1 \cup E_V)^* = E_1 \cup E_V$, suppose that it is not. Then there must be some $f(x_1, x_2, \ldots x_n)$ and $f(y_1, y_2, \ldots y_n)$ appearing in ϕ such that $(f(x_1, x_2, \ldots x_n), f(y_1, y_2, \ldots y_n)) \notin E_1$, but for each $1 \leq i \leq n$, $(x_i, y_i) \in E_1 \cup E_V$. Since E_1 is congruence closed, there must be some i such that $(x_i, y_i) \notin E_1$ and $(x_i, y_i) \in E_V$. But because x_i and y_i appear as arguments of the same function in the same position, and both are in V, we know that the edge (x_i, y_i) must be in the graph $\mathbf{G}$. This means that $\delta_{\mathbf{G}}$ must specify a relationship for x_i and y_i and since $\delta_V \supseteq \delta_{\mathbf{G}}$, the same relationship appears in δ_V. It follows that $(x_i, y_i) \in E_1$, contradicting our earlier assertion. Thus, $E_2 = E_1 \cup E_V$.

We can now show that if $x \neq y \in \phi \cup \delta_V$, then $(x,y) \notin E_2$. We consider two cases.

- First, suppose $x \neq y \in \phi \cup \delta_{\mathbf{G}}$. As mentioned above, we know that $(x,y) \notin E_1$ because $\phi \cup \delta_{\mathbf{G}}$ is satisfiable. We also know that $\delta_V \supseteq \mathfrak{P}^{=}_{\mathtt{euf}}[\![V]\!](\phi \wedge \delta_{\mathbf{G}})$ which means that δ_V must contain $x \neq y$ as well, meaning it cannot contain $x = y$, so $(x,y) \notin E_V$.

- In the other case, we must have $x \neq y \in \delta_V$. Clearly, we cannot also have $x = y \in \delta_V$, so $(x,y) \notin E_V$. The only way that (x,y) could be in E_2 then is if $(x,y) \in$

E_1. Note that by definition, $\mathcal{E}(\mathfrak{P}^{=}_{\mathtt{euf}}[\![V]\!](\phi \wedge \delta_{\mathbf{G}}))$ is exactly equal to $E_1 \cap (V \times V)$. So, if $(x,y) \in E_1$, then we must have $x = y \in \mathfrak{P}^{=}_{\mathtt{euf}}[\![V]\!](\phi \wedge \delta_{\mathbf{G}})$. But $\delta_V \supseteq \mathfrak{P}^{=}_{\mathtt{euf}}[\![V]\!](\phi \wedge \delta_{\mathbf{G}})$, so then we would have $x = y \in \delta_V$ which is not possible.

□

A.3 Proof of Correctness for $\mathfrak{C}_{\mathtt{arr}}[\![\cdot]\!]$

We will prove the correctness of the care function $\mathfrak{C}_{\mathtt{arr}}[\![\cdot]\!]$ by showing that the care-graph spans enough variables in order to completely reduce the reasoning about $T_{\mathtt{arr}}$-satisfiability to the theory of uninterpreted functions $T_{\mathtt{euf}}$, where the signature $\Sigma_{\mathtt{euf}}$ will be crafted for each $\Sigma_{\mathtt{arr}}$-formula.

Given a conjunction of flat $\Sigma_{\mathtt{arr}}$-literals ϕ we will map the signature $\Sigma_{\mathtt{arr}}$ into a signature $\Sigma_{\mathtt{euf}}$ as follows. We will denote as $\approx_a$ the equivalence relation defined, with respect to ϕ, in Section 5. For every array variable $a \in \mathit{vars}_{\mathsf{array}}(\phi)$, we denote with $[a]$ the equivalence representative of a in $\approx_a$. We define the signature $\Sigma_{\mathtt{euf}}$ to operate over the the function symbols

$$\Sigma^{\mathbb{F}}_{\mathtt{euf}} = \{\, f_{[a]} : \mathsf{index} \mapsto \mathsf{elem} \mid a \in \mathit{vars}_{\mathsf{array}}(\phi) \,\} \ .$$

Now, we define a mapping α from conjunctions of $\Sigma_{\mathtt{arr}}$-literals to conjunctions of $\Sigma_{\mathtt{euf}}$-literals as follows

$$\alpha(\phi) = \{\, x = f_{[a]}(i) \mid x \approx_a \mathsf{read}(a,i) \,\} \cup \mathfrak{P}^{=}_{\mathtt{arr}}[\![V]\!](\phi) \ .$$

Note that since V does not contain any array variables, $\alpha(\phi)$ will not contain any array equalities in, only function applications and equalities and dis-equalities over elem and index variables.

Lemma 2. *Let ϕ be a conjunction of flat $\Sigma_{\mathtt{arr}}$ literals, V a set of literals such that $\mathit{vars}_{\mathsf{index}}(\phi) \subseteq V$ and $\mathit{vars}_{\mathsf{array}}(V) = \emptyset$, and let $\mathbf{G} = \mathfrak{C}_{\mathtt{arr}}[\![V]\!](\phi)$. Let $\approx_a$ and $\sim_I$ be the array relations defined in Section 5, but with respect to $\phi \wedge \delta_{\mathbf{G}}$, then*

$$a_1 \sim_I a_2 \wedge x \approx_a \mathsf{read}(a_2,k) \implies \forall i \in I : i \approx_a k \vee i \neq_a k \ .$$

Proof. Consider the (smaller) relations $\approx^{\phi}_a$ and $\sim^{\phi}_I$, as defined in Section 5, but with respect to ϕ. For the purpose of this statement, we notice that the differences in relations $\sim^{\phi}_I$ and $\sim_I$ is not significant. We claim that for every non-empty set of index variables I such that $a \sim_I b$, there is a set of index variables $J \subseteq I$, such that $a \sim^{\phi}_J b$ and for each $i \in I$ there is a $j \in J$ such that $j \approx_a i$.

This holds for the following reasons. First, the new information in $\delta_{\mathbf{G}}$ does not contain any array equalities, and therefore the only way to can extend $\approx_a$, over array terms, is by means of congruence closure, i.e. by rules (CCW1) and (CCW2). Only the first case we would extend $\sim_I$, i.e. with $i \approx_a j$ we would get $a \sim_{\{j\}} b$ where we had $a \sim_{\{i\}} b$. But, this is fine as $i \approx_a j$. It's easy to see that closing $\sim_I$ transitively under unions we still keep the required property.

Now we proceed to prove the main statement. Since we can build $\approx_a$ and $\sim_I$ from $\approx^{\phi}_a$ and $\sim^{\phi}_I$, by adding $\delta_{\mathbf{G}}$ and then closing the relations under the completion rules, we will prove the complete statement by induction on the sequence of completion rule

applications. In generating $\approx_a$ we will always first apply the congruence closure rules, and then all others.

For the base case, notice that by definition of $\mathbf{G} = \mathfrak{C}_{\mathtt{arr}}[\![V]\!](\phi)$ we have

$$a_1 \sim_I^{\phi} a_2 \wedge x \approx_a^{\phi} \mathsf{read}(a_2, k) \implies \forall i \in I : (k,i) \in \mathbf{G} \ .$$

Since $\delta_{\mathbf{G}}$ determines values of equalities or dis-equalities over the pairs of indices in $\mathbf{G}$, by discussion above, we have shown the base case.

Now, assume that $a_1 \sim_I a_2$ and consider the completion rules that can produce a new read term.

$$x \approx_a \mathsf{read}(a_2, j) \wedge j \approx_a k \implies x \approx_a \mathsf{read}(a_2, k) \tag{CCR1}$$

Since, by inductive hypothesis, for all $i \in I$ we have that $i \approx_a j$ or $i \not\approx_a j$, and $j \approx_a k$, we also get that for all $i \in I$ we have that $i \approx_a k$ or $i \not\approx_a k$.

$$a \approx_a \mathsf{write}(a_2, j, y) \implies y \approx_a \mathsf{read}(a_2, j) \tag{WR}$$

Since $\delta_{\mathbf{G}}$ does not contain any write expressions, the only way for this rule to get triggered is if we obtained the left side by congruence closure. But in this case the right hand side is already covered, also by congruence closure.

$$x \approx_a \mathsf{read}(a,k) \wedge a \approx_a \mathsf{write}(a_2, j, y) \wedge k \not\approx_a j \implies x \approx_a \mathsf{read}(a_2, k) \tag{RW1}$$

In this case we have $a_2 \sim_{\{j\}} a$ which, with $a_1 \sim_I a_2$, gives $a_1 \sim_{I \cup \{j\}} a$. Then, by inductive hypothesis we have that for all $i \in I \cup \{j\}$ either $i \approx_a k$ or $i \not\approx_a k$.

$$x \approx_a \mathsf{read}(b,k) \wedge a_2 \approx_a \mathsf{write}(b, j, y) \wedge k \not\approx_a j \implies x \approx_a \mathsf{read}(a_2, k) \tag{RW2}$$

This case is similar to the one just above, and is the last case. □

Lemma 3. *Let ϕ be a conjunction of flat $\Sigma_{\mathtt{arr}}$ literals, V a set of literals such that $vars_{\mathsf{index}}(\phi) \subseteq V$ and $vars_{\mathsf{array}}(V) = \emptyset$. Then*

$$\mathfrak{P}^{=}_{\mathtt{euf}}[\![V]\!](\alpha(\phi)) = \mathfrak{P}^{=}_{\mathtt{arr}}[\![V]\!](\phi) \ .$$

Proof. First, it's easy to see that from definition of α, we have

$$\mathfrak{P}^{=}_{\mathtt{euf}}[\![V]\!](\alpha(\phi)) \supseteq \mathfrak{P}^{=}_{\mathtt{arr}}[\![V]\!](\phi) \ .$$

For the other direction, notice that the function symbols in $\alpha(\phi)$ correspond to arrays that are already equal under $\approx_a$. Since and $\mathfrak{P}^{=}_{\mathtt{arr}}[\![V]\!]$ is also closed under congruence, the other direction is also true, i.e.

$$\mathfrak{P}^{=}_{\mathtt{euf}}[\![V]\!](\alpha(\phi)) \subseteq \mathfrak{P}^{=}_{\mathtt{arr}}[\![V]\!](\phi) \ ,$$

which proves the case. □

Lemma 4. *Let ϕ be a conjunction of flat $\Sigma_{\mathtt{arr}}$ literals, V a set of literals such that $vars_{\mathsf{index}}(\phi) \subseteq V$ and $vars_{\mathsf{array}}(V) = \emptyset$. Let $\mathfrak{C}_{\mathtt{arr}}[\![V]\!](\phi) = \mathbf{G}$, then*

$$\mathfrak{C}_{\mathtt{euf}}[\![V]\!](\alpha(\phi \wedge \delta_{\mathbf{G}})) \subseteq \mathbf{G} \ .$$

Proof. Let $\approx_a^{\phi}$ and $\sim_I^{\phi}$, be the equivalence relations from Section 5 defined with respect to ϕ. Let $\sim_c$ and $\approx_a$ be the relations from Section 4 and Section 5, defined with respect to $\alpha(\phi \wedge \delta_{\mathbf{G}})$ and $\phi \wedge \delta_{\mathbf{G}}$ respectively. We can now see that

$$\begin{aligned}
\mathfrak{C}_{\mathtt{euf}}[\![V]\!](\alpha(\phi \wedge \delta_{\mathbf{G}})) &= \left\{(i,j) \in V \times V \mid f_{[a]}(i) \nsim_c f_{[a]}(j)\right\} \\
&= \{(i,j) \in V \times V \mid b \approx_a c \wedge \mathsf{read}(b,i) \not\approx_a \mathsf{read}(c,j)\} \\
&\subseteq \left\{(i,j) \in V \times V \mid b \sim_I^{\phi} c \wedge \mathsf{read}(b,i) \not\approx_a^{\phi} \mathsf{read}(c,j)\right\} \\
&\subseteq \mathfrak{C}_{\mathtt{arr}}[\![V]\!](\phi) = \mathbf{G} \ .
\end{aligned}$$

This proves the case. □

Lemma 5. *Let ϕ be a conjunction of flat $\Sigma_{\mathtt{arr}}$ literals, V a set of literals such that $vars_{\mathsf{index}}(\phi) \subseteq V$ and $vars_{\mathsf{array}}(V) = \emptyset$, and let $\mathbf{G} = \mathfrak{C}_{\mathtt{arr}}[\![V]\!](\phi)$. Then, for arbitrary arrangements $\delta_{\mathbf{G}}$ and δ_V such that $\mathfrak{P}^{=}_{\mathtt{arr}}[\![V]\!](\phi \wedge \delta_{\mathbf{G}}) \subseteq \delta_V$, it holds that*

- *if $\alpha(\phi \wedge \delta_{\mathbf{G}}) \wedge \delta_V$ is $T_{\mathtt{euf}}$-satisfiable,*
- *then $\phi \wedge \delta_V$ is $T_{\mathtt{arr}}$-satisfiable.*

Proof. Assume that there is a $T_{\mathtt{euf}}$-interpretation $\mathcal{A}$ such that

$$\mathcal{A} \models \alpha(\phi \wedge \delta_{\mathbf{G}}) \wedge \delta_V \ .$$

We will construct a $T_{\mathtt{arr}}$ interpretation $\mathcal{B}$ that satisfies $\phi \wedge \delta_V$. First, denote with $\approx_a$ the equivalence relation (from Section 5) on $\phi \wedge \delta_{\mathbf{G}}$ and $\sim_c$ the equivalence relation (from Section 4)) on $\alpha(\phi \wedge \delta_{\mathbf{G}}) \wedge \delta_V$. We define the domains of $\mathcal{B}$ as

$$\begin{aligned}
B_{\mathsf{index}} &= [vars_{\mathsf{index}}(\phi \wedge \delta_V)]^{\mathcal{A}} \ , \\
B_{\mathsf{elem}} &= [vars_{\mathsf{elem}}(\phi \wedge \delta_V)]^{\mathcal{A}} \ , \\
B_{\mathsf{array}} &= \{ \ f \mid f : B_{\mathsf{index}} \mapsto B_{\mathsf{elem}} \ \} \ .
\end{aligned}$$

As for the interpretations of the index and elem variables for from $\phi \wedge \delta_V$, we let $i^{\mathcal{B}} = i^{\mathcal{A}}$ and $x^{\mathcal{B}} = x^{\mathcal{A}}$. We interpret each array $a \in vars_{\mathsf{array}}(\phi)$ as the corresponding function from $\mathcal{A}$ in a restricted manner. Assuming a distinct element $e_0 \in B_{\mathsf{elem}}$ we interpret each each a so that for $c \in B_{\mathsf{index}}$

$$a^{\mathcal{B}}(c) = \begin{cases} x^{\mathcal{A}} & \text{if } x \sim_c f_{[a]}(i) \text{ and } i^{\mathcal{A}} = c \\ e_0 & \text{otherwise.} \end{cases}$$

The interpretation is well-defined and we claim that it satisfies $\phi \wedge \delta_V$. First, all the equalities and dis-equalities from $\phi \wedge \delta_V$ among the index and the elem variables will clearly be satisfied by construction. All the read applications from ϕ will be satisfied as for each $x = \mathsf{read}(a,i) \in \phi$ we have $x \sim f_{[a]}(i)$ and

$$[x = \mathsf{read}(a,i)]^{\mathcal{B}} = (x^{\mathcal{B}} = a^{\mathcal{B}}(i^{\mathcal{B}})) = (x^{\mathcal{A}} = f^{\mathcal{A}}_{[a]}(i^{\mathcal{A}})) = \mathsf{true} \ .$$

Now we have to prove that the equalities among the array terms are satisfied (we have no dis-equalities). First, for each equality $a = b \in \phi$, we will have that they are related by $\approx_a$, and hence interpreted as the same function in $\mathcal{A}$.

Next, assume $a = \mathsf{write}(b,k,x) \in \phi$ and lets show that the array update at position k is properly reflected, i.e. that

$$i \in \mathit{vars}_{\mathsf{index}}(\phi \wedge \delta_V) \wedge i^{\mathcal{B}} = k^{\mathcal{B}} \implies a^{\mathcal{B}}(i^{\mathcal{B}}) = x^{\mathcal{B}} \ ,$$
$$i \in \mathit{vars}_{\mathsf{index}}(\phi \wedge \delta_V) \wedge i^{\mathcal{B}} \neq k^{\mathcal{B}} \implies a^{\mathcal{B}}(i^{\mathcal{B}}) = b^{\mathcal{B}}(i^{\mathcal{B}}) \ .$$

Since the we know that $a \approx_a \mathsf{write}(b,k,x)$, using the closure rule (WR) we know that $x \approx_a \mathsf{read}(a,k)$, and hence $x \sim_c f_{[a]}(k)$. Therefore the first property holds. For the second property, consider the case when

$$a^{\mathcal{B}}(i^{\mathcal{B}}) = x_1^{\mathcal{A}}, \ \ x_1 \sim_c f_{[a]}(i) \ ,$$
$$b^{\mathcal{B}}(i^{\mathcal{B}}) = x_2^{\mathcal{A}}, \ \ x_2 \sim_c f_{[b]}(i) \ .$$

Since $\sim_c$ is only congruence applications on top of $\approx_a$, we know there are $y_1 \sim_c x_1$, $y_2 \sim_c x_2$, $i_1 \sim_c i$, and $i_2 \sim_c i$ such that

$$y_1 \approx_a f_{[a]}(i_1), \ \ y_1 \approx_a \mathsf{read}(a,i_1) \ ,$$
$$y_2 \approx_a f_{[b]}(i_2), \ \ y_2 \approx_a \mathsf{read}(b,i_2) \ .$$

Now, from Lemma 2 we can conclude that $i_1 \neq_a k$ and $i_2 \neq_a k$. From here, using the rules (RW1) and (RW2) we get that $y_1 \approx_a y_2$ which implies $x_1^{\mathcal{B}} = x_2^{\mathcal{B}}$ and $a^{\mathcal{B}}(i^{\mathcal{B}}) = b^{\mathcal{B}}(i^{\mathcal{B}})$.

In the case when both functions map i to e_0, the equality follows vacuously. The last case, when only one of the functions maps to e_0 is not possible as, for example $y_1 \approx_a \mathsf{read}(a,i_1)$ would imply $i_1 \neq_a k$ which, in turn, would via the read-over-write (RW1) rule imply that $y_1 \approx_a \mathsf{read}(b,i_1)$. This concludes the proof. □

Theorem 3. *Let $T_{\mathtt{arr}}$ be the theory of arrays. $\mathfrak{C}_{\mathtt{arr}}[\![\cdot]\!]$ is a care function for $T_{\mathtt{arr}}$ with respect to the equality propagator $\mathfrak{P}^{=}_{\mathtt{arr}}[\![\cdot]\!]$.*

Proof. Recall that $\mathfrak{C}_{\mathtt{arr}}[\![V]\!]$ is defined to be the trivial care function if V contains any array variables or $\mathit{vars}_{\mathsf{index}}(\phi) \not\subseteq V$. Thus, assume that we are given a conjunction of flat $\Sigma_{\mathtt{arr}}$-literals ϕ and a set of variables V such that V does not contain any array variables and $\mathit{vars}_{\mathsf{index}}(\phi) \subseteq V$. Let $\mathbf{G} = \mathfrak{C}_{\mathtt{arr}}[\![V]\!](\phi)$ and assume that $\phi \wedge \delta_{\mathbf{G}}$ is satisfiable and δ_V is a variable arrangement such that

$$\mathfrak{P}^{=}_{\mathtt{arr}}[\![V]\!](\phi \wedge \delta_{\mathbf{G}}) \subseteq \delta_V \ .$$

Using Lemma 3 we know that

$$\mathfrak{P}^{=}_{\mathtt{euf}}[\![V]\!](\alpha(\phi \wedge \delta_{\mathbf{G}})) = \mathfrak{P}^{=}_{\mathtt{arr}}[\![V]\!](\phi \wedge \delta_{\mathbf{G}}) \subseteq \delta_V \ .$$

From this, we can conclude that $\alpha(\phi \wedge \delta_{\mathbf{G}})$ is $T_{\mathtt{euf}}$-satisfiable. To see why, suppose it is not. Then, by Proposition 1, there must be some dis-equality $x \neq y \in \alpha(\phi \wedge \delta_{\mathbf{G}})$ such that $(x,y) \in \mathcal{E}(()\phi \wedge \delta_{\mathbf{G}})^*$. Furthermore, by the definition of α, it is clear that x and y must both be in V. But then, by the definition of $\mathfrak{P}^{=}_{\mathtt{euf}}[\![\cdot]\!]$, we must have both $x = y \in \mathfrak{P}^{=}_{\mathtt{euf}}[\![V]\!](\alpha(\phi \wedge \delta_{\mathbf{G}}))$ and $x \neq y \in \mathfrak{P}^{=}_{\mathtt{euf}}[\![V]\!](\alpha(\phi \wedge \delta_{\mathbf{G}}))$, meaning that $\mathfrak{P}^{=}_{\mathtt{euf}}[\![V]\!](\alpha(\phi \wedge \delta_{\mathbf{G}}))$ cannot be satisfiable. But we know that $\mathfrak{P}^{=}_{\mathtt{arr}}[\![V]\!](\phi \wedge \delta_{\mathbf{G}})$ is satisfiable, which is a contradiction.

Now, by Lemma 4, we know that

$$\mathbf{G}_1 = \mathfrak{C}_{\mathtt{euf}}[\![V]\!](\alpha(\phi \wedge \delta_{\mathbf{G}})) \subseteq \mathbf{G} \ .$$

Let $\delta_{\mathbf{G}_1}$ be the arrangement over $\mathbf{G}_1$ that is a subset of $\delta_{\mathbf{G}}$. We have $\delta_{\mathbf{G}_1} \subseteq \delta_{\mathbf{G}} \subseteq \delta_V$ and, since $\alpha(\phi \wedge \delta_{\mathbf{G}}) \wedge \delta_{\mathbf{G}_1}$ is $T_{\mathtt{euf}}$-satisfiable, we can deduce by Theorem 2 that $\alpha(\phi \wedge \delta_{\mathbf{G}}) \wedge \delta_V$ is $T_{\mathtt{euf}}$-satisfiable. Finally, by Lemma 5, we conclude that $\phi \wedge \delta_V$ is $T_{\mathtt{arr}}$-satisfiable. □

B 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 [15] and clarified in [11]. 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 $witness_2$. We modify the combination method of Section 3.3 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. $\phi_2' = witness_2(\phi_2)$.
2. We define $V = vars_S(\phi_2')$ instead of $V = vars(\phi_1) \cap vars(\phi_2)$.

We can now state the correctness of the method in the context of polite theories.

Theorem 4. *Let T_i be a Σ_i-theory polite with respect to the set of sorts S_i, and equipped with a care function $\mathfrak{C}_{T_i}[\![\cdot]\!]$ operating with respect to an equality propagator $\mathfrak{P}^{=}_{T_i}[\![\cdot]\!]$, for $i = 1,2$. Let $\Sigma = \Sigma_1 \cup \Sigma_2$, $S = \Sigma_1^{\mathbb{S}} \cap \Sigma_2^{\mathbb{S}}$, $T = T_1 \oplus T_2$ 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 $\phi_2' = witness_{T_2}(\phi_2)$ and $V = vars_S(\phi_2')$. If $S \subseteq S_1 \cap S_2$, then following are equivalent*

1. *ϕ is T-satisfiable;*
2. *there exists an arrangement $\delta_{\mathbf{G}_2}$ over the care graph $\mathbf{G}_2 = \mathfrak{C}_{T_2}[\![V]\!](\phi_2')$ such that the following formulas are T_1- and T_2-satisfiable respectively*

$$\phi_1 \wedge \mathfrak{P}^{=}_{T}[\![V]\!](\phi_1 \wedge \phi_2' \wedge \delta_{\mathbf{G}_2}) \ , \qquad \phi_2' \wedge \mathfrak{P}^{=}_{T}[\![V]\!](\phi_1 \wedge \phi_2' \wedge \delta_{\mathbf{G}_2}) \ .$$

Moreover, T is polite with respect to $S_1 \cup (S_2 \setminus \Sigma_1)$, and has a combined care function $\mathfrak{C}_T[\![\cdot]\!]$ that operates with respect to the combined theory propagator $\mathfrak{P}^{=}_{T}[\![\cdot]\!]$.

Proof. The proof is identical to the one given in Theorem 1 for the case of stably-infinite theories, except that in the last step, instead of relying on the correctness of the standard Nelson-Oppen method, we rely on the correctness of the method for combination of polite theories as described in [15, 11]. □