Noname manuscript No.
(will be inserted by the editor)

Being Careful about Theory Combination

Dejan Jovanović · Clark Barrett

the date of receipt and acceptance should be inserted later

Abstract One of the main shortcomings of traditional methods for combining theories is the complexity of guessing the arrangement of 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.

Keywords theory combination, Nelson-Oppen, satisfiability modulo theories

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].

New York University

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[1] on the running time [16]. Trying to minimize the burden of non-deterministic guessing is thus 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 (as the polite theory combination method requires extending the set of interface variables significantly).

This paper is based on [14] but has been significantly revised and expanded. In particular, it includes proofs of all theorems, provides additional explanations, simplifies definitions and notation, and adds several illustrative examples. The paper is organized as follows. In Section 2 we introduce background and notation. We then 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.

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 assume that each set of predicates P includes the equality predicates $=_\sigma$, for each sort $\sigma \in S$, where we omit the subscript σ when obvious from the context. 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 of

[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 $O(2^{n^2} \times (\mathcal{T}_1(n) + \mathcal{T}_2(n)))$.

signatures $\Sigma_1 = (S_1, F_1, P_1)$ and $\Sigma_2 = (S_2, F_2, P_2)$ (with arities as in Σ_1 and Σ_2). In this paper, we always assume that, except for equality, function and predicate symbols from different theories do not overlap, so that the arities in the union are well-defined. On the other hand, two different theories are allowed to have non-disjoint sets of sorts. 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 $x = y$, $x \neq y$, $x = f(y_1, \dots, y_n)$, $p(y_1, \dots, y_n)$, or $\neg p(y_1, \dots, y_n)$, where $x, y, y_1, \dots, y_n$ 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 $\mathit{vars}_\sigma(\phi)$ the set of variables of sort σ that occur (free) in ϕ. We overload this function in the usual way, $\mathit{vars}_S(\phi)$ denoting variables in ϕ of the sorts in S, and $\mathit{vars}(\phi)$ 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.

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,[2] 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 $\sigma_1 \times \dots \times \sigma_n \times \tau$ as a function $f^{\mathcal{A}} : A_{\sigma_1} \times \dots \times A_{\sigma_n} \to A_\tau$, and each predicate symbol $p \in \Sigma^{\mathbb{P}}$ of arity $\sigma_1 \times \dots \times \sigma_n$ as a subset $p^{\mathcal{A}}$ of $A_{\sigma_1} \times \dots \times A_{\sigma_n}$. The equality predicate $=_\sigma$ is always interpreted as equality in the domain A_σ.

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. 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) $\mathit{vars}(\phi)$. 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.

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]).

[2] In the rest of the paper we will use the calligraphic letters $\mathcal{A}$, $\mathcal{B}$, ... to denote interpretations, and the corresponding subscripted Roman letters A_σ, B_σ, ... to denote the domains of the interpretations.

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 $\mathbf{A} = \{\Sigma\text{-structures } \mathcal{A} \mid \mathcal{A}^{\Sigma_1,\emptyset} \in \mathbf{A_1} \text{ and } \mathcal{A}^{\Sigma_2,\emptyset} \in \mathbf{A_2}\}$.

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.

Definition 3 (Care Graph) Given a set V of variables, a graph $\mathbf{G} = \langle V, E \rangle$ is a *care graph* over V if $E \subseteq V \times V$ and edges exist only between variables of the same sort. A care graph is *trivial* if it contains all possible edges.

Intuitively, if an edge $(x, y) \in E$ is present in a care graph, it means that we are interested in the relationship between the variables x and y.

Definition 4 (Arrangement) Given a set V of variables, an *arrangement* of V is simply the set of equalities and dis-equalities that hold in some interpretation over V. Any (well-sorted) partition of V corresponds to an arrangement in the obvious way (the arrangement includes the set of equalities between variables in the same partitions and the set of well-sorted dis-equalities between variables in different partitions).

Given a care graph $\mathbf{G} = \langle V, E \rangle$, we call $\delta_{\mathbf{G}}$ an *arrangement over* $\mathbf{G}$ if $\delta_{\mathbf{G}}$ is the restriction of some arrangement (as defined above) to the edges in $\mathbf{G}$. More precisely, $\delta_{\mathbf{G}}$ is an arrangement over $\mathbf{G}$ if there exists an arrangement δ of V such that $x = y \in \delta_{\mathbf{G}}$ iff $x = y \in \delta$ and $(x, y) \in E$, and $x \neq y \in \delta_{\mathbf{G}}$ iff $x \neq y \in \delta$ and $(x, y) \in E$. Note that unless $\mathbf{G}$ is trivial, an arrangement over $\mathbf{G}$ is not an arrangement.

The Nelson-Oppen combination theorem states that under quite general conditions, ϕ is satisfiable in $T_1 \oplus T_2$ iff there exists an arrangement δ_V of the shared variables $V = \mathit{vars}(\phi_1) \cap \mathit{vars}(\phi_2)$ such that $\phi_i \cup \delta_V$ is satisfiable in T_i. Sufficient conditions for the theorem to hold are:

- ϕ is quantifier-free,
- the signatures of T_1 and T_2 have no function or predicate symbols in common, and

- T_1 and T_2 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 (Definition 6 from [21]).

Definition 5 (Stable-Infiniteness) Let Σ be a signature, let $S \subseteq \Sigma^{\mathbb{S}}$ be a set of sorts, and let T be a Σ-theory. We say that T is *stably-infinite* with respect to S if for every T-satisfiable quantifier-free Σ-formula ϕ, there exists a T-interpretation $\mathcal{A}$ satisfying ϕ, such that A_σ is infinite for each sort $\sigma \in S$.

Example 1 We will illustrate the Nelson-Oppen method by means of an example. Suppose we would like to check the following formula ϕ for satisfiability:

$$\underbrace{\bigwedge_{1 \le i < j \le 3} f(x_i, x_4) \neq f(x_j, x_4) \wedge}_{\phi_{\texttt{euf}}} \underbrace{\bigwedge_{1 \le i \le 4} (0 \le x_i \wedge x_i \le 1)}_{\phi_{\texttt{lia}}} \ .$$

Assume that the variables $x_1, \ldots, x_4$ are of integer sort and f is an uninterpreted function symbol. The formula ϕ belongs to the language that combines the theory of linear integer arithmetic ($T_{\texttt{lia}}$) and the theory of uninterpreted functions ($T_{\texttt{euf}}$).

In order to apply the Nelson-Oppen method, we first partition the formula ϕ into $\phi_{\texttt{euf}}$ (belonging to $T_{\texttt{euf}}$) and $\phi_{\texttt{lia}}$ (belonging to $T_{\texttt{lia}}$). Note that both of these formulas are satisfiable in their corresponding theories, but to check satisfiability in the combined theory, in accordance with Nelson-Oppen, we must find an arrangement δ of the set of shared variables $V = \{x_1, x_2, x_3, x_4\}$ such that both $\phi_{\texttt{euf}} \cup \delta$ is satisfiable in $T_{\texttt{euf}}$ and $\phi_{\texttt{lia}} \cup \delta$ is satisfiable in $T_{\texttt{lia}}$.

To check this, we must search through the 15 different ways (this is the Bell number B_4) of arranging the variables, and check each one.

$$\begin{aligned}
\delta_V^1 &= \{x_1 = x_2, x_1 = x_3, x_1 = x_4, x_2 = x_3, x_2 = x_4, x_3 = x_4\} \ , \\
\delta_V^2 &= \{x_1 \neq x_2, x_1 \neq x_3, x_1 \neq x_4, x_2 = x_3, x_2 = x_4, x_3 = x_4\} \ , \\
&\vdots \\
\delta_V^{15} &= \{x_1 \neq x_2, x_1 \neq x_3, x_1 \neq x_4, x_2 \neq x_3, x_2 \neq x_4, x_3 \neq x_4\} \ .
\end{aligned}$$

It turns out that none of these arrangements work for both theories, and we may thus conclude that the original formula ϕ is unsatisfiable. To see that ϕ is unsatisfiable, it is enough to notice that $\phi_{\texttt{euf}}$ requires the three variables x_1, x_2, x_3 (but not x_4) to be different, but this is impossible since $\phi_{\texttt{lia}}$ requires these variables to take on values from a set of only two integers.

The important insight for the purpose of this paper is that the variable x_4 is not part of the reasoning that leads to inconsistency and thus, intuitively, it should be sufficient to check only arrangements over $V' = \{x_1, x_2, x_3\}$. The number of different arrangements over V' is much smaller (the Bell number $B_3 = 5$), and thus the complexity of the search would be significantly reduced.

Note that even if the arrangements are explored incrementally, with each theory solver propagating entailed equalities (as is typically done in efficient implementations), effort will still be wasted if any case-split involving x_4 is done. Without any additional information to guide the search, there is no guarantee that this wasted effort can be avoided.

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.

To simplify the presentation (as well as the formal proofs), we present the method in its non-deterministic flavor, following the approach of [20]. We will first define the equality propagator and the care function, and then proceed to presenting and proving correctness of the combination method.

Definition 6 (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:

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

where $\mathit{vars}(\mathfrak{P}_T[\![V]\!](\phi)) \subseteq V$ 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 (note also that for a complete propagator, properties 3 and 4 are consequences of properties 1 and 2).

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 7 (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

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

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

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

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 is 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 8 (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.

The main feature of a care function is that when it returns an empty graph, this guarantees that ϕ can be satisfied regardless of the relationships between the remaining variables. In other cases, notice that the definition does not specify (beyond requirement 2) anything about what the care graph should contain. This is because no additional conditions are required for correctness. However, to be effective, a care function should return a care graph which (to the extent that is efficiently possible) contains only edges corresponding to pairs of variables which, if set equal or dis-equal, could affect the satisfiability of the formula ϕ.

Example 2 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 maximal care graph, i.e.,

$$\mathfrak{C}_0[\![V]\!](\phi) = \langle V, E_0\rangle\ , \text{ where } (x, y) \in E_0 \text{ iff } \begin{cases} x \in V, y \in V, \\ x \text{ and } y \text{ have the same sort,} \\ x = y \notin \mathfrak{P}_T[\![V]\!](\phi), \text{ and} \\ x \neq y \notin \mathfrak{P}_T[\![V]\!](\phi)\ . \end{cases}$$

Notice that $\mathfrak{C}_0[\![\cdot]\!]$ trivially satisfies the conditions of Definition 8 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 an equality 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 $i = 1, 2$. This step is identical to the first step in the standard Nelson-Oppen combination method.

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, and let δ_V be an arrangement (chosen non-deterministically) of V. 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 $\delta_{\mathbf{G}}$ is the arrangement over $\mathbf{G}$ obtained by restricting δ_V to the edges in $\mathbf{G}$.

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 operator $\mathcal{G}$ is increasing, a fix-point always exists. Moreover, it is in our interest to choose a minimal such fix-point, which we can obtain by doing a fix-point iteration starting from $\mathbf{G}_0 = \langle V, \emptyset \rangle$ ($\mathcal{G}$ preserves the property of being a care graph). Another important observation is that for any fix-point $\mathbf{G}_2$ (with respect to the chosen value of δ_V) of the operator $\mathcal{G}$ above, we must have that the care function application in (1) returns an empty graph. This follows from the fact that $\mathbf{G}_2$ is a fix-point and the observation 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 3 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 outputs of the purification phase, and let V be the set of variables shared by ϕ_1 and ϕ_2. Let δ_V be the arrangement of V chosen in the **Arrange** phase, and let $\mathbf{G}_2$ be the fix-point graph. Since $\mathfrak{C}_{T_2}[\![\cdot]\!]$ is a trivial care function, the arrangement $\delta_{\mathbf{G}_2}$ over $\mathbf{G}_2$ must be equivalent to δ_V. Then, since the equality propagators simply keep the input equalities and dis-equalities over V, and all relationships between variables in V are determined by $\delta_{\mathbf{G}_2}$, the combined propagator will simply return δ_V and we will thus check $\phi_1 \cup \delta_V$ and $\phi_2 \cup \delta_V$ for satisfiability in the **Check** phase. 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 propagator $\mathfrak{P}_{T_i}[\![\cdot]\!]$, 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, with $V = \mathit{vars}(\phi_1) \cap \mathit{vars}(\phi_2)$. If $\Sigma_1^{\mathbb{S}} \cap \Sigma_2^{\mathbb{S}} = S_1 \cap S_2$, then the following are equivalent:*

1. *ϕ is T-satisfiable;*

2. there exists a care graph $\mathbf{G}_2$ *and an arrangement* $\delta_{\mathbf{G}_2}$ *over* $\mathbf{G}_2$ *such that* $\mathbf{G}_2$ *is a fix-point solution of* (1), *and 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) ⇒ (2) : Suppose $\phi = \phi_1 \cup \phi_2$ is T-satisfiable in a T-interpretation $\mathcal{A}$. Let δ_V be the arrangement of V satisfied by $\mathcal{A}$, and let $\mathbf{G}_2$ be the trivial care graph over V. It is easy to see that $\mathbf{G}_2$ is a fix-point solution of (1) (with respect to arrangement δ_V), and that $\delta_{\mathbf{G}_2} = \delta_V$. Then, because $\mathcal{A}$ satisfies ϕ_1, ϕ_2, and 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) ⇐ (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 of 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 $\mathfrak{C}_{T_2}[\![V]\!](\phi_2 \cup \mathfrak{P}_T[\![V]\!](\phi_1 \cup \phi_2 \cup \delta_{\mathbf{G_2}})) = \langle V, \emptyset \rangle$. 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 an arrangement of 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].

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. 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]. A theory can be combined with any other theory if it is *polite* with respect to the set of shared sorts. The notion of politeness depends on two other important properties: *smoothness* and *finite witnessability.*

Definition 9 (Smoothness) Let Σ be a signature, let $S \subseteq \Sigma^{\mathbb{S}}$ be a set of sorts, and let T be a Σ-theory. We say that T is *smooth* with respect to S if:

- for every T-satisfiable quantifier-free Σ-formula ϕ,
- for every T-interpretation $\mathcal{A}$ satisfying ϕ,
- for all choices of cardinal numbers κ_σ, such that $\kappa_\sigma \geq |A_\sigma|$ for all $\sigma \in S$,

there exists a T-interpretation $\mathcal{B}$ satisfying ϕ such that $|B_\sigma| = \kappa_\sigma$, for all $\sigma \in S$.

Being able to combine two interpretations from different theories mainly depends on the ability to bring the domains of the shared sorts to the same size. This is where stable-infiniteness helps in the Nelson-Oppen framework: it ensures that the domains of the shared sorts can have the same infinite cardinalities. Smoothness gives us more flexibility in resizing structures upwards. On the other hand, finite-witnessability allows more flexibility in finding "minimal" structures.

Definition 10 (Finite Witnessability) Let Σ be a signature, let $S \subseteq \Sigma^{\mathbb{S}}$ be a set of sorts, and let T be a Σ-theory. We say that T is *finitely witnessable* with respect to S if there exists a computable function, *witness*, which, for every quantifier-free Σ-formula ϕ, returns a quantifier-free Σ-formula $\psi = \mathit{witness}(\phi)$ such that

- ϕ and $(\exists \overrightarrow{w})\psi$ are T-equivalent, where $\overrightarrow{w} = \mathit{vars}(\psi) \setminus \mathit{vars}(\phi)$ are fresh variables;
- if $\psi \wedge \delta_V$ is T-satisfiable, for an arrangement δ_V, where V is a set of variables of sorts in S, then there exists a T-interpretation $\mathcal{A}$ satisfying $\psi \wedge \delta_V$ such that $A_\sigma = [\mathit{vars}_\sigma(\psi \wedge \delta_V)]^{\mathcal{A}}$, for all $\sigma \in S$,

where the notation $[U]^{\mathcal{A}}$ indicates the set $\{ v^{\mathcal{A}} \mid v \in U \}$.

Definition 11 (Politeness) Let Σ be a signature, let $S \subseteq \Sigma^{\mathbb{S}}$ be a set of sorts, and let T be a Σ-theory. We say that T is *polite* with respect to S if it is both smooth and finitely witnessable with respect to S.

The other prerequisites for the polite combination method are the same as for the standard combination method, i.e. we keep all the assumptions on the theories T_1 and T_2 with the exception of the stable-infiniteness. Instead, we assume that the theory T_2 is polite with respect to a set of sorts S_2 with $\Sigma_1^{\mathbb{S}} \cap \Sigma_2^{\mathbb{S}} \subseteq S_2$, and is equipped with a witness function $\mathit{witness}_2$. We can now 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 propagator $\mathfrak{P}_{T_i}[\![\cdot]\!]$, 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 $\Sigma_1^{\mathbb{S}} \cap \Sigma_2^{\mathbb{S}} \subseteq S_2$, 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 $S_1 \cup (S_2 \setminus \Sigma_1^{\mathbb{S}})$.

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 [12,13,18].

4 Theory of Uninterpreted Functions

The theory of uninterpreted functions, over a signature $\Sigma_{\texttt{euf}}$, is the theory $T_{\texttt{euf}} = (\Sigma_{\texttt{euf}}, \mathbf{A})$, where $\mathbf{A}$ is simply the class of all $\Sigma_{\texttt{euf}}$-structures. Conjunctions of literals in this theory can be decided for satisfiability 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 $\Sigma_{\texttt{euf}}$ contains no predicate symbols, but the extension to the case with predicate symbols is straightforward.

4.1 Equality Propagator

Let ϕ be a set of flat literals and V a set of variables. We write $\mathcal{E}(\phi)$ to denote the smallest equivalence relation over the terms occurring in ϕ containing $\{(x,t) \mid x = t \in \phi\}$. We also write $\mathcal{E}_c(\phi)$ for the smallest congruence relation[3] over terms in ϕ containing the same base set $\{(x,t) \mid x = t \in \phi\}$. We define a dis-equality (modulo congruence) relation $\mathcal{N}_c(\phi)$ as the smallest relation satisfying

$$(x,x') \in \mathcal{E}_c(\phi) \text{ and } (y,y') \in \mathcal{E}_c(\phi) \text{ and } x' \neq y' \in \phi \Longrightarrow (x,y) \in \mathcal{N}_c(\phi) \ .$$

Now, we define the equality propagator as

$$\begin{aligned} \mathfrak{P}_{\texttt{euf}}[\![V]\!](\phi) = &\{\ x = y \mid x,y \in V,\ (x,y) \in \mathcal{E}_c(\phi)\ \} \\ &\cup \{\ x \neq y \mid x,y \in V,\ (x,y) \in \mathcal{N}_c(\phi)\ \} \ . \end{aligned}$$

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

Example 4 Given the set of flat literals $\phi = \{\ x = z,\ y = f(a),\ z \neq f(a)\ \}$, the equality propagator would return

$$\mathfrak{P}_{\texttt{euf}}[\![x,y]\!](\phi) = \{\ x = x,\ y = y,\ x \neq y,\ y \neq x\ \} \ .$$

In practice, of course, an implementation of such a propagator does not need to bother with the propagation of trivial equalities, such as $x = x$, or symmetric versions of the same (dis-)equality.

4.2 Care Function

We will define the care function based on the observation that, during construction of the congruence closure, we only care about equalities between pairs of variables that occur as arguments of the same function symbol, in the same position. Again, let V be a set of variables and let ϕ be a set of flat literals. Assume also that ϕ only contains function symbols from $F = \{f_1, f_2, \ldots, f_n\} \subseteq \Sigma^{\mathbb{F}}_{\texttt{euf}}$.

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 from V that could trigger an application of congruence. More precisely, $(x,y) \in E_f$ iff:

[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 $1 \leq i \leq n$, $(x_i,y_i) \in \mathcal{E}_c(\phi)$, then $(f(x_1,\ldots,x_n), f(y_1,\ldots,y_n)) \in \mathcal{E}_c(\phi)$.

1. $x \in V$ and $y \in V$,
2. $(x, y) \notin \mathcal{E}_c(\phi)$ and $(x, y) \notin \mathcal{N}_c(\phi)$,
3. there exist terms $t_1 = f(x_1, \ldots, x_i, \ldots, x_k)$ and $t_2 = f(y_1, \ldots, y_i, \ldots, y_k)$ in ϕ such that:
   (a) for some $1 \leq i \leq k$, $(x, x_i) \in \mathcal{E}_c(\phi)$ and $(y, y_i) \in \mathcal{E}_c(\phi)$;
   (b) $(t_1, t_2) \notin \mathcal{E}_c(\phi)$;
   (c) for $1 \leq j \leq k$, $(x_j, y_j) \notin \mathcal{N}_c(\phi)$.

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

Example 5 Consider the following sets of literals

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

and corresponding sets of shared variables

$$V_1 = \{x_1, x_2\}\ , \qquad V_2 = \{x_1, x_2, y_1, y_2\}\ , \qquad V_3 = \{x_1, x_2, y_2, z_2\}\ .$$

Each of the individual formulas ϕ_1, ϕ_2, and ϕ_3 are satisfiable in $T_{\mathtt{euf}}$. Let us examine each of them in turn, where for the sake of brevity, during the computation, we only report the non-reflexive, non-symmetric versions of the propagated equalities and relations, and skip the clearly irrelevant ones.

The congruence relations corresponding to the formula ϕ_1 are the following

$$\mathcal{E}_c(\phi_1) = \{\ (x_2, y_1)\ \}\ , \qquad \mathcal{N}_c(\phi_1) = \{\ (f(x_1), f(y_1))\ \}\ .$$

Note that, since y_1 is not a shared variable ($y_1 \notin V_1$), the $T_{\mathtt{euf}}$ propagator will return $\mathfrak{P}_{\mathtt{euf}}[\![V_1]\!](\phi_1) = \emptyset$. Nevertheless, the variable y_1 does occur as an argument of f and since x_2 is a shared variable with $(y_1, x_2) \in \mathcal{E}_c(\phi_1)$, we do consider x_2 important as it is an alias for y_1. We now have two terms $f(x_1)$ and $f(y_1)$ that are not congruent in $\mathcal{E}_c(\phi_1)$ and, to make sure that we can ensure satisfiability, we need to know the relation between x_1 and x_2. By definition, the care function therefore returns the care graph $\mathbf{G}_1 = \langle V_1, \{(x_1, x_2)\}\rangle$.

Moving on to ϕ_2, let's assume that, in addition to ϕ_2, we are also considering the integer arithmetic formula $\phi_2^{\mathtt{lia}} = \{x_1 \leq y_1, x_1 \geq y_1, x_2 \leq y_2\}$, thus checking $\phi_2 \wedge \phi_2^{\mathtt{lia}}$ for satisfiability in the combined theory $T = T_{\mathtt{euf}} \oplus T_{\mathtt{lia}}$. The congruence relations for ϕ_2 alone are the following

$$\mathcal{E}_c(\phi_2) = \{\ (z_1, f(x_1)), (z_2, f(y_1))\ \}\ , \qquad \mathcal{N}_c(\phi_2) = \{\ (g(z_1, x_2), g(z_2, y_2))\ \}\ .$$

Again, from ϕ_2 alone we can not conclude any relationship between the shared variables so, at this point, the propagator will return $\mathfrak{P}_{\mathtt{euf}}[\![V_2]\!](\phi_2) = \emptyset$. But, since we are combining two theories, we will compute the combined propagator by exchanging the propagated equalities and dis-equalities. The propagator for the theory of integer arithmetic might be sophisticated enough to propagate $\mathfrak{P}_{\mathtt{lia}}[\![V_2]\!](\phi_2^{\mathtt{lia}}) = \{x_1 = y_1\}$. If so, this new equality is appended to ϕ_2, obtaining $\phi_2^1 = \phi_2 \cup \{x_1 = y_1\}$ and processed further. With the new information, we can obtain the stronger equivalence relations

$$\mathcal{E}_c(\phi_2^1) = \{\ (z_1, f(x_1)), (z_2, f(y_1)), (x_1, y_1), (f(x_1), f(y_1)), (z_1, z_2)\ \}\ ,$$
$$\mathcal{N}_c(\phi_2^1) = \{\ (g(z_1, x_2), g(z_2, y_2))\ \}\ .$$

The stronger equivalence relations then influence the $T_{\texttt{euf}}$ propagator to return $\mathfrak{P}_{\texttt{euf}}[\![V_2]\!](\phi_2^1) = \{x_1 = y_1\}$ and, in fact, this is the final result of the combined theory propagator $\mathfrak{P}_T[\![V_2]\!](\phi_2) = \{x_1 = y_1\}$.

We can now proceed with computation of the care graph. In the set of shared variables, the pairs (x_1, y_1) and (x_2, y_2) appear as arguments of function terms at the same position. But, we already know the relationship between x_1 and y_1, so the only pair that we are interested is the pair (x_2, y_2), i.e. the care graph we compute is $\mathbf{G}_2 = \langle V_2, \{(x_2, y_2)\}\rangle$. If we now choose the arrangement $\delta_{\mathbf{G}_2} = \{x_2 \neq y_2\}$, the pair $\langle \mathbf{G}_2, \delta_{\mathbf{G}_2}\rangle$ will in fact be a fix-point solution to (1), and we check the following formulas for satisfiability

$$\begin{aligned} T_{\texttt{euf}} &: \phi_2 \wedge x_1 = y_1 \wedge x_2 \neq y_2 \\ T_{\texttt{lia}} &: \phi_2^{\texttt{lia}} \wedge x_1 = y_1 \wedge x_2 \neq y_2 \end{aligned}$$

Both of these formulas are satisfiable, and we can conclude that the original formula $\phi_2 \wedge \phi_2^{\texttt{lia}}$ is satisfiable in the combined theory.

Finally, let's consider the formula ϕ_3 and its $T_{\texttt{lia}}$ counterpart $\phi_3^{\texttt{lia}} = \{x_1 + x_2 + y_2 + z_2 \leq 1\}$. The equivalence relations corresponding to ϕ_3 are

$$\begin{aligned} \mathcal{E}_c(\phi_3) &= \{\ (y_1, f(x_1)), (y_2, f(x_2)), (z_1, g(x_1)), (z_2, g(x_2))\ \} \quad , \\ \mathcal{N}_c(\phi_3) &= \{\ (h(y_1), h(z_1))\ \} \quad . \end{aligned}$$

In this case, none of the individual theory propagators provide us with any additional information about the shared variables, so the combined propagator returns $\mathfrak{P}_T[\![V_3]\!](\phi_3 \wedge \phi_3^{\texttt{lia}}) = \emptyset$. We proceed to compute the care graph by noting that the only pair of shared variables under the same function symbol are x_1 and x_2, and so the care graph we compute is $\mathbf{G}_3^1 = \langle V_3, \{(x_1, x_2)\}\rangle$. We can now choose the arrangement $\delta_{\mathbf{G}_3^1} = \{x_1 = x_2\}$. As this pair is still not a fix-point of (1), we continue with the computation, by considering $\phi_3^1 = \phi_3 \cup \{x_1 = x_2\}$.

With the new information, the stronger equivalence relations are

$$\begin{aligned} \mathcal{E}_c(\phi_3^1) &= \mathcal{E}_c(\phi_3) \cup \{\ (y_1, y_2), (z_1, z_2), (f(x_1), f(x_2)), (g(x_1), g(x_2))\ \} \quad , \\ \mathcal{N}_c(\phi_3^1) &= \{\ (h(y_1), h(z_1))\ \} \quad . \end{aligned}$$

We are now in a similar situation as when considering ϕ_1. The variables y_1 and z_1 are not shared, but under the equivalence relation $\mathcal{E}_c(\phi_3^1)$ are in fact aliases for the shared variables y_2 and z_2, respectively. Since x_2 and z_2 appear as arguments of h in the formula, we now care about the pair (y_2, z_2), so we add it to the care graph to obtain $\mathbf{G}_3^2 = \langle V_3, \{(x_1, x_2), (y_2, z_2)\}\rangle$. We can now choose that y_2 and z_2 are different obtaining the arrangement $\delta_{\mathbf{G}_3^2} = \langle V_3, \{x_1 = x_2, y_2 \neq z_2\}\rangle$. This pair is a fix-point of (1), so we check the following formulas for satisfiability in the individual theories

$$\begin{aligned} T_{\texttt{euf}} &: \phi_3 \wedge x_1 = x_2 \wedge y_2 \neq z_2 \\ T_{\texttt{lia}} &: \phi_3^{\texttt{lia}} \wedge x_1 = x_2 \wedge y_2 \neq z_2 \end{aligned}$$

With both of the formulas satisfiable, we conclude that $\phi_3 \wedge \phi_3^{\texttt{lia}}$ is satisfiable in the combined theory T.

We prove correctness of the care function for $\mathfrak{C}_{\texttt{euf}}[\![\cdot]\!]$ by relying on the following well-known proposition.

Proposition 1 *A set ϕ of flat literals is $T_{\texttt{euf}}$-satisfiable if and only if, for each disequality $x \neq y \in \phi$, we have that $(x, y) \notin \mathcal{E}_c(\phi)$.*

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

Proof Let ϕ be a satisfiable set of flat literals, V a set of variables, and $\mathbf{G} = \mathfrak{C}_{\texttt{euf}}[\![V]\!](\phi) = \langle V, \emptyset \rangle$. Suppose we are given an arrangement δ_V, corresponding to equivalence relation E_V, with $\mathfrak{P}_{\texttt{euf}}[\![V]\!](\phi) \subseteq \delta_V$. We must show that $\phi \wedge \delta_V$ is $T_{\texttt{euf}}$-satisfiable. We know, by Proposition 1 above, that if $x \neq y \in \phi$, then $(x, y) \notin \mathcal{E}_c(\phi)$. In order to prove the theorem it suffices to show that:

$$\text{if } x \neq y \in \phi \cup \delta_V \text{ then } (x, y) \notin \mathcal{E}_c(\phi \cup \delta_V) \ .$$

First, it is clear that, since δ_V is compatible with the propagated equalities from ϕ, we must have that:

$$\text{if } v, w \in V \text{ and } (v, w) \in \mathcal{E}_c(\phi) \text{ then } (v, w) \in E_V \ . \tag{2}$$

We now show that:

$$\text{if } v, w \in V \text{ and } (v, w) \in \mathcal{E}(\mathcal{E}_c(\phi) \cup E_V) \text{ then } (v, w) \in E_V \ . \tag{3}$$

Suppose $v, w \in V$ and $(v, w) \in \mathcal{E}(\mathcal{E}_c(\phi) \cup E_V)$ but $(v, w) \notin E_V$. We know $(v, w) \notin \mathcal{E}_c(\phi)$ by (2). The only other possibility is that there is some transitive chain from v to w using pairs from both $\mathcal{E}_c(\phi)$ and E_V. Let $(t_0, t_1), (t_1, t_2), \ldots, (t_{n-1}, t_n)$ be the smallest such chain (with $v = t_0$ and $w = t_n$). Let (t_i, t_{i+1}) be the first pair such that $t_i \in V$ but $t_{i+1} \notin V$. Such a pair must exist since, by (2), every pair in $\mathcal{E}_c(\phi) \cap (V \times V)$ is also in E_V, so that if $t_k \in V$ for all k, we would have $(t_0, t_n) \in E_V$. Then, let (t_j, t_{j+1}) be first pair such that $j > i$ and $t_j \notin V$ and $t_{j+1} \in V$ (there must be such a pair since $t_n \in V$). Notice that every pair from (t_i, t_{i+1}) to (t_j, t_{j+1}) must be in $\mathcal{E}_c(\phi)$ since each contains a term not in V. But then $(t_i, t_{j+1}) \in \mathcal{E}_c(\phi)$ which contradicts the assumption that $(t_0, t_1), (t_1, t_2), \ldots, (t_{n-1}, t_n)$ is the smallest chain from t_0 to t_n. This establishes (3).

The following property follows:

$$\text{if } (s, t) \in \mathcal{E}(\mathcal{E}_c(\phi) \cup E_V) \text{ then } (s, t) \notin \mathcal{N}_c(\phi) \ . \tag{4}$$

Suppose $(s, t) \in \mathcal{E}(\mathcal{E}_c(\phi) \cup E_V)$ and $(s, t) \in \mathcal{N}_c(\phi)$. Because ϕ is satisfiable, we know that $(s, t) \notin \mathcal{E}_c(\phi)$. So, as above, there must be some transitive chain from s to t using pairs from both $\mathcal{E}_c(\phi)$ and E_V. Let v be the first term in this chain such that $v \in V$ and w the last term in the chain such that $w \in V$. By definition, we must have $(v, w) \in \mathcal{N}_c(\phi)$. It follows that $v \neq w \in \delta_V$ and thus $(v, w) \notin E_V$. But by (3), it follows that $(v, w) \in E_V$. This establishes (4).

We can then conclude:

$$\mathcal{E}_c(\phi \cup \delta_V) = \mathcal{E}(\mathcal{E}_c(\phi) \cup E_V) \ . \tag{5}$$

To see this, note first that by basic properties of equivalence and congruence closures we have that

$$\mathcal{E}_c(\phi \cup \delta_V) = \mathcal{E}_c(\mathcal{E}_c(\phi) \cup \mathcal{E}(\delta_V)) = \mathcal{E}_c(\mathcal{E}_c(\phi) \cup E_V) \ .$$

To see that $\mathcal{E}_c(\mathcal{E}_c(\phi) \cup E_V) = \mathcal{E}(\mathcal{E}_c(\phi) \cup E_V)$, suppose that this is not the case. Then there must exist a pair of function applications $t_1 = f(x_1, x_2, \ldots x_n)$ and $t_2 = f(y_1, y_2, \ldots y_n)$, appearing in ϕ, such that $(t_1, t_2) \notin \mathcal{E}_c(\phi)$ but for each $1 \le j \le n$, $(x_j, y_j) \in \mathcal{E}(\mathcal{E}_c(\phi) \cup E_V)$ (from which it follows by (4) that $(x_j, y_j) \notin \mathcal{N}_c(\phi)$). Since $\mathcal{E}_c(\phi)$ is closed under congruence, there must be some i such that $(x_i, y_i) \notin \mathcal{E}_c(\phi)$ and $(x_i, y_i) \in \mathcal{E}(\mathcal{E}_c(\phi) \cup E_V)$.

But then we must have a chain of equalities connecting x_i to y_i, such that at least one equality comes from E_V. This chain then has to contain at least two variables from V. Let x be the first variable from V, and y the last variable from V, in this equality chain. Since the chains from x_i to x, and y_i to y do not contain any variables from V, all these equalities must come from $\mathcal{E}_c(\phi)$, so it must be that $(x_i, x) \in \mathcal{E}_c(\phi)$ and $(y_i, y) \in \mathcal{E}_c(\phi)$. We can conclude that $(x, y) \notin \mathcal{E}_c(\phi)$ since otherwise we could deduce that $(x_i, y_i) \in \mathcal{E}_c(\phi)$. Additionally, it must be that $(x, y) \notin \mathcal{N}_c(\phi)$, as otherwise we would have $x \neq y \in \delta_V$ and thus $(x, y) \notin E_V$, but we know from (3) that $(x, y) \in E_V$.

But now we can see that x and y satisfy all the requirements necessary to ensure that the edge (x, y) must be in the care graph $\mathbf{G}$, contradicting the fact that it should be empty, which establishes (5).

Finally, we return to the main proof and show that if $x \neq y \in \phi \cup \delta_V$, then $(x, y) \notin \mathcal{E}_c(\phi \cup \delta_V)$. We consider two cases.

- First, suppose $x \neq y \in \delta_V$ (and thus $x, y \in V$). Clearly, we cannot also have $x = y \in \delta_V$, so $(x, y) \notin E_V$. It follows by (3) that $(x, y) \notin \mathcal{E}(\mathcal{E}_c(\phi) \cup E_V)$, and thus, by (5), $(x, y) \notin \mathcal{E}_c(\phi \cup \delta_V)$.
- On the other hand, suppose that $x \neq y \in \phi$. This means that $(x, y) \in \mathcal{N}_c(\phi)$. By (4), it follows that $(x, y) \notin \mathcal{E}(\mathcal{E}_c(\phi) \cup E_V)$, and thus, by (5), $(x, y) \notin \mathcal{E}_c(\phi \cup \delta_V)$.

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

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

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

$$\forall\, a{:}\mathsf{array}.\ \forall\, i{:}\mathsf{index}.\ \forall\, v{:}\mathsf{elem}.\ \mathsf{read}(\mathsf{write}(a, i, v), i) = v \ , \tag{A1}$$

$$\forall\, a{:}\mathsf{array}.\ \forall\, i, j{:}\mathsf{index}.\ \forall\, v{:}\mathsf{elem}.\ i \neq j \rightarrow \mathsf{read}(\mathsf{write}(a, i, v), j) = \mathsf{read}(a, j) \ , \tag{A2}$$

$$\forall\, a, b{:}\mathsf{array}.\ (\forall\, i{:}\mathsf{index}.\ \mathsf{read}(a, i) = \mathsf{read}(b, i)) \rightarrow a = b \ . \tag{A3}$$

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, c are variables of sort array; and w, z are variables of any sort. For a set ϕ of flat $T_{\mathtt{arr}}$-literals, we also define $\alpha(\phi)$ to be the subset of ϕ that does not contain literals of the form $a = \mathsf{write}(b, i, v)$.

5.1 Decision Procedure

Before presenting the equality propagator and care function, it will be helpful to present a simple rule-based decision procedure for $T_{\mathtt{arr}}$. The procedure we present is based on [9], with the main difference that in our procedure, we exclude literals containing write terms from the $T_{\mathtt{euf}}$-satisfiability check as they are not needed and this allows us to have a simpler care function.

Given a set Γ of flat $T_{\mathtt{arr}}$-literals, we define $\mathcal{E}_a(\Gamma)$ as $\mathcal{E}_c(\alpha(\Gamma))$ and, as usual, the corresponding dis-equality relation $\mathcal{N}_a(\Gamma)$ as the smallest relation satisfying:

$$\text{if } (w, w') \in \mathcal{E}_a(\Gamma) \text{ and } (z, z') \in \mathcal{E}_a(\Gamma) \text{ and } w' \neq z' \in \Gamma \text{ then } (w, z) \in \mathcal{N}_a(\Gamma) \enspace .$$

As a matter of notational convenience in this section we write $x \approx_a^{\Gamma}$ for $(x, y) \in \mathcal{E}_a(\Gamma)$ and $x \neq_a^{\Gamma} y$ for $(x, y) \in \mathcal{N}_a(\Gamma)$. The following lemma is a straightforward consequence of the fact that adding additional information can only increase the set of consequences of a set of formulas.

Lemma 1 *Suppose ϕ and Γ are sets of flat $T_{\mathtt{arr}}$-literals with $\phi \subseteq \Gamma$. Then:*

- $s \approx_a^{\phi} t \implies s \approx_a^{\Gamma} t$ *, and*
- $s \neq_a^{\phi} t \implies s \neq_a^{\Gamma} t$.

We now present the inference rules of the decision procedure for $T_{\mathtt{arr}}$. For a set of literals Γ, we write $\Gamma[l_1, \ldots, l_n]$ to denote that literals $l_1, \ldots, l_n$ appear in Γ. For every pair (a, b) of variables from $\mathit{vars}_{\mathsf{array}}(\Gamma)$, we let $k_{a,b}$ be a distinguished fresh variable of sort index. Let $\mathcal{D}_{\mathtt{arr}}$ be the following set of inference rules.

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

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

$$\mathsf{ArrDiseq} \qquad \frac{\Gamma[a \neq b]}{\Gamma, \mathsf{read}(a, k_{a,b}) \neq \mathsf{read}(b, k_{a,b})} \qquad \text{if} \quad \text{not } \mathsf{read}(a, k_{a,b}) \neq_a^{\Gamma} \mathsf{read}(b, k_{a,b})$$

Note that although non-flat literals appear in the conclusions of rules $\mathsf{RIntro2}$ and $\mathsf{ArrDiseq}$, we only 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.*

Proof By sound, we mean that for each rule, the set of literals in the premise is $\mathcal{T}_{\mathtt{arr}}$-satisfiable iff one of the conclusion sets is $\mathcal{T}_{\mathtt{arr}}$-satisfiable. It is not hard to see that the soundness of the $\mathsf{RIntro1}$ rule follows from axiom ($\mathsf{A1}$) of $\mathcal{T}_{\mathtt{arr}}$, the soundness of $\mathsf{RIntro2}$ follows from axiom ($\mathsf{A2}$), and the soundness of $\mathsf{ArrDiseq}$ from axiom ($\mathsf{A3}$).

To see that the rules are terminating, first notice that applying a rule results in a new set Γ which no longer satisfies the side conditions of the rule just applied, so every application of a rule along a derivation branch must involve different "trigger" formulas (the ones in square brackets). Now, no rule introduces array dis-equalities, so it is clear that $\mathsf{ArrDiseq}$ can only be applied a finite number of times. Similarly, no rule introduces a new literal containing write, so $\mathsf{RIntro1}$ can only be applied a finite number of times. Now, suppose we have a set Γ in which both the $\mathsf{RIntro1}$ rule and the $\mathsf{ArrDiseq}$ rule no longer apply, and consider rule $\mathsf{RIntro2}$ which may introduce new read terms. The rule cannot, however, introduce new array or index variables, so there are only a finite number of read terms that can be generated. Each application of $\mathsf{RIntro2}$ merges the equivalence classes of either two index terms or two read terms. Since there are a finite number of both, eventually, no more merges will be possible.

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

Proof Since $\mathcal{T}_{\mathtt{euf}}$ also includes all the structures of $\mathcal{T}_{\mathtt{arr}}$, and $\alpha(\Gamma) \subseteq \Gamma$, the only-if direction is trivial. For the other direction, suppose Γ is a $\mathcal{D}_{\mathtt{arr}}$-saturated set of flat $\mathcal{T}_{\mathtt{arr}}$-literals. Let $\mathcal{A}$ be a maximally diverse $\mathcal{T}_{\mathtt{euf}}$ model of $\alpha(\Gamma)$ (for instance, the $\approx_a^\Gamma$-quotient of the term model) and note that it has the property that for any two terms s and t of the same sort, $s \approx_a^\Gamma t$ iff $s^\mathcal{A} = t^\mathcal{A}$. We will show that Γ is $\mathcal{T}_{\mathtt{arr}}$-satisfiable by constructing a $\mathcal{T}_{\mathtt{arr}}$ interpretation $\mathcal{B}$ that satisfies Γ. We define the domains of $\mathcal{B}$ as

$$\begin{aligned} B_{\mathsf{index}} &= A_{\mathsf{index}} \ , \\ B_{\mathsf{elem}} &= A_{\mathsf{elem}} \ , \\ B_{\mathsf{array}} &= \{\ f \ \mid \ f : B_{\mathsf{index}} \mapsto B_{\mathsf{elem}}\ \} \ . \end{aligned}$$

We further define the interpretations of function symbols as

$$\begin{aligned} \mathsf{read}^\mathcal{B} &= \lambda\, a : B_{\mathsf{array}}.\ \lambda\, i : B_{\mathsf{index}}.\ a(i) \ , \\ \mathsf{write}^\mathcal{B} &= \lambda\, a : B_{\mathsf{array}}.\ \lambda\, i : B_{\mathsf{index}}.\ \lambda\, x : B_{\mathsf{elem}}.\ (\lambda\, j : B_{\mathsf{index}}.\text{if } i = j \text{ then } x \text{ else } a(j)) \ , \\ i^\mathcal{B} &= i^\mathcal{A} \ , \\ x^\mathcal{B} &= x^\mathcal{A} \ . \end{aligned}$$

We interpret each array $a \in \mathit{vars}_{\mathsf{array}}(\Gamma)$ as the corresponding function from $\mathcal{A}$ in a restricted manner. Let e_0 be some distinguished element of B_{elem}. Then

$$a^\mathcal{B} = \lambda\, e : B_{\mathsf{index}}.\ \begin{cases} x^\mathcal{A} & \text{if } x = \mathsf{read}(b, i) \in \Gamma \text{ and } a \approx_a^\Gamma b \text{ and } i^\mathcal{A} = e \\ e_0 & \text{otherwise.} \end{cases}$$

To see that this interpretation is well-defined, suppose that for some variable a, we have both $x = \mathsf{read}(b, i) \in \Gamma$ and $y = \mathsf{read}(c, j) \in \Gamma$, with $a \approx_a^\Gamma b \approx_a^\Gamma c$ and

$i^{\mathcal{A}} = j^{\mathcal{A}} = e$. Clearly, $b^{\mathcal{A}} = c^{\mathcal{A}}$, but then it must be the case that $\mathsf{read}(b,i)^{\mathcal{A}} = \mathsf{read}(c,j)^{\mathcal{A}}$ and so $x^{\mathcal{A}} = y^{\mathcal{A}}$.

It is easy to see that the definitions of read and write satisfy the axioms of $T_{\mathtt{arr}}$. Now, we proceed to show that $\mathcal{B} \models \Gamma$. First, note that by definition, equalities and dis-equalities between variables of sort index or elem are trivially satisfied. Next, consider an equality of the form $x = \mathsf{read}(a,i)$. Since this equality is in Γ, we know by the definition of $a^{\mathcal{B}}$ that $a^{\mathcal{B}}(i^{\mathcal{B}}) = x^{\mathcal{B}}$, so by the definition of read, such equalities must be satisfied. This shows that for terms t of sort index or elem, $t^{\mathcal{B}} = t^{\mathcal{A}}$ and thus if s is a term of the same sort as t, $s \approx_a^{\Gamma} t$ iff $s^{\mathcal{B}} = t^{\mathcal{B}}$. Similarly, it is not hard to see that since $\alpha(\Gamma)$ is satisfiable, we must have that if $s \neq_a^{\Gamma} t$ then $s^{\mathcal{B}} \neq t^{\mathcal{B}}$.

Next, consider equalities and dis-equalities between array-variables. For every dis-equality $a \neq b \in \Gamma$, we know that (because Γ is saturated), $\mathsf{read}(a, k_{a,b}) \neq_a^{\Gamma} \mathsf{read}(b, k_{a,b})$, and thus $\mathsf{read}(a, k_{a,b})^{\mathcal{B}} \neq \mathsf{read}(b, k_{a,b})^{\mathcal{B}}$, from which it is clear that $a^{\mathcal{B}} \neq b^{\mathcal{B}}$. To see that equalities $a = b$ are satisfied, note that we have $a \approx_a^{\Gamma} b$, and thus the definitions of $a^{\mathcal{B}}$ and $b^{\mathcal{B}}$ will yield the same function.

Finally, consider an equality of the form $a = \mathsf{write}(b,i,v)$. Let $f_a = a^{\mathcal{B}}$ and $f_{\mathsf{write}} = \mathsf{write}(b,i,v)^{\mathcal{B}}$. We will show that for all index-elements ι, $f_a(\iota) = f_{\mathsf{write}}(\iota)$. First, suppose that $\iota = i^{\mathcal{B}}$. In this case, it is clear that $f_{\mathsf{write}}(\iota) = v^{\mathcal{B}}$ by the definition of $\mathsf{write}^{\mathcal{B}}$. Also, by the $\mathsf{RIntro1}$ rule (and saturation of Γ), we know that $v \approx_a^{\Gamma} \mathsf{read}(a,i)$ and so $\mathsf{read}(a,i)^{\mathcal{B}} = v^{\mathcal{B}}$. Then, by the definition of $a^{\mathcal{B}}$, we must have $f_a(\iota) = v^{\mathcal{B}}$.

Suppose, on the other hand that $\iota \neq i^{\mathcal{B}}$. Note that by the definition of $\mathsf{write}^{\mathcal{B}}$, this implies that $f_{\mathsf{write}}(\iota) = b^{\mathcal{B}}(\iota)$. Suppose now that we have $x = \mathsf{read}(c,j) \in \Gamma$ with $a \approx_a^{\Gamma} c$ and $j^{\mathcal{B}} = \iota$. In this case, the definition of $a^{\mathcal{B}}$ ensures that $f_a(\iota) = \mathsf{read}(c,j)^{\mathcal{B}}$. Looking at rule $\mathsf{RIntro2}$, we can see that because Γ is saturated and $i^{\mathcal{B}} \neq j^{\mathcal{B}}$, we must have $\mathsf{read}(a,j)^{\mathcal{B}} = \mathsf{read}(b,j)^{\mathcal{B}}$. But the first is equal to $\mathsf{read}(c,j)^{\mathcal{B}}$ by saturation and rule $\mathsf{RIntro1}$, and the second is equal to $b^{\mathcal{B}}(\iota)$ by the definition of $\mathsf{read}^{\mathcal{B}}$. Thus, $f_{\mathsf{write}}(\iota) = f_a(\iota)$. A similar case is when $x = \mathsf{read}(c,j) \in \Gamma$ with $b \approx_a^{\Gamma} c$ and $j^{\mathcal{B}} = \iota$. Here, we have $f_{\mathsf{write}}(\iota) = \mathsf{read}(c,j)^{\mathcal{B}}$ by definition, and we can again conclude that $\mathsf{read}(a,j)^{\mathcal{B}} = \mathsf{read}(b,j)^{\mathcal{B}}$ by rule $\mathsf{RIntro2}$. But the first is equal to $f_a(\iota)$ by the definition of read and the second is equal to $\mathsf{read}(c,j)^{\mathcal{B}}$ and thus to $f_{\mathsf{write}}(\iota)$. In the final case, when neither of the previous cases hold, the definitions of $a^{\mathcal{B}}$ and $b^{\mathcal{B}}$ ensure that $f_a(\iota) = a^{\mathcal{B}}(\iota) = e_0 = b^{\mathcal{B}}(\iota) = f_{\mathsf{write}}(\iota)$.

Since $\mathcal{B}$ satisfies the axioms and each of the literals in Γ, this shows that Γ is $T_{\mathtt{arr}}$-satisfiable.

5.2 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:

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

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

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 $\mathcal{D}'_{\mathtt{arr}}$ to Γ until no more rules apply (for the sake of determinism, assume that rules are applied in order of appearance when there is a choice). We say that Γ' is $\mathcal{D}'_{\mathtt{arr}}$-saturated. We define the equality propagator as:

$$\begin{aligned} \mathfrak{P}_{\mathtt{arr}}[\![V]\!](\Gamma) = {} & \{\ w = z \mid w, z \in V,\ (w,z) \in \mathcal{E}_a(\Gamma')\ \} \\ & \cup \{\ w \neq z \mid w, z \in V,\ (w,z) \in \mathcal{N}_a(\Gamma')\ \}\ . \end{aligned}$$

It is easy to see that $\mathfrak{P}_{\mathtt{arr}}[\![\cdot]\!]$ 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.

5.3 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_a^{ϕ} of pairs of array variables in V that are not yet known equal or dis-equal. Let $V_a = \mathit{vars}_{\mathsf{array}}(V)$. Then,

$$E_a^{\phi} = (V_a \times V_a) \setminus (\mathcal{E}_a(\phi) \cup \mathcal{N}_a(\phi))\ .$$

Next, since the inference rules can introduce new read terms, we compute the smallest set R^{ϕ} that includes all 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 $a = \mathsf{write}(b,i,v) \in \phi$, $\mathsf{read}(c,j) \in \mathsf{R}^{\phi}$, $i \not\approx_a^{\phi} j$, and $a \approx_a^{\phi} c$ or $b \approx_a^{\phi} c$, 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_i^{ϕ} as the set of all pairs (i,j) such that:

- $i \not\approx_a^{\phi} j$ and not $i \neq_a^{\phi} j$
- $\exists\, a, b, c, v.\ a = \mathsf{write}(b,i,v) \in \phi, \mathsf{read}(c,j) \in \mathsf{R}^{\phi},$ and $a \approx_a^{\phi} c$ or $b \approx_a^{\phi} c$.

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

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

$$E = \begin{cases} E_a^\phi & \text{if } E_a^\phi \neq \emptyset, \\ E_i^\phi & \text{if } E_i^\phi \neq \emptyset, \text{ and} \\ E_r^\phi & \text{otherwise.} \end{cases}$$

Note that as defined, E_i^ϕ may include pairs of index variables, one or more of which are not in V. Unfortunately, the care function fails if E_i^ϕ 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_i^ϕ or doing additional case-splitting up front on pairs in E_i^ϕ, adding formulas to ϕ, until $E_i^\phi \subseteq V \times V$.

Example 6 Consider the following constraints involving arrays and bit-vectors of size m, where $\times_m$ denotes unsigned bit-vector multiplication:

$$\bigwedge_{k=1}^{n} \left(\mathsf{read}(a_k, i_k) = \mathsf{read}(a_{k+1}, i_{k+1}) \wedge i_k = x_k \times_m x_{k+1}\right) \quad . \tag{6}$$

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. we use the trivial care function over these variables). Fortunately, in practice it appears that index and element variables are typically shared, and only rarely are array variables shared.

Lemma 2 *Let ϕ be a set of flat T_{arr}-literals, and suppose that $\mathfrak{C}_{\mathsf{arr}}[\![V]\!](\phi) = \langle V, \emptyset\rangle$. If Γ is a satisfiable set of literals obtained from ϕ via a sequence of $\mathcal{D}_{\mathsf{arr}}$-inferences, and $\mathsf{read}(a, i)$ appears in Γ, then $\mathsf{read}(a, i) \in \mathsf{R}^\phi$.*

Proof The proof is by induction on inference rule applications. For the base case, suppose $\Gamma = \phi$. The first rule defining R^ϕ ensures that $\mathsf{read}(a, i) \in \mathsf{R}^\phi$.

For the inductive case, suppose every term $\mathsf{read}(a, i)$ apearing in Γ is in R^ϕ and let Γ' be obtained by applying an inference rule to Γ. Suppose the inference rule is RIntro1. This introduces a term of the form $\mathsf{read}(a, i)$. It also requires that we have an equality $a = \mathsf{write}(b, i, v) \in \Gamma$. But no rule introduces such equalities, so it must have been in ϕ originally. Again, the first rule defining R^ϕ then ensures that $\mathsf{read}(a, i) \in \mathsf{R}^\phi$.

Next, suppose the inference rule is RIntro2. The right branch of this rule may introduce $\mathsf{read}(a, j)$ and $\mathsf{read}(b, j)$. In this case, we know there are equalities $a = \mathsf{write}(b, i, v)$ and $x = \mathsf{read}(c, j)$ in Γ with $i \not\approx_a^\Gamma j$, and either $a \approx_a^\Gamma c$ or $b \approx_a^\Gamma c$. As before, we must have $a = \mathsf{write}(b, i, v) \in \phi$, and by the inductive hypothesis, we know that $\mathsf{read}(c, j) \in \mathsf{R}^\phi$. Furthermore, because $E_a^\phi = \emptyset$, we know that all relationships between array variables are already determined by ϕ, so either $a \approx_a^\phi c$ or $b \approx_a^\phi c$; and we know from Lemma 1 that $i \not\approx_a^\phi j$. We can then see that the second rule defining R^ϕ ensures that $\mathsf{read}(a, j)$ and $\mathsf{read}(b, j)$ are in R^ϕ.

Finally, suppose that the inference rule is ArrDiseq. This rule may introduce $\mathsf{read}(a, k_{a,b})$ and $\mathsf{read}(b, k_{a,b})$. This can only happen if $a \neq b \in \mathit{\Gamma}$. Since no rules introduce dis-equalities between array variables, this implies that $a \neq b \in \phi$, and so the last rule defining R^{ϕ} ensures that $\mathsf{read}(a, k_{a,b}) \in \mathsf{R}^{\phi}$ and $\mathsf{read}(b, k_{a,b}) \in \mathsf{R}^{\phi}$.

Theorem 6 *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 $\mathfrak{P}_{\mathtt{arr}}[\![\cdot]\!]$ for all sets ϕ of literals and V of variables such that $E_i^{\phi} \subseteq V \times V$.*

Proof Assume that we are given a set ϕ of flat $\mathit{\Sigma}_{\mathtt{arr}}$-literals and a set V of variables with $E_i^{\phi} \subseteq V \times V$. Let $\mathfrak{C}_{\mathtt{arr}}[\![V]\!](\phi) = \langle V, \emptyset\rangle$, and assume that ϕ is $T_{\mathtt{arr}}$-satisfiable and δ_V is a variable arrangement such that $\delta_V \supseteq \mathfrak{P}_{\mathtt{arr}}[\![V]\!](\phi)$. Because ϕ is $T_{\mathtt{arr}}$-satisfiable, Theorem 5 ensures we can find a set $\mathit{\Gamma} \supseteq \phi$ such that:

- $\mathit{\Gamma}$ is derivable from ϕ using the rules of $\mathcal{D}_{\mathtt{arr}}$ (applying rules in order of appearance if there is a choice).
- $\mathit{\Gamma}$ is $\mathcal{D}_{\mathtt{arr}}$-saturated,
- $\alpha(\mathit{\Gamma})$ is $T_{\mathtt{euf}}$-satisfiable.

Again, for notational convenience, we will write $s \sim_c t$ and $s \not=_c t$ for $(s,t) \in \mathcal{E}_c(\alpha(\mathit{\Gamma}))$ and $(s,t) \in \mathcal{N}_c(\alpha(\mathit{\Gamma}))$, respectively. Furthermore, notice that by definition we have $s \sim_c t$ iff $s \approx_a^{\mathit{\Gamma}} t$ and $s \not=_c t$ iff $s \not=_a^{\mathit{\Gamma}} t$.

We claim that $\delta_V \supseteq \mathfrak{P}_{\mathsf{euf}}[\![V]\!](\alpha(\mathit{\Gamma}))$. We know that $\delta_V \supseteq \mathfrak{P}_{\mathtt{arr}}[\![V]\!](\phi)$, so it suffices to show that $\mathfrak{P}_{\mathsf{euf}}[\![V]\!](\alpha(\mathit{\Gamma})) = \mathfrak{P}_{\mathtt{arr}}[\![V]\!](\phi)$. Notice that, if $\mathit{\Gamma}'$ is the $\mathcal{D}'_{\mathtt{arr}}$-saturated set obtained starting from ϕ, then by matching up the definitions, it is clear that $\mathfrak{P}_{\mathtt{arr}}[\![V]\!](\phi) = \mathfrak{P}_{\mathsf{euf}}[\![V]\!](\alpha(\mathit{\Gamma}'))$. Thus, it suffices to show that $\mathfrak{P}_{\mathsf{euf}}[\![V]\!](\alpha(\mathit{\Gamma})) = \mathfrak{P}_{\mathsf{euf}}[\![V]\!](\alpha(\mathit{\Gamma}'))$. In fact, we can show that $\mathit{\Gamma} = \mathit{\Gamma}'$. Suppose not. The only way this could happen is if there is some $\mathcal{D}_{\mathtt{arr}}$-derivation starting from ϕ in which rule RIntro2 applies but rules RIntro2a and RIntro2b do not. Let $\mathit{\Gamma}''$ be the first set in the $\mathcal{D}_{\mathtt{arr}}$-derivation from ϕ to $\mathit{\Gamma}$ in which this is the case. In order for the rule to be enabled, we must have $a = \mathsf{write}(b, i, v), x = \mathsf{read}(c, j) \in \mathit{\Gamma}''$. As we have noted before, derivations do not introduce equalities containing applications of write, so we must have $a = \mathsf{write}(b, i, v) \in \phi$. We also know by Lemma 2 that $\mathsf{read}(c, j) \in \mathsf{R}^{\phi}$. We also have $a \approx_a^{\mathit{\Gamma}''} c$ or $b \approx_a^{\mathit{\Gamma}''} c$, so it follows from the fact that $E_a^{\phi} = \emptyset$ that $a \approx_a^{\phi} c$ or $b \approx_a^{\phi} c$. But now, since $E_i^{\phi} = \emptyset$, clearly we must have $i \approx_a^{\phi} j$ or $i \not=_a^{\phi} j$. In the first case, we know that $i \approx_a^{\mathit{\Gamma}''} j$, so rule RIntro2 is not applicable, contradicting our assumption. In the second case, we know that $i \not=_a^{\mathit{\Gamma}''} j$ which means that RIntro2b *is* applicable, which also contradicts our assumption.

Now, let $\mathbf{G} = \langle V, E\rangle = \mathfrak{C}_{\mathsf{euf}}[\![V]\!](\alpha(\mathit{\Gamma}))$ be the T_{euf} care graph based on $\alpha(\mathit{\Gamma})$ (with read treated as an uninterpreted function). We claim that $E = \emptyset$. First note that because $E_a^{\phi} = \emptyset$, we know that for variables $a, b \in \mathit{vars}_{\mathsf{array}}(V)$, either $a \approx_a^{\phi} b$ or $a \not=_a^{\phi} b$ (and thus $a \sim_c b$ or $a \not=_c b$), so it is impossible to have $(a, b) \in E$. Next, notice that since variables of sort elem cannot appear as arguments to functions in $\alpha(\mathit{\Gamma})$, there are no pairs $(x, y) \in E$. Finally, suppose we have a pair of variables (i, j) of sort index such that $(i, j) \in E$. By the definition of $\mathfrak{C}_{\mathsf{euf}}[\![\cdot]\!]$, we know that there exist a, b, i', j' such that

1. $\mathsf{read}(a, i')$ and $\mathsf{read}(b, j')$ appear in $\alpha(\mathit{\Gamma})$,
2. $\mathsf{read}(a, i') \not\approx_a^{\mathit{\Gamma}} \mathsf{read}(b, j')$,
3. not $a \not=_a^{\mathit{\Gamma}} b$, and

4. $i \approx_a^{\Gamma} i'$, and $j \approx_a^{\Gamma} j'$,

We can immediately conclude from property 1 that $\mathsf{read}(a, i') \in \mathsf{R}^{\phi}$ and $\mathsf{read}(b, j') \in \mathsf{R}^{\phi}$ by Lemma 2. Also, property 2 implies that $\mathsf{read}(a, i') \not\approx_a^{\phi} \mathsf{read}(b, j')$ by Lemma 1.

We next consider the implications of property 4. Notice that the only equalities between index variables that could have been introduced during the derivation from ϕ to Γ are those introduced by rule RIntro2. But if this rule is enabled and could introduce $i = j$, then $(i, j) \in E_i^{\phi}$. But we know that $E_i^{\phi} = \emptyset$. It follows that no equalities between index variables are introduced in the derivation. So, if $i \approx_a^{\Gamma} i'$, then $i \approx_a^{\phi} i'$. Similarly, if $j \approx_a^{\Gamma} j'$, then $j \approx_a^{\phi} j'$. But now notice that if we can establish $a \approx_a^{\phi} b$ (see below), it will follow from the definition of E_r^{ϕ} (and the fact that $E_r^{\phi} = \emptyset$) that either $i \approx_a^{\phi} j$ (and thus $i \sim_c j$) or $i \neq_a^{\phi} j$ (and thus $i \neq_c j$). This contradicts our assumption that $(i, j) \in E$. It remains to show $a \approx_a^{\phi} b$. By property 3, we know that it is not the case that $a \neq_a^{\Gamma} b$, and thus by Lemma 1, it is not the case that $a \neq_a^{\phi} b$. But since $E_a^{\phi} = \emptyset$, we must then have $a \approx_a^{\phi} b$.

We have thus established that $E = \emptyset$. Now, because $\delta_V \supseteq \mathfrak{P}_{\mathtt{euf}}[\![V]\!](\alpha(\Gamma))$, by Theorem 3, $\alpha(\Gamma) \cup \delta_V$ must be $T_{\mathtt{euf}}$-satisfiable. But $\alpha(\Gamma) \cup \delta_V = \alpha(\Gamma \cup \delta_V)$, so $\alpha(\Gamma \cup \delta_V)$ is $T_{\mathtt{euf}}$-satisfiable. Finally, since Γ is $\mathcal{D}_{\mathtt{arr}}$-saturated, and δ_V can only add new equalities and disequalities between variables of sort index or elem, it is clear that $\Gamma \cup \delta_V$ must also be $\mathcal{D}_{\mathtt{arr}}$-saturated, so by Theorem 5, $\Gamma \cup \delta_V$ is $T_{\mathtt{arr}}$-satisfiable, from which we can conclude that $\phi \cup \delta_V$ is $T_{\mathtt{arr}}$-satisfiable.

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 significant 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 6 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 of Burstall,

Table 1 Experimental results of the evaluation. For each solver, the first column reports the total time (in seconds) used by that solver on the problem instances it solved. The second column reports the number of instances solved. The best results for each benchmark are in bold.

	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

e.g. [17]). All the benchmarks and the Cvc3 binaries used in the experiments are available from the authors' website.[4]

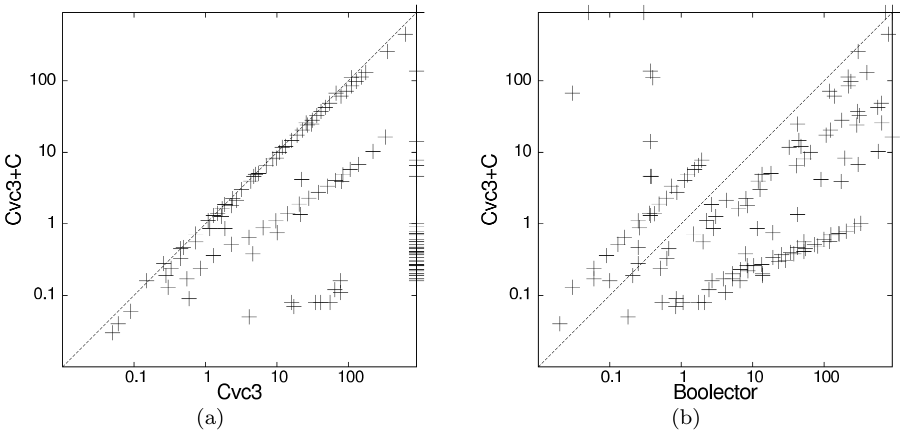
The combined results of our experiments are presented in Table 1, with columns reporting the total time (in seconds) that a solver used on the problem instances it solved (not including time spent on problem instances it was unable to solve), and the number of solved instances. Compared to Cvc3, the 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 that are relevant

[4] http://cs.nyu.edu/~dejan/sharing-is-caring/



(a) (b)

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.

in a specific 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 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. Clark Barrett, Robert Nieuwenhuis, Albert Oliveras, and Cesare 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. Clark Barrett and Cesare Tinelli. CVC3. In *Computer Aided Verification*, volume 4590 of *LNCS*, 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*, volume 5505 of *LNCS*, 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*, volume 5123 of *LNCS*, 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 *5th International Workshop on Satisfiability Modulo Theories*, 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*, volume 4963 of *LNCS*, page 337. Springer, 2008.
9. Leonardo de Moura and Nikolaj Bjørner. Generalized, efficient array decision procedures. In *Formal Methods in Computer-Aided Design, 2009*, pages 45–52. IEEE, November 2009.
10. Bruno Dutertre and Leonardo de Moura. The YICES SMT Solver. *Tool paper at* `http://yices.csl.sri.com/tool-paper.pdf`, 2006.
11. Herbert B. Enderton. *A mathematical introduction to logic.* Academic press New York, 1972.
12. Dejan Jovanović and Clark Barrett. Polite theories revisited. Technical Report TR2010-922, Department of Computer Science, New York University, January 2010.
13. Dejan Jovanović and Clark Barrett. Polite theories revisited. In *Logic for Programming, Artificial Intelligence, and Reasoning*, volume 6397 of *LNCS*, pages 402–416. Springer Berlin / Heidelberg, 2010.
14. Dejan Jovanović and Clark Barrett. Sharing is caring: combination of theories. *Frontiers of Combining Systems*, pages 195–210, 2011.
15. Greg Nelson and Derek C. Oppen. Simplification by cooperating decision procedures. *ACM Transactions on Programming Languages and Systems*, 1(2):245–257, October 1979.
16. Derek C. Oppen. Complexity, convexity and combinations of theories. *Theoretical Computer Science*, 12(3):291–302, 1980.
17. Zvonimir Rakamarić and Alan J. Hu. A Scalable Memory Model for Low-Level Code. In *Verification, Model Checking, and Abstract Interpretation*, volume 5403 of *LNCS*, page 304. Springer-Verlag, 2009.
18. Silvio Ranise, Christophe Ringeissen, and Calogero 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. Robert 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. Cesare Tinelli and Mehdi 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, March 1996.
21. Cesare Tinelli and Calogero Zarba. Combining decision procedures for sorted theories. In *Logic in Artificial Intelligence*, volume 3229 of *LNAI*, pages 641–653. Springer, 2004.