Satisfiability Modulo Free Data Structures Combined with Bridging Functions

Raphaël Berthon, Christophe Ringeissen

Satisfiability Modulo Free Data Structures Combined with Bridging Functions

(Extended Abstract)

Raphaël Berthon[1*] and Christophe Ringeissen[2]

[1] ENS Rennes, France
[2] Inria Nancy - Grand Est and LORIA, France

Abstract

Free Data Structures are finite semantic trees modulo equational axioms that are useful to represent classical data structures such as lists, multisets and sets. We study the satisfiability problem when free data structures are combined with bridging functions. We discuss the possibility to get a combination method à la Nelson-Oppen for these particular non-disjoint unions of theories. In order to handle satisfiability problems with disequalities, we investigate a form of sufficient surjectivity for the bridging functions.

1 Introduction

Solving Satisfiability Modulo Theory (SMT) problems consists in deciding the satisfiability of first order formulas with respect to a given theory. Satisfiability procedures exist for various theories, for instance the equality theory and Presburger arithmetic, but also for many data structures like arrays, lists, multisets, or sets, some of them being successfully handled by general proof techniques [1,2]. To increase the number of theories where the SMT problem can be solved, a natural approach is to proceed in a modular way for the union of theories when a decision procedure is known for each component theory. The Nelson-Oppen method [16] gives a way to solve this problem while combining two theories, but this method requires the component theories to be both stably-infinite (if a formula has a model, it has an infinite model — this ensures that we can find a model with a cardinality suitable for both theories), and to be disjoint (the theories only share the equality symbol). The case where one theory is not stably infinite has led to the study of theories with good properties with respect to the cardinality of the model, with the case of shiny [21], polite [13, 18], or gentle [11] theories. Another challenging problem is to investigate the modular construction of decision procedures for non-disjoint unions of theories [3, 12]. For instance, an interesting case appears when two disjoint theories are connected via a third bridging theory. Bridging functions [8,17,19,20] are good examples of this form of non-disjoint combination.

As SMT solvers are mostly used in (program) verification, they often need to solve problems involving data structure theories (list, sets, multisets, arrays, ...) [6,14]. These data structures may interact with each other, and their elements may be used in computations; the resulting theory being a non-disjoint union. Functions used to represent the transformation of a data structure into another data structure (like calculating the set of elements in a list) are called bridging functions in [19] and abstraction functions in [20]. The task is then to design a decision procedure for these data structures connected by a bridging function.

*This work has been partly done during an internship at Inria Nancy - Grand Est.

The theory of Absolutely Free Data Structures (AFDS) corresponds to the theory of finite syntactic trees axiomatized by Injectivity, Acyclicity, and Disjoint Constructors. The combination with a bridging function can be handled using a locality approach [19] based on the some finite instantiations of the axioms. More recently, a Nelson-Oppen combination method has been proposed for this particular non-disjoint union of theories [8].

Following the work on AFDS [8], a challenging problem is to consider semantic trees instead of syntactic ones. The use of semantic trees is clearly of practical interest, since it allows us to represent more data structures such as multisets or sets. Some contributions in that direction are already reported in [19]. Moreover, combination methods are already known for some individual data structures connected to the integers via a bridging function such as the cardinality for multisets [23] and sets [24]. More generally, we are interested in lifting the combination method given in [8] to handle semantic trees in a uniform way.

Our approach is based on the reduction of the satisfiability in combined theories to the satisfiability in individual theories. Our procedure extends the one used for AFDS [8]. We study Free Data Structures (FDS), defined as trees modulo a set of equational axioms, connected by a bridging function. Our method can only be used for some specific formulas, whose equalities are solved. Given the possibility of having a measure function (notion defined later) to order the (semantic) trees, we will separate our variables in two. The lesser ones for our order will have their value guessed. For the other variables, the specific chosen solved forms and the measure function will ensure the existence of a combined model.

The difficulty of the approach is related to the surjectivity of the bridging function. A form of sufficient surjectivity [20] is needed to get a completeness result in presence of disequalities. We investigate how the surjectivity assumption used for AFDS also applies to FDS. In [8] the focus was on the combination where the source is (the standard interpretation of) AFDS and the target is the theory of integers. In this paper, the aim is to go beyond bridging functions targeting the integers, by considering free data structures as source and target. For instance, the presented approach enables us to consider a bridging function computing the set of elements in a multiset.

This work is the opportunity to reuse the constraint solving techniques known for tree algebras, e.g., unification and matching procedures to solve equalities. In our approach, disunification would be needed [4, 10, 15]. The presence of disequalities is really challenging. Indeed, disunification in initial tree algebras can be undecidable whilst a unification algorithm is known, e.g., for the Associativity-Commutativity [22]. Even if it is impossible to get a general solution, we believe it is interesting to identify decidable classes of equational formulas. In this work, we discuss how to handle some particular disequalities in the case of sufficient surjectivity and when assuming a solver to compute solved forms for the equalities.

2 Preliminaries

We assume the reader has some basic familiarities with equational theories and equational unification [5]. Given a first-order signature Σ and a (countable) set of variables V, the set of Σ-terms over variables V is denoted by $T(\Sigma, V)$. The set of variables in a term t is denoted by $Var(t)$. A term t is *ground* if $Var(t) = \emptyset$. A substitution is an endomorphism of $T(\Sigma, V)$, with only finitely many variables not mapped to themselves. A substitution is denoted by $\sigma = \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_m\}$, its domain is $Dom(\sigma) = \{x_1, \ldots, x_n\}$ and its range is $Ran(\sigma) = \{t_1, \ldots, t_n\}$.

Given a set E of Σ-axioms (i.e., pairs of Σ-terms, denoted by $l = r$), the *equational theory* $=_E$ is the congruence closure of E under the law of substitutivity. By a slight abuse of terminology, E will be often called an equational theory. An axiom $l = r$ is *regular* if $Var(l) = Var(r)$.

A Σ-equality is a pair of Σ-terms denoted by $s = t$. An E-unification problem is a set of Σ-equations, $\phi = \{s_1 = t_1, \ldots, s_m = t_m\}$. The set of variables of ϕ is denoted by $Var(\phi)$. When $t_1, \ldots, t_m$ are ground, ϕ is called a matching problem. A solution to an E-unification problem ϕ, called an *E-unifier*, is a substitution σ such that $s_i\sigma =_E t_i\sigma$ for all $1 \leq i \leq m$. A decision procedure for E-unification is a decision procedure returning *true* if and only if the input E-unification problem admits an E-unifier. E-unification is said to be finitary if any E-unification problem admits a (computable) finite complete set of E-unifiers.

A flat equality is either of the form $t_0 = t_1$ or $t_0 = f(t_1, \ldots, t_n)$ where each term $t_0, \ldots, t_n$ is a variable or a constant. A disequality $t_0 \neq t_1$ is flat when each term t_0, t_1 is a variable or a constant. A flat literal is either a flat equality or a flat disequality. For n distinct variables $x_1, \ldots, x_n$, the set of literals $\{x_i \neq x_j \mid \ i \neq j,\ i, j = 1, \ldots, n\}$ is denoted by $\{x_1 \neq \cdots \neq x_n\}$.

A set of equalities $\phi = \{x_k = t_k\}_{k \in K}$ is a *solved form* if x_k is a variable occurring only once in ϕ for each $k \in K$. Note that a solved form corresponds to a most general E-unifier: it is not necessarily flat, but using flattening, we can obtain an equivalent flat dag solved form. For instance, the flattening of $\{v = c(u(e), y)\}$ is the dag solved form $\{v = c(x, y), x = u(e)\}$. Roughly speaking, a dag solved form is a set of (acyclic) solved equalities leading to a solved form by variable replacement.

3 Free Data Structures

In this paper, we focus on data structure theories constructed from three operators:

- A binary operator c, which is not necessarily absolutely free: it may satisfy some equational properties;
- A unary operator u to construct a singleton structure;
- A constant nil to denote the empty structure.

Definition 1. A *binary constructor-based* signature is of the form

$$\{c : \texttt{struct} \times \texttt{struct} \to \texttt{struct}, nil : \texttt{struct}, u : \texttt{elem} \to \texttt{struct}\}$$

To simplify the notation, we often write $c(e, x)$ instead of $c(u(e), x)$, by assuming that $e, e_1, \ldots, e_n$ denote only `elem`-sorted variables. We may use an infix notation to write $e \cup x$ instead of $\cup(e, x)$.

Definition 2. Let Σ be a binary constructor-based signature. Given an equational Σ-theory E, FDS_E denotes the class of Σ-structures $\mathcal{A}$ such that $\texttt{struct}_{\mathcal{A}} = T(\Sigma \cup \texttt{elem}_{\mathcal{A}})/ =_E$.

Definition 3 (Standard term)**.** Let Σ be a binary constructor-based signature. A `struct`-sorted Σ-term is said to be *standard* if it contains no `struct`-sorted variables.

In this paper, we focus on regular theories whose axioms do not involve the unary operator u. We are interested in the following regular axioms: Associativity (A) $(X \cup Y) \cup Z = X \cup (Y \cup Z)$, Commutativity ($C$) $X \cup Y = Y \cup X$, Right Unit (RU) $X \cup \emptyset = X$, Left Unit (LU) $\emptyset \cup X = X$, and Idempotency (I) $X \cup X = X$. But the Nilpotency (N) $X \oplus X = 0$ is not regular and will not be considered. In the following, we consider any combination of the above regular axioms, such as

$$AU = A \cup LU \cup RU,\ AC = A \cup C,\ ACU = AC \cup LU,\ ACUI = ACU \cup I$$

Besides these regular theories, we also consider two special non-regular ones involving the unary operator u, related to positive integers and booleans.

Given a binary constructor-based signature $\Sigma = \{c : \texttt{struct} \times \texttt{struct} \to \texttt{struct}, nil : \texttt{struct}, u : \texttt{elem} \to \texttt{struct}\}$, a Σ-structure $\mathcal{A}$ is a *standard (positive integer)* $\mathbb{N}$*-interpretation* if $\texttt{struct}_\mathcal{A} = \mathbb{N}$, c is interpreted in $\mathcal{A}$ as the addition, $\mathcal{A}[nil] = 0$ and for any $a \in \texttt{elem}_\mathcal{A}$, $\mathcal{A}[u](a) = 1$. Analogously, a Σ-structure $\mathcal{A}$ is a *standard (boolean)* $\mathbb{B}$*-interpretation* if $\texttt{struct}_\mathcal{A} = \mathbb{B}$, c is interpreted in $\mathcal{A}$ as the disjunction, $\mathcal{A}[nil] = \bot$ and for any $a \in \texttt{elem}_\mathcal{A}$, $\mathcal{A}[u](a) = \top$. Standard $\mathbb{N}$-interpretations and standard $\mathbb{B}$-interpretations can be viewed as models of respectively two free data structures:

$$E_\mathbb{N} = AC(+) \cup \{\forall X.\, X + 0 = X\} \cup \{\forall V, W.\, 1(V) = 1(W)\}$$

$$E_\mathbb{B} = AC(\vee) \cup \{\forall X.\, X \vee \bot = X, X \vee X = X\} \cup \{\forall V, W.\, \top(V) = \top(W)\}$$

Proposition 1. *Any model of $FDS_{E_\mathbb{N}}$ (resp., $FDS_{E_\mathbb{B}}$) is isomorphic to a standard $\mathbb{N}$-interpretation (resp., $\mathbb{B}$-interpretation).*

Obviously, $FDS_{E_\mathbb{N}}$-satisfiability and $FDS_{E_\mathbb{B}}$-satisfiability are decidable.

For an arbitrary equational theory E, FDS_E-satisfiability and E-unification can be related as follows:

Proposition 2. *A set of equalities is satisfiable in FDS_E if and only if it is E-unifiable.*

Moreover, an E-unification algorithm computing most general E-unifiers is sufficient to check E-equality and so to check the satisfiability of disequalities, since an infinite interpretation domain for the `elem` sort can be chosen.

Proposition 3. *FDS_E-satisfiability is decidable if a finitary E-unification algorithm is known.*

4 Combination of Free Data Structures

In a way similar to [8], we consider two theories connected via a bridging theory defining a function f by structural induction over the constructors. In the context of this paper, the two theories are free data structures sharing only the `elem` sort.

Definition 4. ([8]) Let Σ be a binary constructor-based signature, Σ_t a signature such that Σ and Σ_t do not share function symbols. Let f be a new function symbol f with arity $\texttt{struct} \to \texttt{t}$, where $\texttt{t}$ is a sort in Σ_t. Given the signature $\Sigma_f = \Sigma \cup \Sigma_t \cup \{f : \texttt{struct} \to \texttt{t}\}$, a *bridging* Σ_f-theory T_f associated to f has the form:

$$T_f = \left\{ \begin{array}{lcl} f(c(X,Y)) & = & f_c(f(X), f(Y)) \\ f(u(V)) & = & f_u(V) \\ f(nil) & = & f_{nil} \end{array} \right.$$

where $f_c, , f_u, f_{nil}$ are Σ_t-terms of respective arities $2, 1, 0$.

The *term rewrite system F of a bridging theory T_f* is $F = \{f(l) \to r \mid f(l) = r \in T_f\}$.

The term rewrite system (for short, TRS) of a bridging theory T_f is convergent. Let Σ_1 and Σ_2 be two binary constructor-based signatures sharing only the `elem` sort. The non-shared sort in Σ_i is denoted by $\texttt{struct}_i$ for $i = 1, 2$. Let T_f be a bridging Σ_f-theory between signatures $\Sigma = \Sigma_1$ and $\Sigma_t = \Sigma_2$, where $\texttt{struct} = \texttt{struct}_1$ and $\texttt{t} = \texttt{struct}_2$. Assume an equational Σ_i-theory E_i for $i = 1, 2$. The bridging theory T_f is said to be E_1*-compatible in* E_2 if, for any $l = r \in E_1$ we have $f(l) \downarrow_F =_{E_2} f(r) \downarrow_F$.

Example 1. Consider $E_1 = \{(X \cup Y) \cup Z = X \cup (Y \cup Z), X \cup Y = Y \cup X, X \cup \emptyset = X\}$ and $E_2 = E_{\mathbb{N}}$. The theory E_1 corresponds to multisets of elements and the bridging function computing the number of elements in a multiset is given by the bridging theory

$$T_f = \{f(X \cup Y) = f(X) + f(Y), f(\{e\}) = 1, f(\emptyset) = 0\}.$$

T_f is E_1-compatible in E_2 since

$$\begin{array}{l}(f(X) + f(Y)) + f(Z) =_{E_2} f(X) + (f(Y) + f(Z)) \\ f(X) + f(Y) =_{E_2} f(Y) + f(X) \\ f(X) + 0 =_{E_2} f(X)\end{array}$$

Let $E_1' = E_1 \cup \{X \cup X = X\}$. We remark that T_f is not E_1'-compatible in E_2. However, given $E_2' = E_{\mathbb{B}}$, the bridging theory

$$T_g = \{g(X \cup Y) = g(X) \vee g(Y), g(\{e\}) = \top, g(\emptyset) = \bot\}$$

is E_1'-compatible in E_2'. Indeed, E_1' allows us to represent sets of elements. The bridging function g returns $\top$ if the input set is non-empty, and $\bot$ otherwise.

The notion of compatibility is required to get a consistent combined theory:

Definition 5. Assume T_f is E_1-compatible in E_2. $T_f[E_1, E_2]$ denotes the class of Σ_f-structures $\mathcal{A}$ such that $\mathcal{A}^{\Sigma_1} \in FDS_{E_1}$, $\mathcal{A}^{\Sigma_2} \in FDS_{E_2}$, and $\mathcal{A} \models T_f$.

From now on, we assume that $T = T_f[E_1, E_2]$, and $T_i = FDS_{E_i}$ for $i = 1, 2$. By this definition, we have $T = T_1 \cup T_f \cup T_2$.

5 Combination Procedure

The decision procedure works as a classical Nelson-Oppen combination procedure. As usual, the input set of literals is first purified to get a separate form, by introducing fresh variables. Then, we may need to perform some guessing over the fresh variables in order to guarantee the existence of a combined model. In the classical Nelson-Oppen combination procedure, the guessing phase only enumerates the arrangements over the shared variables. In our context, we may need an additional guessing to explore possible values taken by the bridging function. We want to focus on cases where only finitely many guesses are sufficient to get a complete procedure.

Definition 6. A set of literals φ is in *separate form* if $\varphi = \varphi_1 \cup \varphi_{elem} \cup \varphi_2 \cup \varphi_f$ where:

- φ_1 contains only flat $\texttt{struct}_1$-sorted literals
- φ_{elem} contains only `elem`-sorted literals
- φ_2 contains only $\texttt{struct}_2$-sorted literals
- φ_f contains only flat equalities of the form $f_x = f(x)$, where f_x denotes a variable associated to $f(x)$, such that f_x and $f(x)$ occur once in φ_f and each variable of sort $\texttt{struct}_1$ in φ_1 occurs in φ_f.

The sets of literals $\varphi_1 \cup \varphi_{elem}$ and $\varphi_2 \cup \varphi_{elem}$ are called respectively the 1-*component* and the 2-*component* of φ.

It is easy to convert any set of literals into an equisatisfiable separate form φ by introducing fresh variables to denote impure terms. In order to take into account the axioms of T_f, we need additional Σ_2-equalities obtained by encoding the flat equalities in φ_1.

Definition 7. Given a bridging theory T_f, the *target encoding* of a set of flat $\texttt{struct}_1$-sorted equalities φ is the set of $\texttt{struct}_2$-sorted equalities

$$\begin{array}{rcl} CP_\varphi & = & \{f_v = f_c(f_x, f_y) \mid v = c(x,y) \in \varphi\} \\ & & \cup \{f_v = f_u(e) \mid v = u(e) \in \varphi\} \\ & & \cup \{f_v = f_{nil} \mid v = nil \in \varphi\} \\ & & \cup \{f_v = f_x \mid v = x \in \varphi\} \end{array}$$

When a separate form φ contains only flat $\texttt{struct}_1$-equalities corresponding to a dag solved form, the target encoding of this dag solved form is sufficient to reduce T-satisfiability into T_2-satisfiability:

Proposition 4. *Let $\varphi = \varphi_1 \cup \varphi_{elem} \cup \varphi_2 \cup \varphi_f$ be a set of literals in separate form such that φ_1 is a dag solved form. The formula φ is T-satisfiable if and only if $\varphi_{elem} \cup \varphi_2 \cup CP_{\varphi_1}$ is T_2-satisfiable.*

The problem of considering $\texttt{struct}_1$-literals including both equalities and disequalities is much more complicated. We have investigated several possible scenarios.

5.1 Theory of Integers as Target

In a first case, we can assume that the target is the theory of integers, and that the source admits a unification algorithm to solve the equalities. Consider f defines the size of a data structure. We introduce a guessing of finitely many *range constraints* [8] to check the possible values of $f(x)$ for any x occurring in the problem:

- either $f(x) = 0$, and this implies $x = nil$;
- or $f(x) > 0$, and there will be enough distinct $\texttt{struct}_1$-entities to satisfy the $\texttt{struct}_1$-disequalities, by considering that each x is built only with its own distinct element e_x: this is possible since the `elem`-sort can be interpreted by an infinite domain.

This guessing phase leads to a T-equivalent disjunction of separate forms, each of them being T-satisfiable if and only if its 1-component and its 2-component are respectively T_1-satisfiable and T_2-satisfiable.

Proposition 5. *Consider E_1 is included in $AC(c) \cup \{c(X, nil) = X, c(nil, X) = X\}$, $E_2 = E_{\mathbb{N}}$ and $T_f = \{f(c(X,Y)) = f(X) + f(Y), f(u(e)) = 1, f(nil) = 0\}$. For any separate form $\varphi = \varphi_1 \cup \varphi_{elem} \cup \varphi_2 \cup \varphi_f$ such that φ_1 is a dag solved form together with flat disequalities, there exists a computable T-equivalent disjunction of separate forms $\bigvee_{k \in K} \varphi_k$ such that φ is T-satisfiable if and only if there exists some $k \in K$ such that the i-component of φ_k is T_i-satisfiable for each $i \in \{1, 2\}$.*

Example 2. Assume $E_1 = AC(\uplus) \cup LU(\emptyset)$. Consider the separate form

$$\varphi = \left\{ \begin{array}{l} z = x \uplus y, x = e \uplus v, y = e \uplus w, x \neq y \\ f_z = 2, f_z = f_x + f_y, f_x = 1 + f_v, f_y = 1 + f_w \\ f_x = f(x), f_y = f(y), f_z = f(z), f_v = f(v), f_w = f(w) \end{array} \right\}$$

We can check that φ is T-unsatisfiable by considering the following cases: (1) $f_v > 0, f_w > 0$, (2) $f_v = 0, f_w > 0$, (3) $f_v > 0, f_w = 0$, (4) $f_v = 0, f_w = 0$. For the cases (1), (2) and (3), we get the unsatisfiability in T_2. The case (4) $f_v = 0, f_w = 0$ is T_2-satisfiable. Adding $v = \emptyset$ and $w = \emptyset$ to φ_1 leads to $x = \{e\}$ and $y = \{e\}$, which contradicts $x \neq y$.

5.2 Arbitrary Free Data Structures as Target

In the general case, we need a form of sufficient/infinite surjectivity in order to satisfy problems of the form $x_1 \neq \cdots \neq x_n, \quad f(x_1) = \cdots = f(x_n)$. We have studied in [8] the particular case where the source theory is AFDS and the target is the theory of integers. In this paper, we investigate a generalization for the case where the source and the target are both free data structures. To this aim, we introduce a notion of *measure* to order the standard terms of a free data structure.

Definition 8 (Measure)**.** Let Σ be a binary constructor-based signature and E an equational Σ-theory. A *measure* for E is a mapping τ from standard Σ-terms to $\mathbb{N}$ such that

- for any standard Σ-terms s and t, $s =_E t$ implies $\tau(s) = \tau(t)$;
- for any standard Σ-term s and any substitution σ, $\tau(s\sigma) = \tau(s)$;
- for any $k \in \mathbb{N}$ and for $\bowtie$ in $\{=, \geq\}$, the constraint $\tau(x) \bowtie k$ can be equivalently expressed as a constraint in FDS_E of the form $(\exists \bar{v} \;.\; x = t \wedge \varphi)$, where $\bar{v} = Var(t)$, $x \notin \bar{v}$, φ is a conjunction of disequalities between `elem`-variables in $\bar{v}$, and t is standard if $\bowtie$ is $=$.

Example 3. There exists a very natural measure for the following equational theories:

- If $E = A(\cdot) \cup \{X \cdot nil = X, nil \cdot X = X\}$, then τ can be chosen as the number of `elem`-occurrences of a list. The constraints $\tau(x) = k$ and $\tau(x) \geq k$ can be respectively encoded into

$$\exists e_1, \ldots, e_k \;.\; x = u(e_1) \cdots u(e_k) \quad \text{and} \quad \exists y, e_1, \ldots, e_k \;.\; x = u(e_1) \cdots u(e_k) \cdot y$$

- If $E = AC(\uplus) \cup \{X \uplus \emptyset = X\}$, then τ can be chosen as the number of `elem`-occurrences of a multiset. The constraints $\tau(x) = k$ and $\tau(x) \geq k$ can be respectively encoded into

$$\exists e_1, \ldots, e_k \;.\; x = u(e_1) \uplus \cdots \uplus u(e_k) \quad \text{and} \quad \exists y, e_1, \ldots, e_k \;.\; x = u(e_1) \uplus \cdots \uplus u(e_k) \uplus y$$

- If $E = AC(\cup) \cup \{X \cup X = X, X \cup \emptyset = X\}$, then τ can be chosen as the cardinality of a set. The constraints $\tau(x) = k$ and $\tau(x) \geq k$ can be respectively encoded into

$$\exists e_1, \ldots, e_k \;.\; x = u(e_1) \cup \cdots \cup u(e_k) \;\wedge\; e_1 \neq \cdots \neq e_k$$

and

$$\exists y, e_1, \ldots, e_k \;.\; x = u(e_1) \cup \cdots \cup u(e_k) \cup y \;\wedge\; e_1 \neq \cdots \neq e_k$$

- If $E = E_{\mathbb{N}}$, then τ can be chosen as the identity on $\mathbb{N}$. The constraints $\tau(x) = k$ and $\tau(x) \geq k$ can be respectively encoded into $x = k$ and $\exists y \;.\; x = k + y$.

The definition of *gently growing* is introduced in [8] as a way to express a form of sufficient surjectivity. We adapt it to integrate the measure concept defined above. This allows us to consider a target theory which is not necessarily the theory of integers.

Definition 9. Let T be a theory defined as a class of structures in $T_f[E_1, E_2]$ and τ a measure for E_2. For any $\mathcal{A} \in T$ and any standard Σ_2-term s, $F_{\mathcal{A}}^{-1}(s) = \{t \mid \mathcal{A}[f](t) = \mathcal{A}[s]\}$. The bridging function f is *gently growing in T w.r.t τ* if

1. for any $\mathcal{A} \in T$ and any standard Σ_2-term s, $F_{\mathcal{A}}^{-1}(s) \neq \emptyset$;
2. for any $\mathcal{A} \in T$ and any standard Σ_2-terms s and t, $\tau(s) < \tau(t)$ implies $|F_{\mathcal{A}}^{-1}(s)| < |F_{\mathcal{A}}^{-1}(t)|$;
3. for any natural $n > 1$ there exists a computable k such that, for any standard Σ_2-term s,
   - if $\tau(s) \geq k$, then for any $\mathcal{A} \in T$, $|F_{\mathcal{A}}^{-1}(s)| \geq n$;
   - if $\tau(s) < k$, then one can compute a finite non-empty set $F^{-1}(s)$ of standard Σ_1-terms such that

$$T \models f(x) = s \Longleftrightarrow (\exists \bar{v} \,.\, \bigvee_{t \in F^{-1}(s)} x = t) \quad \text{where } \bar{v} = Var(F^{-1}(s))$$

We now introduce a class of satisfiability problems including flat $\texttt{struct}_1$-sorted disequalities together with particular dag solved forms.

Definition 10 (Isolated variable, standard variable)**.** Let φ be a dag solved form and let σ be the substitution corresponding to the solved form obtained by variable replacement. A variable x is *isolated* in φ if $x\sigma$ is a variable. A variable x is *standard* in φ if $x\sigma$ is a standard term.

We consider dag solved forms extended with disequalities $x \neq y$ such that x, y are both standard or both isolated.

Proposition 6. *Consider a theory $T = T_f[E_1, E_2]$ such that f is gently growing in T with respect to a measure τ for E_2. Let φ be any separate form*

$$\varphi = \varphi_1 \cup \varphi_{elem} \cup \varphi_2 \cup \varphi_f$$

such that φ_1 is a dag solved form together with flat disequalities $x \neq y$ where x, y are both standard or both isolated. There exists a computable T-equivalent disjunction of separate forms $\bigvee_{k \in K} \varphi_k$ such that φ is T-satisfiable if and only if there exists some $k \in K$ such that the i-component of φ_k is T_i-satisfiable for each $i \in \{1, 2\}$.

Example 4. Consider the bridging function f computing the set of elements in a multiset. Consider the separate form

$$\varphi = \left\{ \begin{array}{l} v = x \uplus y, x \neq y \\ f_x = f_y, f_v = f_x \cup f_y, f_v = \{e\} \\ f_x = f(x), f_y = f(y), f_v = f(v) \end{array} \right\}$$

Using the formula $\varphi \wedge (f_x = \emptyset \vee f_x = e' \cup v')$, we get the T-satisfiability of φ thanks to the branching $f_x = e' \cup v'$. Indeed, φ is T-satisfiable by considering for instance $x = e \uplus e$ and $y = e \uplus e \uplus e$.

6 Conclusion

In this paper, we have studied the problem of combining satisfiability procedures for the union T of two free data structure theories T_1 and T_2 connected by a bridging theory T_f. We have identified some basic propositions to solve the T-satisfiability problem in a modular way when

the source component of the input is already in solved form. As a pre-processing step, applying a unification algorithm is a natural way to compute the expected solved forms. In the case of Proposition 6, the considered solved forms can be obtained, for instance, by applying a matching algorithm on particular equations with one standard (ground) side.

The (combined) data structure theory T involves a free sort `elem`. The next step is to consider the combinability of T with an `elem`-sorted theory T_{elem}. In the easy case, T and T_{elem} are stably infinite, and we can rely on the classical Nelson-Oppen combination method. Actually, all the theories T_1, T_2 and $T = T_1 \cup T_f \cup T_2$ studied in this paper are stably infinite. In the general case, a politeness property [13, 18] for T would be welcome to a get a modular $T \cup T_{elem}$-satisfiability procedure when T_{elem} is arbitrary (non-necessarily stably infinite). To establish the politeness of a class of T-satisfiability problems, we have to show:

- (finite witnessability) if a formula in the class is T-satisfiable, then it must be satisfiable in a small (computable) model;
- (smoothness) if a formula in the class is satisfiable in a model of T, then it is satisfiable in model of T of any larger cardinality.

The technical propositions stated in this paper will allow us to show the politeness of some particular T-satisfiability problems. Hence, this will be a way to identify significant cases for which a combination method à la Nelson-Oppen will compute the right answer when it returns *satisfiable*.

Acknowledgements. We are very grateful to Paula Chocron and Pascal Fontaine for their collaboration on the design of new combination methods for SMT solving [7–9]. This work on FDS has been made possible thanks to the joint study of the AFDS case [8].

References

[1] A. Armando, M. P. Bonacina, S. Ranise, and S. Schulz. New results on rewrite-based satisfiability procedures. *ACM Trans. Comput. Log.*, 10(1), 2009.

[2] A. Armando, S. Ranise, and M. Rusinowitch. A rewriting approach to satisfiability procedures. *Inf. Comput.*, 183(2):140–164, 2003.

[3] F. Baader and S. Ghilardi. Connecting many-sorted theories. *J. Symb. Log.*, 72(2):535–583, 2007.

[4] F. Baader and K. U. Schulz. Combination techniques and decision problems for disunification. *Theor. Comput. Sci.*, 142(2):229–255, 1995.

[5] F. Baader and W. Snyder. Unification theory. In J. A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, pages 445–532. Elsevier and MIT Press, 2001.

[6] T. Bouton, D. C. B. De Oliveira, D. Déharbe, and P. Fontaine. veriT: an open, trustable and efficient SMT-solver. In *Automated Deduction–CADE-22*, pages 151–156. Springer, 2009.

[7] P. Chocron, P. Fontaine, and C. Ringeissen. A Gentle Non-Disjoint Combination of Satisfiability Procedures. In S. Demri, D. Kapur, and C. Weidenbach, editors, *Proc. of the 7th International Joint Conference on Automated Reasoning, IJCAR*, volume 8562 of *LNCS*, pages 122–136. Springer, 2014.

[8] P. Chocron, P. Fontaine, and C. Ringeissen. A Polite Non-Disjoint Combination Method: Theories with Bridging Functions Revisited. In A. P. Felty and A. Middeldorp, editors, *Proc. Conference on Automated Deduction (CADE)*, volume 9195 of *LNCS*, pages 419–433. Springer, 2015.

[9] P. Chocron, P. Fontaine, and C. Ringeissen. A rewriting approach to the combination of data structures with bridging theories. In C. Lutz and S. Ranise, editors, *Frontiers of Combining Systems (FroCoS)*, volume 9322 of *LNCS*, pages 275–290. Springer, 2015.

[10] H. Comon. Disunification: A survey. In *Computational Logic - Essays in Honor of Alan Robinson*, pages 322–359, 1991.

[11] P. Fontaine. Combinations of theories for decidable fragments of first-order logic. In S. Ghilardi and R. Sebastiani, editors, *Frontiers of Combining Systems (FroCoS)*, volume 5749 of *LNCS*, pages 263–278. Springer, 2009.

[12] S. Ghilardi. Model-theoretic methods in combined constraint satisfiability. *Journal of Automated Reasoning*, 33(3-4):221–249, 2004.

[13] D. Jovanovic and C. Barrett. Polite theories revisited. In C. Fermueller and A. Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning (LPAR'10)*, volume 6397 of *LNCS*, pages 402–416. Springer, 2010.

[14] V. Kuncak. Verifying and synthesizing software with recursive functions - (invited contribution). In *ICALP*, volume 8572 of *Lecture Notes in Computer Science*, pages 11–25. Springer, 2014.

[15] J. Meseguer. Variant-based satisfiability in initial algebras. In C. Artho and P. C. Ölveczky, editors, *Formal Techniques for Safety-Critical Systems - Fourth International Workshop, FTSCS 2015, Paris, France, November 6-7, 2015. Revised Selected Papers*, volume 596 of *Communications in Computer and Information Science*, pages 3–34. Springer, 2015.

[16] G. Nelson and D. C. Oppen. Simplification by cooperating decision procedures. *ACM Trans. on Programming Languages and Systems*, 1(2):245–257, Oct. 1979.

[17] T.-H. Pham and M. W. Whalen. An improved unrolling-based decision procedure for algebraic data types. In E. Cohen and A. Rybalchenko, editors, *Verified Software: Theories, Tools, Experiments - 5th International Conference, VSTTE 2013, Menlo Park, CA, USA, Revised Selected Papers*, volume 8164 of *LNCS*, pages 129–148. Springer, 2014.

[18] S. Ranise, C. Ringeissen, and C. G. Zarba. Combining data structures with nonstably infinite theories using many-sorted logic. In B. Gramlich, editor, *Frontiers of Combining Systems (FroCoS)*, volume 3717 of *LNCS*, pages 48–64. Springer, 2005.

[19] V. Sofronie-Stokkermans. Locality results for certain extensions of theories with bridging functions. In R. A. Schmidt, editor, *Proc. Conference on Automated Deduction (CADE)*, volume 5663 of *LNCS*, pages 67–83. Springer, 2009.

[20] P. Suter, M. Dotta, and V. Kuncak. Decision procedures for algebraic data types with abstractions. In M. V. Hermenegildo and J. Palsberg, editors, *Principles of Programming Languages (POPL)*, pages 199–210. ACM, 2010.

[21] C. Tinelli and C. G. Zarba. Combining non-stably infinite theories. *Journal of Automated Reasoning*, 34(3):209–238, Apr. 2005.

[22] R. Treinen. A new method for undecidablity proofs of first order theories. *J. Symb. Comput.*, 14(5):437–458, 1992.

[23] C. G. Zarba. Combining multisets with integers. In A. Voronkov, editor, *Automated Deduction - CADE-18, 18th International Conference on Automated Deduction, Copenhagen, Denmark*, volume 2392 of *LNCS*, pages 363–376. Springer, 2002.

[24] C. G. Zarba. Combining sets with cardinals. *J. Autom. Reasoning*, 34(1):1–29, 2005.