A Polite Non-Disjoint Combination Method: Theories with Bridging Functions Revisited

Paula Chocron, Pascal Fontaine, Christophe Ringeissen

A Polite Non-Disjoint Combination Method: Theories with Bridging Functions Revisited

(Extended Version)

Paula Chocron[1], Pascal Fontaine[2⋆], and Christophe Ringeissen[2]

[1] IIIA-CSIC, Bellaterra, Catalonia, Spain
[2] INRIA, Université de Lorraine & LORIA, Nancy, France

Abstract. The Nelson-Oppen combination method is ubiquitous in Satisfiability Modulo Theories solvers. However, one of its major drawbacks is to be restricted to disjoint unions of theories. We investigate the problem of extending this combination method to particular non-disjoint unions of theories connected via bridging functions. The motivation is, e.g., to solve verification problems expressed in a combination of data structures connected to arithmetic with bridging functions such as the length of lists and the size of trees. We present a sound and complete combination procedure à la Nelson-Oppen for the theory of absolutely free data structures, including lists and trees. This combination procedure is then refined for standard interpretations. The resulting theory has a nice politeness property, enabling combinations with arbitrary decidable theories of elements.

1 Introduction

Solving the satisfiability problem modulo a theory given as a union of decidable sub-theories naturally calls for combination methods. The Nelson-Oppen combination method [9] is now ubiquitous in SMT (Satisfiability Modulo Theories) solvers. However, this technique imposes strong assumptions on the theories in the combination; in the classical scheme [9, 18], the theories notably have to be signature-disjoint and stably infinite. Many recent advances aim to go beyond these two limitations.

The design of a combination method for non-disjoint unions of theories is clearly a hard task [7, 19]. To stay within the frontiers of decidability, it is necessary to impose restrictions on the theories in the combination; and at the same time, those restrictions should not be such that there is no hope of concrete applications for the combination scheme. For this reason, it is worth exploring specific classes of non-disjoint combinations of theories that appear frequently in

⋆ This work has been partially supported by the project ANR-13-IS02-0001 of the Agence Nationale de la Recherche, by the European Union Seventh Framework Programme under grant agreement no. 295261 (MEALS), and by the STIC AmSud MISMT.

software specification, and for which it would be useful to have a simple combination procedure. An example is the case of shared sets, where sets are represented by unary predicates [4, 21]. In this context, the cardinality operator can also be considered; notice that this operator is a bridging function from sets to natural numbers [24]. In this paper, we investigate the case of bridging functions between data structures and a target theory, for instance, the length of lists, in which case the target theory is a fragment of arithmetic. Here, non-disjointness arises from connecting two disjoint theories via a third theory defining the bridging function. This problem is of prime interest for software verification [6, 15, 16, 25], in particular for the verification of recursive (functional) programs with functions defined by pattern-matching. For instance, a satisfiability procedure for data structures combined with bridging functions is the core reasoning engine of the verification tool Leon targeting Scala programs [17]. To solve instances of this problem, dedicated techniques have been developed [16, 22], and general frameworks, based on locality [15] or superposition [1, 3, 10], are also applicable. In particular, the contributions by Zarba [22], Sofronie-Stokkermans [15], and Suter et al. [16] have given rise to the straight combination approach highlighted in this paper. In [22], Zarba presents a procedure for checking satisfiability of lists with length by using a reduction to arithmetic, and a similar reduction applies to multisets with multiplicity [23]. The motivation was to relax the stably-infiniteness assumption in Nelson-Oppen's procedure, e.g., to be able to consider multisets over a finite domain of elements. In that line of work, Zarba focuses on *standard* interpretations. For instance, the standard interpretation for lists corresponds to the case where lists are interpreted as finite lists of elements. Sofronie-Stokkermans [15] relies on locality properties to show that the definition of the function connecting the theories can be eliminated (using instantiations by ground terms). The subtle problem of restricting interpretations to standard ones is also discussed but, in contrast to our approach, only the case of an infinite domain of elements is considered. In [16], Suter et al. present a dedicated procedure for standard interpretations that is sound and complete for *sufficiently surjective* abstraction functions.

We investigate here an approach by reduction from non-disjoint to disjoint combination. It is an alternative to a non-disjoint combination approach à la Ghilardi [2, 7], for which some assumptions on the shared (target) theory are required. Ghilardi's approach has been applied to combine data structures with fragments of arithmetic, like Integer Offsets [11] and Abelian groups [10]; it is however difficult to go beyond Abelian groups and consider for instance any decidable fragment of arithmetic as a shared theory. The approach by reduction does not impose such limitations, and any (decidable) fragment of arithmetic is suitable for the target (shared) theory. The resulting combination procedure is correct for (*arbitrary* interpretations of) absolutely free data structures. Our correctness proof is not based on locality principles [15], but relies on the construction of a combined model in the line of the Nelson-Oppen procedure. Eventually, the outcome of our approach bears similarities with the locality-based procedure.

Then we focus on the problem of adapting this combination procedure to get a satisfiability procedure for the restricted class of *standard* interpretations of absolutely free data structures. The correctness of the combined satisfiability procedure for standard interpretations is based on a *politeness* property, previously introduced to consider disjoint combinations of some data structure theories with any theory of elements [8, 13]. This paper is a first application of politeness to non-disjoint combinations. The interest of applying politeness is twofold. First, it provides a way to relate satisfiability in standard interpretations to satisfiability in the class of all interpretations. Second, it permits to solve in a modular way the satisfiability problem in the combination of (1) standard interpretations of a data structure theory extended with a bridging function and (2) any arbitrary theory of elements. The resulting combined satisfiability procedure has some similarities with the one studied in [12, 16, 17].

Our combination procedures for arbitrary/standard interpretations are first illustrated on the prominent case of lists with length [6]: this is a simple but meaningful case to grasp the concepts and techniques developed in the paper. But our study is not limited to that particular case, and we show that our combination procedures apply to the general case of trees with bridging functions.

The rest of the paper is organized as follows. Section 2 recalls basic concepts and notations. The combination problem is presented in Section 3 and the related combination procedure in Section 4. In Section 5, we focus on the restriction to standard interpretations for the cases of lists (Sections 5.1–5.2) and trees (Section 5.3), by considering appropriate bridging functions and the combination problem with an arbitrary theory of elements. This paper is an extended version of [5] including additional proofs (Appendix A).

2 Preliminaries

We assume an enumerable set of variables $\mathcal{V}$ and a first-order many-sorted signature Σ given by a set of sorts and sets of function and predicate symbols (equipped with an arity). Nullary function symbols are called constant symbols. We assume that, for each sort σ, the equality "$=_\sigma$" is a logical symbol that does not occur in Σ and that is always interpreted as the identity relation over (the interpretation of) σ; moreover, as a notational convention, we omit the subscript for sorts and we simply use the symbol $=$. The notions of Σ-terms, atomic Σ-formulas and first-order Σ-formulas are defined in the usual way. In particular an atomic formula is either an equality, or a predicate symbol applied to the right number of well-sorted terms. Formulas are built from atomic formulas, Boolean connectives ($\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\equiv$), and quantifiers ($\forall$, $\exists$). A literal is an atomic formula or the negation of an atomic formula. 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. An *arrangement* over a finite set of variables V is a maximal satisfiable set of well-sorted equalities and disequalities $x = y$ or $x \neq y$, with $x, y \in V$. Given a

quantifier-free formula φ, an *arranged form* of φ is any conjunction of φ with an arrangement over the variables in φ. 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 $\{x_1 \neq \cdots \neq x_n\}$. Free variables are defined in the usual way, and the set of free variables of a formula φ is denoted by $\mathit{Var}(\varphi)$. Given a sort σ, $\mathit{Var}_\sigma(\varphi)$ denotes the set of variables of sort σ in $\mathit{Var}(\varphi)$. A formula with no free variables is closed, and a formula without variables is ground. A universal formula is a closed formula $\forall x_1 \ldots \forall x_n.\ \varphi$ where φ is quantifier-free. A (finite) Σ-theory is a (finite) set of closed Σ-formulas. Two theories are disjoint if no predicate symbol or function symbol appears in both respective signatures.

From the semantic side, a *Σ-interpretation* $\mathcal{I}$ comprises a non-empty pairwise disjoint domains D_σ for every sort σ, a sort- and arity-matching total function $\mathcal{I}[f]$ for every function symbol f, a sort- and arity-matching predicate $\mathcal{I}[p]$ for every predicate symbol p, and an element $\mathcal{I}[x] \in D_\sigma$ for every variable x of sort σ. By extension, an interpretation defines a value in D_σ for every term of sort σ, and a truth value for every formula. We may write $\mathcal{I} \models \varphi$ whenever $\mathcal{I}[\varphi] = \top$. A Σ-structure is a Σ-interpretation over an empty set of variables.

A model of a formula (theory) is an interpretation that evaluates the formula (resp. all formulas in the theory) to true. A formula or theory is satisfiable (or consistent) if it has a model; it is unsatisfiable otherwise. A formula G is T-satisfiable if it is satisfiable in the theory T, that is, if $T \cup \{G\}$ is satisfiable. A T-model of G is a model of $T \cup \{G\}$. A formula G is T-unsatisfiable if it has no T-models. A theory T is *stably infinite* if any T-satisfiable set of literals is satisfiable in a model of T whose domain is infinite. A Σ-theory T can be equivalently defined as a pair $T = (\Sigma, \mathbf{A})$, where $\mathbf{A}$ is a class of Σ-structures. We may write $\mathcal{A} \in T$ when $T = (\Sigma, \mathbf{A})$ and $\mathcal{A} \in \mathbf{A}$. Given theories $T_i = (\Sigma_i, \mathbf{A}_i)$ for $i = 1, 2$, the *combination* of T_1 and T_2 is the theory $T_1 \oplus T_2 = (\Sigma_1 \cup \Sigma_2, \mathbf{A})$ where $\mathbf{A}$ is the set of $\Sigma_1 \cup \Sigma_2$-structures $\mathcal{A}$ such that the Σ_i-structure $\mathcal{A}^{\Sigma_i}$ (defined by restricting $\mathcal{A}$ to interpret only symbols in Σ_i) is in $\mathbf{A}_i$ for $i = 1, 2$.

3 The Combination Problem

Consider a many-sorted Σ_s-theory T_s and a many-sorted Σ_t-theory T_t (s and t stand for source and target respectively) such that T_s and T_t have disjoint function symbols (but sorts can be shared by Σ_s and Σ_t). We consider a function f mapping elements from T_s to elements in T_t. This function is defined by some axioms expressed in the signature $\Sigma_s \cup \Sigma_t \cup \{f\}$. The set of axioms defining f is called T_f.

In the following, the theory T_s is the theory of Absolutely Free Data Structures [15] (AFDS, for short) and T_f is a bridging theory connecting AFDS to an arbitrary (target) theory T_t. For simplicity, we only consider Absolutely Free Data Structures without selectors. In Section 5, we will argue that selectors are not mandatory when standard interpretations are considered.

Definition 1. *Consider a set of sorts* `Elem`*, and a sort* `struct` $\notin$ `Elem`*. Let Σ be a signature whose set of sorts is* {`struct`} $\cup$ `Elem` *and whose function symbols*

$c \in \Sigma$ *(called* constructors*) have arities of the form:*

$$c : \mathtt{e}_1 \times \cdots \times \mathtt{e}_m \times \mathtt{struct} \times \cdots \times \mathtt{struct} \to \mathtt{struct}$$

where $\mathtt{e}_1, \ldots, \mathtt{e}_m \in \mathtt{Elem}$*. Consider the following axioms (where upper case letters denote implicitly universally quantified variables)*

$$\begin{array}{ll} (Inj_c) & c(X_1, \ldots, X_n) = c(Y_1, \ldots, Y_n) \Rightarrow \bigwedge_{i=1}^{n} X_i = Y_i \\ (Dis_{c,d}) & c(X_1, \ldots, X_n) \neq d(Y_1, \ldots, Y_m) \\ (Acyc_\Sigma) & X \neq t[X] \text{ if } t \text{ is a non-variable } \Sigma\text{-term} \end{array}$$

The theory of Absolutely Free Data Structures over Σ *is*

$$AFDS_\Sigma = \big(\bigcup_{c \in \Sigma} Inj_c \big) \cup \big(\bigcup_{c,d \in \Sigma, c \neq d} Dis_{c,d} \big) \cup Acyc_\Sigma$$

From now on, T_s is $AFDS_{\Sigma_s}$ (see Appendix A.1 for a T_s-satisfiability procedure).

Example 1. The theory of lists is an example of AFDS where the constructors are *cons* : $\mathtt{elem} \times \mathtt{list} \to \mathtt{list}$ and *nil* : $\mathtt{list}$. Similarly (binary) trees are also a classical AFDS example, where the constructor operator is, e.g., *cons* : $\mathtt{elem} \times \mathtt{tree} \times \mathtt{tree} \to \mathtt{tree}$. The theory of pairs (of numbers) is another example of AFDS where the constructor is *cons* : $\mathtt{num} \times \mathtt{num} \to \mathtt{struct}$.

Given a tuple $\boldsymbol{e}$ of terms of sorts in $\mathtt{Elem}$ and a tuple $\boldsymbol{t}$ of terms of sort $\mathtt{struct}$, the tuple $\boldsymbol{e}, \boldsymbol{t}$ may be written $\boldsymbol{e}; \boldsymbol{t}$ to distinguish terms of sort $\mathtt{struct}$ from the other ones. A bridging theory is a set of equational axioms defining a bridging function by structural induction over a set of constructors.

Definition 2. *Let* Σ *be a signature as given in Definition 1 and let* Σ_t *be a signature such that* Σ *and* Σ_t *have distinct function symbols, and may share sorts, except* $\mathtt{struct}$*. A* bridging *function* $f \notin \Sigma \cup \Sigma_t$ *has arity* $\mathtt{struct} \to \mathtt{t}$ *where* $\mathtt{t}$ *is a sort in* Σ_t*. A* bridging *theory* T_f *associated with a bridging function* f *has the form:*

$$T_f = \bigcup_{c \in \Sigma} \Big\{ \ \forall \boldsymbol{e} \forall t_1, \ldots, t_n.\ f(c(\boldsymbol{e}; t_1, \ldots, t_n)) = f_c(\boldsymbol{e}; f(t_1), \ldots, f(t_n)) \ \Big\}$$

where $f_c(\boldsymbol{x}; \boldsymbol{y})$ *denotes a* Σ_t*-term. When* $\boldsymbol{x}$ *does not occur in* $f_c(\boldsymbol{x}; \boldsymbol{y})$ *for any* $c \in \Sigma$*, we say that* T_f *is* $\mathtt{Elem}$-independent*.*

Remark that the notation $f_c(\boldsymbol{x}; \boldsymbol{y})$ does not enforce all elements of $\boldsymbol{x}; \boldsymbol{y}$ to occur in the term $f_c(\boldsymbol{x}; \boldsymbol{y})$: only elements in $\boldsymbol{x}$ of sort in Σ_t can occur in $f_c(\boldsymbol{x}; \boldsymbol{y})$, and there is no occurrence of $\boldsymbol{x}$ in $f_c(\boldsymbol{x}; \boldsymbol{y})$ in the case of an $\mathtt{Elem}$-independent bridging theory. Throughout the paper, we assume that for any constant c in Σ, f_c denotes a constant in Σ_t, and the equality $f(c) = f_c$ occurs in T_f. For instance, in the case of length of lists, $\ell(nil) = \ell_{nil} = 0$.

Example 2. (Example 1 continued). Many useful bridging theories fall into the above definition such as:

– Length of lists: $\ell(cons(e, y)) = 1 + \ell(y),\ \ell(nil) = 0$
– Sum of lists of numbers: $lsum(cons(e, y)) = e + lsum(y),\ lsum(nil) = 0$
– Sum of pairs of numbers: $psum(cons(e, e')) = e + e'$

Among the above bridging theories, only the length of lists is `Elem`-independent.

4 A Combination Procedure for Bridging Functions

We introduce a combination method for a non-disjoint union of theories $T = T_s \cup T_f \cup T_t$ as stated in Section 3, where the source theory T_s is $AFDS_{\Sigma_s}$, T_t is an arbitrary target Σ_t-theory, and the bridging theory T_f follows Definition 2. It is worth noticing that T_t is not required to be stably infinite. We describe below a decision procedure for checking the T-satisfiability of sets of literals. As usual, the input set of literals is first purified to get a separate form.

Definition 3. *A set of literals φ is in* separate form *if $\varphi = \varphi_{struct} \cup \varphi_{elem} \cup \varphi_t \cup \varphi_f$ where:*

– *φ_{struct} contains only flat literals of forms $x = y$, $x \neq y$ or $x = c(\boldsymbol{e}; x_1, \ldots, x_n)$ where x, $x_1, \ldots, x_n$ and y are variables of sort* `struct` *and c is a constructor*
– *φ_{elem} contains only literals of sorts in $\Sigma_s \backslash (\Sigma_t \cup \{\texttt{struct}\})$*
– *φ_t contains only Σ_t-literals*
– *φ_f contains only flat equalities of the form $f_x = f(x)$, where f_x denotes a variable associated with $f(x)$, such that f_x and $f(x)$ occur once in φ_f and each variable of sort* `struct` *in φ_{struct} occurs in φ_f.*

It is easy to convert any set of literals into an equisatisfiable separate form by introducing fresh variables to denote impure terms.

Unlike classical disjoint combination methods, guessing only one arrangement on the shared variables is not sufficient to get a modular decision procedure.

Definition 4. *Given a set of literals $\varphi = \varphi_{struct} \cup \varphi_{elem} \cup \varphi_t \cup \varphi_f$ in separate form and two arrangements*

– *Γ over the variables of sorts in $\Sigma_s \cap \Sigma_t$ occurring in both φ_{struct} and $\varphi_t \cup \varphi_f$;*
– *Γ' over the variables of sort* `struct` *in $\varphi_{struct} \cup \varphi_f$;*

the combinable separate form *of φ corresponding to Γ, Γ' is $(\varphi_{struct} \cup \Gamma_{struct}) \cup \varphi_{elem} \cup (\varphi_t \cup \Gamma_t) \cup \varphi_f$ where*

$$\begin{aligned} \Gamma_{struct} &= \Gamma \cup \Gamma' \\ \Gamma_t &= \Gamma \cup \{f_x = f_y \mid x = y \in \Gamma'\} \\ &\quad \cup \{f_x = f_c(\boldsymbol{e}; f_{x_1}, \ldots, f_{x_n}) \mid x = c(\boldsymbol{e}; x_1, \ldots, x_n) \in \varphi_{struct}\} \end{aligned}$$

Any separate form extends to finitely many combinable separate forms. Also the separate form is T-equivalent to the disjunction of those combinable separate forms. From now on, we will only consider combinable separate forms and assume that a combinable separate form $\varphi_{struct} \cup \varphi_{elem} \cup \varphi_t \cup \varphi_f$ includes Γ_{struct} and Γ_t respectively in φ_{struct} and φ_t. The T-satisfiability of combinable separate forms can be checked in a modular way (see Appendix A.2 for the correctness proof):

Theorem 1. *A* ***combinable*** *separate form* $\varphi_{struct} \cup \varphi_{elem} \cup \varphi_t \cup \varphi_f$ *is* T*-satisfiable if and only if* $\varphi_{struct} \cup \varphi_{elem}$ *is* T_s*-satisfiable and* φ_t *is* T_t*-satisfiable.*

Notice that φ_f is not used when checking satisfiability: these constraints are indeed encoded within φ_t, according to Definition 4.

Example 3. Consider the theory of (acyclic) lists with a length function ℓ and the set of literals $\varphi = \{x = cons(a, cons(b, z)),\ \ell(x) + 1 = \ell(z)\}$, where the sort for elements is not the sort of integers.

1. **Variable Abstraction and Partition.** Formula φ is separated into
   - $\varphi_{list} : \{y = cons(b, z), x = cons(a, y)\}$
   - $\varphi_{elem} : \emptyset$
   - $\varphi_{int} : \{\ell_x + 1 = \ell_z\}$
   - $\varphi_\ell : \{\ell_x = \ell(x), \ell_y = \ell(y), \ell_z = \ell(z)\}$
2. **Decomposition.** To build the combinable separate form, let Γ_{list} be the only arrangement over the list variables satisfiable together with φ_{list}, i.e. $\{x \neq y \neq z\}$. By Definition 4, Γ_{int} is $\{\ell_y = \ell_z + 1, \ell_x = \ell_y + 1\}$.
3. **Check.** The set $\varphi_{list} \cup \varphi_{elem} \cup \Gamma_{list}$ is satisfiable in the theory of lists. However $\varphi_{int} \cup \Gamma_{int}$ is unsatisfiable in the theory of linear arithmetic (over the integers). The original set of literals φ is thus unsatisfiable.

The next satisfiable formula is used as a running example in Section 5.

Example 4. Consider the following set of literals

$$\varphi = \{x_1 = cons(d, y_1), x_2 = cons(d, y_2), x_1 \neq x_2 \neq y_1 \neq y_2 \neq y_3, \ell(y_2) = \ell(y_3)\}$$

1. **Variable Abstraction and Partition.** Formula φ is separated into
   - $\varphi_{list} : \{x_1 = cons(d, y_1), x_2 = cons(d, y_2), x_1 \neq x_2 \neq y_1 \neq y_2 \neq y_3\}$
   - $\varphi_{elem} : \emptyset$
   - $\varphi_{int} : \{\ell_{y_2} = \ell_{y_3}\}$
   - $\varphi_\ell : \{\ell_{x_1} = \ell(x_1), \ell_{x_2} = \ell(x_2), \ell_{y_1} = \ell(y_1), \ell_{y_2} = \ell(y_2), \ell_{y_3} = \ell(y_3)\}$
2. **Decomposition.** Formula φ_{list} already includes the arrangement $\Gamma_{list} = \{x_1 \neq x_2 \neq y_1 \neq y_2 \neq y_3\}$, and $\Gamma_{int} = \{\ell_{x_1} = \ell_{y_1} + 1, \ell_{x_2} = \ell_{y_2} + 1\}$.
3. **Check.** The set $\varphi_{list} \cup \varphi_{elem} \cup \Gamma_{list}$ is satisfiable in the theory of lists. The set $\varphi_{int} \cup \Gamma_{int}$ is also satisfiable in the theory of linear arithmetic (over the integers), e.g. $\ell_{x_1} = 4, \ell_{x_2} = 3, \ell_{y_1} = 3, \ell_{y_2} = 2, \ell_{y_3} = 2$. Thus φ is satisfiable.

5 Standard Interpretations

Now consider the satisfiability problem modulo data structure theories defined as classes of *standard* structures, where each interpretation domain of `struct` contains only the finite terms generated by the constructors and the elements in the interpretation domains of `Elem`. Standard structures are specific models of the (axiomatized) theories considered in previous sections. We investigate the possibility to get a satisfiability procedure for standard interpretations by applying the combination method of Section 4. We first study the particular case of lists, and then the general case of trees corresponding to the standard interpretations of absolutely free data structures.

5.1 Lists with Length

Definition 5. *Consider the signature* $\Sigma_{list} = \Sigma \cup \{\ell : \texttt{list} \to \texttt{int}\} \cup \Sigma_{int}$ *where* $\Sigma = \{cons : \texttt{elem} \times \texttt{list} \to \texttt{list}, nil : \texttt{list}\}$, $\Sigma_{int} = \{0 : \texttt{int}, 1 : \texttt{int}, + : \texttt{int} \times \texttt{int} \to \texttt{int}, \leq : \texttt{int} \times \texttt{int}\}$, *and* $\texttt{elem} \neq \texttt{int}$. *A* standard *list*-interpretation $\mathcal{A}$ *is a* Σ_{list}*-interpretation satisfying the following conditions:*

- $|A_{\texttt{elem}}| > 1$*;*
- $A_{\texttt{list}} = (A_{\texttt{elem}})^*$ *where* $(A_{\texttt{elem}})^*$ *is the set of finite sequences* $\langle e_1, \ldots, e_n \rangle$ *for* $n \geq 0$ *and* $e_1, \ldots, e_n \in A_{\texttt{elem}}$*;*
- $A_{\texttt{int}} = \mathbb{Z}$ *and* $0, 1, +, \leq$ *are interpreted according to their standard interpretation in* $\mathbb{Z}$*;*
- $\mathcal{A}[nil] = \langle\rangle$*;*
- $\mathcal{A}[cons](e, \langle e_1, \ldots, e_n \rangle) = \langle e, e_1, \ldots, e_n \rangle$*, for* $n \geq 0$ *and* $e, e_1, \ldots, e_n \in A_{\texttt{elem}}$*;*
- $\mathcal{A}[\ell](\langle\rangle) = 0$*;*
- $\mathcal{A}[\ell](\langle e_1, \ldots, e_n \rangle) = n$.

The theory of (standard interpretations) of lists with length *is the pair* $T^{si}_{list} = (\Sigma_{list}, \mathbf{A})$*, where* $\mathbf{A}$ *is the class of all standard list-structures.*

Remark 1. Definition 5 excludes the case of lists built over only one element. In that singular case, the length function ℓ is bijective, which means that any disequality $x \neq y$ between `list`-variables can be equivalently translated into an `int`-disequality $\ell_x \neq \ell_y$. It thus suffices to extend the combination procedure in Section 4 with this additional translation expressing the bijectivity of ℓ. The satisfiability of the `int`-part of the resulting separate form gives a model for the theory of lists on only one element. In the case of lists, and for simplicity, we thus impose the restriction of at least two elements to standard interpretations.

In this paper, we have chosen to define standard interpretations without selectors. Indeed, selectors would be partial functions defined only on non-empty lists, and could be seen as syntactic sugar: any equality $e = car(x)$ (resp. $y = cdr(x)$) can be equivalently expressed as an equality $x = cons(e, x')$ (resp. $x = cons(d, y)$) where x' (resp. d) is a fresh variable. Thus we define T^{si}_{list} without selectors. As shown below, we can relate T^{si}_{list}-satisfiability to satisfiability modulo the **combined** theory of lists with length T_{list} defined as (the class of all the models of) the union of theories $AFDS_\Sigma \cup T_\ell \cup T_\mathbb{Z}$ where $\Sigma = \{cons : \texttt{elem} \times \texttt{list} \to \texttt{list},\ nil : \texttt{list}\}$, $T_\ell = \{\forall e \forall y.\ \ell(cons(e, y)) = 1 + \ell(y),\ \ell(nil) = 0\}$, and $T_\mathbb{Z}$ denotes the theory of linear arithmetic over the integers. Since $T^{si}_{list} \models T_{list}$, a T^{si}_{list}-satisfiable formula is also T_{list}-satisfiable. For the the converse implication, we need to preprocess the formulas. To build a standard interpretation from the model construction of Theorem 1, we need additional arithmetic constraints to state that each value of a length variable corresponds to the length of some finite list. These constraints are used to get witness formulas as defined in [8] for the combination of polite theories [13].

Definition 6 (Finite witnessability). *Let* Σ *be a signature,* S *be a set of sorts in* Σ*,* T *a* Σ*-theory, and* φ *a quantifier-free* Σ*-formula. A quantifier-free* Σ*-formula* ψ *is a* finite witness of φ in T with respect to S *if:*

1. *φ and $(\exists \bar{v})\psi$ are T-equivalent, where $\bar{v} = Var(\psi) \setminus Var(\varphi)$;*
2. *for any arranged form ψ' of ψ, if ψ' is T-satisfiable then there exists a T-interpretation $\mathcal{A}$ satisfying ψ' such that $A_\sigma = \bigcup_{v \in Var_\sigma(\psi')} \mathcal{A}[v]$, for each $\sigma \in S$.*

T is finitely witnessable *with respect to S if there exists a computable function witness such that, for every quantifier-free Σ-formula φ, $witness(\varphi)$ is a finite witness of φ in T with respect to S.*

For the theory T^{si}_{list}, witnesses w.r.t. $\{\texttt{elem}\}$ are derived from *range constraints*: given a set of literals φ in separate form and a natural number n, a *range constraint for φ bounded by n* is a set of literals $\mathfrak{c} = \{\mathfrak{c}(f_x) \mid f_x \in Var_{\texttt{int}}(\varphi_f)\}$ where $\mathfrak{c}(f_x)$ is either $f_x = i$ $(0 \leq i < n)$ or $f_x \geq n$. A range constraint $\mathfrak{c}$ for φ is *satisfiable* if $\varphi_{int} \cup \mathfrak{c}$ is satisfiable in $\mathbb{Z}$. In the case of T^{si}_{list}, the role of range constraints is to perform a guessing of values for length variables. Beyond a limit value n, depending on the input formula, there are enough different lists to satisfy the disequalities between lists, and then to build a standard interpretation.

Proposition 1. *For any set of literals φ in combinable separate form, there exists a finite set of satisfiable range constraints $\mathfrak{C}$ such that*

- *φ is T^{si}_{list}-equivalent to $\bigvee_{\mathfrak{c} \in \mathfrak{C}} (\varphi \wedge \mathfrak{c})$*
- *For any $\mathfrak{c} \in \mathfrak{C}$, $\varphi \wedge \mathfrak{c}$ admits a witness denoted $witness(\varphi \wedge \mathfrak{c})$ such that any arranged form of $witness(\varphi \wedge \mathfrak{c})$ is T^{si}_{list}-satisfiable iff it is T_{list}-satisfiable.*

Proof. Since φ is a combinable separate form, it implies a unique arrangement over $\texttt{list}$-variables. Let m be the number of the corresponding equivalence classes over $\texttt{list}$-variables. We define the bound n used in range constraints as $n = \lceil \log_2(m) \rceil$ to have, for any $i \geq n$, m different lists of length i built over two elements (by definition of T^{si}_{list}, we have at least two elements, the simple case of only one element must be considered separately). The set $\mathfrak{C}$ is defined as the set of all satisfiable range constraints bounded by n. Let us now define the witness of a range constraint $\mathfrak{c}$:

- $witness_{rc}(\{\ell_x = 0\} \cup \mathfrak{c}) = \{x = nil\} \cup witness_{rc}(\mathfrak{c})$
- $witness_{rc}(\{\ell_x = i\} \cup \mathfrak{c}) = \{x = cons(e_1, \ldots cons(e_i, nil)\ldots)\} \cup witness_{rc}(\mathfrak{c})$ if $0 < i < n$, where $e_1, \ldots, e_i$ are fresh $\texttt{elem}$-variables
- $witness_{rc}(\{\ell_x \geq n\} \cup \mathfrak{c}) = witness_{rc}(\mathfrak{c})$

Then, $witness(\varphi \wedge \mathfrak{c}) = (e \neq e') \wedge \varphi \wedge \mathfrak{c} \wedge witness_{rc}(\mathfrak{c})$, where e, e' are two distinct fresh $\texttt{elem}$-variables.

Consider an arbitrary arrangement arr over variables in $witness(\varphi \wedge \mathfrak{c})$. If $witness(\varphi \wedge \mathfrak{c}) \wedge arr$ is T_{list}-satisfiable, then it is possible to construct (by using syntactic unification as in Appendix A.1) a T_{list}-equivalent set of literals φ' whose $\texttt{list}$-part contains only flat disequalities and equalities of the following forms:

(1) flat equalities $v = x$ such that v occurs once in φ',

(2) equalities $x = t$, where t is a nil-terminated list and x occurs once in the equalities of φ',
(3) equalities $x = cons(d, y)$, where x and y cannot be equal to nil-terminated lists (by applying the variable replacement of syntactic unification).

Let us now define a T^{si}_{list}-interpretation. First, the equalities in (1) can be discarded since v occurs once in φ'. The interpretation of variables occurring in (2) directly follows from φ'. It remains to show how to interpret variables occurring in (3). Note that each of these variables has a length greater or equal than n, otherwise it would occur in (2). The solved form φ' defines a (partial) ordering $>$ on these variables: $x > y$ if $x = cons(d, y)$ occurs in φ'. Each minimal variable y with respect to $>$ is interpreted by a fresh nil-terminated list not occurring in φ' whose elements are (the representatives of) e, e', and whose length is the interpretation of ℓ_y (this is possible by definition of n and the fact that $\ell_y \geq n$). Then, the interpretation of non-minimal variables follows from the equalities (3) in φ'. By construction, distinct variables are interpreted by distinct lists. In other words, the `list`-disequalities introduced by arr are satisfied by this interpretation. Furthermore, any equality $\ell_x = \ell(x)$ in φ_ℓ is satisfied by this interpretation since φ is a combinable separate form. Therefore, all literals of φ' are true in this interpretation, and so we have built a T^{si}_{list}-model of $witness(\varphi \wedge \mathfrak{c}) \wedge arr$.

Moreover the above construction is a way to build a T^{si}_{list}-model such that the `elem` sort is interpreted as the set of interpreted `elem`-variables in the witness. So the *witness* function satisfies the requirements of Definition 6.

□

Example 5. Consider the T^{si}_{list}-satisfiability of the combinable separate form built in Example 4: $\varphi = \varphi_\ell \cup \{x_1 = cons(d, y_1), x_2 = cons(d, y_2), x_1 \neq x_2 \neq y_1 \neq y_2 \neq y_3, \ell_{x_1} = \ell_{y_1} + 1, \ell_{x_2} = \ell_{y_2} + 1, \ell_{y_2} = \ell_{y_3}\}$. The five `list`-variables imply that range constraints are bounded by $n = 3$. There are 4^5 possible range constraints (each variable can be equal to 0, 1, 2 or greater than or equal to 3). We now focus on few satisfiable range constraints and their related witnesses (the remaining ones are handled similarly).

1. $\mathfrak{c} = \{\ell_{x_1} = \ell_{x_2} = 1, \ell_{y_1} = \ell_{y_2} = \ell_{y_3} = 0\}$. To obtain a witness of φ and $\mathfrak{c}$, we add $y_1 = y_2 = y_3 = nil$, $x_1 = cons(e_{x_1}, nil)$, and $x_2 = cons(e_{x_2}, nil)$. It follows that $e_{x_1} = e_{x_2} = d$ and $x_1 = x_2$ which contradicts φ.
2. $\mathfrak{c} = \{\ell_{x_1} \geq 3, \ell_{y_1} = \ell_{x_2} = 2, \ell_{y_2} = \ell_{y_3} = 1\}$. The witness leads to
   - $y_1 = cons(e'_{y_1}, cons(e_{y_1}, nil))$, $y_2 = cons(e_{y_2}, nil)$, $y_3 = cons(e_{y_3}, nil)$
   - $x_1 = cons(d, y_1) = cons(d, cons(e'_{y_1}, cons(e_{y_1}, nil)))$
   - $x_2 = cons(d, y_2) = cons(d, cons(e_{y_2}, nil))$

   All the `list`-variables are instantiated by distinct lists, provided the arrangement over `elem`-variables is such that $e_{y_2} \neq e_{y_3}$ and ($e_{y_1} \neq e_{y_2}$ or $e'_{y_1} \neq d$).
3. $\mathfrak{c} = \{\ell_{x_1} = 1, \ell_{y_1} = 0, \ell_{x_2} \geq 3, \ell_{y_2} \geq 3, \ell_{y_3} \geq 3\}$. The related witness is equisatisfiable to $\varphi \cup \mathfrak{c} \cup \{y_1 = nil, e \neq e'\}$, which is satisfiable by considering:
   - $y_2 = cons(e, cons(e, cons(e, nil)))$, $y_3 = cons(e, cons(e, cons(e', nil)))$
   - $y_1 = nil, x_1 = cons(d, nil), x_2 = cons(d, cons(e, cons(e, cons(e, nil))))$

The following sections will demonstrate that witnesses are not only interesting for T^{si}_{list}-satisfiability but also for the combination of T^{si}_{list} with an arbitrary theory for elements. However, when T^{si}_{list} is considered alone, there is a much simpler T^{si}_{list}-satisfiability procedure (see Appendix A.3 for the proof).

Theorem 2. *Let φ be a set of literals in combinable separate form, and let $\mathfrak{C}$ be the finite set of satisfiable range constraints of φ* **bounded by** 1*. The formula φ is T^{si}_{list}-satisfiable iff there exists a satisfiable range constraint $\mathfrak{c} \in \mathfrak{C}$ such that $witness(\varphi \wedge \mathfrak{c})$ is T_{list}-satisfiable.*

5.2 Combining Lists with an Arbitrary Theory of Elements

As shown below, T^{si}_{list} is actually a polite theory, and so it can be combined with an arbitrary disjoint theory of elements, using the combination method designed for polite theories [8,13]. By definition, a polite theory is both finite witnessable and smooth.

Definition 7 (Smoothness & Politeness). *Consider a set $S = \{\sigma_1, \ldots, \sigma_n\}$ of sorts in a signature Σ. A Σ-theory 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 every cardinal number $\kappa_1, \ldots, \kappa_n$ such that $\kappa_i \geq |A_{\sigma_i}|$, for $i = 1, \ldots, n$,*

there exists a T-model $\mathcal{B}$ of φ such that $|B_{\sigma_i}| = \kappa_i$ for $i = 1, \ldots, n$. A Σ-theory T is polite *with respect to S if it is both smooth and finitely witnessable with respect to S.*

The smoothness of the theory of standard interpretations of lists has been shown in [13], and this is preserved while considering the length function. By definition of T^{si}_{list}, any set of elements can be used to build the lists (since ℓ is $\{\texttt{elem}\}$-independent). Hence a T^{si}_{list}-satisfiable formula remains T^{si}_{list}-satisfiable when adding elements (of sort `elem`), and so T^{si}_{list} is smooth. The finite witnessability of T^{si}_{list} is a consequence of Proposition 1.

Proposition 2. *T^{si}_{list} is polite with respect to $\{\texttt{elem}\}$.*

Consider the satisfiability problem in the disjoint combination $T^{si}_{list} \oplus T_{elem}$ where T_{elem} is a mono-sorted Σ_{elem}-theory over the shared sort `elem`. Due to the politeness of T^{si}_{list}, we can directly use the combination method initiated in [13] for polite theories, and this leads to the following result.

Theorem 3. *Let φ be a set of literals in combinable separate form, and let $\mathfrak{C}$ be the finite set of satisfiable range constraints introduced in Proposition 1. The formula φ is $T^{si}_{list} \oplus T_{elem}$-satisfiable iff there exists a satisfiable range constraint $\mathfrak{c} \in \mathfrak{C}$ and an arrangement arr such that (1) $witness(\varphi \wedge \mathfrak{c}) \wedge arr$ is T_{list}-satisfiable and (2) $\varphi_{elem} \wedge arr$ is T_{elem}-satisfiable, where arr is an arrangement over the variables of sort `elem` in $witness(\varphi \wedge \mathfrak{c})$.*

Proof. It follows from the correctness proof of the combination method known for polite theories [8, 13]. □

Example 6. Recall the formula from Example 4 in its combinable separate form and suppose we add a new literal stating that the sum of the lengths of y_1, y_2 and y_3 is three: $\varphi = \varphi_\ell \cup \{x_1 = cons(d, y_1), x_2 = cons(d, y_2), x_1 \neq x_2 \neq y_1 \neq y_2 \neq y_3, \ell_{x_1} = \ell_{y_1} + 1, \ell_{x_2} = \ell_{y_2} + 1, \ell_{y_2} = \ell_{y_3}, \ell_{y_1} + \ell_{y_2} + \ell_{y_3} = 3\}$ and consider the theory of elements $T_{elem} = \{a \neq b, (\forall x : \texttt{elem}.\ x = a \vee x = b)\}$. There are now only two satisfiable range constraints:

1. $\ell_{x_1} \geq 3, \ell_{y_1} \geq 3, \ell_{x_2} = 1, \ell_{y_2} = 0, \ell_{y_3} = 0$, which leads to $\ell_{x_1} = 4$ and $\ell_{y_1} = 3$. But this is T^{si}_{list}-unsatisfiable, as it requires $y_2 = nil$ and $y_3 = nil$, which makes $y_2 \neq y_3$ false.
2. $\ell_{x_1} = 2, \ell_{y_1} = 1, \ell_{x_2} = 2, \ell_{y_2} = 1, \ell_{y_3} = 1$, which implies
   - $y_1 = cons(e_{y_1}, nil)$, $y_2 = cons(e_{y_2}, nil)$, $y_3 = cons(e_{y_3}, nil)$
   - $x_1 = cons(d, cons(e_{y_1}, nil))$, $x_2 = cons(d, cons(e_{y_2}, nil))$

   But this requires $e_{y_1} \neq e_{y_2} \neq e_{y_3}$, which is T_{elem}-unsatisfiable.

Hence φ is $T^{si}_{list} \oplus T_{elem}$-unsatisfiable.

Let us now assume T_{elem} is stably infinite. Since T^{si}_{list} is stably infinite too, the classical Nelson-Oppen combination method applies to $T^{si}_{list} \oplus T_{elem}$ by using the T^{si}_{list}-satisfiability procedure stated in Theorem 2. This leads to a result similar to Theorem 3, where it is sufficient to guess only few particular range constraints.

Proposition 3. *Assume T_{elem} is stably infinite. Let φ be a set of literals in combinable separate form, and let $\mathfrak{C}$ be the finite set of satisfiable range constraints of φ* **bounded by** 1*. The formula φ is $T^{si}_{list} \oplus T_{elem}$-satisfiable iff there exists a satisfiable range constraint $\mathfrak{c} \in \mathfrak{C}$ and an arrangement arr such that (1) $witness(\varphi \wedge \mathfrak{c}) \wedge arr$ is T_{list}-satisfiable and (2) $\varphi_{elem} \wedge arr$ is T_{elem}-satisfiable, where arr is an arrangement over the variables of sort* `elem` *in $witness(\varphi \wedge \mathfrak{c})$.*

In the above proposition, *arr* is an arrangement over the variables of sort `elem` in φ since $Var(\varphi) = Var(witness(\varphi \wedge \mathfrak{c}))$. Indeed $witness(\varphi \wedge \mathfrak{c})$ only extends $\varphi \cup \mathfrak{c}$ with an equality $x = nil$ for each $l_x = 0$ in $\mathfrak{c}$.

5.3 Trees with Bridging Functions over the Integers

The combination method presented for standard interpretations of lists can be extended to standard interpretations of any AFDS theory, as discussed below.

Definition 8. *Consider the signature $\Sigma_{tree} = \Sigma \cup \{f : \texttt{struct} \to \texttt{int}\} \cup \Sigma_{int}$ where Σ and Σ_{int} are signatures respectively as in Definition 1 and Definition 5 such that* `int` $\notin$ `Elem`*, and let T_f be an* `Elem`*-independent bridging theory as in Definition 2. A* standard *tree*-interpretation *$\mathcal{A}$ is a Σ_{tree}-interpretation satisfying the following conditions:*

- *$A_{\texttt{struct}}$ is the set of Σ-terms of sort* `struct` *built with elements in $(A_{\texttt{e}})_{\texttt{e} \in \texttt{Elem}}$;*

- $A_{\mathtt{int}} = \mathbb{Z}$ *and* $0, 1, +, \leq$ *are interpreted according to their standard interpretation in* $\mathbb{Z}$*;*
- $\mathcal{A}[c] = c$ *for each constant constructor* $c \in \Sigma$*;*
- $\mathcal{A}[c](\boldsymbol{e}, t_1, \ldots, t_n) = c(\boldsymbol{e}, t_1, \ldots, t_n)$ *for each non-constant constructor* $c \in \Sigma$*, tuple* $\boldsymbol{e}$ *of elements in* $(A_{\mathtt{e}})_{\mathtt{e} \in \mathtt{Elem}}$*, and* $t_1, \ldots, t_n \in A_{\mathtt{struct}}$*;*
- $\mathcal{A}[f](c) = f_c$ *for each constant constructor* $c \in \Sigma$*;*
- $\mathcal{A}[f](c(\boldsymbol{e}, t_1, \ldots, t_n)) = f_c(\boldsymbol{e}, \mathcal{A}[f](t_1), \ldots, \mathcal{A}[f](t_n))$ *for each non-constant constructor* $c \in \Sigma$*, tuple* $\boldsymbol{e}$ *of elements in* $(A_{\mathtt{e}})_{\mathtt{e} \in \mathtt{Elem}}$*, and* $t_1, \ldots, t_n \in A_{\mathtt{struct}}$*.*

The theory of (standard interpretations) of trees with bridging function f *is the pair* $T^{si}_{tree} = (\Sigma_{tree}, \mathbf{A})$*, where* $\mathbf{A}$ *is the class of all standard tree-structures.*

Let T_{tree} be the **combined** theory of trees with the bridging function f defined as (the class of all the models of) the union of theories $AFDS_\Sigma \cup T_f \cup T_{\mathbb{Z}}$. If a formula is T^{si}_{tree}-satisfiable, then it is also T_{tree}-satisfiable. For the converse implication, we proceed like for lists by introducing witnesses. Witnesses can easily be computed when f is the height or the size of trees. Hence, in a way analogous to what has been done for lists (cf. Proposition 1), there is a method to reduce T^{si}_{tree}-satisfiability to T_{tree}-satisfiability. As shown below, T^{si}_{tree} is polite, and so we can obtain a $T^{si}_{tree} \oplus T_{elem}$-satisfiability procedure by combining the satisfiability procedures for T_{tree} and T_{elem} (analogous to Theorem 3). The following assumptions enable us to extend the proofs developed for lists to the case of trees.

Definition 9. *Let* T *be a theory defined as a class of standard tree-structures. For any* $\mathcal{A} \in T$*, let* $F^{-1}_{\mathcal{A}}(n) = \{t \mid \mathcal{A}[f](t) = n\}$*. The bridging function* f *is* gently growing in T *if*

1. *for any* $n \in \mathbb{Z}$ *and any* $\mathcal{A} \in T$*,* $F^{-1}_{\mathcal{A}}(n) \neq \emptyset \iff n \geq 0$*;*
2. *for any* $n \geq 0$ *and any* $\mathcal{A} \in T$*,* $|F^{-1}_{\mathcal{A}}(n)| < |F^{-1}_{\mathcal{A}}(n+1)|$*;*
3. *there exists a computable function* $b : \mathbb{N} \to \mathbb{N}$ *such that for any* $n > 1$ *and any* $\mathcal{A} \in T$*,* $|F^{-1}_{\mathcal{A}}(b(n))| \geq n$*;*
4. *for any* $n \geq 0$*, one can compute a finite non-empty set* $F^{-1}(n)$ *of terms with variables of sorts in* `Elem` *such that*

$$T \models f(x) = n \iff (\exists \bar{v} \; . \bigvee_{t \in F^{-1}(n)} x = t) \qquad \textit{where } \bar{v} = \mathit{Var}(F^{-1}(n))$$

Proposition 4. *Let* $\Sigma = \{cons : \mathtt{elem} \times \mathtt{struct} \times \cdots \times \mathtt{struct} \to \mathtt{struct},\ nil : \mathtt{struct}\}$*. Assume that cons is of arity strictly greater than* 2 *and consider the following bridging theories:*

- *Size of trees:* $sz(cons(e, y_1, \ldots, y_n)) = 1 + \sum_{i=1}^{n} sz(y_i),\ sz(nil) = 0$
- *Height of trees:* $ht(cons(e, y_1, \ldots, y_n)) = 1 + \max_{i \in [1,n]} ht(y_i),\ ht(nil) = 0$

If $f = sz$ *or* $f = ht$*, then* f *is gently growing in* T^{si}_{tree}*.*

To prove the above proposition, the function b of Definition 9 can be defined as the identity over $\mathbb{N}$, but it is possible to get a better bound, e.g., thanks to Catalan numbers [25] for the size of trees. When *cons* is of arity 2, sz and ht coincide with the length ℓ. In that case, ℓ is gently growing in T^{si}_{list} that corresponds to $T^{si}_{tree} \cup \{\exists v, v' : \texttt{elem} \;.\; v \neq v'\}$.

Proposition 5. *If f is gently growing in T^{si}_{tree}, then T^{si}_{tree} is polite w.r.t.* `Elem`.

Proof. See Appendix A.4. □

Theorem 3 (for lists) can be rephrased for trees and gives:

Corollary 1. *Assume f is gently growing in T^{si}_{tree}. Let $T^{si}_{tree} \oplus T_{elem}$ be a disjoint combination where T_{elem} is a many-sorted Σ_{elem}-theory over the sorts in* `Elem`. *$T^{si}_{tree} \oplus T_{elem}$-satisfiability is decidable if T_{elem}-satisfiability is decidable.*

Consider a theory T^{si}_{tree} as in Proposition 4 where $f = sz$ or $f = ht$. Similarly to Theorem 2, T^{si}_{tree}-satisfiability reduces to T_{tree}-satisfiability by guessing only range constraints bounded by 1. If T_{elem} is stably infinite, then we get a combination method for $T^{si}_{tree} \oplus T_{elem}$-satisfiability as in Proposition 3.

6 Conclusion

This paper describes (Section 4) a non-deterministic combination method à la Nelson-Oppen for unions of theories including absolutely free data structures connected to target theories via bridging functions. Similarly to the classical Nelson-Oppen method, implementations of this non-deterministic combination method should be based not on guessings but on more practical refinements. But this lightweight approach is in the line with disjoint combination procedures embedded in SMT solvers, and is thus amenable to integration in those tools.

We reuse the notions of witness and politeness (Section 5), already introduced for non-stably infinite disjoint combinations, to adapt satisfiability procedures to standard interpretations. Hence, the combination method for polite theories is applicable to combine the theory of standard interpretations of lists (trees) with an arbitrary disjoint theory for elements. The case of standard interpretations of lists (trees) over integer elements has not been tackled but can be solved using the approach discussed for a stably infinite theory of elements.

To complete this work, we are currently investigating more (data structure) theories with bridging functions for which the combination method of Section 4 is sound and complete. Another natural continuation consists in considering standard interpretations modulo non-absolutely free constructors [15], e.g., associative-commutative operators to specify multisets.

Acknowledgments: We are grateful to Jasmin Blanchette and to the anonymous reviewers for many constructive remarks.

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. F. Baader and S. Ghilardi. Connecting many-sorted theories. *J. Symb. Log.*, 72(2):535–583, 2007.
3. P. Baumgartner and U. Waldmann. Hierarchic superposition with weak abstraction. In *Automated Deduction - CADE-24 - 24th International Conference on Automated Deduction, Lake Placid, NY, USA*, volume 7898 of *LNCS*, pages 39–57. Springer, 2013.
4. P. Chocron, P. Fontaine, and C. Ringeissen. A Gentle Non-Disjoint Combination of Satisfiability Procedures. In *Proc. of the 7th International Joint Conference on Automated Reasoning, IJCAR*. Springer, 2014. Extended version available as Inria Research Report, cf. `http://hal.inria.fr/hal-00985135`.
5. P. Chocron, P. Fontaine, and C. Ringeissen. A Polite Non-Disjoint Combination Method: Theories with Bridging Functions Revisited. In *Proc. Conference on Automated Deduction (CADE)*, LNCS. Springer, 2015. To appear.
6. P. Fontaine, S. Ranise, and C. G. Zarba. Combining lists with non-stably infinite theories. In F. Baader and A. Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning (LPAR'04)*, volume 3452 of *LNCS*, pages 51–66. Springer-Verlag, 2005.
7. S. Ghilardi. Model-theoretic methods in combined constraint satisfiability. *Journal of Automated Reasoning*, 33(3-4):221–249, 2004.
8. 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.
9. G. Nelson and D. C. Oppen. Simplification by cooperating decision procedures. *ACM Trans. on Programming Languages and Systems*, 1(2):245–257, Oct. 1979.
10. E. Nicolini, C. Ringeissen, and M. Rusinowitch. Combinable extensions of Abelian groups. In R. A. Schmidt, editor, *Proc. Conference on Automated Deduction (CADE)*, volume 5663 of *LNCS*, pages 51–66. Springer, 2009.
11. E. Nicolini, C. Ringeissen, and M. Rusinowitch. Combining satisfiability procedures for unions of theories with a shared counting operator. *Fundam. Inform.*, 105(1-2):163–187, 2010.
12. 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.
13. S. Ranise, C. Ringeissen, and C. G. Zarba. Combining data structures with non-stably infinite theories using many-sorted logic. In B. Gramlich, editor, *Frontiers of Combining Systems (FroCoS)*, volume 3717 of *LNCS*, pages 48–64. Springer, 2005.
14. R. E. Shostak. A practical decision procedure for arithmetic with function symbols. *J. ACM*, 26(2):351–360, 1979.
15. 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.
16. 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.

17. P. Suter, A. S. Köksal, and V. Kuncak. Satisfiability modulo recursive programs. In E. Yahav, editor, *Static Analysis - 18th International Symposium, SAS 2011, Venice, Italy*, volume 6887 of *LNCS*, pages 298–315. Springer, 2011.
18. C. Tinelli and M. T. Harandi. A new correctness proof of the Nelson–Oppen combination procedure. In F. Baader and K. U. Schulz, editors, *Frontiers of Combining Systems (FroCoS)*, Applied Logic, pages 103–120. Kluwer Academic Publishers, Mar. 1996.
19. C. Tinelli and C. Ringeissen. Unions of non-disjoint theories and combinations of satisfiability procedures. *Theoretical Comput. Sci.*, 290(1):291–353, Jan. 2003.
20. D. Tran, C. Ringeissen, S. Ranise, and H. Kirchner. Combination of convex theories: Modularity, deduction completeness, and explanation. *J. Symb. Comput.*, 45(2):261–286, 2010.
21. T. Wies, R. Piskac, and V. Kuncak. Combining theories with shared set operations. In S. Ghilardi and R. Sebastiani, editors, *Frontiers of Combining Systems (FroCoS)*, volume 5749 of *LNCS*, pages 366–382. Springer, 2009.
22. C. G. Zarba. Combining lists with integers. In *International Joint Conference on Automated Reasoning (Short Papers), Technical Report DII 11/01*, pages 170–179. University of Siena, 2001.
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.
25. T. Zhang, H. B. Sipma, and Z. Manna. Decision procedures for term algebras with integer constraints. *Inf. Comput.*, 204(10):1526–1574, 2006.

A Appendix

Let us introduce this technical section with some classical notations related to terms. The set of Σ-terms (of sort σ) is denoted by $T(\Sigma)$ ($T_\sigma(\Sigma)$). A finite set of variables V can be seen as a set of constants, and we may consider the signature $\Sigma \cup V$ and the set of terms $T(\Sigma \cup V)$. Given a set of equalities E, the relation $=_E$ denotes the *equational theory of* E which is defined as the smallest relation including E which is closed by reflexivity, symmetry, transitivity, congruence and substitivity. For any term t, $[\![t]\!]$ denotes the equivalence class of t modulo $=_E$, and $T(\Sigma \cup V)/=_E$ is $\{[\![t]\!] \mid t \in T(\Sigma \cup V)\}$.

A.1 A Satisfiability Procedure Modulo AFDS

The theory AFDS has some very good properties with respect to the satisfiability problem. First, it is a convex theory, as is any theory defined by a set of Horn clauses. Second, it is a Shostak theory [14], which means that it admits a solver and a canonizer, denoted respectively *solve* and *canon*. A satisfiability procedure modulo a Shostak theory can be constructed by using both the solver and the canonizer [20]. Given an input set of literals divided into a set of equalities Γ and a set of disequalities Δ, the procedure works as follows:

Solve: It reports unsatisfiability if $solve(\Gamma)$ returns the unsatisfiable output $false$. Otherwise, let μ be the substitution returned by $solve(\Gamma)$.

Canonize: It reports unsatisfiability if there exists some disequality $s \neq t \in \Delta$ such that $canon(s\mu) = canon(t\mu)$. Otherwise, return satisfiable.

Moreover, an equality between variables $x = y$ is entailed by a satisfiable input if and only if $canon(x\mu) = canon(y\mu)$.

In the case of AFDS, *solve* is given by syntactic unification, and *canon* is simply the identity.

Del :	$\{x = x\} \cup \Gamma \vdash \Gamma$
Dec :	$\{x = c(x_1, \ldots, x_n), x = c(x'_1, \ldots, x'_n)\} \cup \Gamma$
	$\vdash \{x = c(x_1, \ldots, x_n), x_1 = x'_1, \ldots, x_n = x'_n\} \cup \Gamma$ if $c \in \Sigma$
Clash :	$\{x = c(x_1, \ldots, x_n), x = d(y_1, \ldots, y_m)\} \cup \Gamma \vdash \bot$
	if $c, d \in \Sigma, c \neq d$
Cycle :	$\{x = t_1[x_1], \ldots, x_{n-1} = t_n[x]\} \cup \Gamma \vdash \bot$
	if $t_1, \ldots, t_n$ are Σ-terms of depth 1
Merge :	$\{x = y\} \cup \Gamma \vdash \{x \mapsto y\}(\Gamma) \cup \{x = y\}$
	if $x, y \in Var(\Gamma), x \neq y$

Fig. 1. Syntactic unification over flat equalities

Proposition 6. *Let $\varphi = \Gamma \cup \Delta$ be a set of flat Σ_s-literals such that Γ (resp. Δ) is the set of equalities (resp. disequalities) in φ. If φ is $AFDS_{\Sigma_s}$-satisfiable, then φ is satisfiable in an $AFDS_{\Sigma_s}$-interpretation $T(\Sigma_s \cup V)/ =_E$, where V is the set of variables in φ and E is the (dag) solved form of Γ computed by the syntactic unification algorithm given in Figure 1.*

Proof. In Figure 1, we adapt a standard syntactic unification algorithm to maintain equalities in flat form. This syntactic unification algorithm is used to compute the flat (directed acyclic graph) solved form E of Γ. Consider the interpretation $\mathcal{A}$ whose domain A is $T(\Sigma_s \cup V)/ =_E$ and such that constructors $c \in \Sigma_s$ are interpreted as expected: $\mathcal{A}[c]([\![\boldsymbol{e}]\!]; [\![t_1]\!], \ldots, [\![t_n]\!]) = [\![c(\boldsymbol{e}; t_1, \ldots t_n)]\!]$ and $\mathcal{A}[v] = [\![v]\!]$ for each $v \in V$. By this definition, $\mathcal{A}$ is a model of $AFDS_{\Sigma_s}$, and $\mathcal{A}$ satisfies E, as well as the set of equalities in φ. Moreover, $\mathcal{A}$ satisfies all the disequalities in φ, otherwise it would contradict the assumption that φ is $AFDS_{\Sigma_s}$-satisfiable. Hence, we can conclude that $\mathcal{A}$ satisfies φ. □

A.2 Correctness of the Combination Procedure

Lemma 1 (Soundness). *Let $\varphi = \varphi_{struct} \cup \varphi_{elem} \cup \varphi_t \cup \varphi_f$ be a set of literals in separate form. If φ is T-satisfiable, then there exist some Γ_{struct} and Γ_t as introduced in Definition 4 such that:*

1. *$\varphi_{struct} \cup \varphi_{elem} \cup \Gamma_{struct}$ is T_s-satisfiable*

2. $\varphi_t \cup \Gamma_t$ *is* T_t*-satisfiable.*

Proof. Let $\mathcal{M}$ be a T-interpretation satisfying φ. Consider the set of variables V'_{struct} defined as the set of variables of sort `struct` in φ plus the set of variables occurring in both φ_{struct} and $\varphi_t \cup \varphi_f$. The sets of literals Γ_{struct} and Γ_t are obtained as in Definition 4 with the following arrangements Γ, Γ':

- Γ is the arrangement over the variables of shared sorts in both φ_{struct} and $\varphi_t \cup \varphi_f$ such that $x = y \in \Gamma$ if and only if $\mathcal{M}[x] = \mathcal{M}[y]$;
- Γ' is the arrangement over the variables of sort `struct` in $\varphi_{struct} \cup \varphi_f$ such that $x = y \in \Gamma'$ if and only if $\mathcal{M}[x] = \mathcal{M}[y]$.

Clearly, (1) holds since $\mathcal{M}$ is an $AFDS_{\Sigma_s}$-interpretation satisfying φ_{struct}, φ_{elem} and also Γ_{struct} by construction.

To check (2), notice that $\mathcal{M}$ is a T_t-interpretation satisfying φ_t. By construction, $\mathcal{M}$ satisfies also the arrangement $\Gamma \subseteq \Gamma_t$. For any equality $e \in \Gamma_t \backslash \Gamma$, we have that $T \models \varphi \Rightarrow e$, and so e is satisfied by $\mathcal{M}$. □

Lemma 2 (Completeness). *Let* $\varphi = \varphi_{struct} \cup \varphi_{elem} \cup \varphi_t \cup \varphi_f$ *be a set of literals in separate form. If there exist some* Γ_{struct} *and* Γ_t *as introduced in Definition 4 such that:*

1. $\varphi_{struct} \cup \varphi_{elem} \cup \Gamma_{struct}$ *is* T_s*-satisfiable*
2. $\varphi_t \cup \Gamma_t$ *is* T_t*-satisfiable*

then φ *is* T*-satisfiable.*

Proof. Consider the set S of sorts shared by Σ_s and Σ_t. Let us first assume $S = \emptyset$ (which is the interesting case for Section 5).

According to Proposition 6 (Appendix A.1), there exists a term-generated $AFDS_{\Sigma_s}$-interpretation $\mathcal{A}$ satisfying $\varphi_{struct} \cup \varphi_{elem} \cup \Gamma_{struct}$, whose domain is precisely $T(\Sigma_s \cup V)/=_E$ where V is the set of variables in $\varphi_{struct} \cup \varphi_{elem} \cup \Gamma_{struct}$ and E is a finite set of flat equalities. The interpretation in $\mathcal{A}$ of constructors $c \in \Sigma_s$ is as follows: $\mathcal{A}[c]([\![\boldsymbol{e}]\!]; [\![t_1]\!], \ldots, [\![t_n]\!]) = [\![c(\boldsymbol{e}; t_1, \ldots t_n)]\!]$ and $\mathcal{A}[v] = [\![v]\!]$ for each $v \in V$. Second, let $\mathcal{B}$ be a T_t-interpretation satisfying $\varphi_t \cup \Gamma_t$. Consider the following set of variables:

- $V_{struct} = Var_{\mathtt{struct}}(\varphi_{struct} \cup \varphi_f)$
- $V_{elem} = Var(\varphi_{elem})$
- $V_t = Var(\varphi_t)$

Note that V_{struct}, V_{elem} and V_t are pairwise disjoint, and $V = V_{struct} \cup V_{elem}$.

We are now ready to define an interpretation $\mathcal{M}$. First, we specify the domains. Let $M_{\mathtt{t}} = B_{\mathtt{t}}$ for any sort $\mathtt{t} \in \Sigma_t$ and $M_{\mathtt{s}} = A_{\mathtt{s}}$ for any $\mathtt{s} \in \Sigma_s$. Hence $M_{\mathtt{struct}}$ is $T_{\mathtt{struct}}(\Sigma_s \cup V)/=_E$. We consider the following interpretation in $\mathcal{M}$:

- for each $u \in V_t$, $\mathcal{M}[u] = \mathcal{B}[u]$
- for each $x \in V_{struct} \cup V_{elem}$, $\mathcal{M}[x] = [\![x]\!]$

– the interpretation of constructors $c \in \Sigma_s$ is defined on the equivalence classes in the usual way[3] : $\mathcal{M}[c](\mathcal{M}[\boldsymbol{e}]; [\![t_1]\!], \ldots, [\![t_n]\!]) = [\![c(\boldsymbol{e}; t_1, \ldots t_n)]\!]$
– the interpretation of the symbols in Σ_t is the same as the one in $\mathcal{B}$
– the interpretation of the function f is defined recursively over the equivalence classes in $M_{\texttt{struct}}$ as follows:
  - If we consider the equivalence class of some $x \in V_{struct}$, then $\mathcal{M}[f]([\![x]\!]) = \mathcal{B}[f_x]$.
  - Otherwise, the equivalence class must consist of just one constructed element. If $[\![c(\boldsymbol{e}; t_1, \ldots t_n)]\!]$ is an equivalence class of this form, then

$$\mathcal{M}[f]([\![c(\boldsymbol{e}; t_1, \ldots, t_n)]\!]) = f_c(\mathcal{M}[\boldsymbol{e}]; \mathcal{M}[f]([\![t_1]\!]), \ldots, \mathcal{M}[f]([\![t_n]\!]))$$

Now we need to show that $\mathcal{M}$ is a T-interpretation satisfying $\varphi_{struct} \cup \varphi_{elem} \cup \varphi_t \cup \varphi_f$. The sets of literals $\varphi_{struct} \cup \varphi_{elem}$ and φ_t are clearly satisfied by $\mathcal{M}$, as they are respectively satisfied by $\mathcal{A}$ and $\mathcal{B}$ and we preserve these interpretations. It remains to check that φ_f is satisfied by $\mathcal{M}$. For any $x \in V_{struct}$, we have that $\mathcal{M}[f](\mathcal{M}[x]) = \mathcal{M}[f]([\![x]\!]) = \mathcal{B}[f_x] = \mathcal{M}[f_x]$, and so $\varphi_f = \bigcup_{x \in V_{struct}} \{f_x = f(x)\}$ is satisfied by $\mathcal{M}$.

Then we still need to prove that $\mathcal{M} \models T$. By construction of $\mathcal{M}$, we have that $\mathcal{M} \models AFDS_{\Sigma_s}$ and $\mathcal{M} \models T_t$. To prove that $\mathcal{M} \models T_f$, let us analyze the different equivalence classes of $M_{\texttt{struct}}$:

– If we consider the equivalence class of some $x \in V_{struct}$, Γ_{struct} contains the literal $f_x = f_c(\boldsymbol{e}; f_{x_1}, \ldots, f_{x_n})$ if $x = c(\boldsymbol{e}; x_1, \ldots, x_n)$ occurs in φ_{struct}. This literal is satisfied by $\mathcal{B}$, and since $\mathcal{M}[f]([\![v]\!]) = \mathcal{B}[f_v]$ for any $v \in V_{struct}$, the axioms of T_f must hold.
– Otherwise, we recursively define $\mathcal{M}[f]$ by resorting to the definition given by the axioms of T_f, so they hold by construction.

Consider now a non-empty set of shared sorts $S \subseteq \texttt{Elem}$. In that case, there also exists a larger T_s-model $\mathcal{A}$ satisfying $\varphi_{struct} \cup \varphi_{elem} \cup \Gamma_{struct}$ such that

– $A_\sigma = B_\sigma$ for each sort σ in S
– $\mathcal{A}^\Sigma$ is $T(\Sigma \cup V \cup D)/ =_E$ for an appropriate set of elements D of sorts in S

This particular model $\mathcal{A}$ exists thanks to the arrangement Γ (over S) occurring in both Γ_{struct} and Γ_t. Then, the construction of $\mathcal{M}$ is just like in the case $S = \emptyset$, and this leads to a T-model $\mathcal{M}$ satisfying φ. □

A.3 Standard Interpretations of Lists with Length

Theorem 2. *Let φ be a set of literals in combinable separate form, and let $\mathfrak{C}$ be the finite set of satisfiable range constraints of φ* **bounded by** 1*. The formula φ is T^{si}_{list}-satisfiable iff there exists a satisfiable range constraint $\mathfrak{c} \in \mathfrak{C}$ such that $witness(\varphi \wedge \mathfrak{c})$ is T_{list}-satisfiable.*

[3] The interpretation of constructors is well-defined since $\mathcal{A}$ and $\mathcal{B}$ satisfy the same arrangement over variables of sorts in $\Sigma_s \cap \Sigma_t$ occurring in both φ_{struct} and $\varphi_t \cup \varphi_f$.

Proof. Given a satisfiable range constraint $\mathfrak{c}$ such that there exists a T_{list}-model of $witness(\varphi \wedge \mathfrak{c})$, we show the existence of a T^{si}_{list}-model of $\varphi \wedge \mathfrak{c}$.

Similarly to the proof of Proposition 1 and by using syntactic unification as in Appendix A.1, $witness(\varphi \wedge \mathfrak{c})$ is equivalent to a set of literals φ' whose `list` part contains only flat disequalities and equalities of the following forms:

(1) flat equalities $v = x$ such that v occurs once in φ',
(2) equalities $x = t$, where t is a *nil*-terminated list and x occurs once in the equalities of φ',
(3) equalities $x = cons(d, y)$, where x and y cannot be equal to *nil*-terminated lists (by applying the variable replacement of syntactic unification).

Let us detail how to interpret `list`-variables. For variables occurring in (2), the interpretation is obvious. Minimal variables are defined as in the proof of Proposition 1. Each of these minimal variables has a length greater or equal than 1, otherwise it would occur in (2). For the minimal variables, we use the interpretation satisfying $\mathfrak{c}$ to consider lists of appropriate **strictly positive** lengths and containing **fresh** (distinct) elements. For non-minimal variables, the interpretation is inductively defined by the equalities of the form (3) occurring in φ. By this construction, different `list`-variables are still interpreted by distinct lists. Moreover, any equality $\ell_x = \ell(x)$ in φ_ℓ is satisfied by this interpretation since φ is a combinable separate form. Therefore, all literals of φ' are true in this interpretation, and so a T^{si}_{list}-model of φ' has been constructed. It is a T^{si}_{list}-model of φ since φ' is T^{si}_{list}-equivalent to $\varphi \wedge \mathfrak{c}$. □

A.4 Politeness of Trees

Given any sort $\mathtt{e}$ in $\mathtt{Elem}$, we define the theory $T^{\geq \kappa}_{\mathtt{e}}$ of at least κ element(s) of sort $\mathtt{e}$ as follows: $T^{\geq \kappa}_{\mathtt{e}} = \{\exists v_1, \ldots, v_\kappa : \mathtt{e} \;.\; v_1 \neq \ldots \neq v_\kappa\}$ for any $\kappa \geq 2$, and $T^{\geq 1}_{\mathtt{e}} = \emptyset$. A *cardinality mapping* is a mapping $\boldsymbol{\kappa} : \mathtt{Elem} \to \mathbb{N}^+$. For any cardinality mapping $\boldsymbol{\kappa}$, let $T^{\geq \boldsymbol{\kappa}}_{\mathtt{Elem}} = \bigcup_{\mathtt{e} \in \mathtt{Elem}} T^{\geq \boldsymbol{\kappa}(\mathtt{e})}_{\mathtt{e}}$, and $T^{\geq \boldsymbol{\kappa}}_{tree} = T^{si}_{tree} \oplus T^{\geq \boldsymbol{\kappa}}_{\mathtt{Elem}}$. By definition, $T^{si}_{tree} = T^{\geq \mathbf{1}}_{tree}$ where $\mathbf{1}$ is the (lowest) cardinality mapping such that $\mathbf{1}(\mathtt{e}) = 1$ for any $\mathtt{e} \in \mathtt{Elem}$. According to Definition 5, T^{si}_{list} corresponds to $T^{\geq \mathbf{2}}_{tree}$ such that $\mathtt{Elem} = \{\mathtt{elem}\}$, $\mathbf{2}(\mathtt{elem}) = 2$, $\Sigma = \{cons : \mathtt{elem} \times \mathtt{list} \to \mathtt{list},\ nil : \mathtt{list}\}$, and f is the length ℓ.

Proposition 7. *Let $\boldsymbol{\kappa}$ be any cardinality mapping. Assume f is gently growing in $T^{\geq \boldsymbol{\kappa}}_{tree}$. For any set of literals φ in combinable separate form, there exists a finite set of satisfiable range constraints $\mathfrak{C}$ such that*

- *φ is $T^{\geq \boldsymbol{\kappa}}_{tree}$-equivalent to $\bigvee_{\mathfrak{c} \in \mathfrak{C}} (\varphi \wedge \mathfrak{c})$*
- *For any $\mathfrak{c} \in \mathfrak{C}$, $\varphi \wedge \mathfrak{c}$ admits a witness denoted $witness(\varphi \wedge \mathfrak{c})$ such that any arranged form of $witness(\varphi \wedge \mathfrak{c})$ is $T^{\geq \boldsymbol{\kappa}}_{tree}$-satisfiable iff it is T_{tree}-satisfiable.*

Proof. The proof of Proposition 1 can be adapted by using the assumptions (2) and (3) of Definition 9. By assumption, there exist computable functions b and F^{-1} as given in Definition 9.

Since φ is a combinable separate form, it is obtained by an arrangement over `struct`-variables. Let m be the number of the corresponding equivalence classes over `struct`-variables. We define the bound n used in range constraints as $n = b(m)$. The set $\mathfrak{C}$ is defined as the set of all satisfiable range constraints bounded by n. Let us now define the witness of a range constraint $\mathfrak{c}$:

- $witness_{rc}(\{f_x = i\} \cup \mathfrak{c}) = \bigvee_{t \in F^{-1}(i)}(x = t \wedge witness_{rc}(\mathfrak{c}))$ if $0 \leq i < n$, where variables in t are fresh variables of sort in `Elem`;
- $witness_{rc}(\{f_x \geq n\} \cup \mathfrak{c}) = witness_{rc}(\mathfrak{c})$.

Then, we define $witness(\varphi \wedge \mathfrak{c})$ equal to

$$\bigwedge_{\mathtt{e} \in \mathtt{Elem}} \Big(\bigwedge_{v \in W_\mathtt{e}} (v = v) \wedge \bigwedge_{v, v' \in W_\mathtt{e}, v \neq v'} (v \neq v') \Big) \wedge \varphi \wedge \mathfrak{c} \wedge witness_{rc}(\mathfrak{c})$$

where $W_\mathtt{e}$ denotes a set of $\boldsymbol{\kappa}(\mathtt{e})$ variable(s)[4] of sort $\mathtt{e}$ for any $\mathtt{e} \in \mathtt{Elem}$. The construction of a $T^{\geq \boldsymbol{\kappa}}_{tree}$-interpretation is analogous to the construction given in Proposition 1 for lists, by using terms in $T_{\mathtt{struct}}(\Sigma, \bigcup_{\mathtt{e} \in \mathtt{Elem}} W_\mathtt{e})$ corresponding to instances of terms in $\bigcup_{n \geq 0} F^{-1}(n)$, instead of using nil-terminated lists.

Given a T_{tree}-satisfiable arranged form of $witness(\varphi \wedge \mathfrak{c})$, let φ' be the equivalent formula obtained as in Proposition 1 by solving the `struct`-equalities with syntactic unification. Again, there are enough distinct terms to interpret differently the minimal `struct`-variables in φ', thanks to the function b. Then, the interpretation of the other `struct`-variables follows from φ'. With this interpretation and by using the injectivity of constructors in Σ, we can prove by structural induction that all flat `struct`-disequalities are satisfied. The `struct`-variables also occur in the subset φ_f of φ'. Since φ is a combinable separate form, φ_f is satisfied too, and so we have built a $T^{\geq \boldsymbol{\kappa}}_{tree}$-model of φ' (equivalently, of the given arranged form of $witness(\varphi \wedge \mathfrak{c})$). □

Since the smoothness proof for T^{si}_{list} can be directly extended to $T^{\geq \boldsymbol{\kappa}}_{tree}$, we thus obtain the following politeness result for $T^{\geq \boldsymbol{\kappa}}_{tree}$.

Corollary 2. *Let $\boldsymbol{\kappa}$ be any cardinality mapping. If f is gently growing in $T^{\geq \boldsymbol{\kappa}}_{tree}$, then $T^{\geq \boldsymbol{\kappa}}_{tree}$ is polite with respect to* `Elem`.

[4] Trivial equalities $v = v$ are used to introduce fresh variables denoting elements. Actually, trivial equalities of sort $\mathtt{e}$ can be omitted when $\boldsymbol{\kappa}(\mathtt{e}) > 1$: in that case, the non-empty conjunction of disequalities $v \neq v'$ of sort $\mathtt{e}$ is sufficient.