Journal of Automated Reasoning (2020) 64:97–134
https://doi.org/10.1007/s10817-019-09512-4

Politeness and Combination Methods for Theories with Bridging Functions

Paula Chocron[1] · **Pascal Fontaine**[2] · **Christophe Ringeissen**[3]

Received: 15 December 2016 / Accepted: 8 January 2019 / Published online: 22 January 2019

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 defined by connecting disjoint theories via bridging functions. A possible application is 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 method 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. In addition, we have identified a class of polite data structure theories for which the combination method remains sound and complete. This class includes all the subtheories of absolutely free data structures (e.g, the empty theory, injectivity, projection). Again, the politeness property holds for any theory in this class, which can thus be combined with bridging functions and arbitrary decidable theories of elements. This illustrates the significance of politeness in the context of non-disjoint combinations of theories.

Keywords Satisfiability modulo theories · Combination method · Non-disjoint theories · Bridging functions

This work has been partially supported by the European Research Council (ERC) starting Grant Matryoshka (713999).

✉ Christophe Ringeissen
Christophe.Ringeissen@loria.fr

1 IIIA-CSIC, Bellaterra, Catalonia, Spain

2 INRIA, Université de Lorraine and LORIA, Nancy, France

3 INRIA and LORIA, Nancy, France

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 [15] is now ubiquitous in Satisfiability Modulo Theories (SMT) solvers. However, this technique imposes strong assumptions on the theories in the combination; in the classical scheme [15,27], 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 [12,28]. 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 be permissive enough to accommodate concrete applications of the combination scheme. For this reason, it is worth exploring specific classes of non-disjoint combinations of theories that appear frequently in software specification, and for which it would be useful to have a simple combination procedure. The case of sets, possibly represented by shared unary predicates, is a motivating example [8,30]. When considering the data structure of sets, the cardinality operator is a natural bridging function from sets to natural numbers [33]. The length of lists is another classical example of a bridging function between a data structure of lists and a target theory of arithmetic. For these combinations, 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 [11,22,24,34], 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 [25]. To solve instances of this problem, dedicated techniques have been developed [24,31], and general frameworks, based on non-disjoint combination [3,12], superposition [1,7,16] or locality [22] are also applicable. The superposition calculi provide elegant and uniform ways to build satisfiability procedures for data structures [1,2], possibly extended with bridging functions [7,14,16,17]. Then, the resulting satisfiability procedures can be combined using a non-disjoint combination approach à la Ghilardi [12]. This blend of superposition and combination has been applied to unions of data structure theories sharing some particular fragments of arithmetic, like integer offsets [17] and Abelian groups [16]; it is however difficult to go beyond Abelian groups and consider for instance any decidable fragment of arithmetic as a shared theory.

The results by Zhang et al. [34], Zarba [31], Sofronie-Stokkermans [22] and Suter et al. [24] have given rise to the straight combination approach highlighted in this paper. In [34], Zhang et al. investigate the problem of extending the theory of finite trees with a length function, by showing a decision procedure for the quantifier-free extended theory and more generally a quantifier elimination procedure. The satisfiability procedure given in [34] for quantifier-free formulas relies on a reduction to arithmetic. The challenging case appears when the trees are generated by a finite set of constants. To solve that case, the reduction must incorporate counting constraints because there are only finitely many distinct trees with the same given length. In [31], 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 [32]. The motivation was to relax the stably-infiniteness assumption in Nelson–Oppen's procedure, in particular, to be able to consider data structures over a finite domain of elements, where the elements correspond to the constants in the setting of [34]. In both [34] and [31,32], the authors focus 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 [22] relies on locality properties of axiomatized data structures 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. In [22], it is mentioned that the case of an infinite set of constants is easy, and counting constraints are used as in [34] to deal with a finite set of constants. In [24], Suter et al. present a dedicated procedure for standard interpretations that is sound and complete for *sufficiently surjective* abstraction functions.

To solve the satisfiability problem in unions of theories connected with bridging functions, we investigate here an approach by reduction from non-disjoint to disjoint combination. This approach does not impose any limitation on the target theory, and so any (decidable) fragment of arithmetic is suitable. The resulting combination procedure is correct for absolutely free data structures. Our correctness proof is not based on locality principles [22], though it bears similarities with it. The proof relies instead on the construction of a combined model in the line of the Nelson–Oppen procedure.

Building on this combination procedure, we then focus on 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 [13,20]. Intuitively, a polite theory satisfies some form of finite model property and is smooth, i.e. any model can be extended to models of greater cardinality. This paper is a first application of politeness to non-disjoint combinations. The benefit 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 is instrumental 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) an arbitrary theory of elements. The resulting satisfiability procedure has some similarities with the one studied in [19,24,25], but thanks to politeness, it is expressed as a clean combination procedure.

Our combination procedures for arbitrary/standard interpretations are first illustrated on the prominent case of lists with length [11]. This is a simple but meaningful case to grasp the concepts and techniques developed in the paper. We later show that our combination procedures apply to the general case of trees with bridging functions.

Another contribution of this paper is to identify a class of data structure theories for which our first combination method remains complete. In this class, theories are many-sorted, with disjoint sorts to denote respectively the data instances and the structure instances. When the source theory is in this class, the target theory can be arbitrary, due to the fact that we are focusing on data structure theories that also fulfill the politeness property. Hence, the second contribution can be considered as another way to extend the use of polite theories to some simple non-disjoint combinations. The class of data structure theories is of practical interest since it includes well-known finitely axiomatized theories for which a rewriting approach to satisfiability can be successfully applied [1,2]. In this class, one can find the theory of equality, the theory of (acyclic) lists/trees, the theory of absolutely free data structures (with or without selectors).

The completeness proof of our combination method requires the construction of a combined model from the models available in the component theories. For that purpose, we introduce the notion of *polished* theory, for which a satisfiable input admits particular term-generated models modulo a congruence relation E, where the generators and E are derived from the terms of the input. The originality of our *rewriting* approach is to define a bridging

theory as a convergent term rewrite system F, and to analyze the interplay between F and E. The careful study of $F \cup E$ leads to the construction of the combined model.

The paper extends and improves two previous shorter versions considering respectively standard interpretations [9] and axiomatized data structure theories [10]. The new notion of polished theory represents a significant improvement with respect to the class of basic data structure theories studied in [10]. Polished theories allow us to get a clear connection with politeness. As a consequence, we can now consider arbitrary target theories instead of stably infinite ones as in [10], especially in presence of selectors. Moreover, we show that combinations produce theories that remain polished and can be further combined in the same way. This provides an elegant solution to chain several bridging theories, whereas it was previously not clear in [10] that the resulting combined theories remain suitable for further combinations.

Section 2 recalls basic concepts and notations and Sect. 3 introduces the theory of absolutely free data structures. The combination problem and the related combination procedure are presented in Sect. 4. In Sect. 5, we focus on the restriction to standard interpretations for the cases of lists (Sects. 5.1–5.2) and trees (Sect. 5.3), by considering appropriate bridging functions and the combination problem with an arbitrary theory of elements. In Sect. 6, we introduce the class of *polished* theories. By using a rewriting approach, we prove in Sect. 6.2 the completeness of the combination procedure (given in Sect. 4) for this class of theories. Section 7 discusses in details the connections of our contributions with existing works.

2 Preliminaries

2.1 Terms and Equational Theories

We assume a first-order many-sorted signature Σ given by a set of sorts and sets of function and predicate symbols (equipped with an arity), together with a denumerable set of sorted variables $\mathcal{V}$. Nullary function symbols are called constant symbols. A Σ-term is a term built over the signature Σ with variables in $\mathcal{V}$. A ground Σ-term is a Σ-term without variables. The set of ground Σ-terms (of sort σ) is denoted by $T(\Sigma)$ (resp. $T_\sigma(\Sigma)$). Given a set of constants C disjoint from Σ, the signature $\Sigma \cup C$ is called a *constant expansion* of Σ if sorts of C belong to Σ; constants in C are said to be *free*.

We assume that, for each sort σ, the equality symbol "$=_\sigma$" 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 $=$.

An equality is a pair of terms (of same sort), denoted by $s = t$. 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 substitutivity. For any term t, the equivalence class of t modulo $=_E$ is denoted by $[\![t]\!]_E$ or simply $[\![t]\!]$ if E is clear from the context. Given a constant expansion $\Sigma \cup C$, the set of equivalence classes of ground $(\Sigma \cup C)$-terms modulo $=_E$ is denoted by $T(\Sigma \cup C)/=_E$ and by a slight abuse of notation, the corresponding $(\Sigma \cup C)$-structure defined in the usual way is also denoted by $T(\Sigma \cup C)/=_E$. A term rewrite system R is a set of oriented equalities. A convergent term rewrite system R is defined in the usual way [4] and implies, for any term t, the existence and the unicity of its normal form $t\downarrow_R$ which is the same for all terms of an equivalence class modulo $=_R$.

2.2 Formulas

The notions of 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. For any predicate $p \in \Sigma$, a literal $p(t_1, \ldots, t_n)$ or $\neg p(t_1, \ldots, t_n)$ is flat when each term $t_1, \ldots, t_n$ is a variable or a constant. 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 Σ-formula φ and a set S of sorts in Σ, a *S-sorted arranged form* of φ is any conjunction of φ with an arrangement over the S-sorted 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 $Var(\varphi)$. Given a sort σ, $Var_\sigma(\varphi)$ denotes the set of variables of sort σ in $Var(\varphi)$. 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 or function symbols occur in both respective signatures.

2.3 Semantics

From the semantic side, a *Σ-interpretation* $\mathcal{I}$ comprises non-empty pairwise disjoint domains $\mathcal{I}[\sigma]$ 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 $\mathcal{I}[x] \in \mathcal{I}[\sigma]$ for every variable $x \in \mathcal{V}$ of sort σ. By extension, an interpretation defines a value in $\mathcal{I}[\sigma]$ 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. The unsatisfiable formula $\bot$ is used to denote the empty clause, i.e., the empty disjunction of literals. 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. Given a signature Σ and a set of sorts S in Σ, a Σ-theory T is *stably infinite with respect to S* if any T-satisfiable set of literals is satisfiable in a model $\mathcal{A}$ of T whose domain $\mathcal{A}[\sigma]$ is infinite for any $\sigma \in S$. A Σ-theory is said to be stably infinite if it is stably infinite with respect to the set of all sorts in Σ. 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 a Σ-structure $\mathcal{A}$ and a signature $\Sigma' \subseteq \Sigma$, $\mathcal{A}^{\Sigma'}$ is the Σ'-structure defined by restricting $\mathcal{A}$ to interpret only symbols in Σ'. Given a Σ-theory T and a signature $\Sigma' \subseteq \Sigma$, $T^{\Sigma'}$ is the Σ'-theory $(\Sigma', \mathbf{A}^{\Sigma'})$ where $\mathbf{A}^{\Sigma'}$ is the class of Σ'-structures $\mathcal{A}^{\Sigma'}$ such that $\mathcal{A} \in T$. Given theories $T_i = (\Sigma_i, \mathbf{A}_i)$ for $i = 1, 2$, the *combination* of T_1 and T_2 is the theory $(\Sigma_1 \cup \Sigma_2, \mathbf{A})$ where $\mathbf{A}$ is the set of $\Sigma_1 \cup \Sigma_2$-structures $\mathcal{A}$ such that the Σ_i-structure $\mathcal{A}^{\Sigma_i}$ is in $\mathbf{A}_i$ for $i = 1, 2$. When theories are defined as sets of closed formulas like in Sect. 6, the combination corresponds to the union of theories and so the union operator $\cup$ is used to combine them. We also use the union operator $\cup$ to denote the combination of theories defined as classes of structures, i.e., $T_1 \cup T_2 = (\Sigma_1 \cup \Sigma_2, \mathbf{A})$.

3 Absolutely Free Data Structures

The theory of Absolutely Free Data Structures (AFDS for short) [22] is convenient to capture usual constructor-based data structures, e.g. lists and trees.

Definition 1 (*Absolutely free data structures*) Consider a set of sorts $\texttt{Elem}$, and a sort $\texttt{struct} \notin \texttt{Elem}$. Let Σ be a signature whose set of sorts is $\{\texttt{struct}\} \cup \texttt{Elem}$ and whose function symbols $c \in \Sigma$ (called *constructors*) have arities of the form:

$$c : \sigma_1 \times \cdots \times \sigma_m \times \texttt{struct} \times \cdots \times \texttt{struct} \rightarrow \texttt{struct}$$

where $\sigma_1, \ldots, \sigma_m \in \texttt{Elem}$. 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

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

We do not consider yet selectors, e.g., *car* and *cdr* for lists. Handling selectors is easy with standard interpretations (of lists and trees) as discussed in Sect. 5, but requires some care with axiomatized theories (Sect. 6). Also notice that Definition 1 above is predicate-free, hence, every Σ-literal is either an equality or a disequality.

Example 1 The theory of lists is an example of AFDS where the constructors are $cons : \texttt{elem} \times \texttt{struct} \rightarrow \texttt{struct}$ and $nil : \texttt{struct}$. Similarly (binary) trees are also a classical AFDS example, with the constructor operators $cons : \texttt{elem} \times \texttt{struct} \times \texttt{struct} \rightarrow \texttt{struct}$ and $nil : \texttt{struct}$. The theory of pairs (of numbers) is another example of AFDS, with the constructor $cons : \texttt{num} \times \texttt{num} \rightarrow \texttt{struct}$. □

The theory AFDS has nice properties with respect to the satisfiability problem. Like any Horn theory, AFDS is convex [26]. Thanks to this, a satisfiability procedure modulo AFDS can consider separately the set of equalities, and each of the disequalities. Given an input set of flat literals divided into a set of equalities Γ and a set of disequalities Δ, the procedure works as follows:

1. It applies the rules in Fig. 1 on Γ exhaustively, to compute a solved form E. If $E = \bot$, then the procedure reports unsatisfiability. Otherwise, E is a set of equalities leading through variable replacement to an idempotent substitution μ.
2. The procedure reports unsatisfiability if there is some disequality $x \neq y \in \Delta$ such that $x\mu = y\mu$. Otherwise, it reports satisfiability.

As a side note, remark that AFDS is a Shostak theory [21], which means that it admits a solver (the syntactic unification procedure in Fig. 1) and a canonizer which is simply the identity. As illustrated above, a satisfiability procedure modulo a Shostak theory can be constructed by using both the solver and the canonizer [29]. Also, as usual for Shostak theories, equality entailment is easily checked by canonizing the output of the solver.

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 1 *Let $\varphi = \Gamma \cup \Delta$ be a set of flat Σ-literals with Γ and Δ respectively the sets of equalities and disequalities in φ. If φ is $AFDS_\Sigma$-satisfiable, then φ is satisfiable in an $AFDS_\Sigma$-interpretation $T(\Sigma \cup V)/=_E$, where V is the set of variables in φ and E is the solved form of Γ computed by the syntactic unification algorithm given in Fig. 1.*

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

4 The Combination Problem for Bridging Functions

Consider a many-sorted Σ_s-theory T_s (where s stands for source). In this paper, the set of sorts in Σ_s is $\{\texttt{struct}\} \cup \texttt{Elem}$ with $\texttt{struct} \notin \texttt{Elem}$, and Σ denotes the subsignature of Σ_s containing only the *constructor* symbols $c : \sigma_1 \times \cdots \times \sigma_n \to \texttt{struct}$, with $\sigma_1, \ldots, \sigma_n \in \{\texttt{struct}\} \cup \texttt{Elem}$. Similarly to Definition 1 and without loss of generality, we assume that each constructor in Σ is of the form $c : \sigma_1 \times \cdots \times \sigma_m \times \texttt{struct} \times \cdots \times \texttt{struct} \to \texttt{struct}$, with $\sigma_1, \ldots, \sigma_m \in \texttt{Elem}$. Given a tuple $\mathbf{e}$ of terms of sorts in $\texttt{Elem}$ and a tuple $\mathbf{t}$ of terms of sort $\texttt{struct}$, the tuple $\mathbf{e}, \mathbf{t}$ may be written $\mathbf{e}; \mathbf{t}$ to distinguish terms of sort $\texttt{struct}$ from the other ones, e.g. to denote a term $c(\mathbf{e}; \mathbf{t})$.

In addition to the Σ_s-theory T_s, we consider a Σ_t-theory T_t (where t stands for target) such that T_s and T_t are disjoint and the set of sorts shared by Σ_s and Σ_t are included in $\texttt{Elem}$. A bridging theory T_f connecting T_s to T_t is a set of equational axioms defining a bridging function f by structural induction over the constructors in Σ. A similar notion is sometimes called *catamorphism* in the literature, e.g., in [19].

Definition 2 (*Bridging theory*) 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 $\texttt{struct}$. A *bridging* function $f \notin \Sigma \cup \Sigma_t$ has arity $\texttt{struct} \to \texttt{t}$ where $\texttt{t}$ is a sort in Σ_t. A *bridging* theory T_f associated with a bridging function f has the form:

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

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

Observe that the notation $f_c(\mathbf{x}; \mathbf{y})$ does not enforce all elements of $\mathbf{x}; \mathbf{y}$ to occur in the term $f_c(\mathbf{x}; \mathbf{y})$. In particular $f_c(\mathbf{x}; \mathbf{y})$ may only depend on elements in $\mathbf{x}$ of sort in Σ_t. Or the bridging theory may be `Elem`-independent, in which case $f_c(\mathbf{x}; \mathbf{y})$ does not depend at all on $\mathbf{x}$. By definition, for any constant c in Σ there is an equality $f(c) = f_c$ in T_f and for simplicity f_c is assumed to be a constant in Σ_t. For instance, in the case of length of lists, $\ell(nil) = \ell_{nil} = 0$.

Example 2 (*Example 1 continued*). Many useful 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 these bridging theories, only the length of lists is `Elem`-independent. □

We introduce a combination method for a non-disjoint union of theories $T = T_s \cup T_f \cup T_t$ where the bridging theory T_f follows Definition 2. 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 (*Separate form*) A set of literals φ is in *separate form* if $\varphi = \varphi_s \cup \varphi_t \cup \varphi_f$ where:

- φ_s contains only Σ_s-literals such that its `struct`-sorted subterms only occur in flat literals;
- 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 $Var_{\texttt{struct}}(\varphi_s) = Var_{\texttt{struct}}(\varphi_f)$.

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

Example 3 Consider the theory of (acyclic) lists with a length function ℓ. The constructors of lists are $cons : \texttt{elem} \times \texttt{struct} \to \texttt{struct}$ and $nil : \texttt{struct}$, where `elem` is distinct from the sort for integers. Assume φ is the set of literals $\{x = cons(a, cons(b, z)),\ \ell(x)+1 = \ell(z)\}$. By purification, φ is transformed into the separate form $\varphi_s \cup \varphi_{int} \cup \varphi_\ell$ where:

- $\varphi_s = \{y = cons(b, z), x = cons(a, y)\}$,
- $\varphi_{int} = \{\ell_x + 1 = \ell_z\}$,
- $\varphi_\ell = \{\ell_x = \ell(x), \ell_y = \ell(y), \ell_z = \ell(z)\}$.

□

Unlike classical disjoint combination methods, it is not sufficient to guess just one arrangement on the shared variables to get a modular decision procedure. Notably it is necessary to include information derived from the bridging theory.

Definition 4 (*Combinable separate form*) Given a set of literals in separate form $\varphi = \varphi_s \cup \varphi_t \cup \varphi_f$ and two arrangements

- α over the variables of sorts in $\Sigma_s \cap \Sigma_t$ occurring in φ_s;
- α' over the variables of sort `struct` in φ_s;

the *combinable separate form* extending φ with α, α' is

$$\varphi(\alpha, \alpha') = (\varphi_s \cup \alpha' \cup \alpha) \cup (\varphi_t \cup \alpha \cup CP_E) \cup \varphi_f$$

where E is the set of Σ-equalities in $\varphi \cup \alpha \cup \alpha'$, and CP_E is the *target encoding* of E on the bridging theory T_f, defined as the set of Σ_t-literals

$$\begin{aligned} CP_E = &\{f_{x'} = f_c(\mathbf{e}; f_{x_1}, \ldots, f_{x_n}) \mid c(\mathbf{e}; x_1, \ldots, x_n) = x' \in E\} \\ &\cup \{f_{x'} = f_x \mid x =_{\mathtt{struct}} x' \in E\} \end{aligned}$$

Since φ is in separate form, the target encoding CP_E contains a Σ_t-equality for each `struct`-sorted equality in E. It results from the superposition of equalities in E into the axioms of T_f. Thus, CP_E can be viewed as a set of critical pairs, a classical notion used in the completion of term rewrite systems [4].

Example 4 Consider the separate form $\varphi_s \cup \varphi_{int} \cup \varphi_\ell$ from Example 3. For the arrangement $\alpha' = \{x \neq y \neq z\}$, the target encoding CP_E is $\{\ell_y = \ell_z + 1, \ell_x = \ell_y + 1\}$ where $E = \varphi_s$, and the corresponding combinable separate form is $(\varphi_s \cup \{x \neq y \neq z\}) \cup (\varphi_{int} \cup \{\ell_y = \ell_z + 1, \ell_x = \ell_y + 1\}) \cup \varphi_\ell$. For the arrangement $\alpha' = \{x = y, x \neq z\}$, the corresponding combinable separate form is $(\varphi_s \cup \{x = y, x \neq z\}) \cup (\varphi_{int} \cup \{\ell_x = \ell_y, \ell_y = \ell_z + 1, \ell_x = \ell_y + 1\}) \cup \varphi_\ell$. □

Proposition 2 *Any separate form is T-equivalent to a finite disjunction of combinable separate forms.*

Proof Let φ be a separate form. Consider all the finitely many possible arrangements α, α' as given in Definition 4. We have that $T \models \varphi \Leftrightarrow \bigvee_{\alpha,\alpha'}(\varphi \cup \alpha \cup \alpha')$. Let $\varphi(\alpha, \alpha') = \varphi \cup \alpha \cup \alpha' \cup CP_E$ where E is the set of Σ-equalities in $\varphi \cup \alpha \cup \alpha'$. By definition, $\varphi(\alpha, \alpha')$ is the combinable separate form extending φ by α, α'. Since $T \models (\varphi \cup \alpha \cup \alpha') \Rightarrow CP_E$, we have that $\varphi(\alpha, \alpha')$ is T-equivalent to $\varphi \cup \alpha \cup \alpha'$, and so $T \models \varphi \Leftrightarrow \bigvee_{\alpha,\alpha'} \varphi(\alpha, \alpha')$. □

From now on, we will only consider combinable separate forms and assume that a combinable separate form $\varphi_s \cup \varphi_t \cup \varphi_f$ includes $\alpha \cup \alpha'$ and $\alpha \cup CP_E$ respectively in φ_s and φ_t for some arrangements α, α'.

In Sect. 6, we investigate a class of source theories T_s (including $AFDS_\Sigma$) where the T-satisfiability of any combinable separate form φ can be checked in a modular way, by considering the T_s-satisfiability of φ_s and the T_t-satisfiability of φ_t. Notice that φ_f is not used when checking satisfiability: these constraints are indeed now encoded within φ_t, according to Definition 4. The proof of this modular result is given below for the particular case where T_s is $AFDS_\Sigma$. Even if it is subsumed by a similar proof presented for a more general case in Sect. 6, we believe it is interesting to provide a first simplified version in the case of AFDS.

Theorem 1 *Let $T = T_s \cup T_f \cup T_t$, where $T_s = AFDS_\Sigma$, T_t shares only sorts with T_s and T_f is a bridging theory. A combinable separate form $\varphi_s \cup \varphi_t \cup \varphi_f$ is T-satisfiable if and only if φ_s is T_s-satisfiable and φ_t is T_t-satisfiable.*

Proof The soundness (only-if direction) is obvious since T_s and T_t are included in T. To prove the completeness (if-direction), consider the set S of sorts shared by Σ_s and Σ_t, and the following sets of variables:

- $V = Var(\varphi_s)$,
- $V_{\mathtt{struct}} = Var_{\mathtt{struct}}(\varphi_s)$,

– $V_t = \{x \mid x \in Var_\sigma(\varphi_s \cup \varphi_t \cup \varphi_f), \sigma \text{ is a sort in } \Sigma_t\}$.

Note that $V \cap V_t$ is the set of S-sorted variables in φ_s.

According to Proposition 1, there exists a term-generated T_s-interpretation $\mathcal{H}$ satisfying φ_s such $\mathcal{H}^\Sigma$ is $T(\Sigma \cup V)/=_E$ where E is the finite set of flat equalities occurring in φ_s. Second, let $\mathcal{B}$ be a T_t-interpretation satisfying φ_t. Given $\mathcal{H}$ and $\mathcal{B}$, there exists another T_s-interpretation $\mathcal{A}$ satisfying φ_s such that

– $\mathcal{A}[\sigma]$ coincides with $\mathcal{B}[\sigma]$ 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 due to the arrangement α over $V \cap V_t$, and the fact that the sorts `Elem` in Σ_s can be considered as uninterpreted in T_s since there are no function symbols in Σ_s of arity $\sigma_1 \cdots \sigma_n \to \sigma$ with $\sigma \in$ `Elem`.

We are now ready to define an interpretation $\mathcal{M}$. First, we specify the domains. Let $\mathcal{M}[\sigma] = \mathcal{B}[\sigma]$ for any sort $\sigma \in \Sigma_t$ and $\mathcal{M}[\sigma] = \mathcal{A}[\sigma]$ for any $\sigma \in \Sigma_s$. Hence $\mathcal{M}$[`struct`] is $T_{\texttt{struct}}(\Sigma \cup V \cup D)/=_E$. We consider the following interpretation in $\mathcal{M}$:

– for each $u \in V_t$, $\mathcal{M}[u] = \mathcal{B}[u]$ and for each $x \in V$, $\mathcal{M}[x] = [\![x]\!]$; this is well-defined due to the arrangement α over $V \cap V_t$
– the interpretation of constructors $c \in \Sigma_s$ is defined on the equivalence classes in the usual way: $\mathcal{M}[c](\mathcal{M}[\mathbf{e}]; [\![t_1]\!], \ldots, [\![t_n]\!]) = [\![c(\mathbf{e}; t_1, \ldots t_n)]\!]$
– 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 $\mathcal{M}$[`struct`] as follows:
  – If it is the equivalence class of some $x \in V_{\texttt{struct}}$, then $\mathcal{M}[f]([\![x]\!]) = \mathcal{B}[f_x]$.
  – Otherwise, the equivalence class must consist of just one constructed element. If $[\![c(\mathbf{e}; t_1, \ldots t_n)]\!]$ is an equivalence class of this form, then

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

Now we need to show that $\mathcal{M}$ is a T-interpretation satisfying $\varphi_s \cup \varphi_t \cup \varphi_f$. The sets of literals φ_s and φ_t are clearly satisfied by $\mathcal{M}$, since 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 $x \in V_{\texttt{struct}}$, we have that $\mathcal{M}[f](\mathcal{M}[x]) = \mathcal{M}[f]([\![x]\!]) = \mathcal{B}[f_x] = \mathcal{M}[f_x]$, and so $\varphi_f = \bigcup_{x \in V_{\texttt{struct}}} \{f_x = f(x)\}$ is satisfied by $\mathcal{M}$.

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

– If we consider the equivalence class of some $x \in V_{\texttt{struct}}$, φ_t contains the literal $f_x = f_c(\mathbf{e}; f_{x_1}, \ldots, f_{x_n})$ if $x = c(\mathbf{e}; x_1, \ldots, x_n)$ occurs in φ_s. This literal is satisfied by $\mathcal{B}$, and since $\mathcal{M}[f]([\![v]\!]) = \mathcal{B}[f_v]$ for any $v \in V_{\texttt{struct}}$, 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.

□

For simplicity, the combination method given by Theorem 1 is presented in a non-deterministic way, guessing two arrangements α and α'. Since $AFDS_\Sigma$ is a convex theory, it is also possible to get a more deductive method by replacing the guessing of α' with the computation of all `struct`-sorted equalities between variables which are entailed by $\varphi_s \cup \alpha$. Then, the resulting (combined) deductive method is similar to the one obtained by applying

the locality approach [22,23]. The arrangement α is used to take into account the possible non-convexity of T_t. By assumption, T_t is not necessarily stably infinite with respect to the set of sorts `Elem`. This can be assumed without loss of completeness because the theory $AFDS_\Sigma$ is indeed polite with respect to `Elem` (cf. Sect. 6). Consequently, a satisfiability procedure for $AFDS_\Sigma \cup T_t$ can be obtained by applying a Nelson–Oppen combination method [13,20], when T_t is an arbitrary theory sharing only sorts in `Elem` with $AFDS_\Sigma$. In the same way, Theorem 1 provides a Nelson–Oppen combination method for $AFDS_\Sigma \cup T_t$ extended with some bridging theory T_f connecting $AFDS_\Sigma$ to T_t.

Example 5 Consider the theory of (acyclic) lists with a length function ℓ and the separate form $\varphi_s \cup \varphi_{int} \cup \varphi_\ell$ given in Example 3. Let α' be the only arrangement over the list variables such that $\alpha' \cup \varphi_s$ is satisfiable, i.e. $\{x \neq y \neq z\}$. By Definition 4, CP_E is $\{\ell_y = \ell_z + 1, \ell_x = \ell_y + 1\}$ since $E = \varphi_s$. The set $\varphi_s \cup \alpha'$ is satisfiable in the theory of lists. However $\varphi_{int} \cup CP_E$ 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 Sect. 5.

Example 6 Consider again the theory of (acyclic) lists with a length function ℓ, and the following set of literals $\varphi = \{x_1 = cons(d, y_1), x_2 = cons(d, y_2), x_1 \neq x_2 \neq y_1 \neq y_2 \neq y_3, \ell(y_2) = \ell(y_3)\}$. By purification, φ is transformed into the separate form $\varphi_s \cup \varphi_{int} \cup \varphi_\ell$ where:

- $\varphi_s = \{x_1 = cons(d, y_1), x_2 = cons(d, y_2), x_1 \neq x_2 \neq y_1 \neq y_2 \neq y_3\}$,
- $\varphi_{int} = \{\ell_{y_2} = \ell_{y_3}\}$,
- $\varphi_\ell = \{\ell_{x_1} = \ell(x_1), \ell_{x_2} = \ell(x_2), \ell_{y_1} = \ell(y_1), \ell_{y_2} = \ell(y_2), \ell_{y_3} = \ell(y_3)\}$.

Formula φ_s already includes the arrangement $\alpha' = \{x_1 \neq x_2 \neq y_1 \neq y_2 \neq y_3\}$. The target encoding is $CP_E = \{\ell_{x_1} = \ell_{y_1} + 1, \ell_{x_2} = \ell_{y_2} + 1\}$ since E is the set of equalities in φ_s. The set φ_s is satisfiable in the theory of lists. The set $\varphi_{int} \cup CP_E$ 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. According to the proof of Theorem 1, φ is satisfiable in a model $\mathcal{M}$ such that $\mathcal{M}[\texttt{struct}] = T_{\texttt{struct}}(\Sigma \cup V)\,/{=_E}$ for $\Sigma = \{cons, nil\}$ and $V = \{d, x_1, x_2, y_1, y_2, y_3\}$, $\mathcal{M}[\texttt{elem}] = \{[\![d]\!]\}$, and $\mathcal{M}[\texttt{int}] = \mathbb{Z}$. The function $\mathcal{M}[\ell] : \mathcal{M}[\texttt{struct}] \to \mathcal{M}[\texttt{int}]$ is defined recursively as follows:

- $\mathcal{M}[\ell]([\![x_1]\!]) = 4$, $\mathcal{M}[\ell]([\![x_2]\!]) = \mathcal{M}[\ell]([\![y_1]\!]) = 3$, $\mathcal{M}[\ell]([\![y_2]\!]) = \mathcal{M}[\ell]([\![y_3]\!]) = 2$, and $\mathcal{M}[\ell]([\![nil]\!]) = 0$;
- $\mathcal{M}[\ell]([\![cons(d, Y)]\!]) = 1 + \mathcal{M}[\ell]([\![Y]\!])$.

As another example, consider the combinable separate form $\psi = \psi_s \cup \psi_{int} \cup \psi_\ell$ where:

- $\psi_s = \{x_1 = cons(d, y_1), x_2 = cons(d, y_2), x_1 \neq x_2 \neq y_1 \neq y_2 \neq y_3\}$,
- $\psi_{int} = \{\ell_{y_2} = \ell_{y_3}, \ell_{x_1} = \ell_{y_1} + 1, \ell_{x_2} = \ell_{y_2} + 1, \ell_{y_2} = \ell_{y_1} - \ell_{x_1}\}$,
- $\psi_\ell = \{\ell_{x_1} = \ell(x_1), \ell_{x_2} = \ell(x_2), \ell_{y_1} = \ell(y_1), \ell_{y_2} = \ell(y_2), \ell_{y_3} = \ell(y_3)\}$.

Again, ψ_s is satisfiable in the theory of lists, and ψ_{int} is also satisfiable in the theory of linear arithmetic (over the integers), where necessarily $\ell_{x_2} = 0, \ell_{y_2} = -1, \ell_{y_3} = -1$. Thus, ψ is satisfiable in a model such that the range of ℓ includes necessarily -1 and ℓ maps both nil and x_2 to 0. Thus, this model does not correspond to a standard interpretation of lists, where the length of any list is necessarily positive and nil is the unique list whose length is 0. To avoid this kind of non-desirable models, we study in Sect. 5 the satisfiability problem in standard interpretations of lists. □

The example below discusses the case of lists over the integers where the sort of integers is shared by the theory of lists and the theory of integers.

Example 7 Consider the theory of (acyclic) lists over the integers with a length function ℓ. In that case, the constructors of lists are $cons : \texttt{int} \times \texttt{struct} \rightarrow \texttt{struct}$ and $nil : \texttt{struct}$, where `int` denotes the sort for integers. Assume the separate form $\varphi = \varphi_s \cup \varphi_{int} \cup \varphi_\ell$ where:

- $\varphi_s = \{x = cons(\ell_x, z), y = cons(\ell_y, z), x \neq y\}$,
- $\varphi_{int} = \emptyset$,
- $\varphi_\ell = \{\ell_x = \ell(x), \ell_y = \ell(y), \ell_z = \ell(z)\}$.

Let α' be the only arrangement over the list variables such that $\varphi_s \cup \alpha'$ is satisfiable, i.e. $\{x \neq y \neq z\}$. By Definition 4, CP_E is $\{\ell_x = \ell_z + 1, \ell_y = \ell_z + 1\}$. Let α be the only arrangement over the `int`-sorted variables in φ_s such that $\varphi_{int} \cup \alpha \cup CP_E$ is satisfiable, i.e. $\{\ell_x = \ell_y\}$. Then $\varphi_s \cup \alpha' \cup \alpha$ is unsatisfiable. Consequently, all combinable separate forms of φ are unsatisfiable, and so φ is unsatisfiable. □

5 Standard Interpretations

Now consider the satisfiability problem modulo data structure theories defined as classes of *standard* structures: each interpretation domain of `struct` contains only the (infinite set of) finite terms generated by the constructors and the elements in the interpretation domains of `Elem`. Thanks to this natural assumption on the domains, these standard structures faithfully capture the concept of the data structures, while remaining models of the (axiomatized) theories considered in previous sections. We investigate satisfiability procedures for standard structures based on the combination method of Sect. 4. We first study the case of lists, and then the general case of trees corresponding to the standard interpretations of absolutely free data structures. Both cases involve a theory of integers defined as follows.

Definition 5 (*Theory of integers*) Given a signature Σ_{int} including $\{0 : \texttt{int}, 1 : \texttt{int}, + : \texttt{int} \times \texttt{int} \rightarrow \texttt{int}, \leq : \texttt{int} \times \texttt{int}\}$, the theory of integers $T_{\mathbb{Z}}$ is $(\Sigma_{int}, \{\mathcal{A}\})$ where $\mathcal{A}$ is the Σ_{int} interpretation such that $\mathcal{A}[\texttt{int}] = \mathbb{Z}$ and symbols in Σ_{int} are interpreted according to their standard interpretation in $\mathbb{Z}$.

In the following, we assume the existence of a $T_{\mathbb{Z}}$-satisfiability procedure.

5.1 Lists with Length

Definition 6 (*Standard list-interpretation*) Consider a Σ_{int}-theory of integers $T_{\mathbb{Z}}$ as in Definition 5, a signature $\Sigma = \{cons : \texttt{elem} \times \texttt{struct} \rightarrow \texttt{struct}, nil : \texttt{struct}\}$ such that $\texttt{elem} \neq \texttt{int}$, and let $\Sigma_{list} = \Sigma \cup \{\ell : \texttt{struct} \rightarrow \texttt{int}\} \cup \Sigma_{int}$. A *standard list-interpretation* $\mathcal{A}$ is a Σ_{list}-interpretation satisfying the following conditions:

- $\mathcal{A}[\texttt{struct}] = (\mathcal{A}[\texttt{elem}])^*$ where $(\mathcal{A}[\texttt{elem}])^*$ is the set of all finite sequences $\langle e_1, \ldots, e_n \rangle$ for $n \geq 0$ and $e_1, \ldots, e_n \in \mathcal{A}[\texttt{elem}]$;
- $\mathcal{A}[nil] = \langle\rangle$;
- $\mathcal{A}[cons](e, \langle e_1, \ldots, e_n \rangle) = \langle e, e_1, \ldots, e_n \rangle$, for $n \geq 0$ and $e, e_1, \ldots, e_n \in \mathcal{A}[\texttt{elem}]$;
- $\mathcal{A}[\ell](\langle e_1, \ldots, e_n \rangle) = n$, and in particular $\mathcal{A}[\ell](\langle\rangle) = 0$;
- $\mathcal{A}^{\Sigma_{int}} \in T_{\mathbb{Z}}$.

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

Remark 1 The sorts `elem` and `int` in Definition 6 are distinct. As a consequence `elem` can be interpreted as an arbitrary set of elements, possibly finite or infinite. The case of lists over the integers is discussed in Remark 3.

Remark 2 Definition 6 does not mention selectors *car* and *cdr*. A standard list-interpretation $\mathcal{A}$ with selectors *car*, *cdr* would follow the additional conditions:

- $car_{\mathcal{A}}(\langle e_1, \ldots, e_n\rangle) = e_1$, for each $n > 0$ and $e_1, \ldots, e_n \in \mathcal{A}[\texttt{elem}]$;
- $cdr_{\mathcal{A}}(\langle e_1, \ldots, e_n\rangle) = \langle e_2, \ldots, e_n\rangle$, for each $n > 0$ and $e_1, \ldots, e_n \in \mathcal{A}[\texttt{elem}]$.

Thus, selectors *car* and *cdr* are defined only on non-empty lists, and can 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. For this reason, we have chosen to define standard interpretations of lists without selectors.

We show below that T^{si}_{list}-satisfiability can be reduced 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 $\Sigma = \{cons : \texttt{elem} \times \texttt{struct} \to \texttt{struct},\ nil : \texttt{struct}\}$, and $T_{\ell} = \{\forall e \forall y.\ \ell(cons(e, y)) = 1 + \ell(y),\ \ell(nil) = 0\}$. Since $T^{si}_{list} \models T_{list}$, a T^{si}_{list}-satisfiable formula is also T_{list}-satisfiable. However, a T_{list}-satisfiable formula is not necessarily T^{si}_{list}-satisfiable, as illustrated by the formula ψ in Example 6. To tackle this problem, the solution consists in guessing the different forms of standard lists, using the length function as an abstraction to denote empty and non-empty lists. Thanks to arithmetic constraints stating that each positive value of a length variable ℓ_x corresponds to the length of some finite list x, it is possible to guarantee the existence of a standard model. 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 $\rho = \{\rho(\ell_x) \mid \ell_x \in Var_{\texttt{int}}(\varphi_{\ell})\}$ where $\rho(\ell_x)$ is either $\ell_x = i$ $(0 \leq i < n)$ or $\ell_x \geq n$. A range constraint ρ is *feasible* for φ if $\varphi_{int} \cup \rho$ is $T_{\mathbb{Z}}$-satisfiable. The set $\mathcal{R}_n(\varphi)$ is defined as the set of all range constraints bounded by n that are feasible for φ. For instance, given the bound $n = 1$ and the formula ψ introduced in Example 6, it is easy to check that $\mathcal{R}_1(\psi) = \emptyset$. The bound $n = 1$ suffices to get a T^{si}_{list}-satisfiability procedure:

Proposition 3 *For any combinable separate form φ and any $\rho \in \mathcal{R}_1(\varphi)$, let $w(\varphi \wedge \rho)$ be the formula $\varphi \cup \rho \cup \{x = nil \mid \ell_x = 0 \in \rho\}$. Any combinable separate form φ is T^{si}_{list}-satisfiable iff there exists a feasible range constraint $\rho \in \mathcal{R}_1(\varphi)$ such that $w(\varphi \wedge \rho)$ is T_{list}-satisfiable.*

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

By using syntactic unification as in Sect. 3, $w(\varphi \wedge \rho)$ is equivalent to a set of literals φ' whose `struct` 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 `struct`-variables. For variables occurring in (2), the interpretation is obvious. The solved form φ' defines a (partial) ordering $>$ on variables occurring in (3): $x > y$ if $x = cons(d, y)$ occurs in φ'. Each minimal variable with respect to $>$ has a

length greater or equal than 1, otherwise it would occur in (2). For the minimal variables, we use the interpretation satisfying ρ 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 `struct`-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 $\varphi \wedge \rho$. □

Remark 3 The proof of Proposition 3 also holds when considering lists over the integers with the length function. Let $T_{list[\mathbb{Z}]}$ be the theory introduced in Example 7 and the related theory of standard interpretations $T^{si}_{list[\mathbb{Z}]}$ obtained from the definition of T^{si}_{list} by replacing `elem` with `int`. Just like in Proposition 3, satisfiability in $T^{si}_{list[\mathbb{Z}]}$ can be reduced to satisfiability in $T_{list[\mathbb{Z}]}$ since the domain of $\mathbb{Z}$ is infinite. Actually, the proof of Proposition 3 is perfectly suitable also in this case. After guessing range constraints bounded by $n = 1$, the combination method introduced in Sect. 4 can be applied to get a satisfiability procedure in $T_{list[\mathbb{Z}]}$.

The theory T^{si}_{list} can be divided in two complementary subtheories where the length function behaves completely differently:

1. the theory of lists built over only one element,
$$T^{=1}_{list} = T^{si}_{list} \cup \{\exists v : \texttt{elem}\ \forall x : \texttt{elem}.\ x = v\},$$
2. and the theory of lists built over at least two elements,
$$T^{\geq 2}_{list} = T^{si}_{list} \cup \{\exists v, v' : \texttt{elem}.\ v \neq v'\}.$$

The T^{si}_{list}-satisfiability problem can be obviously divided into two cases since a formula is T^{si}_{list}-satisfiable if and only if it is $T^{=1}_{list}$-satisfiable or $T^{\geq 2}_{list}$-satisfiable. In the singular case of $T^{=1}_{list}$, the length function ℓ is a bijection between the lists built over only one element and $\mathbb{N}$. Thus, a $T^{=1}_{list}$-satisfiability procedure can be easily obtained by adding to a combinable separate form $\varphi(\alpha, \alpha')$ the target constraint

$$\bigcup_{x \neq y \in \alpha'} \{\ell_x \neq \ell_y\} \cup \bigcup_{\ell_x \in Var_{\texttt{int}}(\varphi_\ell)} \{\ell_x \geq 0\}$$

to encode the bijectivity of ℓ. Then, such a separate form is $T^{=1}_{list}$-satisfiable if and only if its `int`-part is $T_{\mathbb{Z}}$-satisfiable. The problem of $T^{=1}_{list}$-satisfiability being solved, it remains now to study the $T^{\geq 2}_{list}$-satisfiability problem, where there are at least two elements in standard interpretations. The $T^{\geq 2}_{list}$-satisfiability problem can be solved by another finite and complete guessing of values for list lengths. Compared to the guessing used for T^{si}_{list}-satisfiability in Proposition 3, this new guessing leads to a $T^{\geq 2}_{list}$-satisfiability procedure with the property of being more generally combinable. It is a complete guessing because beyond a limit value n, that now depends on the input formula, there are enough different lists to build a $T^{\geq 2}_{list}$-model satisfying constraints of the form $x_1 \neq \cdots \neq x_m \wedge \ell(x_1) \geq n \wedge \cdots \wedge \ell(x_m) \geq n$.

Proposition 4 *For any set of literals φ in combinable separate form, there exists a bound n such that*

– *φ is $T^{\geq 2}_{list}$-equivalent to $\bigvee_{\rho \in \mathcal{R}_n(\varphi)}(\varphi \wedge \rho)$*

– *For any $\rho \in \mathcal{R}_n(\varphi)$, there exists a formula denoted by $witness(\varphi \wedge \rho)$ such that $\varphi \wedge \rho$ is $T_{list}^{\geq 2}$-equivalent to $(\exists \bar{v}) witness(\varphi \wedge \rho)$ for the set of variables $\bar{v} = Var(witness(\varphi \wedge \rho)) \setminus Var(\varphi \wedge \rho)$, and any $\{\texttt{elem}\}$-sorted arranged form of $witness(\varphi \wedge \rho)$ is $T_{list}^{\geq 2}$-satisfiable iff it is T_{list}-satisfiable.*

Proof Since φ is a combinable separate form, it implies a unique arrangement over `struct`-variables. Let m be the number of the corresponding equivalence classes over `struct`-variables. In order to have m different lists of length at least n, it is sufficient to define the bound n of range constraints as $n = \lceil \log_2(m) \rceil$. Let us now define the witness of a range constraint ρ:

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

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

Consider an arbitrary arrangement arr over the `elem`-variables occurring in $witness(\varphi \wedge \rho)$. Similarly to the proof of Proposition 3 and by using syntactic unification as in Sect. 3, if $witness(\varphi \wedge \rho) \wedge arr$ is T_{list}-satisfiable then it is possible to construct a T_{list}-equivalent set of literals φ' whose `struct`-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 $T_{list}^{\geq 2}$-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). As in the proof of Proposition 3, 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 `struct`-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 $T_{list}^{\geq 2}$-model of $witness(\varphi \wedge \rho) \wedge arr$. □

Example 8 Consider the $T_{list}^{\geq 2}$-satisfiability of the combinable separate form built in Example 6: $\varphi = \varphi_\ell \cup \{x_1 = cons(d, y_1), x_2 = cons(d, y_2), x_1 \neq x_2 \neq y_1 \neq y_2 \neq y_3, \ell_{x_1} = \ell_{y_1} + 1, \ell_{x_2} = \ell_{y_2} + 1, \ell_{y_2} = \ell_{y_3}\}$. The five distinct `struct`-variables imply that range constraints are bounded by $n = \lceil \log_2(5) \rceil = 3$. There are $4^5 = 1024$ possible range constraints since each variable can be equal to 0, 1, 2 or greater than or equal to 3. We now focus on a few feasible range constraints and their related witness, the remaining ones are handled similarly.

1. $\rho = \{\ell_{x_1} = \ell_{x_2} = 1, \ell_{y_1} = \ell_{y_2} = \ell_{y_3} = 0\}$. To obtain a witness of φ and ρ, 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. $\rho = \{\ell_{x_1} \geq 3, \ell_{y_1} = \ell_{x_2} = 2, \ell_{y_2} = \ell_{y_3} = 1\}$. A possible witness for this range constraint would comprise
   - $y_1 = cons(e'_{y_1}, cons(e_{y_1}, nil))$
   - $y_2 = cons(e_{y_2}, nil)$
   - $y_3 = cons(e_{y_3}, nil)$
   - $x_1 = cons(d, y_1) = cons(d, cons(e'_{y_1}, cons(e_{y_1}, nil)))$
   - $x_2 = cons(d, y_2) = cons(d, cons(e_{y_2}, nil))$

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

□

In the following section we will prove that the $T^{\geq 2}_{list}$-satisfiability procedure of Proposition 4 is useful for the combination of $T^{\geq 2}_{list}$ with an arbitrary theory for elements, whereas the T^{si}_{list}-satisfiability procedure of Proposition 3 is restricted to the combination of T^{si}_{list} with a stably infinite theory for elements.

5.2 Combining Lists with an Arbitrary Theory of Elements

We here show that $T^{\geq 2}_{list}$ 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 [13, 20]. By definition, a polite theory is both finitely witnessable and smooth.

Definition 7 (*Polite theory*) 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 $\kappa_i \geq |\mathcal{A}[\sigma_i]|$, for $i = 1, \ldots, n$,

there exists a T-model $\mathcal{B}$ of φ such that $|\mathcal{B}[\sigma_i]| = \kappa_i$ for $i = 1, \ldots, n$.

Given a Σ-theory T, 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 S-sorted arranged form ψ' of ψ, if ψ' is T-satisfiable then there exists a T-model $\mathcal{A}$ of ψ' such that $\mathcal{A}[\sigma] = \{\mathcal{A}[v] \mid v \in Var_\sigma(\psi')\}$, 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*(φ) is a finite witness of φ in T with respect to S.

A Σ-theory T is *polite* with respect to S if it is both smooth and finitely witnessable with respect to S.

Proposition 5 *$T_{list}^{\geq 2}$ is polite with respect to* $\{\texttt{elem}\}$.

Proof The smoothness of the theory of standard interpretations of lists has been shown in [20], and this is preserved when considering the length function. By definition of $T_{list}^{\geq 2}$, any set of elements can be used to build the lists (since ℓ is $\{\texttt{elem}\}$-independent). Hence, any $T_{list}^{\geq 2}$-satisfiable formula remains $T_{list}^{\geq 2}$-satisfiable when augmenting the domain of sort elem with new elements, and so $T_{list}^{\geq 2}$ is smooth.

To show the finite witnessability of $T_{list}^{\geq 2}$, consider the *witness* function defined for Proposition 4. For any combinable separate form φ, the formula

$$\bigvee_{\rho \in \mathcal{R}_n(\varphi)} witness(\varphi \wedge \rho)$$

is a finite witness of φ in $T_{list}^{\geq 2}$ with respect to $\{\texttt{elem}\}$. Indeed, the $T_{list}^{\geq 2}$-model built in the proof of Proposition 4 interprets the elem sort as the set of interpreted elem-variables occurring in that formula. □

Consider the satisfiability problem in the disjoint combination $T_{list}^{\geq 2} \cup T_{elem}$ where T_{elem} is a Σ_{elem}-theory sharing only the sort elem with $T_{list}^{\geq 2}$. Due to the politeness of $T_{list}^{\geq 2}$, we can directly use the combination method known for polite theories [13,20], and this leads to the following result.

Theorem 2 *Let T_{elem} be a Σ_{elem}-theory sharing only the sort* elem *with $T_{list}^{\geq 2}$. For any combinable separate form φ and any finite set φ_{elem} of Σ_{elem}-literals, the formula $\varphi \wedge \varphi_{elem}$ is $T_{list}^{\geq 2} \cup T_{elem}$-satisfiable iff there exists a range constraint $\rho \in \mathcal{R}_n(\varphi)$ and an arrangement arr such that (1) $witness(\varphi \wedge \rho) \wedge arr$ is T_{list}-satisfiable and (2) $\varphi_{elem} \wedge arr$ is T_{elem}-satisfiable, where $witness(\varphi \wedge \rho)$ is the formula defined for Proposition 4 and arr is an arrangement over the variables of sort* elem *in $witness(\varphi \wedge \rho)$.*

Proof It follows from the correctness proof of the combination method known for polite theories [13,20], using the finite witness of a combinable separate form given in the proof of Proposition 5.

To conclude the proof, Proposition 4 shows that $witness(\varphi \wedge \rho) \wedge arr$ is $T_{list}^{\geq 2}$-satisfiable iff it is T_{list}-satisfiable. □

Thus, the $T_{list}^{\geq 2} \cup T_{elem}$-satisfiability problem is NP-decidable if the T_{elem}-satisfiability problem is NP-decidable. Indeed, in the combination procedure of Theorem 2, the guessing of range constraints and the guessing of arrangements can be done in nondeterministic polynomial time; the *witness* function is computable in polynomial time; and the satisfiability problems in $AFDS_{\Sigma}$ and in $T_{\mathbb{Z}}$ are NP-decidable.

Example 9 Consider the combinable separate form of Example 8 and suppose we add a new literal stating that the sum of the lengths of y_1, y_2 and y_3 is 3, i.e., $\varphi = \varphi_\ell \cup \{x_1 = cons(d, y_1), x_2 = cons(d, y_2), x_1 \neq x_2 \neq y_1 \neq y_2 \neq y_3, \ell_{x_1} = \ell_{y_1} + 1, \ell_{x_2} = \ell_{y_2} + 1, \ell_{y_2} = \ell_{y_3}, \ell_{y_1} + \ell_{y_2} + \ell_{y_3} = 3\}$. Consider also the theory of elements $T_{elem} = \{\exists a, b : \texttt{elem}.\ a \neq b \wedge \forall x : \texttt{elem}.\ x = a \vee x = b\}$. As in Example 8, there are five struct-variables and so range constraints are bounded by $n = 3$. Among the 4^5 range constraints, most of them are not feasible. It is easy to see that any range constraint such that $\ell_{y_2} \geq 2$ is not feasible. There are only two feasible range constraints, obtained by considering $\ell_{y_2} = 0$ or $\ell_{y_2} = 1$:

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 $T_{list}^{\geq 2}$-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 $T_{list}^{\geq 2} \cup T_{elem}$-unsatisfiable. □

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

Proposition 6 *Let T_{elem} be a Σ_{elem}-theory sharing only the sort* elem *with T_{list}^{si} and such that T_{elem} is stably infinite with respect to* {elem}*. For any combinable separate form φ, and any finite set φ_{elem} of Σ_{elem}-literals, the formula $\varphi \wedge \varphi_{elem}$ is $T_{list}^{si} \cup T_{elem}$-satisfiable iff there exists a feasible range constraint $\rho \in \mathcal{R}_1(\varphi)$ and an arrangement arr such that (1) $w(\varphi \wedge \rho) \wedge arr$ is T_{list}-satisfiable and (2) $\varphi_{elem} \wedge arr$ is T_{elem}-satisfiable, where $w(\varphi \wedge \rho)$ is defined in Proposition 3 and arr is an arrangement over the variables in $Var(\varphi) \cap Var(\varphi_{elem})$.*

In the above proposition, arr is an arrangement over elem-sorted variables where $Var(\varphi) = Var(w(\varphi \wedge \rho))$.

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.

Definition 8 (*Standard tree-interpretation*) Consider a Σ_{int}-theory of integers $T_{\mathbb{Z}}$ as in Definition 5, a signature $\Sigma_{tree} = \Sigma \cup \{f : \texttt{struct} \rightarrow \texttt{int}\} \cup \Sigma_{int}$ where Σ is a signature as in Definition 1 with 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:

- $\mathcal{A}$[struct] is the set of Σ-terms of sort struct built with Elem-sorted elements in A;
- $\mathcal{A}[c] = c$ for each constant constructor $c \in \Sigma$;
- $\mathcal{A}[c](\mathbf{e}, t_1, \ldots, t_n) = c(\mathbf{e}, t_1, \ldots, t_n)$ for each non-constant constructor $c \in \Sigma$, tuple $\mathbf{e}$ of Elem-sorted elements in A, and $t_1, \ldots, t_n \in \mathcal{A}$[struct];
- $\mathcal{A}[f](c) = f_c$ for each constant constructor $c \in \Sigma$;
- $\mathcal{A}[f](c(\mathbf{e}, t_1, \ldots, t_n)) = f_c(\mathbf{e}, \mathcal{A}[f](t_1), \ldots, \mathcal{A}[f](t_n))$ for each non-constant constructor $c \in \Sigma$, tuple $\mathbf{e}$ of Elem-sorted elements in A, and $t_1, \ldots, t_n \in \mathcal{A}$[struct];
- $\mathcal{A}^{\Sigma_{int}} \in T_{\mathbb{Z}}$.

The *theory of (standard interpretations) of trees with bridging function* f is the pair $T_{tree}^{si} = (\Sigma_{tree}, \mathbf{A})$, 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 $AFDS_{\Sigma} \cup T_f \cup T_{\mathbb{Z}}$. In a way analogous to what has been done for lists (cf. Proposition 4), there is a method to reduce T^{si}_{tree}-satisfiability to T_{tree}-satisfiability. Similarly to lists, we introduce finite witnesses which can easily be computed when f is the height or the size of trees.

The next definition captures the assumptions used to extend the proof of Proposition 4 developed for lists to the case of trees. Let us first introduce some additional notations related to the range of the bridging function f. Given a theory T defined as a class of standard $tree$-structures and any $\mathcal{A} \in T$, let $F^{-1}_{\mathcal{A}}(n) = \{t \mid \mathcal{A}[f](t) = n\}$. By definition of T, the bridging theory T_f is `Elem`-independent. Consequently, the set $Ran(f) = \{n \mid F^{-1}_{\mathcal{A}}(n) \neq \emptyset\}$ remains identical for all $\mathcal{A} \in T$. A *range recognizer* is a Σ_{int}-formula $\nu = (\exists \tilde{j} \,.\, \nu')$ such that ν' is quantifier-free and ν has a unique free variable called the parameter of ν. When this parameter is instantiated in ν by some `int`-sorted term t, the resulting formula is denoted by $\nu(t)$.

Definition 9 (*Gently growing function*) Let T be a theory defined as a class of standard $tree$-structures. The bridging function f is *gently growing in* T if

1. there exists a range recognizer ν such that $Ran(f)$ is equal to the set $\{n \mid n \in \mathbb{N} \text{ and } T_{\mathbb{Z}} \models \nu(n)\}$;
2. for any $n, m \in Ran(f)$ and any $\mathcal{A} \in T$, $n \leq m \Longrightarrow |F^{-1}_{\mathcal{A}}(n)| \leq |F^{-1}_{\mathcal{A}}(m)|$;
3. there exists a computable function $b : \mathbb{N} \to \mathbb{N}$ such that for any $k > 1$ and any $\mathcal{A} \in T$, $|F^{-1}_{\mathcal{A}}(b(k))| \geq k$;
4. for any $n \in Ran(f)$, one can compute a finite non-empty set $F^{-1}(n)$ of terms with variables of sorts in `Elem` such that

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

Remark 4 Definition 9 strongly relates to the notion of sufficient surjectivity defined in [24] via two ingredients: a cardinality constraint together with a finite set of shapes. When f is gently growing in T and $Ran(f) = \mathbb{N}$, the sufficient surjectivity can be expressed as the T-valid formula

$$f(x) \geq b(k) \vee \left(\exists \bar{v}. \bigvee_{t \in S_k} x = t \right)$$

where $S_k = \bigcup_{0 \leq n < b(k)} F^{-1}(n)$ and $\bar{v} = Var(S_k)$, for any $k > 1$. According to this disjunctive formula, $n \geq b(k)$ plays the role of the cardinality constraint implying $|F^{-1}_{\mathcal{A}}(n)| \geq k$ for any $\mathcal{A} \in T$, and the remaining disjuncts provide the finite set of shapes. For simplicity, Definition 9(2) includes a monotonicity assumption that allows us to use $n \geq b(k)$ as a simple uniform cardinality constraint. Despite its name, sufficient surjectivity does not imply surjectivity. To overcome this problem, [19] advocates the need of an additional assumption stating that $Ran(f)$ can be given by a range recognizer in the target theory, namely ν in Definition 9(1). The case $\nu = (n \geq 0)$ is sufficient to consider the classical bridging functions discussed in the example below.

Example 10 Let us assume that $T_{\mathbb{Z}}$ denotes the theory of linear integer arithmetic extended with the max function. Consider $\Sigma = \{cons : \texttt{elem} \times \texttt{struct} \times \cdots \times \texttt{struct} \to \texttt{struct},\ nil : \texttt{struct}\}$, and the bridging theories corresponding to the size and the height of trees:

- $T_{sz} = \left\{ sz(cons(e, y_1, \ldots, y_m)) = 1 + \sum_{i=1}^{m} sz(y_i),\ sz(nil) = 0 \right\}$
- $T_{ht} = \left\{ ht(cons(e, y_1, \ldots, y_m)) = 1 + \max_{i \in [1,m]} ht(y_i),\ ht(nil) = 0 \right\}$

where m is the number of `struct` occurrences in the input sorts of *cons*.
If $m > 1$ then sz (resp. ht) is gently growing in T^{si}_{tree}, assuming $T_f = T_{sz}$ (resp. $T_f = T_{ht}$) in the definition of T^{si}_{tree}. To state this result, 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 [34] for the size of trees. Since $Ran(sz) = Ran(ht) = \mathbb{N}$, $\nu = (n \geq 0)$ is a suitable range recognizer for both sz and ht.
When $m = 1$ (i.e., $cons$: `elem` × `struct` → `struct`), sz and ht coincide with the length of lists ℓ. According to Sect. 5.2, ℓ is gently growing in $T^{\geq 2}_{list}$, that is, $T^{si}_{tree} \cup \{\exists v, v' :$ `elem`$.\ v \neq v'\}$. The length function ℓ is neither gently growing in $T^{=1}_{list}$ due to Definition 9(3), nor in T^{si}_{list} since $T^{=1}_{list} \subset T^{si}_{list}$. □

Definition 9 is general enough to encompass bridging functions which are not surjective between trees and the set of natural numbers, provided that a range recognizer is known. A simple motivating example is given below.

Example 11 Consider $\Sigma = \{cons$: `elem` × `struct` → `struct`, nil : `struct`$\}$, the bridging theory $T_f = \{\forall e \forall y.\ f(cons(e, y)) = 2 + f(y),\ f(nil) = 0\}$, and the corresponding theory T^{si}_{tree}. Let $T = T^{si}_{tree} \cup \{\exists v, v'$: `elem`$.\ v \neq v'\}$. In this theory T, the range of f is the set of even natural numbers, and f is gently growing, with ν, b and F^{-1} defined as follows:

- $\nu = (\exists j.\ n = 2j)$,
- $b(k) = 2 \log_2(k)$,
- $F^{-1}(0) = \{nil\}$ and for any strictly positive even number, $F^{-1}(n) = \{cons(e_1, \ldots cons(e_{n/2}, nil) \ldots)\}$.

In Definition 9, the use of a range recognizer requires a generalization of the notion of range constraint introduced in the particular case of lists with length.

Definition 10 (*Range constraint*) Assume f is gently growing in a theory T with ν as range recognizer. Given a set of literals φ in separate form and a natural number n, a *T-range constraint for φ bounded by n* is a Σ_{int}-formula $\rho = \bigwedge_{f_x \in Var_{\mathtt{int}}(\varphi_f)} \nu(f_x) \wedge \rho(f_x)$ where $\rho(f_x)$ is either $f_x = i$ $(0 \leq i < n)$ or $f_x \geq n$. A T-range constraint ρ is *feasible* for φ if $\varphi_{int} \wedge \rho$ is $T_{\mathbb{Z}}$-satisfiable. The set $R_{T,n}(\varphi)$ is defined as the set of all T-range constraints bounded by n that are feasible for φ.

In the rest of this section, T is clear from the context, and so a T-range constraint is simply called range constraint. Accordingly, $\mathcal{R}_{T,n}(\varphi)$ is abbreviated into $\mathcal{R}_n(\varphi)$.

We are now ready to generalize the proof of Proposition 4 where at least two elements are assumed. Like in $T^{\geq 2}_{list}$, some additional constraints on the minimal cardinality of elements also have to be considered. Given any sort σ in `Elem`, we define the theory $T^{\geq \kappa}_{\sigma}$ of at least κ elements of sort σ as follows: $T^{\geq \kappa}_{\sigma} = \{\exists v_1, \ldots, v_\kappa : \sigma.\ v_1 \neq \cdots \neq v_\kappa\}$ for any $\kappa \geq 2$, and $T^{\geq 1}_{\sigma} = \emptyset$. A *cardinality mapping* is a mapping $\boldsymbol{\kappa}$: `Elem` → $\mathbb{N}^+$. For any cardinality mapping $\boldsymbol{\kappa}$, let $T^{\geq \boldsymbol{\kappa}}_{\mathtt{Elem}} = \bigcup_{\sigma \in \mathtt{Elem}} T^{\geq \boldsymbol{\kappa}(\sigma)}_{\sigma}$, and $T^{\geq \boldsymbol{\kappa}}_{tree} = T^{si}_{tree} \cup T^{\geq \boldsymbol{\kappa}}_{\mathtt{Elem}}$. By definition, $T^{si}_{tree} = T^{\geq \mathbf{1}}_{tree}$ where $\mathbf{1}$ is the (lowest) cardinality mapping such that $\mathbf{1}(\sigma) = 1$ for any $\sigma \in$ `Elem`. According to Definition 6, $T^{\geq 2}_{list}$ corresponds to $T^{\geq \mathbf{2}}_{tree}$ such that `Elem` $= \{$`elem`$\}$, $\mathbf{2}$(`elem`$) = 2$, $\Sigma = \{cons$: `elem` × `struct` → `struct`, nil : `struct`$\}$, and f is the length ℓ.

Proposition 7 *Let κ be any cardinality mapping. Assume f is gently growing in $T_{tree}^{\geq\kappa}$. For any set of literals φ in combinable separate form, there exists a bound n such that*

- *φ is $T_{tree}^{\geq\kappa}$-equivalent to $\bigvee_{\rho\in\mathcal{R}_n(\varphi)}(\varphi\wedge\rho)$*
- *For any $\rho\in\mathcal{R}_n(\varphi)$, there exists a formula denoted by $witness(\varphi\wedge\rho)$ such that $\varphi\wedge\rho$ is $T_{tree}^{\geq\kappa}$-equivalent to $(\exists\bar{v})witness(\varphi\wedge\rho)$ for the set of variables $\bar{v}=Var(witness(\varphi\wedge\rho))\setminus Var(\varphi\wedge\rho)$, and any* `Elem`*-sorted arranged form of $witness(\varphi\wedge\rho)$ is $T_{tree}^{\geq\kappa}$-satisfiable iff it is T_{tree}-satisfiable.*

Proof By assumption, there exist computable functions b and F^{-1} as given in Definition 9. The proof of Proposition 4 can be adapted by using b and F^{-1}.

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 $n=b(m)$. Let us now define the witness of a range constraint ρ:

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

Then, we define $witness(\varphi\wedge\rho)$ as

$$\bigwedge_{\sigma\in\mathtt{Elem}}\left(\bigwedge_{v\in W_\sigma}(v=v)\wedge\bigwedge_{v,v'\in W_\sigma,v\neq v'}(v\neq v')\right)\wedge\varphi\wedge\rho\wedge witness_{rc}(\rho)$$

where W_σ denotes a set of $\kappa(\sigma)$ variables[1] of sort σ for any $\sigma\in$ `Elem`. The construction of a $T_{tree}^{\geq\kappa}$-interpretation is analogous to the construction given in Proposition 4 for lists, by using terms in $T_{\mathtt{struct}}(\Sigma,\bigcup_{\sigma\in\mathtt{Elem}}W_\sigma)$ corresponding to instances of terms in $\bigcup_{n\geq0}F^{-1}(n)$, instead of using *nil*-terminated lists.

Given a T_{tree}-satisfiable `Elem`-sorted arranged form of $witness(\varphi\wedge\rho)$, let φ' be the equivalent formula obtained as in Proposition 4 by solving the `struct`-sorted 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. The interpretation is thus a $T_{tree}^{\geq\kappa}$-model of φ', or equivalently, of the given `Elem`-sorted arranged form of $witness(\varphi\wedge\rho)$. □

Since the finite witnessability and smoothness proofs for $T_{list}^{\geq2}$ can be directly extended to $T_{tree}^{\geq\kappa}$, the following politeness result holds for $T_{tree}^{\geq\kappa}$, as well as for T_{tree}^{si}.

Proposition 8 *Let κ be any cardinality mapping. If f is gently growing in $T_{tree}^{\geq\kappa}$, then $T_{tree}^{\geq\kappa}$ is polite with respect to* `Elem`*.*

Proof By definition of $T_{tree}^{\geq\kappa}$, any set of elements can be used to build the trees, provided that, for each $\sigma\in$ `Elem`, the number of σ-sorted elements is greater or equal than $\kappa(\sigma)$.

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

Hence, any $T_{tree}^{\geq\kappa}$-satisfiable formula remains $T_{tree}^{\geq\kappa}$-satisfiable when adding elements of sorts in $\texttt{Elem}$, and so $T_{tree}^{\geq\kappa}$ is smooth.

To show the finite witnessability of $T_{tree}^{\geq\kappa}$, we can use the *witness* function defined for Proposition 7. For any combinable separate form φ, consider the disjunction $\bigvee_{\rho\in\mathcal{R}_n(\varphi)} witness(\varphi\wedge\rho)$. For this disjunction, one can observe that the $T_{tree}^{\geq\kappa}$-model built in the proof of Proposition 7 interprets each $\sigma \in \texttt{Elem}$ as the set of its interpreted σ-sorted variables. Thus, this disjunction is a finite witness of φ in $T_{tree}^{\geq\kappa}$ with respect to $\texttt{Elem}$. □

Theorem 2 (for lists) can be generalized to trees:

Theorem 3 *Assume f is gently growing in $T_{tree}^{\geq\kappa}$. Let T_{elem} be a Σ_{elem}-theory sharing only the sorts in $\texttt{Elem}$ with $T_{tree}^{\geq\kappa}$. For any combinable separate form φ and any finite set φ_{elem} of Σ_{elem}-literals, the formula $\varphi \wedge \varphi_{elem}$ is $T_{tree}^{\geq\kappa} \cup T_{elem}$-satisfiable iff there exists a range constraint $\rho \in \mathcal{R}_n(\varphi)$ and an arrangement arr such that (1) $witness(\varphi \wedge \rho) \wedge arr$ is T_{tree}-satisfiable and (2) $\varphi_{elem} \wedge arr$ is T_{elem}-satisfiable, where $witness(\varphi \wedge \rho)$ is the formula defined for Proposition 7 and arr is an arrangement over the $\texttt{Elem}$-sorted variables in $witness(\varphi \wedge \rho)$.*

Proof In a way similar to Theorem 2, the combination method known for polite theories [13,20] can be applied to $T_{tree}^{\geq\kappa} \cup T_{elem}$, using the finite witness of a combinable separate form given in the proof of Proposition 8. Then, Proposition 7 shows that $witness(\varphi \wedge \rho) \wedge arr$ is $T_{tree}^{\geq\kappa}$-satisfiable iff it is T_{tree}-satisfiable. □

Consequently, the $T_{tree}^{\geq\kappa} \cup T_{elem}$-satisfiability problem can be shown NP-decidable if, in the related combination procedure, the guessing of range constraints can be done in nondeterministic polynomial time; the *witness* function is computable in polynomial time; and the T_{elem}-satisfiability problems is NP-decidable. For example, it is easy to see that the first two conditions can be met in the particular case of $T_{list}^{\geq 2}$. Even if it seems difficult to get witness functions computable in polynomial time in the general case, NP-decidability is easy to prove in some more noteworthy cases. Consider a theory T_{tree}^{si} as in Example 10 where $T_f = T_{sz}$ or $T_f = T_{ht}$. Similarly to Proposition 3, T_{tree}^{si}-satisfiability reduces to T_{tree}-satisfiability by guessing only range constraints bounded by $n = 1$ and by computing the corresponding formulas thanks to the function w. In the related decision procedure, the guessing of range constraints can be done in nondeterministic polynomial time and the function w is computable in polynomial time. Thus the T_{tree}^{si}-satisfiability problem is NP-decidable. Similarly to Proposition 6, the combination procedure for $T_{tree}^{si} \cup T_{elem}$-satisfiability is simpler when T_{elem} is stably infinite with respect to $\texttt{Elem}$. Again, the form of this procedure shows that the $T_{tree}^{si} \cup T_{elem}$-satisfiability problem is NP-decidable if the T_{elem}-satisfiability problem is NP-decidable. When T_{elem} is not stably infinite with respect to $\texttt{Elem}$, the sizes of witnesses might explode with a negative impact on complexity.

Example 12 Consider the theory of standard interpretations of (binary) trees T_{tree}^{si} with constructors $\Sigma = \{cons : \texttt{elem} \times \texttt{struct} \times \texttt{struct} \rightarrow \texttt{struct},\ nil : \texttt{struct}\}$, the size function sz as defined in Example 10, and the theory of elements with only one object: $T_{elem} = \{\forall x : \texttt{elem}.\ x = d\}$. Let $\varphi = \{x_1 = cons(d, y, nil), x_2 = cons(d, nil, y), sz(x_3) \leq 2, sz(y) = 1, x_1 \neq x_2 \neq x_3 \neq y \neq nil\}$. The combinable separate form of φ is as follows:

- $\varphi_s = \{x_1 = cons(d, y, z), x_2 = cons(d, z, y), z = nil, x_1 \neq x_2 \neq x_3 \neq y \neq z\}$
- $\varphi_{int} = \{sz_{x_1} = 1 + sz_y + sz_z, sz_{x_2} = 1 + sz_z + sz_y, sz_{x_3} \leq 2, sz_y = 1, sz_z = 0\}$
- $\varphi_{sz} = \{sz_{x_1} = sz(x_1), sz_{x_2} = sz(x_2), sz_{x_3} = sz(x_3), sz_y = sz(y), sz_z = sz(z)\}$

In [34], Catalan numbers are used to get additional counting constraints for the particular case where the trees are generated by finitely many constants. In our setting, Catalan numbers

are also applicable to get the bound of range constraints. The n-th Catalan number is the number of structurally different trees with n nodes, this is, the amount of different trees that can be built with one element. The formula to compute the n-th Catalan number is given by $C_n = \frac{1}{n+1} \times \binom{2n}{n}$. To find the bound, we look for the first n such that C_n is greater than the number of different trees in the formula. Our combinable separate form involving five different trees, we use $n = 3$ since $C_2 = 2$ and $C_3 = 5$. There are $4^5 = 1024$ possible range constraints, since each variable can be equal to 0, 1, 2 or greater than or equal to 3. However only three of them are feasible:

- $sz_{x_1} = sz_{x_2} = sz_{x_3} = 2, sz_y = 1, sz_z = 0$. It is T^{si}_{tree}-unsatisfiable: $y = cons(d, nil, nil)$, thus x_1 and x_2 are the two possible trees of size 2, whereas x_3 should also be a tree of size 2, different to both x_1 and x_2.
- $sz_{x_1} = sz_{x_2} = 2, sz_{x_3} = sz_y = 1, sz_z = 0$. It is T^{si}_{tree}-unsatisfiable since $x_3 = y = cons(d, nil, nil)$.
- $sz_{x_1} = sz_{x_2} = 2, sz_{x_3} = 0, sz_y = 1, sz_z = 0$. It is T^{si}_{tree}-unsatisfiable since $x_3 = z = nil$.

Hence φ is T^{si}_{tree}-unsatisfiable.

Assume the signature Σ includes an additional constant of sort `struct`, say a, such that $sz(a) = 0$. Since there are now more trees of the same size, the same bound n still works even it is not optimal. Then, the formula φ becomes T^{si}_{tree}-satisfiable by considering for instance the feasible range constant $sz_{x_1} = sz_{x_2} = 2, sz_{x_3} = 0, sz_y = 1, sz_z = 0$ which leads to a satisfiable witness, e.g., $y = cons(d, nil, nil)$, $z = nil$, $x_3 = a$, $x_2 = cons(d, nil, cons(d, nil, nil))$, and $x_1 = cons(d, cons(d, nil, nil), nil)$.

Back to the original signature Σ including only the nil constant, let us now consider the same formula but using the height function instead of the size: $\varphi' = \{x_1 = cons(d, nil, y), x_2 = cons(d, y, nil), ht(x_3) \leq 2, ht(y) = 1, x_1 \neq x_2 \neq x_3 \neq y \neq nil\}$. The combinable separate form remains the same (replacing sz by ht) except for φ'_{int} which becomes $\{ht_{x_1} = 1 + max(ht_y, ht_z), ht_{x_2} = 1 + max(ht_z, ht_y), ht_{x_3} \leq 2, ht_y = 1, ht_z = 0\}$. The number of trees with height n (built with only one element) can be computed using a simple formula: $H_n = H^2_{n-1} + 2 \times H_{n-1} \times (\sum_{i=0}^{n-2} H_i)$, where $H_1 = H_0 = 1$. With five different trees (i.e., x_1, x_2, x_3, y, z), and since $H_2 = 3$ and $H_3 = 21$, we use $n = 3$. From the possible range constraints, the following ones are feasible:

- $ht_{x_1} = ht_{x_2} = 2, ht_y = 1, ht_z = 0$ with $ht_{x_3} = 0$ or $ht_{x_3} = 1$. These are not T^{si}_{tree}-satisfiable for reasons analogous to the size case.
- $ht_{x_1} = ht_{x_2} = ht_{x_3} = 2, ht_y = 1, ht_z = 0$. Considering the witness $x_3 = cons(d, cons(d, nil, nil), cons(d, nil, nil))$, it is T^{si}_{tree}-satisfiable.

Therefore φ' is T^{si}_{tree}-satisfiable. □

6 Axiomatized Data Structures

The previous section focused on standard theories, that is, theories corresponding precisely to standard data structures such as finite lists and trees. This involves restricting the class of models of AFDS, axiomatized in Sect. 3. It is also sometimes useful to weaken AFDS to a subset of its axioms. We now consider again axiomatized theories as a mean to represent data structures. The signature not only comprises constructors, but may also include defined symbols, e.g. selectors, bridging functions, or functions over elements. In the end of the section, we show that this class of theories is stable with respect to combination with theories for elements or bridging functions. The related combination procedure will be the

one presented in Sect. 4 for AFDS, and so it is not based on a guessing of range constraints as for standard interpretations. In the theories considered below, any disequality between variables, say $x \neq y$, is indeed easy to satisfy due to the existence of a term-generated model where x and y are interpreted by different free constants.

6.1 Data Structure Theories

In this section, we now investigate the possibility to consider source theories axiomatized by some of the axioms of AFDS. In addition, we allow selectors and the related projection axioms.

Definition 11 (*Data structure signature with free sorts*) Consider a set of sorts `Elem` and a disjoint sort `struct`. A *data structure signature* Σ_s is a signature on $\{\texttt{struct}\} \cup \texttt{Elem}$, including (but not restricted to)

- Σ, the signature of *constructor* symbols $c : \sigma_1 \times \cdots \times \sigma_n \to \texttt{struct}$, with $\sigma_1, \ldots, \sigma_n \in \{\texttt{struct}\} \cup \texttt{Elem}$;
- Σ', the signature of *selector* symbols $s_i^c : \texttt{struct} \to \sigma_i$, with $c : \sigma_1 \times \cdots \times \sigma_n \to \texttt{struct}$ a non-constant constructor in Σ.

A sort of `Elem` is *free* (in Σ_s) if it occurs only in the arities of functions symbols in $\Sigma \cup \Sigma'$. Given a signature Σ_t sharing only sorts with Σ_s, a consistent Σ_t-theory *shares only free sorts with* a consistent Σ_s-theory if the sorts shared by Σ_s and Σ_t are free in Σ_s.

To simplify the notation, we assume from now on that constructor symbols $c \in \Sigma$ have arities $c : \sigma_1 \times \cdots \times \sigma_m \times \texttt{struct} \times \cdots \times \texttt{struct} \to \texttt{struct}$ as in Definition 1, that is, all `Elem`-sorted arguments come first.

Notice that among all selectors corresponding to the constructors, a data structure signature may only contain a few of them. Furthermore a data structure signature Σ_s may contain, in addition to constructors in Σ and selectors in Σ', various other symbols, e.g. operators on elements and bridging functions: in general, $\Sigma \cup \Sigma' \subseteq \Sigma_s$. This section provides the necessary tools to build new expressive data structure theories by combination, starting with a simple data structure theory with only free sorts.

Definition 12 (*Data structure theories*) Consider a data structure signature Σ_s and the set of axioms

$$Proj_{\Sigma'} = \left\{ \forall x_1 \ldots x_n.\ s_i^c(c(x_1, \ldots, x_n)) = x_i \mid s_i^c \in \Sigma' \right\}$$

The class of Data Structure Theories $\mathbf{DST}^+$ comprises all theories T_s such that T_s is the union of $Proj_{\Sigma'}$ and any subset of axioms among Inj_c, $Dis_{c,d}$, $Acyc_\Sigma$ as given in Definition 1.

The class $\mathbf{DST}^+$ includes theories of practical interest worth considering for non-disjoint combinations with bridging functions. It contains the theory of Absolutely Free Data Structures, possibly with selectors, but also, for instance, the theory of equality, or simply, injective functions. It appears that those theories satisfy a model-theoretic property instrumental to prove the completeness of the combination procedure. They admit some particular Herbrand models similar to the ones we can build for the theory of equality. This property captures data structure theories that can be somehow reduced to the theory of equality. One could alternatively use the locality approach [23] to get a reduction to the theory of equality through a finite instantiation of axioms. But our model-based approach eases the construction of a model for data structures extended with bridging functions.

Rather than considering the satisfiability of a set of literals modulo a theory, we explore, in an equivalent way, the consistency of the theory extension including the set of (ground) literals. This however requires to extend the signature with free constants. We focus on particular formulas that will be witnessable in a way similar to Definition 7.

Definition 13 (*Witnessable extension*) Consider a data structure signature Σ_s including the signature Σ of constructors and the signature Σ' of selectors, together with a finite set of constants C such that $\Sigma_s \cup C$ is a constant expansion of Σ_s. Given a Σ_s-theory T_s, a *witnessable T_s-extension* is a $\Sigma_s \cup C$-theory $T_s \cup G$ where

- C includes a constant of sort σ for each sort σ in $\mathtt{Elem} \cup \{\mathtt{struct}\}$;
- G is a finite set of ground $\Sigma_s \cup C$-literals such that its $\mathtt{struct}$-sorted subterms only occur in flat literals.

By flattening, any finite set of Σ_s-literals φ is T_s-equivalent to a sentence $(\exists \bar{v})G$, where $T_s \cup G$ corresponds to a witnessable T_s-extension, considering variables in G as free constants. We focus on theories admitting models defined on structures of terms generated by some constructors and the free constants occurring in $T_s \cup G$.

The model-theoretic properties of $\mathbf{DST}^+$ theories are essential for combinations: models can be generated from some of their symbols (i.e., the constructors). The following definition captures these properties:

Definition 14 (*Polished theory*) A consistent Σ_s-theory T_s is *polished* if, for any witnessable T_s-extension $T_s \cup G$ on signature $\Sigma_s \cup C$, we have:

(i) If $T_s \cup G$ is consistent, it admits a model $\mathcal{H}$ such that $\mathcal{H}^{\Sigma \cup C}$ is $T(\Sigma \cup C \cup D)/=_E$ where E is a finite set of ground flat $\Sigma \cup C$-equalities defined as the set of $\Sigma \cup C$-equalities in G plus some additional equalities between constants of C occurring in G, and D is a set of fresh elements of non-free $\mathtt{Elem}$-sorts of Σ_s. Such a model $\mathcal{H}$ is called a *basic Herbrand model*.

(ii) For any sets D and D' of fresh elements, respectively of non-free $\mathtt{Elem}$-sorts and free sorts (in Σ_s), if $T_s \cup G$ admits a basic Herbrand model $\mathcal{H}_1$ such that $\mathcal{H}_1^{\Sigma \cup C}$ is $T(\Sigma \cup C \cup D)/=_E$, then it also admits a model $\mathcal{H}_2$ such that $\mathcal{H}_2^{\Sigma \cup C}$ is $T(\Sigma \cup C \cup D \cup D')/=_E$.

A basic Herbrand model is constructed on the subsignature Σ of T_s. This introduces a natural distinction between *constructors* in Σ and defined symbols in $\Sigma_s \setminus \Sigma$. The constructors build the domain of the basic Herbrand model, while the defined symbols are interpreted as operators on this domain. For instance, the defined symbols in AFDS are the selectors. The motivation for such a definition is to capture the fact that the theory is tolerant to combinations with other theories, and in particular, that there is some freedom for choosing domains with arbitrary cardinality for free $\mathtt{Elem}$-sorts. Polished theories are indeed polite, but they furthermore have the property that the domain is term-generated modulo the set E of $\Sigma \cup C$-equalities in G when G corresponds to a S-sorted arranged form for the set S of free sorts in $\mathtt{Elem}$.

Proposition 9 *Polished theories are polite with respect to the set of free sorts in* $\mathtt{Elem}$.

Proof Consider a polished Σ_s-theory T_s (cf. Definition 14), and let S be the set of sorts that are free in Σ_s. Assume any set φ of Σ_s-literals such that, considering variables of φ as free constants, $T_s \cup \varphi$ can be viewed as a witnessable T_s-extension. For any S-sorted arranged form φ' of φ, $T_s \cup \varphi'$ corresponds to a witnessable T_s-extension which is consistent if and

only if φ' is T_s-satisfiable. By Definition 14(i), if φ' is T_s-satisfiable, then there exists a T_s-interpretation $\mathcal{H}$ of φ' such that $\mathcal{H}[\sigma] = \bigcup_{v \in Var_\sigma(\varphi')} \mathcal{H}[v]$ for each $\sigma \in S$. Hence, φ is a finite witness of itself in T_s with respect to S.

Smoothness with respect to S is a consequence of Definition 14(ii). □

The set E of equalities used to build a basic Herbrand model can be obtained by guessing or in a deductive way. In a guessing approach, the set E would be maximal (obtained from an arrangement) and in that case no additional equality would be needed. An attractive approach is to get directly the right E thanks to a deductive process. This is possible for theories T_s in $\mathbf{DST}^+$ since a saturation-based calculus (see Fig. 2) provides a T_s-satisfiability procedure together with a saturated set of literals including E and yielding a basic Herbrand model.

Proposition 10 *Theories in* $\mathbf{DST}^+$ *are polished.*

Proof Consider first the finite witnessability property given in Definition 14(i). Assume T_s is any Σ_s-theory in $\mathbf{DST}^+$ and $\mathfrak{T}_s = T_s \cup G$ is any witnessable T_s-extension. In that case, $\Sigma_s = \Sigma \cup \Sigma'$.

To check the consistency of $\mathfrak{T}_s$, we can use a (simplified) superposition calculus. It can be viewed as an abstract congruence closure procedure for the theory of equality extended with additional inference rules to take into account the axioms in T_s. In Fig. 2, we provide a version of this calculus instantiated for the case of $AFDS^+ = AFDS_\Sigma \cup Proj_{\Sigma'}$. This inference system is parameterized by an ordering $>$ on constants in C. Notice that there is a one to one correspondence between the axioms of $AFDS^+$ and inference rules of this calculus. A satisfiability procedure for a theory comprising only a subset of $AFDS^+$ is simply obtained by removing the inference rules corresponding to the missing axioms. For instance, if we omit $\mathbf{Inj}_c$, $\mathbf{Dis}_{c,d}$, $\mathbf{Acyc}_\Sigma$ and $\mathbf{Proj}_{c,i}$, we retrieve the inference system for the satisfiability problem in the theory of equality.

Given the input G, the calculus terminates, and computes a finite saturated set of flat literals, say G_*. If G_* does not contain the empty clause, the theory $\mathfrak{T}_s = T_s \cup G$ is consistent. Along the lines of the model-generation technique for superposition calculi [5], the set of equalities in G_* defines a convergent term rewrite system R helpful to build a model. Formally, let $R = \{c_1 \to c_2 \mid c_1 = c_2 \in G_*,\ c_1, c_2 \in C, c_1 > c_2\} \cup \{f(c_1, \ldots, c_n) \to c_{n+1} \mid f(c_1, \ldots, c_n) = c_{n+1} \in G_*\}$.

We consider the structure $\mathcal{H}_1$ of R-normal forms in $T(\Sigma \cup C)$ together with an interpretation of selectors in Σ', and H_1 the domain of $\mathcal{H}_1$. By Definition 13, it is possible to choose, for each sort $\sigma \in \texttt{Elem} \cup \{\texttt{struct}\}$, an arbitrary but fixed constant $\mathfrak{u}_\sigma \in C_\sigma$ in R-normal form. Using this constant $\mathfrak{u}_\sigma$, any selector $s_i^c : \texttt{struct} \to \sigma$ in Σ' is interpreted in $\mathcal{H}_1$ as follows:

- For any `struct`-sorted normal form which is a constant $x \in C$, $\mathcal{H}_1[s_i^c](x) = x'$ if $s_i^c(x) \downarrow_R = x' \in C$, otherwise, $\mathcal{H}_1[s_i^c](x) = \mathfrak{u}_\sigma$.
- For any `struct`-sorted normal form which is a term $t = c(t_1, \ldots, t_n)$, $\mathcal{H}_1[s_i^c](t) = t_i$.
- For any other `struct`-sorted normal form t, $\mathcal{H}_1[s_i^c](t) = \mathfrak{u}_\sigma$.

To show that $\mathcal{H}_1 \models T_s$, consider the set of constants $C_G \subseteq C$ occurring in G which are R-normal forms. By definition, $C_G \subseteq H_1$. We can check that:

- For any axiom ψ in T_s and any assignment in H_1 such that all terms in ψ are assigned to values in C_G, ψ evaluates to true in $\mathcal{H}_1$. Otherwise, it would contradict that G_* is saturated.
- For any axiom ψ in $AFDS_\Sigma \cup Proj_{\Sigma'}$ and any assignment in H_1 such that some term in ψ is assigned to a value in $H_1 \backslash C_G$, ψ evaluates to true in $\mathcal{H}_1$.

$$
\begin{array}{ll}
\textbf{Sup}: & x = x', x = y \vdash x' = y \quad \text{if } x > x', x > y \\
\textbf{Cong1}: & x_j = x'_j, x = f(\ldots, x_j, \ldots) \vdash x = f(\ldots, x'_j, \ldots) \quad \text{if } x_j > x'_j \\
\textbf{Cong2}: & x = f(x_1, \ldots, x_n), x' = f(x_1, \ldots, x_n) \vdash x = x' \\
\textbf{Param}: & x = x', x \neq y \vdash x' \neq y \quad \text{if } x > x', x > y \\
\textbf{Ref}: & x \neq x \vdash \bot \\
\textbf{Inj}_c: & x = c(x_1, \ldots, x_n), x = c(x'_1, \ldots, x'_n) \vdash x_1 = x'_1 \ldots x_n = x'_n \ \text{if } c \in \Sigma \\
\textbf{Dis}_{c,d}: & x = c(x_1, \ldots, x_n), x = d(y_1, \ldots, y_m) \vdash \bot \ \text{if } c, d \in \Sigma, c \neq d \\
\textbf{Acyc}_\Sigma: & x = t_1[x_1], \ldots, x_{n-1} = t_n[x] \vdash \bot \ \text{if } t_1, \ldots, t_n \text{ are } \Sigma\text{-terms of depth 1} \\
\textbf{Proj}_{c,i}: & x = c(x_1, \ldots, x_n) \vdash x_i = s_i^c(x)
\end{array}
$$

Fig. 2 T_s-satisfiability procedure

Since T_s is included in $AFDS_\Sigma \cup Proj_{\Sigma'}$, all axioms in T_s evaluate to true for any assignment in H_1, i.e., $\mathcal{H}_1 \models T_s$. Clearly, $\mathcal{H}_1 \models G$, and so $\mathcal{H}_1$ is a model of $\mathfrak{T}_s = T_s \cup G$.

It remains to introduce an equational theory E that follows Definition 14. Let E be the set of $\Sigma \cup C$-equalities in G plus the set of C-equalities in G_*. It is easy to check that for any $\Sigma \cup C$-terms s, t, we have $s =_E t$ if and only if $s\downarrow_R = t\downarrow_R$:

- Assume $s =_E t$. Since $E \subseteq G_*$, we have $s =_R t$ and so $s\downarrow_R = t\downarrow_R$.
- Conversely, any rule $l \to r$ in R used to R-normalize s and t is such that $l =_E r$, and so $s\downarrow_R = t\downarrow_R$ implies $s =_E t$.

Consequently, the model $\mathcal{H}_1$ constructed above is indeed in the desired form.

Consider now the (smoothness) property given in Definition 14(ii). Since all `Elem`-sorts are free in T_s, there are no fresh elements of non-free `Elem`-sorts. Hence, the set D in Definition 14(ii) is empty. So we have to show the existence of a model $\mathcal{H}_2$ of $\mathfrak{T}_s$ such that its $\Sigma \cup C$-reduct $\mathcal{H}_2^{\Sigma\cup C}$ is $T(\Sigma \cup C \cup D')/=_E$, where D' is any set of fresh elements of `Elem`-sorts. Let $\mathcal{H}_2$ be the $(\Sigma' \cup \Sigma \cup C)$-structure defined by $T(\Sigma \cup C \cup D')/=_E$ together with an interpretation of selectors in Σ', and H_2 the domain of $\mathcal{H}_2$. By definition, $H_1 \subseteq H_2$. Any selector $s_i^c : \texttt{struct} \to \sigma$ in Σ' is interpreted in $\mathcal{H}_2$ as follows:

- for any `struct`-sorted $a \in H_1$, $\mathcal{H}_2[s_i^c](a) = \mathcal{H}_1[s_i^c](a)$;
- for any `struct`-sorted $a' \in H_2 \backslash H_1$, $\mathcal{H}_2[s_i^c](a') = a'_i$ if $a' = \mathcal{H}_2[c](a'_1, \ldots, a'_n)$, otherwise $\mathcal{H}_2[s_i^c](a') = \mathbb{u}_\sigma$.

By construction, $\mathcal{H}_2^{\Sigma\cup C}$ is $T(\Sigma \cup C \cup D')/=_E$. We can check that $\mathcal{H}_2$ is a model of T_s:

- Let ψ be any axiom in T_s. By assigning all variables in ψ to values in H_1, ψ evaluates to true in $\mathcal{H}_2$ since $\mathcal{H}_1$ is a model of T_s.
- Let ψ be any axiom in $AFDS_\Sigma \cup Proj_{\Sigma'}$. By assigning some variable in ψ to a value in $H_2 \backslash H_1$, ψ evaluates to true in $\mathcal{H}_2$.

Since T_s is included in $AFDS_\Sigma \cup Proj_{\Sigma'}$, all axioms in T_s evaluate to true for any assignment in H_2, which means that $\mathcal{H}_2$ is a model of T_s. Furthermore, all literals in G are also true in $\mathcal{H}_2$, and so $\mathcal{H}_2$ is a model of $\mathfrak{T}_s = T_s \cup G$. □

Since any polished Σ_s-theory is polite with respect to the set of sorts that are free in Σ_s, it can be combined with an arbitrary disjoint theory of elements whose sorts are free in Σ_s, and a satisfiability procedure for the resulting combined theory is provided by the combination procedure known for polite theories [13,20]. In the following, we show that combining a polished theory with a target theory and a bridging theory is a way to build a new polished theory T. A T-satisfiability procedure is given by the combination procedure presented in Sect. 4.

6.2 Completeness of the Combination Procedure

The combination of a polished source theory T_s with a target theory T_t (and a bridging theory T_f) results in an extension of T_s with new function symbols in Σ_t, and a new bridging function f. To build on the politeness of T_s, we assume that T_t provides a theory of elements for `Elem`-sorts that are free in T_s. The assumption on signatures used in Sect. 4 corresponds to a particular case where the Σ_s-theory T_s is the polished theory $AFDS_\Sigma$, in which all `Elem`-sorts are free.

In what follows, we study the satisfiability problem modulo $T = T_s \cup T_f \cup T_t$ where T_f is a bridging theory between a polished theory T_s and a theory T_t sharing only free sorts with T_s (cf. Definition 11). The combination procedure described in Sect. 4 is sound and complete also here. We prove the completeness of the combination procedure thanks to a combined model constructed using rewriting techniques. Given a bridging function $f : \texttt{struct} \to \texttt{t}$ where `t` is a sort from the target theory, a bridging theory provides a convergent term rewrite system F such that for any term s of sort `struct`, its normal form $f(s)\downarrow_F$ corresponds to a term that can be interpreted in a model of the target theory. To prove completeness, we carefully study the interplay between the equational theory E related to a basic Herbrand model and the term rewrite system F.

For convenience, we will consider theory extensions including the sets of ground literals rather than handling literals and theories separately.

Assumption 1 (*Input theories*) Let T_s be a polished Σ_s-theory and T_t a Σ_t-theory sharing only free sorts with T_s. Let C and C_t be two finite sets of constants such that $\Sigma_s \cup C$ and $\Sigma_t \cup C_t$ are constant expansions of Σ_s and Σ_t, respectively, where $C_\sigma \subseteq (C_t)_\sigma$ for any sort σ occurring in $\Sigma_s \cap \Sigma_t$. Let α be an arrangement over $C \cap C_t$.

1. $\mathfrak{T}_s$ is a consistent $\Sigma_s \cup C$-theory including α, corresponding to a witnessable T_s-extension. It admits a basic Herbrand model $\mathcal{H}$ such that $\mathcal{H}^{\Sigma \cup C}$ is $T(\Sigma \cup C \cup D)\,/\!=_E$.
2. $\mathfrak{T}_t$ is a consistent $\Sigma_t \cup C_t$-theory defined as the union of T_t and some finite set of ground $\Sigma_t \cup C_t$-literals including α.

Notice that the arrangement α is over the set of constants in C whose sorts are shared by Σ_s and Σ_t. From now on, we consider that Assumption 1 holds.

A bridging theory T_f (Definition 2) is an equational theory that can naturally be oriented as a term rewrite system F.

Proposition 11 *Let T_f be a bridging theory as introduced in Definition 2, and let $\mathfrak{T}_F = T_f \cup \{f(x) = f_x \mid x \in C_{\texttt{struct}}\}$. The term rewrite system $F = \{f(l) \to r \mid f(l) = r \in \mathfrak{T}_F\}$ is convergent and satisfies the following properties:*

- $f(c(\mathbf{e}; t_1, \ldots, t_n))\downarrow_F = f_c(\mathbf{e}; f(t_1)\downarrow_F, \ldots, f(t_n)\downarrow_F)$ *for any non-constant constructor* $c \in \Sigma$;
- $f(c)\downarrow_F = f_c$ *for any constant* c *in* Σ, *where* f_c *is a constant in* Σ_t;
- $f(x)\downarrow_F = f_x$ *for any* `struct`*-sorted constant* $x \in C$, *where* $f_x \in C_t$.

Example 13 Consider the length function ℓ over lists of integers, and let $\Sigma = \{cons : \texttt{int} \times \texttt{struct} \to \texttt{struct}, nil : \texttt{struct}\}$, $\Sigma' = \emptyset$, $C = \{x, y, z, e, e'\}$. The theory $\mathfrak{T}_s = \{x = cons(e, y), y = cons(e', z), z = nil, x \neq z, e \neq e'\}$ is a consistent witnessable T_s-extension for the empty Σ-theory $T_s = \emptyset$, which is polished since it belongs to $\mathbf{DST}^+$. Assume the theory of integers T_t, a finite set of constants $C_t \supseteq \{\ell_x, \ell_y, \ell_z, e, e'\}$, and $\mathfrak{T}_t = T_t \cup \{e \neq e'\}$. Given the bridging theory

$$T_\ell = \{\ell(cons(X, Y)) = 1 + \ell(Y), \ell(nil) = 0\},$$

the corresponding rewrite system is $F = \{\ell(cons(X, Y)) \to 1+\ell(Y), \ell(nil) \to 0\} \cup \{\ell(x) \to \ell_x, \ell(y) \to \ell_y, \ell(z) \to \ell_z\}$. □

While building a $\mathfrak{T}_s \cup \mathfrak{T}_F \cup \mathfrak{T}_t$-model, to get a well-defined interpretation for the bridging function f, we need a $\mathfrak{T}_t$-model in which f returns the same value for all E-equal input terms of sort `struct`. This motivates the following definition of *E-compatibility*. Below, a *constructor-based* term denotes a term only built over constructors in Σ together with constants, and such that all `struct`-sorted free constants occurring in the term belong to C.

Definition 15 (*E-Compatibility*) F is *E-compatible* in a model $\mathcal{A}$ of $\mathfrak{T}_t$ if for any constructor-based terms s and t, $s =_E t \Rightarrow \mathcal{A}[f(s)\downarrow_F] = \mathcal{A}[f(t)\downarrow_F]$.

Proposition 12 *Under Assumption 1, $\mathfrak{T}_s \cup \mathfrak{T}_F \cup \mathfrak{T}_t$ is consistent if F is E-compatible in a model of $\mathfrak{T}_t$.*

Proof We know that F is E-compatible in a model $\mathcal{A}$ of $\mathfrak{T}_t$, and there exists a model $\mathcal{H}$ of $\mathfrak{T}_s$ such that $\mathcal{H}^{\Sigma \cup C}$ is $T(\Sigma \cup C \cup D \cup D')/=_E$ where

- D' is a set of elements of shared sorts,
- $\mathcal{H}[\sigma] = \mathcal{A}[\sigma]$ for each shared sort.

Given $\mathcal{A}$ and $\mathcal{H}$, let us define an interpretation $\mathcal{M}$ as follows. The domains of $\mathcal{M}$ are:

- $\mathcal{M}[\sigma] = \mathcal{A}[\sigma]$ for any sort σ in Σ_t
- $\mathcal{M}[\sigma] = \mathcal{H}[\sigma]$ for any sort σ in Σ_s

The function symbols are interpreted as follows:[2]

- For any g in $\Sigma_t \cup C_t$, $\mathcal{M}[g] = \mathcal{A}[g]$ and for any g in $\Sigma_s \cup C$, $\mathcal{M}[g] = \mathcal{H}[g]$; this is well-defined due to the arrangement α over $C \cap C_t$
- For any constructor-based term t, $\mathcal{M}[f](\llbracket t \rrbracket) = \mathcal{A}[f(t)\downarrow_F]$

$\mathcal{M}$ is well-defined due to the assumption that F is E-compatible in $\mathcal{A}$. Let us check that $\mathcal{M}$ is a model of $\mathfrak{T}_s \cup \mathfrak{T}_F \cup \mathfrak{T}_t$.

- $\mathcal{M}^{\Sigma_s \cup C} = \mathcal{H}$, which is a model of $\mathfrak{T}_s$ by assumption.
- $\mathcal{M}^{\Sigma_t \cup C_t} = \mathcal{A}$, which is a model of $\mathfrak{T}_t$ by assumption.
- For any constructor-based term t, we have that

$$\mathcal{M}[f(t)] = \mathcal{M}[f](\llbracket t \rrbracket) = \mathcal{A}[f(t)\downarrow_F] = \mathcal{M}[f(t)\downarrow_F]$$

by definition of $\mathcal{M}$. Therefore $\mathcal{M}$ is a model of $\mathfrak{T}_F$.

□

The missing piece of the method is to provide a way to check the E-compatibility of F in a model of $\mathfrak{T}_t$. In the following, we show that this property can be reduced to a $\mathfrak{T}_t$-satisfiability problem.

Proposition 13 *F is E-compatible in a model of $\mathfrak{T}_t$ if the theory $\mathfrak{T}_t \cup CP_E$ is consistent, where CP_E is the target encoding of E (Definition 4).*

Proof Let $\mathcal{A}$ be a model of $\mathfrak{T}_t \cup CP_E$. Let R be the convergent term rewrite system associated to E. Since $\alpha \subseteq \mathfrak{T}_t$, $\mathcal{A}$ satisfies α, and we have that $\mathcal{A}[e\downarrow_R] = \mathcal{A}[e]$ for any constant e of sort in $\Sigma_s \cap \Sigma_t$. We first prove by structural induction that for any constructor-based term u, $\mathcal{A}[f(u\downarrow_R)\downarrow_F] = \mathcal{A}[f(u)\downarrow_F]$.
(Inductive case) Assume $u = c(\mathbf{e}; u_1, \ldots, u_n)$.

[2] For any constructor-based term t, $\llbracket t \rrbracket$ is the equivalence class of t modulo $=_E$.

– If $c(\mathbf{e}; u_1, \ldots, u_n)\downarrow_R = c(\mathbf{e}\downarrow_R; u_1\downarrow_R, \ldots, u_n\downarrow_R)$, then we have:

$$\begin{aligned}
&\mathcal{A}[f(c(\mathbf{e}; u_1, \ldots, u_n)\downarrow_R)\downarrow_F] \\
&= \mathcal{A}[f(c(\mathbf{e}\downarrow_R; u_1\downarrow_R, \ldots, u_n\downarrow_R))\downarrow_F] \\
&= \mathcal{A}[f_c(\mathbf{e}\downarrow_R;\ f(u_1\downarrow_R)\downarrow_F, \ldots,\ f(u_n\downarrow_R)\downarrow_F)] \\
&= f_c(\mathcal{A}[\mathbf{e}\downarrow_R]; \mathcal{A}[f(u_1\downarrow_R)\downarrow_F], \ldots, \mathcal{A}[f(u_n\downarrow_R)\downarrow_F]) \\
&= f_c(\mathcal{A}[\mathbf{e}]; \mathcal{A}[f(u_1\downarrow_R)\downarrow_F], \ldots, \mathcal{A}[f(u_n\downarrow_R)\downarrow_F]) \\
&= f_c(\mathcal{A}[\mathbf{e}]; \mathcal{A}[f(u_1)\downarrow_F], \ldots, \mathcal{A}[f(u_n)\downarrow_F]) \\
&= \mathcal{A}[f_c(\mathbf{e};\ f(u_1)\downarrow_F, \ldots,\ f(u_n)\downarrow_F)] \\
&= \mathcal{A}[f(c(\mathbf{e}; u_1, \ldots, u_n))\downarrow_F]
\end{aligned}$$

– Otherwise, $c(\mathbf{e}; u_1, \ldots, u_n)\downarrow_R$ is necessarily a constant x', and the terms $u_1\downarrow_R, \ldots, u_n\downarrow_R$ are constants $x_1, \ldots, x_n$. By assumption on $\mathcal{A}$, we have

$$\begin{aligned}
\mathcal{A}[f(x')\downarrow_F] &= \mathcal{A}[f_{x'}] \\
&= \mathcal{A}[f_c(\mathbf{e};\ f_{x_1}, \ldots,\ f_{x_n})] \\
&= \mathcal{A}[f_c(\mathbf{e};\ f(x_1)\downarrow_F, \ldots,\ f(x_n)\downarrow_F)] \\
&= \mathcal{A}[f_c(\mathbf{e};\ f(u_1\downarrow_R)\downarrow_F, \ldots,\ f(u_n\downarrow_R)\downarrow_F)] \\
&= f_c(\mathcal{A}[\mathbf{e}]; \mathcal{A}[f(u_1\downarrow_R)\downarrow_F], \ldots, \mathcal{A}[f(u_n\downarrow_R)\downarrow_F]) \\
&= f_c(\mathcal{A}[\mathbf{e}]; \mathcal{A}[f(u_1)\downarrow_F], \ldots, \mathcal{A}[f(u_n)\downarrow_F]) \\
&= \mathcal{A}[f_c(\mathbf{e};\ f(u_1)\downarrow_F, \ldots,\ f(u_n)\downarrow_F)] \\
&= \mathcal{A}[f(c(\mathbf{e}; u_1, \ldots, u_n))\downarrow_F]
\end{aligned}$$

(Base case) Assume u is a constant x. If $x\downarrow_R = x$, then $f(x\downarrow_R)\downarrow_F = f(x)\downarrow_F$, and so $\mathcal{A}[f(x\downarrow_R)\downarrow_F] = \mathcal{A}[f(x)\downarrow_F]$. Otherwise, we have $x\downarrow_R = x'$. Then, by assumption on $\mathcal{A}$, we have $\mathcal{A}[f(x')\downarrow_F] = \mathcal{A}[f_{x'}] = \mathcal{A}[f_x] = \mathcal{A}[f(x)\downarrow_R]$.

To conclude the proof, let s and t be any constructor-based terms. If $s =_E t$, then $s\downarrow_R = t\downarrow_R$ and $\mathcal{A}[f(s)\downarrow_F] = \mathcal{A}[f(s\downarrow_R)\downarrow_F] = \mathcal{A}[f(t\downarrow_R)\downarrow_F] = \mathcal{A}[f(t)\downarrow_F]$. This means F is E-compatible in the model $\mathcal{A}$ of $\mathfrak{T}_t$. □

Example 14 (*Example 13 continued*) For the given $\mathfrak{T}_s$, we have that $E = \{x = cons(e, y), y = cons(e', z), z = nil\}$ and so $CP_E = \{\ell_x = 1 + \ell_y, \ell_y = 1 + \ell_z, \ell_z = 0\}$. Since $\mathfrak{T}_t \cup CP_E$ is consistent, $\mathfrak{T}_s \cup \mathfrak{T}_F \cup \mathfrak{T}_t$ is consistent thanks to Propositions 12 and 13. □

As a side remark, in the trivial case of $F = \{f(x_k) \to f_{x_k}\}_{k\in K}$, the combination becomes disjoint, and the consistency of $\mathfrak{T}_s \cup \mathfrak{T}_F \cup \mathfrak{T}_t$ corresponds to the consistency of the union of three disjoint theories, including the theory of equality for f.

Propositions 12 and 13 are instrumental to prove the completeness of the combination procedure. To relate the case of polished theories with the procedure presented in Sect. 4, it is sufficient to notice that separate forms correspond to witnessable T_s-extensions. Then, the following result subsumes Theorem 1:

Theorem 4 *Let $T = T_s \cup T_f \cup T_t$, where T_s is a polished theory, T_t shares only free sorts with T_s and T_f is a bridging theory. A combinable separate form $\varphi_s \cup \varphi_t \cup \varphi_f$ is T-satisfiable if and only if φ_s is T_s-satisfiable and φ_t is T_t-satisfiable.*

Proof Soundness is straightforward just like in Sect. 4. Let us focus on the completeness. Consider $\varphi = \varphi_s \cup \varphi_t \cup \varphi_f$ and the sets of variables $V = Var(\varphi_s)$, and $V_t = \{x \mid x \in$

$Var_\sigma(\varphi), \sigma$ is a sort in $\Sigma_t\}$. We can consider without loss of generality that V includes a variable v for each sort in $\texttt{Elem} \cup \{\texttt{struct}\}$ (it can be enforced by adding the trivial equality $v = v$ to the input).

By assumption, φ is a combinable separate form and so there exist arrangements α, α' as introduced in Definition 4 such that $\alpha \cup \alpha' \subseteq \varphi_s$ and $\alpha \cup CP_E \subseteq \varphi_t$. By viewing φ as a set of ground literals in a constant expansion of $\Sigma_s \cup \Sigma_f \cup \Sigma_t$, we can introduce the same theories $\mathfrak{T}_s$, $\mathfrak{T}_t$ and $\mathfrak{T}_F$ as in Assumption 1 and Proposition 11. Consider $C = V$ and $C_t = V_t$. Then, the theories are defined as follows:

- the $\Sigma_s \cup C$-theory $\mathfrak{T}_s$ is $T_s \cup \varphi_s$,
- the $\Sigma_t \cup C_t$-theory $\mathfrak{T}_t$ is $T_t \cup \varphi_t$,
- $\mathfrak{T}_F = T_f \cup \varphi_f$.

By assumption, φ_s is T_s-satisfiable and φ_t is T_t-satisfiable. Since $\varphi_t = \varphi_t \cup CP_E, \varphi_t \cup CP_E$ is T_t-satisfiable. Thus, φ_s is T_s-satisfiable and $\varphi_t \cup CP_E$ is T_t-satisfiable. Equivalently, $\mathfrak{T}_s$ and $\mathfrak{T}_t \cup CP_E$ are consistent. By applying Propositions 12 and 13, we get that $\mathfrak{T}_s \cup \mathfrak{T}_F \cup \mathfrak{T}_t$ is consistent, and so $T_s \cup T_f \cup T_t \cup \varphi$ is consistent, or equivalently, φ is T-satisfiable. □

The following example illustrates the application of the procedure when the target theory is not stably infinite with respect to the sorts that are free in the signature of the source theory.

Example 15 Let T_t be the theory of Booleans and a theory of binary trees over Booleans, with $\texttt{Elem} = \{\texttt{bool}\}$, constructors $\Sigma = \{nil : \texttt{struct}, cons : \texttt{bool} \times \texttt{struct} \times \texttt{struct} \rightarrow \texttt{struct}\}$, and selectors $val, left, right$, formally defined by $T_s = \{val(cons(I, X, Y)) = I, left(cons(I, X, Y)) = X, right(cons(I, X, Y)) = Y\}$. Assume the bridging theory for the function $and : \texttt{struct} \rightarrow \texttt{bool}$ is $T_{and} = \{and(nil) = true,\ and(cons(I, X, Y)) = I \wedge and(X) \wedge and(Y)\}$.

Let $T = T_s \cup T_{and} \cup T_t$, and consider the T-satisfiability problem $\varphi = \{v_1 \neq v_2, v_1 \neq v_3, x = cons(e, y, z), v_1 = val(y), v_3 = val(z), and(x) \neq v_3, val(x) = true, v_2 = and(y), and(z) = true\}$, or in separate form:

- $\varphi_s = \{x = cons(e, y, z), val(x) = b, v_1 = val(y), v_3 = val(z)\}$
- $\varphi_t = \{v_1 \neq v_2, v_1 \neq v_3, and_x \neq v_3, v_2 = and_y, and_z = true, b = true\}$
- $\varphi_{and} = \{and_x = and(x), and_y = and(y), and_z = and(z)\}$

The T-unsatisfiability of φ follows from Theorem 4:

- Assume an arrangement α containing $e = b$. In the target theory, the equality $e = b$ implies $e = true$ since $b = true$ is in φ_t. Since $e = and_z = true$, the equality $and_x = e \wedge and_y \wedge and_z$ in CP_E reduces to $and_x = and_y$. Then, $and_x \neq v_3$ becomes $v_2 \neq v_3$ since $and_x = and_y = v_2$. Finally, $\{v_1 \neq v_2 \neq v_3\}$ is T_t-unsatisfiable.
- If α is an arrangement containing $e \neq b$, then $\varphi_s \cup \alpha$ is T_s-unsatisfiable.

□

6.3 Modularity Results

The combined model constructed for the proof of Theorem 4 is a basic Herbrand model of the combined theory T, actually showing that T remains a polished theory, albeit with fewer free sorts. Compared to the source polished theory T_s, the combined polished theory T includes an extended theory of elements. To state this modularity result, let us introduce the *data signature* of a polished Σ_s-theory T_s defined as the subsignature Ω of Σ_s comprising all function symbols $f : \sigma_1 \times \cdots \times \sigma_n \rightarrow \sigma_{n+1}$, with $\sigma_1, \ldots, \sigma_{n+1} \in \texttt{Elem}$. Then, the Ω-theory $T_s^e = T_s^\Omega$ is called the *data theory* of T_s.

Proposition 14 *The combined theory T from Theorem 4 is a polished theory whose data theory is $T_s^e \cup T_t$.*

Proof The fact that T is a polished theory is a consequence of the model construction used in the proof of Theorem 4. Let us prove that the data theory of T is $T_s^e \cup T_t$. The combined model built in Theorem 4 shows that any model of $T_s^e \cup T_t$ can be expanded to a model of T, and conversely any model of T can be reduced to a model of $T_s^e \cup T_t$.

Let Ω be the data signature of T_s. For any $\Omega \cup \Sigma_t$-sentence φ, we can now show that $T_s^e \cup T_t \models \varphi$ iff $T \models \varphi$. This is proved by contradiction. □

Theorem 4 shows that a new polished theory is built by considering simultaneously two "dimensions": the addition of a target theory, and the addition of a bridging theory. It is also possible to consider these two dimensions separately, as discussed below.

The combined model construction seen in Sect. 6.2 also holds in the case $T_f = \emptyset$ and $T = T_s \cup T_t$. Similarly to the case of a non-empty bridging theory T_f, T is polished and a T-satisfiability procedure is provided by the combination method, where for any separate form φ, the sets of literals φ_f and CP_E are both empty. Since any separate form is its own witness, we retrieve exactly the combination method known for unions of polite theories T_s and arbitrary theories T_t [13,20], already applied in Sect. 5. Thus, this leads to the following modularity result:

Corollary 1 *The class of polished theories with a decidable satisfiability problem is closed by combination with a decidable theory sharing only free sorts.*

It is important to notice that the number of free sorts strictly decreases when a polished theory is combined with a theory sharing only free sorts, if there is indeed at least one shared free sort. Hence, a polished theory can be repeatedly combined with such theories but only finitely many times before reaching a final "fully instantiated" polished theory with an empty set of free sorts.

A combination $T_s \cup T_f \cup T_t$ is said to be *direct* when T_s is polished, $T_t = T_s^e$ and T_f is a bridging theory. In that case $T_s^e \cup T_t = T_s^e$ and the combined model construction seen in Sect. 6.2 also holds in a simplified version not relying on the politeness of T_s.

Corollary 2 *The class of polished theories with a decidable satisfiability problem is closed by direct combination with a bridging theory.*

The above modular approach is particularly useful when several bridging functions are defined on the same target theory. To build the combined theory in that case, the target theory is first added to the data structure theory, and then each bridging theory is added in an incremental way.

Notice that there is no restriction on the (decidable) target theory. Actually, T_t could be also a polished theory obtained from a previous application of the combination method. Consider the case

$$T = (T_{tree} \cup T_t) \cup T_{sz}$$

where T_{tree} denotes a polished theory of trees and T_{sz} denotes the bridging theory defining the size of trees thanks to a target theory

$$T_t = (T_{list} \cup T_{\mathbb{Z}}) \cup T_\ell$$

corresponding to a theory of lists extended with a bridging function ℓ computing the length of lists. Applying twice the combination method is a way to build a T-satisfiability procedure

where T corresponds to the union of two disjoint polished theories extended with their respective bridging functions to $T_{\mathbb{Z}}$:

$$T = (T_{tree} \cup T_{sz} \cup T_{\mathbb{Z}}) \cup (T_{list} \cup T_{\ell} \cup T_{\mathbb{Z}}).$$

In the same vein, the combination method applied twice yields a satisfiability procedure for a theory of lists of trees extended with tree size sz and list length ℓ. The above examples illustrate the generality of our combination method. To conclude this discussion on the applications of Theorem 4, we may remark that any Σ_s-theory T_s can be considered as a polished theory for $\Sigma = \emptyset$. In that case, T_f reduces to the theory of equality over the function symbol $f : \texttt{struct} \to \texttt{t}$, and our combination method leads to a satisfiability procedure for a disjoint union of (many-sorted) theories $T_s \cup T_f \cup T_t$.

7 Related Work

7.1 Axiomatized Data Structures

One particular aspect of our work is that the sorts for elements and for the data structure are distinct. This is crucial for our politeness results. Data structure theories $\mathbf{DST}^+$ do not include function or constant symbols over sorts for elements; these are supposed to be defined by an additional, separate, theory of elements. In [34] the elements are denoted using a finite or infinite set of constants. In our work, any separate theory can be used to define how elements are interpreted.

The theory of Absolutely Free Data Structures is essentially the theory of finite trees. Syntactic unification thus naturally provides a solver for the equalities. For instance, the procedure given in [24] is based on syntactic unification to solve equations over trees, while disequalities are processed one by one thanks to the convexity of this theory. The theory of finite of trees is indeed a typical example of a Shostak theory. We build on this to present a clean, abstract satisfiability procedure, by applying a solver (for equalities) together with a canonizer (for disequalities). In practice, an efficient implementation of this satisfiability procedure would use (bidirectional) congruence closure, similarly to [18,34] for the satisfiability of ground literals.

While Absolutely Free Data Structures are nicely handled by a solver and a canonizer, other axiomatized theories in $\mathbf{DST}^+$ do not fit this schema, but their decision procedures can still be described in an abstract way as inference systems. The abstract congruence closure procedure given in Fig. 2 for theories in $\mathbf{DST}^+$ is defined as a superposition calculus. As illustrated for instance in [6], superposition calculi are well-suited to develop abstract decision procedures for axiomatized data structure theories. In the same vein, a dedicated superposition calculus has been developed in [17] for an AFDS theory whose constructors are the successor and 0, and a standard superposition calculus is applied in [2], e.g, for a theory of lists defined by projection axioms $\{car(cons(E, Y)) = E, cdr(cons(E, Y)) = Y\}$. In our framework, selectors car and cdr, are partially defined by these projection axioms. Then, the extension to a total function is achieved similarly to [23] by using an arbitrary but fixed constant (of sort $\texttt{elem}$ for $car : \texttt{struct} \to \texttt{elem}$, and of sort $\texttt{struct}$ for $cdr : \texttt{struct} \to \texttt{struct}$). In [34], car and cdr are defined as functions from trees to trees such that $car(nil) = cdr(nil) = nil$. Such a definition is not possible here since distinct sorts are used for the elements and for the data structure.

Our combination approach is strongly inspired by the locality approach, and many notions and model constructions we use are borrowed from [22,23]. All the theories in $\mathbf{DST}^+$ correspond to subtheories of Absolutely Free Data Structures successfully considered in [23]. In our many-sorted framework, the output sort for the data structure, say `struct`, is distinguished from the input sorts for elements (`Elem`) that are possibly shared with the target theory. This crucial hypothesis enables new politeness results, leading to satisfiability procedures for unions of theories sharing sorts that are not covered by [23], e.g., the one in Example 15. With respect to the locality approach, we here trade the expressiveness of bridging theories for a simpler combination schema. In our context, the bridging theory T_f is exhaustive in the sense that f is defined for each constructor in Σ. Moreover, T_f is simply defined by a set of equalities. In [23], further bridging theories are successfully considered, by using guarded equalities and by relaxing the exhaustivity assumption to allow constants in Σ that are undefined for f. Extending our approach to bridging theories expressed by guarded equalities may be done at the cost of using a conditional term rewrite system in the completeness proof of Sect. 6.2.

To prove locality properties, a classical approach [22,23] amounts to build (total) models from some particular *weak partial models*. A couple of models built in this paper can be related to models used in [23] for showing locality results:

- the combined model constructed in the proof of Theorem 1 corresponds to the model used in the proof of Theorem 13 in [23], showing that the non-disjoint combination $AFDS_\Sigma \cup T_f \cup T_t$ satisfies some locality property with respect to the disjoint combination $AFDS_\Sigma \cup T_t$;
- the term-generated model constructed in the proof of Proposition 10 corresponds to the model used in the proof of Theorem 10 in [23], showing that theories in $\mathbf{DST}^+$ are local.

The decision procedures developed here and in [22,23] are correct for essentially the same reasons. Thus, proofs are naturally based on similar model constructions.

7.2 Standard Interpretations

The restriction to standard interpretations is presented in [22,23] as a restriction to term-generated models built without `struct`-sorted free constants, where additional constraints are used to express for instance the fact that the length of any list is positive. As noticed in [22, 23], these constraints are not sufficient when the domain for elements is finite and `struct`-sorted disequalities are allowed in satisfiability problems. Actually, the main difficulty is to decide the satisfiability of a formula

$$(SC) \quad x_1 \neq \cdots \neq x_n \wedge f(x_1) = \cdots = f(x_n) = v$$

where $n > 1$, $x_1, \ldots, x_n$ are `struct`-variables and f is a bridging function with arity $f : \texttt{struct} \to \texttt{t}$. To tackle this problem, one can explore various assumptions on f.

- First, if f is bijective, then the formula SC is unsatisfiable.
- Second, if f is infinitely surjective, the formula SC is satisfiable in general, except for some particular values of f. In that case, there are usually infinitely many `struct`-elements mapped by f to the same value, except for some particular cases.
- Sufficient surjectivity [19,24] is a generalization of infinite surjectivity where the formula SC is satisfiable because there are at least n distinct `struct`-elements mapped by f to the same value, except for some particular values of f.

– Our notion of *gently growing* function refines the case of a sufficiently surjective function for $\mathtt{t} = \mathtt{int}$. In that case, the formula SC is satisfiable if v is large enough (greater or equal than $b(n)$ according to Definition 9), and possibly unsatisfiable for strictly smaller values. From our point of view, the restriction $\mathtt{t} = \mathtt{int}$ allows us to express sufficient surjectivity in a simple way by using natural numbers. When f is gently growing in some theory T, any T-satisfiability problem can be reduced to a satisfiability problem in $AFDS_\Sigma \cup T_\mathbb{Z}$ by a non-deterministic procedure guessing finitely many range constraints. This reduction can be viewed as a way to build a finite witness of any T-satisfiability problem, showing that T is polite. Hence, T is combinable with an arbitrary (non-necessarily stably infinite) theory of elements T_{elem} thanks to a combination method à la Nelson–Oppen [13,20].

Our approach for standard interpretations presents some similarities with the ad hoc decision procedure presented in [34] for the particular case of trees combined with integer constraints via the standard length function. In [34], the theory of trees includes selectors, where a selector for a given type α is defined as expected for trees of type α and as the identity otherwise. In the case of lists, the two possible types are *nil* and *cons*, and $car(nil) = cdr(nil) = nil$. This definition of selectors requires an additional predicate to check the type of a term. The approach followed in [34] includes a type completion, to guess whether a list is of type *nil* or *cons*. This guessing is sound and complete in [34] since the underlying theory includes the axiom of extensionality (called $\mathtt{IsC}$ in [22,23])

$$\forall x.\ x \neq nil \Rightarrow x = cons(car(x), cdr(x)).$$

Similarly, the formula

$$\forall x.\ x \neq nil \Rightarrow \exists e, y.\ x = cons(e, y)$$

stating that any list is constructed, holds in the case of standard interpretations, and the guessing of range constraints actually provides a guessing of types: the length 0 corresponds to *nil*, and any length ≥ 1 corresponds to the *cons* type.

In [34], the decision procedure for the existential fragment is based on a reduction to $T_\mathbb{Z}$ via the computation of a *length constraint completion*. This is sufficient when the length function is infinitely surjective due to an infinite constant domain. We state a similar result in Proposition 3, where the length constraint completion corresponds to the target encoding for the length function (cf. Definition 4) together with range constraints bounded by $n = 1$. To go beyond the case of infinite surjective length, [34] also introduces a general notion of *relativized length constraint completion* to capture the existence of a reduction in $T_\mathbb{Z}$. In [34], an algorithm is given to compute the relativized length constraint completion in the particular case of a finite constant domain of cardinality n. To reuse this relativized length constraint completion in our framework, we would have to consider the particular combination $T^{si}_{tree} \cup T_{elem}$ where $T_{elem} = \{\exists e_1, \ldots, e_n.\ e_1 \neq \cdots \neq e_n \wedge \forall x.\ x = e_1 \vee \cdots \vee x = e_n\}$. Our approach, based on a reduction to $AFDS_\Sigma \cup T_\mathbb{Z}$ instead of a reduction to $T_\mathbb{Z}$ as in [34], is more flexible and is suitable for a combination with any arbitrary theory of elements T_{elem}. The possibility of a richer theory on the constant domain, say T_{elem}, has been briefly outlined in [34] (cf. Sect. 5.5). The idea, recasted in our framework, would be to reduce any $T^{si}_{tree} \cup T_{elem}$-satisfiability problem into a $T_\mathbb{Z} \cup T_{elem}$-satisfiability problem, and then to apply a Nelson–Oppen combination method for $T_\mathbb{Z} \cup T_{elem}$. In this paper, we show that a combination method à la Nelson–Oppen is applicable when combining T^{si}_{tree} with any arbitrary theory of elements T_{elem} since T^{si}_{tree} is indeed a polite theory.

8 Conclusion

This paper describes (Sect. 4) a non-deterministic combination method à la Nelson–Oppen for unions of constructor-based theories connected to target theories via bridging functions. Similarly to the classical Nelson–Oppen method, implementations of this non-deterministic method should be based on practical refinements of the guessing phases. 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 (Sect. 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.

To go beyond the case of absolutely free data structures, we have investigated in Sect. 6 more data structure theories with bridging functions. The combination method of Sect. 4 is indeed sound and complete for a large class of source data structure theories, ranging from the theory of equality to the theory of absolutely free data structures. Thanks to the politeness of these source theories, one can consider any arbitrary target theory, including a non-stably infinite one sharing some sorts with the source. Hence, we have identified two significant applications of politeness to non-disjoint combinations of theories. First, we have studied theories defined as classes of standard interpretations. Second, we have introduced the class of polished theories, including well-known axiomatized theories. Using the finite axiomatization of these theories, the satisfiability problem can be solved by applying an off-the-shelf equational theorem prover [1,2].

We envision several further investigations. First, we would like to consider the case of non-absolutely free constructors, e.g., associative-commutative constructors. Second, we want to continue the study of saturation-based satisfiability procedures as a mean to build finite witnesses of polite theories. By introducing polished theories, we have focused on a basic case related to absolutely free data structures. We believe it is possible to go further, for instance to cope with data structure theories that are non-convex over the `struct` sort.

Finally, to go beyond the considered bridging axioms, a natural continuation is to identify other "simple" connecting axioms that could be compiled into a combination method à la Nelson–Oppen.

Acknowledgements We are very grateful to the reviewers of this paper and of the previous related conference papers for their insightful remarks: the paper has been improved significantly thanks to their comments. Pascal Fontaine would also like to thank Jasmin C. Blanchette for discussions, encouragements and financial support through his ERC Grant.

References

1. Armando, A., Bonacina, M.P., Ranise, S., Schulz, S.: New results on rewrite-based satisfiability procedures. ACM Trans. Comput. Log. **10**(1), 4 (2009)
2. Armando, A., Ranise, S., Rusinowitch, M.: A rewriting approach to satisfiability procedures. Inf. Comput. **183**(2), 140–164 (2003)
3. Baader, F., Ghilardi, S.: Connecting many-sorted theories. J. Symb. Log. **72**(2), 535–583 (2007)
4. Baader, F., Nipkow, T.: Term Rewriting and All That. Cambridge University Press, Cambridge (1998)
5. Bachmair, L., Ganzinger, H.: Rewrite-based equational theorem proving with selection and simplification. J. Log. Comput. **4**(3), 217–247 (1994)
6. Barrett, C., Shikanian, I., Tinelli, C.: An abstract decision procedure for a theory of inductive data types. JSAT **3**(1–2), 21–46 (2007)

7. Baumgartner, P., Waldmann, U.: Hierarchic superposition with weak abstraction. In: Bonacina, M.P. (ed.) 24th International Conference on Automated Deduction (CADE-24), Lake Placid, NY, USA, volume 7898 of LNCS, pp. 39–57. Springer (2013)
8. Chocron, P., Fontaine, P., Ringeissen, C.: A Gentle non-disjoint combination of satisfiability procedures. In: Demri, S., Kapur, D., Weidenbach, C. (eds.) Proceedings of the 7th International Joint Conference on Automated Reasoning, IJCAR, volume 8562 of LNCS, pp. 122–136. Springer (2014)
9. Chocron, P., Fontaine, P., Ringeissen, C.: A polite non-disjoint combination method: theories with bridging functions revisited. In: Felty, A.P., Middeldorp, A. (eds.) 25th International Conference on Automated Deduction (CADE-25), Berlin, Germany, volume 9195 of LNCS, pp. 419–433. Springer (2015)
10. Chocron, P., Fontaine, P., Ringeissen, C.: A rewriting approach to the combination of data structures with bridging theories. In: Lutz, C., Ranise, S. (eds.) Frontiers of Combining Systems (FroCoS), volume 9322 of LNCS, pp. 275–290. Springer (2015)
11. Fontaine, P., Ranise, S., Zarba, C.G.: Combining lists with non-stably infinite theories. In Baader, F., Voronkov, A. (eds.) Logic for Programming, Artificial Intelligence, and Reasoning (LPAR'04), volume 3452 of LNCS, pp. 51–66. Springer (2005)
12. Ghilardi, S.: Model-theoretic methods in combined constraint satisfiability. J. Autom. Reason. **33**(3–4), 221–249 (2004)
13. Jovanovic, D., Barrett, C.: Polite theories revisited. In: Fermueller, C., Voronkov, A. (eds.) Logic for Programming, Artificial Intelligence, and Reasoning (LPAR'10), volume 6397 of LNCS, pp. 402–416. Springer (2010)
14. Kruglov, E., Weidenbach, C.: Superposition decides the first-order logic fragment over ground theories. Math. Comput. Sci. **6**(4), 427–456 (2012)
15. Nelson, G., Oppen, D.C.: Simplification by cooperating decision procedures. ACM Trans. Program. Lang. Syst. **1**(2), 245–257 (1979)
16. Nicolini, E., Ringeissen, C., Rusinowitch, M.: Combinable extensions of Abelian groups. In: Schmidt, R.A. (ed.) 22nd International Conference on Automated Deduction (CADE-22), Montreal, Canada, volume 5663 of LNCS, pp. 51–66. Springer (2009)
17. Nicolini, E., Ringeissen, C., Rusinowitch, M.: Combining satisfiability procedures for unions of theories with a shared counting operator. Fundam. Inform. **105**(1–2), 163–187 (2010)
18. Oppen, D.C.: Reasoning about recursively defined data structures. J. ACM **27**(3), 403–411 (1980)
19. Pham, T., Gacek, A., Whalen, M.W.: Reasoning about algebraic data types with abstractions. J. Autom. Reason. **57**(4), 281–318 (2016)
20. Ranise, S., Ringeissen, C., Zarba, C.G.: Combining data structures with nonstably infinite theories using many-sorted logic. In: Gramlich, B. (ed.) Frontiers of Combining Systems (FroCoS), volume 3717 of LNCS, pp. 48–64. Springer (2005)
21. Shostak, R.E.: A practical decision procedure for arithmetic with function symbols. J. ACM **26**(2), 351–360 (1979)
22. Sofronie-Stokkermans, V.: Locality results for certain extensions of theories with bridging functions. In: Schmidt, R.A. (ed.) 22nd International Conference on Automated Deduction (CADE-22), Montreal, Canada, volume 5663 of LNCS, pp. 67–83. Springer (2009)
23. Sofronie-Stokkermans, V.: Automated reasoning in extensions of theories of constructors with recursively defined functions and homomorphisms. In: Ball, T., Giesl, J., Hähnle, R., Nipkow, T. (eds.) Interaction versus Automation: The Two Faces of Deduction, number 09411 in Dagstuhl Seminar Proceedings. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, Germany (2010)
24. Suter, P., Dotta, M., Kuncak, V.: Decision procedures for algebraic data types with abstractions. In: Hermenegildo, M.V., Palsberg, J. (eds.) Principles of Programming Languages (POPL), pp. 199–210. ACM, New York (2010)
25. Suter, P., Köksal, A.S., Kuncak, V.: Satisfiability modulo recursive programs. In: Yahav, E. (ed.) 18th International Symposium on Static Analysis (SAS), Venice, Italy, volume 6887 of LNCS, pp. 298–315. Springer (2011)
26. Tinelli, C.: Cooperation of background reasoners in theory reasoning by residue sharing. J. Autom. Reason. **30**(1), 1–31 (2003)
27. Tinelli, C., Harandi, M.T.: A new correctness proof of the Nelson–Oppen combination procedure. In: Baader, F., Schulz, K.U. (eds.) Frontiers of Combining Systems (FroCoS). Applied Logic, pp. 103–120. Kluwer Academic Publishers, Dordrecht (1996)
28. Tinelli, C., Ringeissen, C.: Unions of non-disjoint theories and combinations of satisfiability procedures. Theor. Comput. Sci. **290**(1), 291–353 (2003)
29. Tran, D., Ringeissen, C., Ranise, S., Kirchner, H.: Combination of convex theories: modularity, deduction completeness, and explanation. J. Symb. Comput. **45**(2), 261–286 (2010)

30. Wies, T., Piskac, R., Kuncak, V.: Combining theories with shared set operations. In: Ghilardi, S., Sebastiani, R. (eds.) Frontiers of Combining Systems (FroCoS), volume 5749 of LNCS, pp. 366–382. Springer (2009)
31. Zarba, C.G.: Combining lists with integers. In: International Joint Conference on Automated Reasoning (Short Papers), Technical Report DII 11/01, pp. 170–179. University of Siena (2001)
32. Zarba, C.G.: Combining multisets with integers. In: Voronkov, A. (ed.) 18th International Conference on Automated Deduction (CADE-18), Copenhagen, Denmark, volume 2392 of LNCS, pp. 363–376. Springer (2002)
33. Zarba, C.G.: Combining sets with cardinals. J. Autom. Reason. **34**(1), 1–29 (2005)
34. Zhang, T., Sipma, H.B., Manna, Z.: Decision procedures for term algebras with integer constraints. Inf. Comput. **204**(10), 1526–1574 (2006)