Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI-21)
Sister Conferences Best Papers Track

Politeness for the Theory of Algebraic Datatypes (Extended Abstract) *

Ying Sheng[1] , **Yoni Zohar**[1] , **Christophe Ringeissen**[2] , **Jane Lange**[1,4] , **Pascal Fontaine**[2,3] and **Clark Barrett**[1]
[1]Stanford University
[2]Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France
[3]Université de Liège, Belgium
[4]MIT
ying1123@stanford.edu, yoniz@cs.stanford.edu, Christophe.Ringeissen@loria.fr, jlange20@stanford.edu, Pascal.Fontaine@uliege.be, barrett@cs.stanford.edu

Abstract

Algebraic datatypes, and among them lists and trees, have attracted a lot of interest in automated reasoning and Satisfiability Modulo Theories (SMT). Since its latest stable version, the SMT-LIB standard defines a theory of algebraic datatypes, which is currently supported by several mainstream SMT solvers. In this paper, we study this particular theory of datatypes and prove that it is strongly polite, showing also how it can be combined with other arbitrary disjoint theories using polite combination. Our results cover both inductive and finite datatypes, as well as their union. Our proof uses a new, simple, and natural notion of additivity, that enables deducing strong politeness from (weak) politeness.

1 Introduction

Algebraic datatypes are extremely common in many programming languages. Reasoning about them is therefore crucial for modeling and verifying programs. Various decision procedures for algebraic datatypes have been and continue to be developed and employed by formal reasoning tools such as theorem provers and Satisfiability Modulo Theories (SMT) solvers [Barrett *et al.*, 2007; Reynolds *et al.*, 2018; Reynolds and Blanchette, 2017; Kovács *et al.*, 2017; Gutiérrez and Meseguer, 2017; Meseguer, 2018; Hojjat and Rümmer, 2017; Bonacina and Echenim, 2007; Armando *et al.*, 2009]. SMT solvers employ a combination framework that derives a decision procedure for combined theories by using the decision procedure for each theory as a black box. This is useful, as many applications require reasoning about combined theories. For example, the verification of a simple program may require reasoning about arithmetic, bit-vectors, datatypes, etc. To integrate a new theory into such a framework, it suffices to focus on the decision procedure of the new theory, and its interface to the generic combination framework. The interface is what we focus on in this paper.

The traditional combination method of Nelson and Oppen [Nelson and Oppen, 1979] is applicable for the combination of the theory of datatypes with other theories, as long as the other theories are *stably infinite*. Some theories of interest, however, are not stably infinite, the most notable one being the theory of fixed-width bit-vectors, which is commonly used for modeling and verifying both hardware and software. To be able to perform combinations with such theories, a more general combination method was designed [Ranise *et al.*, 2005], which relies on *polite theories*. This notion was later strengthened to *strongly polite theories* [Jovanovic and Barrett, 2010], which are needed to properly handle variable arrangements in the theory combination mechanism. Strongly polite theories can be combined with any other disjoint decidable theories. While strong politeness was already proven for several useful theories (such as equality, arrays, sets, multisets [Ranise *et al.*, 2005]), strong politeness of algebraic datatypes remained an unanswered question.

The main contribution of this paper is an affirmative answer to this question. The main challenge in proving politeness is producing a *witness* function that preserves equivalence but allows for minimal satisfying models. We introduce a witness function that essentially "guesses" the right constructors of variables without an explicit constructor in the formula. We show how to "shrink" any model of a formula that is the output of this function to obtain a minimal model. The witness function, as well as the model construction, can be used by any SMT solver that includes the theory of datatypes and implements polite theory combination. We introduce and use the notion of additive witnesses, which allows us to prove politeness and conclude strong politeness. We further study the theory of datatypes beyond politeness and extend a decision procedure for a subset of this theory presented in [Chocron *et al.*, 2020] to support the full theory.

2 Datatypes and Politeness

2.1 The Theory of Datatypes

We use the definition of the theory of algebraic datatypes as it appears in the SMT-LIB 2 standard [Barrett *et al.*, 2010]. Our

*Originally published as Politeness for the Theory of Algebraic Datatypes at the 10th edition of the International Joint Conference on Automated Reasoning (IJCAR) in Paris, France.

formalization is based on [Barrett *et al.*, 2007], but adjusted to suit our investigation of politeness. For formal definitions, we refer the reader to [Sheng *et al.*, 2020]. Here we provide only a summary of the definitions.

A signature Σ consists of sort, function and predicate symbols. We use $vars_\sigma(\varphi)$ to denote the set of variables with sort σ in formula φ. $\mathcal{F}_\Sigma$ denotes the set of function symbols of Σ and $\mathcal{S}_\Sigma$ denotes the set of sorts in Σ. A *datatypes signature* further splits the set of sorts into a set of element sorts denoted $\mathbf{Elem}$, and a set of struct sorts denoted $\mathbf{Struct}$, with finitely many constructors (used in formulas like $y = c(w_1, ..., w_k)$) and selectors (used in formulas like $y = s_{c,i}(x)$) as function symbols, and testers (used in formulas like $is_c(x)$) as predicate symbols. In what follows, $\mathcal{CO}$ stands for the constructors in Σ and $\mathcal{SE}$ for its selectors.

Interpretations and satisfaction are defined as usual. For a Σ-interpretation $\mathcal{A}$, $\gamma^\mathcal{A}$ denotes the interpretation of γ in $\mathcal{A}$, where γ can be a sort, a variable, or a function/predicate symbol, or a set of these. A Σ-theory is a class of Σ-interpretations.

Intuitively, the interpretations in the theory of datatypes correspond to term models generated by the constructors. The interpretation of testers is as expected, and the interpretation of selectors is also as expected for terms that were built with the corresponding constructor (otherwise, selectors are freely interpreted). We use $\mathcal{T}_\Sigma$ to denote the theory of datatypes based on the datatypes signature Σ.

Example 1. *The signature Σ_{list} has sorts* $\mathbf{elem}$ *and* $\mathbf{list}$*. Its function symbols are cons of arity* $(\mathbf{elem} \times \mathbf{list}) \to \mathbf{list}$*, nil of arity* $\mathbf{list}$*, car of arity* $\mathbf{list} \to \mathbf{elem}$ *and cdr of arity* $\mathbf{list} \to \mathbf{list}$*. Its predicate symbols are is_{nil} and is_{cons}, both of arity* $\mathbf{list}$*. It is a datatypes signature, with* $\mathbf{Elem} = \{\mathbf{elem}\}$, $\mathbf{Struct} = \{\mathbf{list}\}$, $\mathcal{CO} = \{nil, cons\}$ *and* $\mathcal{SE} = \{car, cdr\}$*. car represents the head of the list and cdr represents its tail. nil represents the empty list.*

In the theory of datatypes, a sort σ is *inductive* if for any natural number k, there is a term with sort σ that applies k nested constructors. We call such a term a struct term with depth k. For example, $cons(a, cons(a, cons(a, nil)))$ has depth 3. A non-inductive sort is called *finite*.

2.2 Politeness

Given two theories T_1 and T_2, polite theory combination requires that T_1 and T_2 agree on the cardinality of their respective models, and also there must be an agreement between T_1 and T_2 on the interpretation of formulas built over the equality symbol. The following definition will be useful.

Definition 1 (Arrangement)**.** *Let V be a finite set of variables whose sorts are in S and $\{V_\sigma \mid \sigma \in S\}$ a partition of V such that V_σ is the set of variables of sort σ in V. We say that a formula δ is an* arrangement of V *if $\delta = \bigwedge_{\sigma \in S}(\bigwedge_{(x,y)\in E_\sigma}(x = y) \wedge \bigwedge_{(x,y)\notin E_\sigma}(x \neq y))$, where E_σ is some equivalence relation over V_σ for each $\sigma \in S$.*

Definition 2 (Smoothness)**.** *Let Σ be a signature, S a subset of sorts in Σ, and T a Σ-theory. T is* smooth *w.r.t. S if for every quantifier-free formula ϕ, T-interpretation $\mathcal{A} \models \phi$, and function κ from S to the class of cardinals such that $\kappa(\sigma) \geq \left|\sigma^\mathcal{A}\right|$ for every $\sigma \in S$, there exists a Σ-interpretation $\mathcal{A}'$ that satisfies ϕ with $\left|\sigma^{\mathcal{A}'}\right| = \kappa(\sigma)$ for every $\sigma \in S$.*

Let ϕ be a quantifier-free (QF) Σ-formula. We say that a function $wtn : QF(\Sigma) \to QF(\Sigma)$ is a *strong witness for T w.r.t. S* if for every $\phi \in QF(\Sigma)$ we have that:

1. ϕ and $\exists \overrightarrow{w}.\ wtn(\phi)$ are T-equivalent for $\overrightarrow{w} = vars(wtn(\phi)) \setminus vars(\phi)$; and
2. if $wtn(\phi) \wedge \delta_V$ is T-satisfiable, for an arrangement δ_V, where V is a set of variables whose sorts are in S, then there exists a T-interpretation $\mathcal{A}$ satisfying $wtn(\phi) \wedge \delta_V$ such that $\sigma^\mathcal{A} = vars_\sigma(wtn(\phi) \wedge \delta_V)^\mathcal{A}$, for all $\sigma \in S$. We also say $\Gamma = wtn(\phi) \wedge \delta_V$ is finitely witnessed for T w.r.t. S and $\mathcal{A}$ is a finite witness of Γ for T w.r.t. S.

Correspondingly, the definition of a *witness for T w.r.t. S* is obtained by dismissing V and δ_V, and then replacing every occurrence of $wtn(\phi) \wedge \delta_V$ with $wtn(\phi)$.

Definition 3 (Finitely Witnessable)**.** *The theory T is* (strongly) finitely witnessable *w.r.t. S if there exists a (strong) witness for T w.r.t. S which is computable.*

Definition 4 (Polite)**.** *T is called* (strongly) polite w.r.t. *S if it is smooth and (strongly) finitely witnessable w.r.t. S.*

3 Additive Witnesses

It was shown in [Jovanovic and Barrett, 2010] that politeness is not sufficient for the proof of the polite combination method from [Ranise *et al.*, 2005]. Strong politeness was introduced to fix the problem. In this section we offer a simple (yet useful) criterion for the equivalence of the two notions. Let Σ and S denote an arbitrary signature and a subset of its sorts, and let T denote an arbitrary Σ-theory.

Definition 5 (Additivity)**.** *Let $f : QF(\Sigma) \to QF(\Sigma)$. We say that f is S-additive for T if $f(f(\phi) \wedge \varphi)$ and $f(\phi) \wedge \varphi$ are T-equivalent and have the same set of S-sorted variables for every $\phi, \varphi \in QF(\Sigma)$, provided that φ is a conjunction of flat literals such that every term in φ is a variable whose sort is in S. When T is clear from the context, we say that f is S-additive.*

We show that additive witnesses are strong:

Proposition 1. *Let wtn be a witness for T w.r.t. S. If wtn is S-additive then it is a strong witness for T w.r.t. S.*

The concept of additive witnesses will be useful in the following section, where we define a witness and instead of proving that it is a strong witness, we only need to prove that it is additive.

4 Politeness for the SMT-LIB 2 Theory of Datatypes

Let Σ be a datatypes signature with $\mathcal{S}_\Sigma = \mathbf{Elem} \uplus \mathbf{Struct}$ and $\mathcal{F}_\Sigma = \mathcal{CO} \uplus \mathcal{SE}$. In this section, we prove that $\mathcal{T}_\Sigma$ is strongly polite with respect to $\mathbf{Elem}$. In Section 4.1, we consider theories with only inductive sorts; we consider theories with only finite sorts in Section 4.2. We combine them in Section 4.3, where arbitrary theories of datatypes are considered. For smoothness, however, it is straightforward to show

that the $\mathbf{Elem}$ domain of a given interpretation can always be augmented without changing satisfiability of quantifier-free formulas, for both finite and inductive sorts.

Lemma 1. *$\mathcal{T}_\Sigma$ is smooth w.r.t.* $\mathbf{Elem}$.

4.1 Inductive Datatypes

In this section, we assume that all sorts in $\mathbf{Struct}$ are inductive. To prove finite witnessability, we now introduce an additive witness function. Following arguments from [Ranise *et al.*, 2005], it suffices to define the witness only for conjunctions of flat literals. Similarly, it suffices to show a function $wtn_i(\phi)$ that is finitely witnessed for ϕ which is a conjunction of flat literals. Essentially, our witness guesses possible constructors for variables whose constructors are not explicit in the input formula.

Definition 6 (A Witness for $\mathcal{T}_\Sigma$)**.** *Let ϕ be a quantifier-free conjunction of flat Σ-literals. $wtn_i(\phi)$ is obtained from ϕ by performing the following steps:*

1. *For any literal of the form $y = s_{c,i}(x)$ such that $x = c(\overrightarrow{u_1}, y, \overrightarrow{u_2})$ does not occur in ϕ and $x = d(\overrightarrow{u_d})$ does not occur in ϕ for any $\overrightarrow{u_1}, \overrightarrow{u_2}, \overrightarrow{u_d}$, we conjunctively add $x = c(\overrightarrow{u_1}, y, \overrightarrow{u_2}) \vee (\bigvee_{d \neq c} x = d(\overrightarrow{u_d}))$ with fresh $\overrightarrow{u_1}, \overrightarrow{u_2}, \overrightarrow{u_d}$, where c and d range over $\mathcal{CO}$.*
2. *For any literal of the form $is_c(x)$ such that $x = c(\overrightarrow{u})$ does not occur in ϕ for any $\overrightarrow{u}$, we conjunctively add $x = c(\overrightarrow{u})$ with fresh $\overrightarrow{u}$.*
3. *For any literal of the form $\neg is_c(x)$ such that $x = d(\overrightarrow{u_d})$ does not occur in ϕ for any $d \neq c$ and $\overrightarrow{u_d}$, we conjunctively add $\bigvee_{d \neq c} x = d(\overrightarrow{u_d})$, with fresh $\overrightarrow{u_d}$.*
4. *For any sort $\sigma \in \mathbf{Elem}$ such that ϕ does not include a variable of sort σ we conjunctively add a literal $x = x$ for a fresh variable x of sort σ.*

The requirement for absence of literals before adding literals or disjunctions to ϕ is used to ensure additivity of wtn_i. And indeed:

Lemma 2. *wtn_i is* $\mathbf{Elem}$*-additive.*

Further, it can be verified that:

Lemma 3. *Let ϕ be a conjunction of flat literals. ϕ and $\exists \overrightarrow{w}.\Gamma$ are $\mathcal{T}_\Sigma$-equivalent, where $\Gamma = wtn_i(\phi)$ and $\overrightarrow{w} = vars(\Gamma) \setminus vars(\phi)$.*

The remainder of this section is dedicated to the proof of the following lemma:

Lemma 4 (Finite Witnessability)**.** *Let ϕ be a conjunction of flat literals. Then, $\Gamma = wtn_i(\phi)$ is finitely witnessed for $\mathcal{T}_\Sigma$ with respect to* $\mathbf{Elem}$.

Suppose that Γ is $\mathcal{T}_\Sigma$-satisfiable, and let $\mathcal{A}$ be a satisfying $\mathcal{T}_\Sigma$-interpretation. We define a $\mathcal{T}_\Sigma$-interpretation $\mathcal{B}$ as follows, and then show that $\mathcal{B}$ is a finite witness of Γ for $\mathcal{T}_\Sigma$ w.r.t. $\mathbf{Elem}$. First, for every $\sigma \in \mathbf{Elem}$, we set $\sigma^\mathcal{B} = vars_\sigma(\Gamma)^\mathcal{A}$, and for every variable $e \in vars_\sigma(\Gamma)$, we set $e^\mathcal{B} = e^\mathcal{A}$. The interpretations of $\mathbf{Struct}$-sorts, testers, and constructors are uniquely determined by the theory. It is left to define the interpretation of $\mathbf{Struct}$-variables in $\mathcal{B}$, as well as the interpretation of the selectors (the interpretation of selectors is fixed by the theory only when applied to the "right" constructor). We do this in several steps:

Step 1 – Simplifying Γ: since ϕ is a conjunction of flat literals, Γ is a conjunction whose conjuncts are either flat literals or disjunctions of flat literals (introduced in Items 1 and 3 of Definition 6). Since $\mathcal{A} \models \Gamma$, $\mathcal{A}$ satisfies exactly one disjunct of each such disjunction. We can thus obtain a formula Γ_1 from Γ by replacing every disjunction with the disjunct that is satisfied by $\mathcal{A}$. Notice that $\mathcal{A} \models \Gamma_1$ and that it is a conjunction of flat literals. Let Γ_2 be obtained from Γ_1 by removing any literal of the form $is_c(x)$ and any literal of the form $\neg is_c(x)$. Let Γ_3 be obtained from Γ_2 by removing any literal of the form $x = s_{c,i}(y)$. For convenience, we denote Γ_3 by Γ'. Obviously, $\mathcal{A} \models \Gamma'$, and Γ' is a conjunction of flat literals without selectors and testers.

Step 2 – Working with Equivalence Classes: We would like to preserve equalities between $\mathbf{Struct}$-variables from $\mathcal{A}$. To this end, we group all variables in $vars(\Gamma)$ into equivalence classes according to their interpretation in $\mathcal{A}$. Let $\equiv_\mathcal{A}$ denote an equivalence relation over $vars(\Gamma)$ such that $x \equiv_\mathcal{A} y$ iff $x^\mathcal{A} = y^\mathcal{A}$. We denote by $[x]$ the equivalence class of x. Let α be an equivalence class; then, $\alpha^\mathcal{A} = \{x^\mathcal{A} \mid x \in \alpha\}$ is a singleton. Identifying this singleton with its only element, we have that $\alpha^\mathcal{A}$ denotes $a^\mathcal{A}$ for an arbitrary element a of the equivalence class α.

Step 3 – Ordering Equivalence Classes: We would also like to preserve disequalities between $\mathbf{Struct}$-variables from $\mathcal{A}$. Thus, we introduce a relation $\prec$ over the equivalence classes, such that $\alpha \prec \beta$ if $y = c(w_1, \ldots, w_n)$ occurs as one of the conjuncts in Γ' for some $w_1, \ldots, w_n$, $y \in \beta$, $w_k \in \alpha$ and $c \in \mathcal{CO}$. Call an equivalence class α *nullary* if $\mathcal{A} \models is_c(x)$ for some $x \in \alpha$ and nullary constructor c. Call an equivalence class α *minimal* if $\beta \not\prec \alpha$ for every β. Notice that each nullary equivalence class is minimal. The relation $\prec$ induces a directed acyclic graph (DAG), denoted G. The vertices are the equivalence classes. Whenever $\alpha \prec \beta$, we draw an edge from vertex α to β.

Step 4 – Interpretation of Equivalence Classes: We define $\alpha^\mathcal{B}$ for every equivalence class α. Then, $x^\mathcal{B}$ is simply defined as $[x]^\mathcal{B}$, for every $\mathbf{Struct}$-variable x. Let m be the number of equivalence classes, l the number of minimal equivalence classes, r the number of nullary equivalence classes, and $\alpha_1, \ldots, \alpha_m$ a topological sort of G, such that all minimal classes occur before all others, and the first r classes are nullary. Let d be the length of the longest path in G. We define $\alpha_i^\mathcal{B}$ by induction on i.

1. If $0 < r$ and $i \leq r$ then α_i is a nullary class and so we set $\alpha_i^\mathcal{B} = \alpha_i^\mathcal{A}$.
2. If $r < i \leq l$ then α_i is minimal and not nullary. Let σ be the sort of variables in α_i. If $\sigma \in \mathbf{Elem}$, then all variables in the class have already been defined. Otherwise, $\sigma \in \mathbf{Struct}$. In this case, we define $\alpha_i^\mathcal{B}$ to be the interpretation of an arbitrary struct term with depth strictly greater than $\max\{depth(\alpha_j^\mathcal{B}) \mid 0 < j < i\} + d$ (here $\max \emptyset = 0$).
3. If $i > l$ then we set $\alpha_i^\mathcal{B} = c(\beta_1^\mathcal{B}, \ldots, \beta_n^\mathcal{B})$ for the unique equivalence classes $\beta_1, \ldots, \beta_n \subseteq \{\alpha_1, \ldots, \alpha_{i-1}\}$ and

c such that $y = c(x_1, \ldots, x_n)$ occurs in Γ' for some $y \in \alpha_i$ and $x_1 \in \beta_1, \ldots, x_n \in \beta_n$.

Since Σ is a datatypes signature in which all $\mathbf{Struct}$-sorts are inductive, the second case of the definition is well-defined. Further, the topological sort ensures $\beta_1, \ldots, \beta_n$ exist, and the partition into equivalence classes ensures that they are unique.

Step 5 – Interpretation of Selectors: Let $s_{c,i} \in \mathcal{SE}$ for $c : \sigma_1 \times \ldots \times \sigma_n \to \sigma$, $1 \leq i \leq n$ and $a \in \sigma^{\mathcal{B}}$. If $a \in is_c^{\mathcal{B}}$, we must have $a = c(a_1, \ldots, a_n)$ for some $a_1 \in \sigma_1^{\mathcal{B}}, \ldots, a_n \in \sigma_n^{\mathcal{B}}$. We then set $s_{c,i}^{\mathcal{B}}(a) = a_i$. Otherwise, we consider two cases. If $x^{\mathcal{B}} = a$ for some $x \in vars(\Gamma)$ such that $y = s_{c,i}(x)$ occurs in Γ_2 for some y, we set $s_{c,i}^{\mathcal{B}}(a) = y^{\mathcal{B}}$. Otherwise, $s_{c,i}^{\mathcal{B}}(a)$ is set arbitrarily.

Now that $\mathcal{B}$ is defined, it is left to show that it is a finite witness of Γ for $\mathcal{T}_\Sigma$ w.r.t. $\mathbf{Elem}$. By construction, $\sigma^{\mathcal{B}} = vars_\sigma(\Gamma)^{\mathcal{B}}$ for every $\sigma \in \mathbf{Elem}$. $\mathcal{B}$ also preserves the equalities and disequalities in $\mathcal{A}$, and by considering every shape of a literal in Γ' we can prove that $\mathcal{B} \models \Gamma'$. Our interpretation of the selectors then ensures that:

Lemma 5. $\mathcal{B} \models \Gamma$.

Lemma 5, together with the definition of the domains of $\mathcal{B}$, gives us that $\mathcal{B}$ is a finite witness of Γ for $\mathcal{T}_\Sigma$ w.r.t. $\mathbf{Elem}$, and so Lemma 4 is proven. As a corollary of Lemmas 1, 2 and 4, strong politeness is obtained.

Theorem 1. *If Σ is a datatypes signature and all sorts in $\mathbf{Struct}_\Sigma$ are inductive, then $\mathcal{T}_\Sigma$ is strongly polite w.r.t. $\mathbf{Elem}_\Sigma$.*

4.2 Finite Datatypes

In this section, we assume that all sorts in $\mathbf{Struct}$ are finite.

For finite witnessability, we define the following witness, that guesses the construction of each $\mathbf{Struct}$-variables until a fixpoint is reached. For every quantifier-free conjunction of flat Σ-literals ϕ, define the sequence $\phi_0, \phi_1, \ldots$, such that $\phi_0 = \phi$, and for every $i \geq 0$, ϕ_{i+1} is obtained from ϕ_i by conjoining it with a disjunction $\bigvee_{c \in \mathcal{CO}} x = c(w_1^c, \ldots, w_{n_c}^c)$ for fresh $w_1^c, \ldots, w_{n_c}^c$, where x is some arbitrary $\mathbf{Struct}$-variable in ϕ_i such that there is no literal of the form $x = c(y_1, \ldots, y_n)$ in ϕ_i for any constructor c and variables $y_1, \ldots, y_n$, if such x exists. Since $\mathbf{Struct}$ only has finite sorts, this sequence becomes constant at some ϕ_k.

Definition 7 (A Witness for $\mathcal{T}_\Sigma$)**.** *$wtn_f(\phi)$ is ϕ_k for the minimal k such that $\phi_k = \phi_{k+1}$.*

As in Section 4.1, we have that wtn_f is a strong witness for $\mathcal{T}_\Sigma$ w.r.t. $\mathbf{Elem}$ and hence:

Theorem 2. *If Σ is a datatypes signature and all sorts in $\mathbf{Struct}_\Sigma$ are finite, then $\mathcal{T}_\Sigma$ is strongly polite w.r.t. $\mathbf{Elem}_\Sigma$.*

4.3 Combining Finite and Inductive Datatypes

For the general case, let Σ be a datatypes signature. We prove that $\mathcal{T}_\Sigma$ is strongly polite w.r.t. $\mathbf{Elem}$ by showing that there are datatype signatures $\Sigma_1, \Sigma_2 \subseteq \Sigma$ such that $\mathcal{T}_\Sigma = \mathcal{T}_{\Sigma_1} \oplus \mathcal{T}_{\Sigma_2}$, and then using Theorem 1 from [Jovanovic and Barrett, 2010] about the preservation of politeness when combining polite theories.

Theorem 3. *If Σ is a datatypes signature then $\mathcal{T}_\Sigma$ is strongly polite w.r.t. $\mathbf{Elem}_\Sigma$.*

5 Axiomatizations

In this section, we show how to reduce any $\mathcal{T}_\Sigma$-satisfiability problem to a satisfiability problem modulo an axiomatized theory of trees. The latter can be decided using syntactic unification [Baader *et al.*, 2001].

Let Σ be a datatypes signature. The set $TREE_\Sigma^*$ of axioms contains all the axioms as follows for $c, d \in \mathcal{CO}, c \neq d$: (where free variables are implicitly universally quantified, t is a non-variable term containing x that is selector-free, and $\sigma \in \mathbf{Struct}$).

$(Dis)\ c(x_1, \ldots, x_n) \neq d(Y_1, \ldots, Y_m)$ $\quad$ $(Acyc)\ x \neq t[x]$
$(Is_1)\ is_c(c(\overrightarrow{x}))$ $\quad$ $(Is_2)\ \neg is_c(d(\overrightarrow{y}))$
$(Ext_1)\ \exists \overrightarrow{y}\ .\ is_c(x) \to x = c(\overrightarrow{y})$
$(Ext_2)\ \vee_{c \in \mathcal{CO}}\ is_c(x)$
$(Proj)\ s_{c,i}(c(x_1, \ldots, x_n)) = x_i, i \in [1, n]$
$(Inj)\ c(x_1, \ldots, x_n) = c(y_1, \ldots, y_n) \to \bigwedge_{i=1}^n x_i = y_i$

Let $TREE_\Sigma$ be the set obtained from $TREE_\Sigma^*$ by dismissing Ext_1 and Ext_2. $TREE_\Sigma$ is a generalization of the theory of Absolutely Free Data Structures (AFDS) from [Chocron *et al.*, 2020] to many-sorted signatures with selectors and testers. In what follows we identify $TREE_\Sigma$ (and $TREE_\Sigma^*$) with the class of structures that satisfy them when there is no ambiguity.

Proposition 2. *Every $TREE_\Sigma^*$-unsatisfiable formula is $\mathcal{T}_\Sigma$-unsatisfiable.*

The following result shows that any $\mathcal{T}_\Sigma$-satisfiability problem can be reduced to a $TREE_\Sigma$-satisfiability problem. This leads to a $\mathcal{T}_\Sigma$-satisfiability procedure.

Proposition 3. *Let Σ be a finite datatype signature and φ any conjunction of flat Σ-literals including an arrangement over the variables in φ. Then, there exists a Σ-formula φ' such that:*

1. *φ and $\exists \overrightarrow{w}\ .\ \varphi'$ are $\mathcal{T}_\Sigma$-equivalent, where $\overrightarrow{w} = vars(\varphi') \backslash vars(\varphi)$.*
2. *φ' is $\mathcal{T}_\Sigma$-satisfiable iff φ' is $TREE_\Sigma$-satisfiable.*

Proposition 3 can be easily lifted to any conjunction of Σ-literals φ by flattening and then guessing all possible arrangements over the variables. Further, $\exists \overrightarrow{w}\ .\ \varphi'$ and φ are not only $\mathcal{T}_\Sigma$-equivalent but also $TREE_\Sigma^*$-equivalent. As a consequence, Proposition 3 also holds when stated using $TREE_\Sigma^*$ instead of $\mathcal{T}_\Sigma$.

6 Conclusion

In this paper, we have studied the theory of algebraic datatypes, as it is defined by the SMT-LIB 2 standard. Our investigation included both finite and inductive datatypes. We have proved that this theory is strongly polite, making it amenable for combination with other theories using the polite combination method. Our proofs used the notion of additive witnesses, also introduced in this paper. We concluded by extending existing axiomatizations to support this theory of datatypes.

References

[Armando *et al.*, 2009] Alessandro Armando, Maria Paola Bonacina, Silvio Ranise, and Stephan Schulz. New results on rewrite-based satisfiability procedures. *ACM Trans. Comput. Log.*, 10(1):4:1–4:51, 2009.

[Baader *et al.*, 2001] Franz Baader, Wayne Snyder, Paliath Narendran, Manfred Schmidt-Schauß, and Klaus U. Schulz. Unification theory. In John Alan Robinson and Andrei Voronkov, editors, *Handbook of Automated Reasoning (in 2 volumes)*, pages 445–532. Elsevier and MIT Press, 2001.

[Barrett *et al.*, 2007] Clark W. Barrett, Igor Shikanian, and Cesare Tinelli. An abstract decision procedure for a theory of inductive data types. *Journal on Satisfiability, Boolean Modeling and Computation*, 3(1-2):21–46, 2007.

[Barrett *et al.*, 2010] Clark Barrett, Aaron Stump, and Cesare Tinelli. The SMT-LIB Standard: Version 2.0. In A. Gupta and D. Kroening, editors, *Proceedings of the 8th International Workshop on Satisfiability Modulo Theories (Edinburgh, UK)*, 2010.

[Bonacina and Echenim, 2007] Maria Paola Bonacina and Mnacho Echenim. Rewrite-based satisfiability procedures for recursive data structures. *Electron. Notes Theor. Comput. Sci.*, 174(8):55–70, 2007.

[Chocron *et al.*, 2020] Paula Chocron, Pascal Fontaine, and Christophe Ringeissen. Politeness and combination methods for theories with bridging functions. *J. Autom. Reasoning*, 64(1):97–134, 2020.

[Gutiérrez and Meseguer, 2017] Raúl Gutiérrez and José Meseguer. Variant-based decidable satisfiability in initial algebras with predicates. In Fabio Fioravanti and John P. Gallagher, editors, *Logic-Based Program Synthesis and Transformation - 27th International Symposium, LOPSTR 2017, Namur, Belgium, October 10-12, 2017, Revised Selected Papers*, volume 10855 of *Lecture Notes in Computer Science*, pages 306–322. Springer, 2017.

[Hojjat and Rümmer, 2017] Hossein Hojjat and Philipp Rümmer. Deciding and interpolating algebraic data types by reduction. In Tudor Jebelean, Viorel Negru, Dana Petcu, Daniela Zaharie, Tetsuo Ida, and Stephen M. Watt, editors, *19th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, SYNASC 2017, Timisoara, Romania, September 21-24, 2017*, pages 145–152. IEEE Computer Society, 2017.

[Jovanovic and Barrett, 2010] Dejan Jovanovic and Clark W. Barrett. Polite theories revisited. In Christian G. Fermüller and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning - 17th International Conference, LPAR-17, Yogyakarta, Indonesia, October 10-15, 2010. Proceedings*, volume 6397 of *Lecture Notes in Computer Science*, pages 402–416. Springer, 2010. Extended technical report is available at http://theory.stanford.edu/~barrett/pubs/JB10-TR.pdf.

[Kovács *et al.*, 2017] Laura Kovács, Simon Robillard, and Andrei Voronkov. Coming to terms with quantified reasoning. In Giuseppe Castagna and Andrew D. Gordon, editors, *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages, POPL 2017, Paris, France, January 18-20, 2017*, pages 260–270. ACM, 2017.

[Meseguer, 2018] José Meseguer. Variant-based satisfiability in initial algebras. *Sci. Comput. Program.*, 154:3–41, 2018.

[Nelson and Oppen, 1979] Greg Nelson and Derek C. Oppen. Simplification by cooperating decision procedures. *ACM Trans. Program. Lang. Syst.*, 1(2):245–257, 1979.

[Ranise *et al.*, 2005] Silvio Ranise, Christophe Ringeissen, and Calogero G. Zarba. Combining data structures with nonstably infinite theories using many-sorted logic. In Bernhard Gramlich, editor, *Frontiers of Combining Systems, 5th International Workshop, FroCoS 2005, Vienna, Austria, September 19-21, 2005, Proceedings*, volume 3717 of *Lecture Notes in Computer Science*, pages 48–64. Springer, 2005. Extended technical report is available at https://hal.inria.fr/inria-00070335/.

[Reynolds and Blanchette, 2017] Andrew Reynolds and Jasmin Christian Blanchette. A decision procedure for (co)datatypes in SMT solvers. *J. Autom. Reasoning*, 58(3):341–362, 2017.

[Reynolds *et al.*, 2018] Andrew Reynolds, Arjun Viswanathan, Haniel Barbosa, Cesare Tinelli, and Clark W. Barrett. Datatypes with shared selectors. In Didier Galmiche, Stephan Schulz, and Roberto Sebastiani, editors, *Automated Reasoning - 9th International Joint Conference, IJCAR 2018, Held as Part of the Federated Logic Conference, FloC 2018, Oxford, UK, July 14-17, 2018, Proceedings*, volume 10900 of *Lecture Notes in Computer Science*, pages 591–608. Springer, 2018.

[Sheng *et al.*, 2020] Ying Sheng, Yoni Zohar, Christophe Ringeissen, Jane Lange, Pascal Fontaine, and Clark W. Barrett. Politeness for the theory of algebraic datatypes. In Nicolas Peltier and Viorica Sofronie-Stokkermans, editors, *Automated Reasoning - 10th International Joint Conference, IJCAR 2020, Paris, France, July 1-4, 2020, Proceedings, Part I*, volume 12166 of *Lecture Notes in Computer Science*, pages 238–255. Springer, 2020.