On automated reasoning about recursively defined functions and homomorphisms

Viorica Sofronie-Stokkermans

Max Planck Institut für Informatik, Saarbruecken, Germany
sofronie@mpi-inf.mpg.de

Abstract. We study possibilities of reasoning about extensions of base theories with functions which satisfy certain recursion (or homomorphism) properties. Our focus is on emphasizing possibilities of hierarchical and modular reasoning in such extensions and combinations thereof. We present practical applications in verification and cryptography.

Keywords. Combinations of decision procedures, Hierarchical reasoning, Recursive functions, Homomorphisms.

1 Introduction

In this paper we study possibilities of reasoning in extensions of theories with functions which satisfy certain recursion (or homomorphism) axioms. This type of axioms is very important in verification – for instance in situations in which we need to reason about functions defined by certain forms of primitive recursion, such as for instance the function computing the size of a tree formed using a binary constructor c and a constant c_0:

$$\begin{cases} \mathsf{size}(c_0) = 1 \\ \mathsf{size}(c(x_1, x_2)) = 1 + \mathsf{size}(x_1) + \mathsf{size}(x_2) \end{cases}$$

and in cryptography, where one may need to model homomorphism axioms of the form

$$\forall x, y, z(\mathsf{encode}_z(x * y) = \mathsf{encode}_z(x) * \mathsf{encode}_z(y)).$$

Decision procedures for recursive data structures exist. In [13], Oppen gave a PTIME decision procedure for absolutely free data structures based on bidirectional closure; methods which use rewriting and/or basic equational reasoning were given e.g. by Barrett et al. [2] and Bonacina and Echenim [3]. Some extensions of theories with recursively defined functions and homomorphisms have also been studied. In [1], Armando, Rusinowitch, and Ranise give a decision procedure for a theory of homomorphisms. In [19], Zhang, Manna and Sipma give a decision procedure for the extension of a theory of term structures with a recursively defined length function. In [8] tail recursive definitions are studied. It is proved that tail recursive definitions can be expressed by shallow axioms and therefore define so-called “)-5.8887(stably local extensions”. Locality properties have

Dagstuhl Seminar Proceedings 09411
”Interaction versus Automation: The two Faces of Deduction”
http://drops.dagstuhl.de/opus/volltexte/2010/2424

also been studied in a series of papers on the analysis of cryptographic protocols (cf. e.g. [4,5,6]).

In this paper we show that many extensions with recursive definitions (or with generalized homomorphism properties) satisfy locality conditions. This allows us to significantly extend existing results on reasoning about functions defined using certain forms of recursion, or satisfying homomorphism properties [1,8,19], and at the same time shows how powerful and widely applicable the concept of local theory (extension) is in automated reasoning. As a by-product, the methods we use provide a possibility of presenting in a different light (and in a different form) locality phenomena studied in cryptography in [4,5,6]; we believe that they will allow to better separate rewriting from proving, and thus to give simpler proofs.

The main results are summarized below:

(1) We show that the theory of absolutely free constructors is local, and locality is preserved also in the presence of selectors. These results are consistent with existing decision procedures for this theory [13] which use a variant of bi-directional closure in a graph formed starting from the subterms of the set of clauses whose satisfiability is being checked.
(2) We show that, under certain assumptions, extensions of the theory of absolutely free constructors with functions satisfying a certain type of recursion axioms satisfy locality properties, and show that for functions with values in an ordered domain we can combine recursive definitions with boundedness axioms without sacrificing locality. We also address the problem of only considering models whose data part is the *initial* term algebra of such theories.
(3) We analyze conditions which ensure that similar results can be obtained if we relax some assumptions about the absolute freeness of the underlying theory of data types, and illustrate the ideas on an example from cryptography.

The locality results we establish allow us to reduce the task of reasoning about the class of recursive functions we consider to reasoning in the underlying theory of data structures (possibly combined with the theories associated with the co-domains of the recursive functions). This paper is an extended version of [18].

Structure of the paper. In Section 2 we present the results on local theory extensions and hierarchical reasoning in local theory extensions needed in the paper. We start Section 3 by considering theories of absolutely free data structures, and extensions of such theories with selectors. We prove locality results for such theories, and for variants thereof in which the acyclicity axioms are ommitted for some of the constructors. In Section 4 we consider extensions of theories of absolutely free constructors with functions defined using certain types of recursion axioms (we also consider functions having values in a different – e.g. numeric – domain). We show that in these cases locality results can also be established. In Section 5 we show that similar results can be obtained if we relax some assumptions about the absolute freeness of the underlying theory of data types. In Section 6 we illustrate the ideas on a simple example from cryptography.

2 Preliminaries

We start with presenting some definitions and results needed in the paper.

2.1 Theories and theory extensions

We will consider theories over possibly many-sorted signatures $\Pi = (S, \Sigma, \mathsf{Pred})$, where S is a set of sorts, Σ a set of function symbols, and Pred a set of predicate symbols. For each function $f \in \Sigma$ (resp. predicate $P \in \mathsf{Pred}$), we denote by $a(f) = s_1, \dots, s_n \rightarrow s$ (resp. $a(P) = s_1, \dots, s_n$) its arity, where $s_1, \dots, s_n, s \in S$, and $n \geq 0$. In the one-sorted case we will simply write $a(f) = n$ (resp. $a(P) = n$).

First-order theories are sets of formulae (closed under logical consequence), typically the set of all consequences of a set of axioms. When referring to a theory, we can also consider the set of all its models. We here consider theories specified by their sets of axioms, but – usually when talking about local extensions of a theory – we will refer to a theory, and mean the set of all its models.

Theory extensions. We here also consider *extensions of theories*, in which the signature is extended by new *function symbols* (i.e. we assume that the set of predicate symbols remains unchanged in the extension[1]). Let $\mathcal{T}_0$ be an arbitrary theory with signature $\Pi_0 = (S, \Sigma_0, \mathsf{Pred})$. We consider extensions $\mathcal{T}_1$ of $\mathcal{T}_0$ with signature $\Pi = (S, \Sigma, \mathsf{Pred})$, where the set of function symbols is $\Sigma = \Sigma_0 \cup \Sigma_1$. We assume that $\mathcal{T}_1$ is obtained from $\mathcal{T}_0$ by adding a set $\mathcal{K}$ of (universally quantified) clauses in the signature Π.

2.2 Total and partial models

Let $\Pi = (S, \Sigma, \mathsf{Pred})$. A *partial Π-structure* is a structure

$$(\{A_s\}_{s \in S}, \{f_A\}_{f \in \Sigma}, \{P_A\}_{P \in \mathsf{Pred}})$$

in which for every $f \in \Sigma$, with $a(f) = s_1, \dots, s_n \rightarrow s$, f_A is a (possibly partially defined) function from $A_{s_1} \times \dots \times A_{s_n}$ to A_s, and for every $P \in \mathsf{Pred}$ with arity $a(P) = s_1 \dots s_n$, $P_A \subseteq A_{s_1} \times \dots \times A_{s_n}$.

Definition 1 *A* weak Π-embedding *between partial structures* $A = (\{A_s\}_{s \in S}, \{f_A\}_{f \in \Sigma}, \{P_A\}_{P \in \mathsf{Pred}})$ *and* $B = (\{B_s\}_{s \in S}, \{f_B\}_{f \in \Sigma}, \{P_B\}_{P \in \mathsf{Pred}})$ *is an S-sorted family* $i = (i_s)_{s \in S}$ *of injective maps* $i_s : A_s \rightarrow B_s$ *which is an embedding w.r.t.* Pred*, s.t. if* $a(f) = s_1, \dots, s_n \rightarrow s$ *and* $f_A(a_1, \dots, a_n)$ *is defined then* $f_B(i_{s_1}(a_1), \dots, i_{s_n}(a_n))$ *is defined and* $i_s(f_A(a_1, \dots, a_n)) = f_B(i_{s_1}(a_1), \dots, i_{s_n}(a_n))$.

We now define truth and satisfiability in partial structures of Π-literals and (sets of) clauses with variables in a set X.

[1] In a many-sorted framework we can regard predicates as functions of boolean output sort, thus the framework presented here can, in fact, be also used for considering extensions with new predicate symbols.

Definition 2 *If A is a partial structure, $\beta : X \to A$ is a valuation*[2] *and $L = (\neg)P(t_1, \dots, t_n)$ is a literal (with $P \in \mathsf{Pred} \cup \{=\}$) we say that $(A, \beta) \models_w L$ if*

(i) either $\beta(t_i)$ are all defined and $(\neg)P_A(\beta(t_1), \dots, \beta(t_n))$ is true in A, or
(ii) $\beta(t_i)$ is not defined for some argument t_i of P.

Weak satisfaction of clauses ($(A, \beta) \models_w C$) is defined in the usual way. A is a weak partial model *of a set $\mathcal{K}$ of clauses if $(A, \beta) {\models_w} C$ for every $\beta : X \to A$ and every clause $C \in \mathcal{K}$.*

Definition 3 *A* weak partial model *of $\mathcal{T}_0 \cup \mathcal{K}$ is a weak partial model of $\mathcal{K}$ whose reduct to Π_0 is a total model of $\mathcal{T}_0$.*

2.3 Local theories and local theory extensions

Local theories. The notion of *local theory* was introduced in [9,10] by Givan and McAllester. A *local theory* is a set of Horn clauses $\mathcal{K}$ such that, for any ground Horn clause C, $\mathcal{K} \models C$ only if already $\mathcal{K}[C] \models C$ (where $\mathcal{K}[C]$ is the set of instances of $\mathcal{K}$ in which all terms are subterms of ground terms in either $\mathcal{K}$ or $\mathcal{C}$). The size of $\mathcal{K}[G]$ is polynomial in the size of G for a fixed $\mathcal{K}$. Since satisfiability of sets of ground Horn clauses can be checked in linear time, it follows that for local theories, validity of ground Horn clauses can be checked in polynomial time. Givan and McAllester proved that every problem which is decidable in PTIME can be encoded as an entailment problem of ground clauses w.r.t. a local theory [10]. In [7], Ganzinger established a link between proof theoretic and semantic concepts for polynomial time decidability of uniform word problems which had already been studied in algebra. He defined two notions of locality for equational Horn theories, and established relationships between these notions of locality and corresponding semantic conditions, referring to embeddability of partial algebras into total algebras.

Local theory extensions. We now consider extensions of theories in which the signature is extended by new *function symbols*.

Let $\mathcal{T}_0$ be an arbitrary theory with signature $\Pi_0 = (S, \Sigma_0, \mathsf{Pred})$. We consider extensions $\mathcal{T}_1$ of $\mathcal{T}_0$ with signature $\Pi = (S, \Sigma, \mathsf{Pred})$, where the set of function symbols is $\Sigma = \Sigma_0 \cup \Sigma_1$. We assume that $\mathcal{T}_1$ is obtained from $\mathcal{T}_0$ by adding a set $\mathcal{K}$ of (universally quantified) clauses in the signature Π.

Consider the following condition (in what follows we refer to sets G of ground clauses and assume that they are in the signature $\Pi^c = (S, \Sigma \cup \Sigma_c, \mathsf{Pred})$, where Σ_c is a set of new constants):

(Loc) For every finite set G of ground clauses $\mathcal{T}_1 \cup G \models \perp$ iff $\mathcal{T}_0 \cup \mathcal{K}[G] \cup G$ has no weak partial model with all terms in $\mathsf{st}(\mathcal{K}, G)$ defined

where if T is a set of terms, $\mathcal{K}[T]$ is the set of instances of $\mathcal{K}$ in which all terms starting with a symbol in Σ_1 are in T, and $\mathcal{K}[G] := \mathcal{K}[\mathsf{st}(\mathcal{K}, G)]$, where $\mathsf{st}(\mathcal{K}, G)$ is the family of all subterms of ground terms in $\mathcal{K}$ or G.

[2] We denote the canonical extension to terms of a valuation $\beta : X \to A$ again by β.

Definition 4 *We say that an extension* $\mathcal{T}_0 \subseteq \mathcal{T}_1$ *is* local *if it satisfies condition* (Loc). *We say that it is* local for clauses with a property P *if it satisfies the locality conditions for all ground clauses* G *with property* P.

A more general locality condition (ELoc) refers to situations when $\mathcal{K}$ consists of formulae $(\Phi(x_1, \dots, x_n) \vee C(x_1, \dots, x_n))$, where $\Phi(x_1, \dots, x_n)$ is a *first-order* Π_0*-formula* with free variables $x_1, \dots, x_n$, and $C(x_1, \dots, x_n)$ is a *clause* in the signature Π. The free variables $x_1, \dots, x_n$ of such an axiom are considered to be universally quantified [14].

(ELoc) For every formula $\Gamma = \Gamma_0 \cup G$, where Γ_0 is a Π_0^c-sentence and G is a finite set of ground Π^c-clauses, $\mathcal{T}_1 \cup \Gamma \models\perp$ iff $\mathcal{T}_0 \cup \mathcal{K}[G] \cup \Gamma$ has no weak partial model in which all terms in $\mathsf{st}(\mathcal{K}, G)$ are defined.

A more general notion, namely Ψ-locality of a theory extension (in which the instances to be considered are described by a closure operation Ψ) is introduced in [11].

Definition 5 *Let* $\mathcal{K}$ *be a set of clauses. Let* $\Psi_{\mathcal{K}}$ *be a closure operation associating with any set* T *of ground terms a set* $\Psi_{\mathcal{K}}(T)$ *of ground terms such that all ground subterms in* $\mathcal{K}$ *and* T *are in* $\Psi_{\mathcal{K}}(T)$. *Let* $\Psi_{\mathcal{K}}(G) := \Psi_{\mathcal{K}}(\mathsf{st}(\mathcal{K}, G))$. *We say that the extension* $\mathcal{T}_0 \subseteq \mathcal{T}_0 \cup \mathcal{K}$ *is* Ψ*-local if it satisfies:*

(Loc^{Ψ}) *for every finite set* G *of ground clauses,* $\mathcal{T}_0 \cup \mathcal{K} \cup G \models\perp$ *iff* $\mathcal{T}_0 \cup \mathcal{K}[\Psi_{\mathcal{K}}(G)] \cup G$ *has no weak partial model in which all terms in* $\Psi_{\mathcal{K}}(G)$ *are defined.*

(ELoc^{Ψ}) is defined analogously. In (Ψ-)local theories and extensions satisfying (ELoc^{Ψ}), hierarchical reasoning is possible.

Theorem 6 ([14,11]) *Let* $\mathcal{K}$ *be a set of clauses. Assume that* $\mathcal{T}_0 \subseteq \mathcal{T}_1 = \mathcal{T}_0 \cup \mathcal{K}$ *is a* Ψ*-local theory extension, and that for every finite set* T *of terms* $\Psi_{\mathcal{K}}(T)$ *is finite. For any set* G *of ground clauses, let* $\mathcal{K}_0 \cup G_0 \cup \mathsf{Def}$ *be obtained from* $\mathcal{K}[\Psi_{\mathcal{K}}(G)] \cup G$ *by flattening and purification*[3]. *Then the following are equivalent:*

(1) G *is satisfiable w.r.t.* $\mathcal{T}_1$.
(2) $\mathcal{T}_0 \cup \mathcal{K}[\Psi_{\mathcal{K}}(G)] \cup G$ *has a partial model with all terms in* $\mathsf{st}(\mathcal{K}, G)$ *defined.*
(3) $\mathcal{T}_0 \cup \mathcal{K}_0 \cup G_0 \cup \mathsf{Con}[G]_0$ *has a (total) model, where*

$$\mathsf{Con}[G]_0 = \{\bigwedge_{i=1}^{n} c_i = d_i \rightarrow c = d \mid f(c_1, \dots, c_n) = c, f(d_1, \dots, d_n) = d \in \mathsf{Def}\}.$$

[3] $\mathcal{K}[\Psi_{\mathcal{K}}(G)] \cup G$ can be flattened and purified by introducing, in a bottom-up manner, new constants c_t for subterms $t = f(g_1, \dots, g_n)$ with $f \in \Sigma_1$, g_i ground $\Sigma_0 \cup \Sigma_c$-terms (where Σ_c is a set of constants which contains the constants introduced by flattening, resp. purification), together with corresponding definitions $c_t = t$. We obtain a set of clauses $\mathcal{K}_0 \cup G_0 \cup \mathsf{Def}$, where Def consists of ground unit clauses of the form $f(g_1, \dots, g_n) = c$, where $f \in \Sigma_1$, c is a constant, $g_1, \dots, g_n$ are ground $\Sigma_0 \cup \Sigma_c$-terms, and $\mathcal{K}_0$ and G_0 are $\Sigma_0 \cup \Sigma_c$-clauses. Flattening and purification preserve satisfiability and unsatisfiability w.r.t. total algebras, and w.r.t. partial algebras in which all ground subterms which are flattened are defined [14]. In what follows, we explicitly indicate the sorts of the constraints in Def by using indices, i.e. $\mathsf{Def} = \bigcup_{s \in S} \mathsf{Def}_s$.

A similar hierarchical reduction to satisfiability tests in $\mathcal{T}_0$ can be proved for theory extensions satisfying conditions (ELoc and (ELoc^{Ψ} and sets Γ of Π-formulae satisfying the conditions in the definition of condition (ELoc.

Theorem 6 allows us to transfer decidability and complexity results from the theory $\mathcal{T}_0$ to the theory $\mathcal{T}_1$:

Theorem 7 ([14]) *Assume that the extension $\mathcal{T}_0 \subseteq \mathcal{T}_1$ satisfies condition (Loc^{Ψ}) – where Ψ has the property that $\Psi(T)$ is finite for every finite T – and that every variable in any clause of $\mathcal{K}$ occurs below some function symbol from Σ_1.*

(1) If testing satisfiability of ground clauses in $\mathcal{T}_0$ is decidable, then so is testing satisfiability of ground clauses in $\mathcal{T}_1$.
(2) Assume that the complexity of testing the satisfiability w.r.t. $\mathcal{T}_0$ of a set of ground clauses of size m can be described by a function $g(m)$. Let G be a set of $\mathcal{T}_1$-clauses such that $\Psi_{\mathcal{K}}(G)$ has size n. Then the complexity of checking the satisfiability of G w.r.t. $\mathcal{T}_1$ is of order $g(n^k)$, where k is the maximum number of free variables in a clause in $\mathcal{K}$ (but at least 2).

A similar transfer of decidability and parameterized complexity results can be obtained for theory extensions satisfying condition (ELoc) or (ELoc^{Ψ})

Theorem 8 ([11]) *Assume that $\mathcal{K}$ consists of axioms of the form $\overline{C} = (\Phi_C(\overline{x}) \vee C(\overline{x}))$, where $\Phi_C(\overline{x})$ is in a fragment (class of formulae) $\mathcal{F}$ of $\mathcal{T}_0$ and $C(\overline{x})$ is a Π-clause, and $\Gamma = \Gamma_0 \wedge G$, where Γ_0 is a formula in $\mathcal{F}$ without free variables, and G is a set of ground Π^c-clauses, both containing constants in Σ_c. Assume that the theory extension $\mathcal{T}_0 \subseteq \mathcal{T}_1$ satisfies (ELoc), or (ELoc^{Ψ}).*

Satisfiability of formulae of the form $\Gamma_0 \cup G$ as above w.r.t. $\mathcal{T}_1$ is decidable provided $\mathcal{K}[G]$ (resp. $\mathcal{K}[\Psi_{\mathcal{K}}(G)]$) is finite and $\mathcal{K}_0 \cup G_0 \cup \Gamma_0 \cup N_0$ belongs to a decidable fragment of $\mathcal{T}_0$.

Locality allows us to obtain parameterized decidability and complexity results:

Case 1: If for each $\overline{C} = \Phi_C(\overline{x}) \vee C(\overline{x}) \in \mathcal{K}$ all free variables occur below some extension symbol, then $\mathcal{K}[G]$ (resp. $\mathcal{K}[\Psi_{\mathcal{K}}(G)]$) contains only formulae of the form $\Phi_C(\overline{g}) \vee C(\overline{g})$, where $\overline{g}$ consists of ground Σ_0-terms, so $\mathcal{K}_0 \cup G_0 \cup \Gamma_0 \cup N_0 \in \mathcal{F}_g$, the class obtained instantiating all free variables of formulae in $\mathcal{F}$ with ground Σ_0-terms.

Decidability and complexity: If checking satisfiability for the class $\mathcal{F}_g$ w.r.t. $\mathcal{T}_0$ is decidable, then checking satisfiability of goals of the form above w.r.t. $\mathcal{T}_1$ is decidable. Assume that the complexity of a decision procedure for the fragment $\mathcal{F}_g$ of $\mathcal{T}_0$ is $g(n)$ for an input of size n. Let m be the size of $\mathcal{K}_0 \cup G_0 \cup \Gamma_0 \cup N_0$. Then the complexity of proving satisfiability of $\Gamma_0 \cup G$ w.r.t. $\mathcal{T}_1$ is of order $g(m)$.

For local extensions, $\mathcal{K}_0 = (\mathcal{K}[G])_0$; the size m of $\mathcal{K}_0 \cup G_0 \cup \Gamma_0 \cup N_0$ is of order $|G|^k$ for some $2 \leq k \in \mathbb{Z}$ for a fixed $\mathcal{K}$ (at least quadratic because of N_0).

Similarly for Ψ-local extensions (with $\mathsf{st}(\mathcal{K}, G)$ replaced by $\Psi_{\mathcal{K}}(G)$, and $|G|^k$ replaced by $|\Psi(G)|^k$).

Case 2: If not all free variables in $\mathcal{K}$ occur below an extension symbol, then the instances in $\mathcal{K}[G]$ (resp. $\mathcal{K}[\Psi_{\mathcal{K}}(G)]$) contain free variables, so $\mathcal{K}_0 \cup G_0 \cup \Gamma_0 \cup N_0$ is in the universal closure $\forall \mathcal{F}$ of $\mathcal{F}$. The decidability and complexity remarks above apply w.r.t. the complexity of checking satisfiability of formulae in the fragment $\forall \mathcal{F}$ of $\mathcal{T}_0$ with constants in Σ_c (viewed as existentially quantified variables).

2.4 Examples of local extensions

The locality of an extension can either be proved directly, or by proving embeddability of partial into total models. We give here some examples which will be used later in the paper.

Theorem 9 ([14,16,11,17]) *The following theory extensions are local:*

(1) Any extension of a theory with free function symbols;
(2) Extensions of any base theory $\mathcal{T}_0$ with functions satisfying axioms of the form

$$\mathsf{GBounded}(f) \qquad \bigwedge_{i=1}^n (\phi_i(\overline{x}) \rightarrow s_i \leq f(\overline{x}) \leq t_i)$$

where Π_0 contains a sort s for which a reflexive binary relation $\leq$ exists, s_i, t_i are Σ_0-terms of sort s and ϕ_i are Π_0-formulae s.t. for $i \neq j$, $\phi_i \wedge \phi_j \models_{\mathcal{T}_0} \perp$, and $\mathcal{T}_0 \models \forall \overline{x}(\phi_i(\overline{x}) \rightarrow s_i(\overline{x}) \leq t_i(\overline{x}))$.

3 Theories of constructors and selectors

Let $\mathsf{AbsFree}_{\Sigma_0} = (\bigcup_{c \in \Sigma_0} (\mathsf{Inj}_c) \cup (\mathsf{Acyc}_c)) \cup \bigcup_{\substack{c,d \in \Sigma \\ c \neq d}} \mathsf{Disjoint}(c, d)$, where:

$$\begin{array}{rll}
(\mathsf{Inj}_c) & c(x_1, \dots, x_n) = c(y_1, \dots, y_n) \rightarrow \bigwedge\limits_{i=1}^n x_i = y_i & \\
(\mathsf{Acyc}_c) & c(t_1, \dots, t_n) \neq x \text{ if } x \text{ occurs in some } t_i & \\
\mathsf{Disjoint}(c, d) & c(x_1, \dots, x_n) \neq d(y_1, \dots, y_k) & \text{if } c \neq d
\end{array}$$

Note that (Acyc_c) is an axiom schema (representing an infinite set of axioms).

Theorem 10 *The following theories are local:*

(1) Theories of one constructor:
  (a) The theory $\mathsf{AbsFree}_c$ of an absolutely free constructor.
  (b) The theory (Inj_c) of an injective constructor.
  (c) Theories $\mathsf{AbsFree}_c \cup \bigcup_{i=1}^n \mathsf{Sel}(s_i, c)$ of an absolutely free constructor c and selectors $s_1, \dots, s_n$ corresponding to c, satisfying the axioms:

$$\mathsf{Sel}(s_i, c) \qquad \forall x, x_1, \dots, x_n \quad x = c(x_1, \dots, x_n) \rightarrow s_i(x) = x_i.$$

  (d) Theories $(\mathsf{Inj}_c) \cup \bigcup_{i=1}^n \mathsf{Sel}(s_i, c)$ of an injective constructor c and selectors $s_1, \dots, s_n$ corresponding to c.

(2) Theories of several constructors:

(a) The theory $\mathsf{AbsFree}_{\Sigma_0}$ *of absolutely free constructors in* Σ_0.
(b) Any theory $\mathsf{AbsFree}_{\Sigma_0 \setminus \Sigma}$ *obtained from* $\mathsf{AbsFree}_{\Sigma_0}$ *by dropping the acyclicity condition for a set* $\Sigma \subseteq \Sigma_0$ *of constructors.*
(c) $\mathcal{T} \cup \mathsf{Sel}(\Sigma')$*, where* $\mathcal{T}$ *is one of the theories in 2(a) or 2(b), and* $\mathsf{Sel}(\Sigma') = \bigcup_{c \in \Sigma'} \bigcup_{i=1}^{n} \mathsf{Sel}(s_i^c, c)$ *axiomatizes a family of selectors* $s_1^c, \dots, s_n^c$*, where* $n = a(c)$*, corresponding to constructors* $c \in \Sigma' \subseteq \Sigma_0$.

In addition, $\mathcal{K} = \mathsf{AbsFree}_{\Sigma_0} \cup \mathsf{Sel}(\Sigma_0) \cup \mathsf{IsC}$*, where*

$$(\mathsf{IsC}) \quad \forall x \bigvee_{c \in \Sigma_0} x = c(s_1^c(x), \dots, s_{a(c)}^c(x))$$

has the property that for every set G *of ground* $\Sigma_0 \cup \mathsf{Sel} \cup \Sigma_c$*-clauses (where* Σ_c *is a set of additional constants),* $\mathcal{K} \wedge G \models \perp$ *iff* $\mathcal{K}[\Psi(G)] \wedge G \models \perp$*, where* $\Psi(G) = \mathsf{st}(G) \cup \bigcup_{a \in \Sigma_c \cap \mathsf{st}(G)} \bigcup_{c \in \Sigma_0} (\{s_i^c(a) \mid 1 {\leq} i {\leq} a(c)\} \cup \{c(s_1^c(a), \dots, s_n^c(a))\})$.

Proof: (1) The result is proved by showing that every weak partial model of the axioms for (a)–(d) weakly embeds into a total model of the axioms. The locality then follows from the link between embeddability and locality established in [7].

(1.a) Let P be a partial algebra which weakly satisfies the axioms $(\mathsf{Inj}_c) \cup (\mathsf{Acyc}_c)$ of an absolutely free constructor. Starting from the elements of P seen as (different) constants, we define the following term rewrite system:

$$c(p_1, \dots, p_n) \rightarrow p \text{ if } c_P(p_1, \dots, p_n) \text{ is defined and equal to } p.$$

It is easy to see that this term rewrite system is convergent, as it is noetherian (since the terms are finite, there are no infinite rewrite chains) and there are no critical pairs, and that the canonical form $t{\downarrow}$ of a term t is an element in P iff all the subterms of t are defined in P (this can be proved by induction on the length of the term, taking into account the form of rewrite rules).
Let $T_c(P)$ be the term $\{c\}$-algebra freely generated by the elements of P. Let $\equiv_P$ be the equivalence relation generated by the rewrite relation $\rightarrow$ on $T_c(P)$, and let $i : P \rightarrow T_c(P)/{\equiv_P}$ be the canonical projection, which associates with every element of P its equivalence class. Note that the canonical extension of i to terms associates with every term t the equivalence class $[t{\downarrow}]$ of its canonical form $t{\downarrow}$ modulo $\rightarrow$ in $T_c(P)$. We show that

(i) i is a (weak) embedding and
(ii) $T_c(P)/{\equiv_P}$ satisfies the axiom Inj_c and
(iii) $T_c(P)/{\equiv_P}$ satisfies the axiom Acyc_c.

(i) The injectivity of i is a consequence of the fact that every element in P is already in canonical form. Thus, for every $p, q \in P$ if $i(p) = i(q)$ then $[p{\downarrow}] = [q{\downarrow}]$, so $p = p{\downarrow} = q{\downarrow} = q$. Assume now that $c_P(p_1, \dots, p_n)$ is defined in P and equals p. Then $[c(p_1, \dots, p_n)] = [p]$. Therefore, $i(c_P(p_1, \dots, p_n)) = [p] = [c(p_1, \dots, p_n)] = c(i(p_1), \dots, i(p_n))$.

(ii) We show that Inj_c holds. Let $t_1, \dots, t_n, s_1, \dots, s_n \in T_c(P)$ be such that $[c(t_1, \dots, t_n)] = [c(s_1, \dots, s_n)]$ (i.e. the canonical forms of the two terms w.r.t.

$\rightarrow$ are equal). We prove that $[t_i] = [s_i]$ for all $i = 1, \dots, n$. We proceed by induction on the length of the (common) canonical form of $c(t_1, \dots, t_n)$ and $c(s_1, \dots, s_n)$. Assume first that $c(t_1, \dots, t_n)\!\downarrow = c(s_1, \dots, s_n)\!\downarrow = p \in P$. Then $c_P(t_{1P}, \dots, t_{nP})$ is defined, so is $c_P(s_{1P}, \dots, s_{nP})$, and they are both equal to p. As P weakly satisfies Inj_c it follows that t_{iP} and s_{iP} are defined and equal for all i, hence $[t_i] = [s_i]$, for all $i = 1, \dots, n$. Assume now that $c_P(t_{1P}, \dots, t_{nP})$ is not defined. Then $c_P(s_{1P}, \dots, s_{nP})$ is undefined and $c(t_1, \dots, t_n)\!\downarrow = c(t_1\!\downarrow, \dots, t_n\!\downarrow)$, $c(s_1, \dots, s_n)\!\downarrow = c(s_1\!\downarrow, \dots, s_n\!\downarrow)$, and therefore $t_i\!\downarrow = s_i\!\downarrow$ for all $i = 1, \dots, n$, i.e. $[s_i] = [t_i]$ for $i = 1, \dots, n$.

(iii) We now prove that Acyc_c holds. Let $X = \{x_1, \dots, x_m\}$ be the variables in $c(t_1, \dots, t_n)$, and let $x = x_i$ be one of these variables. Let $v : X \rightarrow T_c(P)/\!\equiv_P$ be defined by $v(x_j) = [s_j]$, where $s_1, \dots, s_m \in T_c(P)$. Assume $[c(v(t_1), \dots, v(t_n))] = [s_i]$, i.e. (if we denote $v(t_i)$ by t'_i) $[c(t'_1, \dots, t'_n)] = [s_i]$ and $[s_i] = [s'_j]$ for some j and some subterm s'_j of t'_j. We show that if $t = c(t'_1, \dots, t'_n)$ and s is a subterm of some t'_i then t and s must have different canonical forms. We proceed by induction on the length of the canonical form of $c(t'_1, \dots, t'_n)$.

Assume first that $c(t'_1, \dots, t'_n)\!\downarrow = p \in P$. Then $c_P(t'_{1P}, \dots, t'_{nP})$ is defined (hence also s_{iP} is defined). As P weakly satisfies the axiom schema Acyc_c, it follows that $c_P(t'_{1P}, \dots, t'_{nP}) \neq s_{iP}$. Contradiction.

If $c_P(t_{1P}, \dots, t_{nP})$ is not defined, then $c(t'_1, \dots, t'_n)\!\downarrow = c(t'_1\!\downarrow, \dots, t'_n\!\downarrow) = s_i\!\downarrow$. Therefore, s_{iP} is also undefined in P, and $s_i\!\downarrow = c(u_1, \dots, u_n)$, where $u_j = t_j\!\downarrow$ for all $j = 1, \dots, n$. We know that $[s_i] = [s'_j]$ for some j and some subterm s'_j of t'_j. Then $s_i\!\downarrow = c(u_1, \dots, u_n) = s'_j\!\downarrow$. As s_{iP} is undefined P, s'_j is also undefined in P. As s'_j is a subterm of t'_j, $u_j = t'_j\!\downarrow$, and s'_j is undefined in P, we know that the canonical form of s'_j, $c(u_1, \dots, u_n)$, is a subterm of the canonical form $c(t'_1\!\downarrow, \dots, t'_n\!\downarrow)$ of t'_j, i.e. that u_j is a proper subterm of $t'_j\!\downarrow$. By the induction hypothesis, $[t'_j] \neq [u_j]$, so it follows that $c([t'_1], \dots, [t'_n]) \neq c([u_1], \dots, [u_k]) = [s_i]$.

(1.b) Follows from the proof of (1.a), since for proving that $T_c(P)/\!\equiv_P$ satisfies Inj_c only the fact that P weakly satisfies Inj_c is needed.

(1.c) Assume that on P also partial selectors $s_1, \dots, s_n$ are defined. We extend them to total functions on $T_c(P)/\!\equiv_P$ by defining

$$\overline{s}^c_i(x) = \begin{cases} [t_i] & \text{if } x = [c(t_1, \dots, t_n)] \\ [s^c_i(p)] & \text{if } x = [p], p \in P \text{ and } s^c_i(p) \text{ is defined} \\ a_c & \text{otherwise,} \end{cases}$$

where a_c is an arbitrary, but fixed, element of $T_c(P)/\!\equiv_P$.

We first show that $\overline{s}^c_i$ is well-defined. Assume that $[c(t_1, \dots, t_n)] = [p]$ and $s^c_i(p)$ is defined. Then p is the canonical form of $c(t_1, \dots, t_n)$ w.r.t. the term rewrite system defined starting from P at the beginning of the proof of 1(a), so there exist $p_i \in P$ such that $t_i\!\downarrow = p_1$ and $c_P(p_1, \dots, p_n) = p$. Since P is a weak model of the selector axioms, it follows that $s^c_i(p) = p_i = t_i\!\downarrow$.

Assume now that $[c(t_1, \dots, t_n)] = [c(s_1, \dots, s_n)]$. Then, by injectivity of c, $[t_i] = [s_i]$.

From the definition, it is easy to see that $\overline{s}_i^c$ satisfies all selector axioms.

We need to show that the canonical projection $i : P \rightarrow T_c(P)/{\equiv_P}$ is also a weak homomorphism w.r.t. the selectors s_i^c. This follows from the fact that if $s_i^c(p)$ is defined in P then $i(s_i^c(p)) = [s_i^c(p)] = \overline{s}_i^c([p])$.

(1.d) Follows from the proof of (1.c) which only uses the injectivity of c.

(2) can be proved analogously. All the constructions are similar in all cases (2.a), (2.b) and (2.c). In addition to (1) we need to show that $T_\Sigma(P)/{\equiv_P}$ has the property that for $c \neq d \in \Sigma$, $c([t_1], \dots, [t_n]) \neq d([s_1], \dots, [s_m])$ for all terms $t_1, \dots, t_n, s_1, \dots, s_m$. Assume that there exist terms $t_1, \dots, t_n, s_1, \dots, s_m$ such that $c([t_1], \dots, [t_n]) = [c(t_1, \dots, t_n)] = [d(s_1, \dots, s_n)] = d([s_1], \dots, [s_m])$. Thus, $c(t_1, \dots, t_n)$ and $d(s_1, \dots, s_n)$ have the same canonical form. This can happen only if the canonical form is an element $p \in P$. Thus, t_{iP}, s_{jP} are all defined in P and $c_P(t_{1P}, \dots, t_{nP}) = p = d_P(s_{1P}, \dots, s_{mP})$. This means that there exist elements in P which contradict axiom ($\mathsf{Disjoint}(c, d)$). Contradiction.

In order to prove the last claim in the theorem, assume that $\mathcal{K}[\Psi(G)] \wedge G$ has a weak partial model P in which all terms in $\Psi(G)$ are defined. Let $\overline{P} = \{t_P \mid t \in T_{\Sigma_0}(\Sigma_c \cap \mathsf{st}(G))\}$. Let $c_{\overline{P}}(t_P^1, \dots, t_P^n)$ be defined iff $c_P(t_P^1, \dots, t_P^n)$ is defined (and in this case they are equal). It is easy to see that $\overline{P}$ is itself a weak partial model of $\mathcal{K}[\Psi(G)] \wedge G$ in which all terms in $\Psi(G)$ are defined. We complete $\overline{P}$ as showed previously to a total model $T_{\Sigma_0}(\overline{P})/{\equiv_{\overline{P}}}$ of $\mathsf{AbsFree}_{\Sigma_0} \cup \mathsf{Sel}(\Sigma_0)$. We show that it also satisfies IsC. Let $x \in T_{\Sigma_0}(\overline{P})/{\equiv_{\overline{P}}}$. Then $x = [t]$, where $t \in T_{\Sigma_0}(\overline{P})$, i.e. $x = t$, where $t \in T_{\Sigma_0}(\{a_P \mid a \in \Sigma_c\})$. If $t = c(t_1, \dots, t_n)$ for some $c \in \Sigma_0$ then obviously $x = [t] = [c(s_1^c(t), \dots, s_n^c(t))]$. Assume now that $t = a_P \in \Sigma_c \cap \mathsf{st}(G)$. Since we know that P weakly satisfies $\mathsf{IsC}[\Psi(G)]$, and $c(s_1^c(a), \dots, s_n^c(a)) \in \Psi(G)$ for every $c \in \Sigma_c$, $a_P = c_P(s_1^c(a_P), \dots, s_n^c(a_P))$ for some $c \in \Sigma_0$. Thus, $T_{\Sigma_0}(\overline{P})/{\equiv_{\overline{P}}}$ is a model of IsC. $\square$

The reduction to the pure theory of equality made possible by Theorem 10 is very similar to Oppen's method [13] for deciding satisfiability of ground formulae for free recursive data structures by bi-directional closure. Quantifier elimination (cf. [13]) followed by the reduction enabled by Theorem 10 can be used to obtain a decision procedure for the first-order theory of absolutely free constructors axiomatized by $\mathsf{AbsFree}_{\Sigma_0} \cup \mathsf{Sel}(\Sigma_0) \cup \mathsf{IsC}$.

4 Theories of absolutely free constructors and recursively defined functions

We consider extensions of $\mathsf{AbsFree}_{\Sigma_0}$ with new function symbols, possibly with codomain of a different sort, i.e. theories over the signature $S = \{d, s_1, \dots, s_n\}$, where d is the "data" sort; we do not impose any restriction on the nature of the sorts in s_i (some may be equal to d). The function symbols are:

- constructors $c \in \Sigma$ (arity $d^n \rightarrow d$), and corresponding selectors s_i^c (arity $d \rightarrow d$);
- all functions Σ_{s_i} in the signature of the theory of sort s_i, for $i = 1, \dots, n$;

– for every $1 \leq i \leq n$, a set Σ_i of functions of sort $d \to s_i$.

In what follows we will analyze certain such extensions for which decision procedures for ground satisfiability exist[4]. We will assume for simplicity that $S = \{d, s\}$, where d is the "data" sort and s is a different sort (output sort for some of the recursively defined functions).

Let $\mathcal{T}_s$ be a theory of sort s. We consider extensions of the disjoint combination of $\mathsf{AbsFree}_{\Sigma_0}$ and $\mathcal{T}_s$ with functions in a set $\Sigma = \Sigma_1 \cup \Sigma_2$, where the functions in Σ_1 have arity $d \to d$ and those in Σ_2 have arity $d \to s$. We will make the following notational conventions:

– If f has sort $d \to b$, with $b \in S$, we denote its output sort b by $o(f)$.
– $\Sigma_{o(f)}$ denotes Σ_0 if $o(f) = d$, or Σ_s if $o(f) = s$,
– $\mathcal{T}_{o(f)}$ is the theory $\mathsf{AbsFree}_{\Sigma_0}$ if $o(f) = d$, or $\mathcal{T}_s$ if $o(f) = s$.

For every $f \in \Sigma$ we assume that a subset $\Sigma_r(f) \subseteq \Sigma_0$ is specified (a set of constructors for which recursion axioms for f exist).

We consider theories of the form $\mathcal{T} = \mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathsf{Rec}_\Sigma$, where $\mathsf{Rec}_\Sigma = \bigcup_{f \in \Sigma} \mathsf{Rec}_f$ is a set of axioms of the form:

$$\mathsf{Rec}_f \quad \left\{ \begin{array}{rl} f(k) = k_f & \\ f(c(x_1, \dots, x_n)) = g^{c,f}(f(x_1), \dots, f(x_n)) \end{array} \right.$$

where k, c range over all constructors in $\Sigma_r(f) \subseteq \Sigma_0$, with $a(k) = 0, a(c) = n$, k_f are ground $\Sigma_{o(f)}$-terms and the functions $g^{c,f}$ are expressible by $\Sigma_{o(f)}$-terms.

We also consider extensions with a new set of functions satisfying definitions by guarded recursion of the form $\mathsf{Rec}^g_\Sigma = \bigcup_{f \in \Sigma} \mathsf{Rec}^g_f$:

$$\mathsf{Rec}^g_f \quad \left\{ \begin{array}{rl} f(k) = k_f & \\ f(c(x_1, \dots, x_n)) = & \left\{ \begin{array}{ll} g^{c,f}_1(f(x_1), \dots, f(x_n)) & \text{if } \phi_1(f(x_1), \dots, f(x_n)) \\ \dots & \\ g^{c,f}_k(f(x_1), \dots, f(x_n)) & \text{if } \phi_k(f(x_1), \dots, f(x_n)) \end{array} \right. \end{array} \right.$$

where k, c range over all constructors in $\Sigma_r(f) \subseteq \Sigma_0$, with $a(k) = 0, a(c) = n$, k_f are ground $\Sigma_{o(f)}$-terms and the functions $g^{c,f}_i$ are expressible by $\Sigma_{o(f)}$-terms, and $\phi_i(x_1, \dots, x_n)$ are $\Sigma_{o(f)}$-formulae with free variables $x_1, \dots, x_n$, where $\phi_i \wedge \phi_j \models_{\mathcal{T}_{o(f)}} \perp$ for $i \neq j$.

Definition 1. *A definition of type* Rec_f *is* exhaustive *if* $\Sigma_r(f) = \Sigma_0$ *(i.e.* Rec_f *contains recursive definitions for terms starting with any* $c \in \Sigma_0$*). A definition of type* Rec^g_f *is* exhaustive *if* $\Sigma_r(f) = \Sigma_0$ *and for every definition, the disjoint guards* $\phi_1, \dots, \phi_n$ *are exhaustive, i.e.* $\mathcal{T}_{o(f)} \models \forall \overline{x}(\phi_1(\overline{x}) \vee \dots \vee \phi_n(\overline{x}))$. Quasi-exhaustive *definitions are defined similarly, by allowing that* $\Sigma_0 \backslash \Sigma_r(f)$ *may contain constants (but no function of arity greater than, or equal to 1).*

[4] In this paper we only focus on the problem of checking the satisfiability of sets of ground clauses, although it appears that when adding axiom IsC decision procedures for larger fragments can be obtained using arguments similar to those used in [19].

4.1 Examples

We illustrate the type of recursive definitions we consider on the following examples.

Example 1 *Let $\Sigma_0 = \{c_0, c\}$ with $a(c_0) = 0, a(c) = n$. Let $\mathcal{T}_0 = \mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s$ be the disjoint, many-sorted combination of the theory $\mathsf{AbsFree}_{\Sigma_0}$ (sort d) and $\mathcal{T}_{\mathsf{num}}$, the theory of natural numbers with addition (sort num).*

(1) A size function can be axiomatized by $\mathsf{Rec}_{\mathsf{size}}$:

$$\begin{cases} \mathsf{size}(c_0) = 1 \\ \mathsf{size}(c(x_1, \dots, x_n)) = 1 + \mathsf{size}(x_1) + \dots + \mathsf{size}(x_n) \end{cases}$$

(2) A depth function can be axiomatized by the following definition $\mathsf{Rec}^g_{\mathsf{depth}}$:

$$\begin{cases} \mathsf{depth}(c_0) = 1 \\ \mathsf{depth}(c(x_1, \dots, x_n)) = 1 + \mathsf{max}\{\mathsf{depth}(x_1), \dots, \mathsf{depth}(x_n)\} \end{cases}$$

This definition is of type Rec^g because although $\mathsf{max}\{\mathsf{depth}(x_1), \dots, \mathsf{depth}(x_n)\}$ cannot be expressed as a term function, the condition

$$\mathsf{depth}(c(x_1, \dots, x_n)) = 1 + \mathsf{max}\{\mathsf{depth}(x_1), \dots, \mathsf{depth}(x_n)\}$$

can alternatively be expressed as:

$$\mathsf{depth}(c(x_1, \dots, x_n)) = \begin{cases} 1 + \mathsf{depth}(x_1) & \textit{if } \mathsf{depth}(x_1) \geq \mathsf{depth}(x_j) \forall j, 1 < j \leq n \\ \dots & \\ 1 + \mathsf{depth}(x_i) & \textit{if } \mathsf{depth}(x_i) \geq \mathsf{depth}(x_j) \forall j, i < j \leq n \\ & \textit{and } \mathsf{depth}(x_i) > \mathsf{depth}(x_j) \forall j, 1 \leq j \leq i-1 \\ \dots & \\ 1 + \mathsf{depth}(x_n) & \textit{if } \mathsf{depth}(x_n) > \mathsf{depth}(x_j) \forall j, 1 \leq j \leq n-1 \end{cases}$$

Example 2 *Let $\Sigma_0{=}\{c_0, d_0, c\}$ with $a(c_0) = a(d_0) = 0, a(c) = n$, and let $\mathcal{T}_0 = \mathsf{AbsFree}_{\Sigma_0} \cup \mathsf{Bool}$ be the disjoint combination of the theories $\mathsf{AbsFree}_{\Sigma_0}$ (sort d) and Bool, having as model the two-element Boolean algebra $\mathsf{B}_2{=}(\{\mathsf{t}, \mathsf{f}\}, \sqcap, \sqcup, \neg)$ (sort bool) with a function has_{c_0} with output of sort bool, defined by $\mathsf{Rec}_{\mathsf{has}_{c_0}}$:*

$$\begin{cases} \mathsf{has}_{\mathsf{c}_0}(c_0) = \mathsf{t} \\ \mathsf{has}_{\mathsf{c}_0}(d_0) = \mathsf{f} \\ \mathsf{has}_{\mathsf{c}_0}(c(x_1, \dots, x_n)) = \bigsqcup_{i=1}^n \mathsf{has}_{\mathsf{c}_0}(x_i) \ (\bigsqcup \textit{ is the supremum operation in } \mathsf{B}_2). \end{cases}$$

4.2 Problem

We analyze the problem of testing satisfiability of conjunctions G of ground unit $\Sigma_0 \cup \Sigma_1 \cup \Sigma_2 \cup \Sigma_c$-clauses, where Σ_c is a set of new constants:

$$(\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathsf{Rec}^{[g]}_{\Sigma_1} \cup \mathsf{Rec}^{[g]}_{\Sigma_2}) \wedge G \models \perp$$

(If $\Sigma_2{=}\emptyset$, $\mathcal{T}_s$ can be omitted.) In what follows we use the abbreviations $\Sigma = \Sigma_1 \cup \Sigma_2$, $\mathsf{Rec}^g_\Sigma = \mathsf{Rec}^g_{\Sigma_1} \cup \mathsf{Rec}^g_{\Sigma_2}$, and $\mathsf{Rec}_\Sigma = \mathsf{Rec}_{\Sigma_1} \cup \mathsf{Rec}_{\Sigma_2}$.

4.3 Preprocessing: Formula simplification

The form of the ground formulae to be considered can be simplified as follows:

Lemma 11 *For every set G of ground unit $\Sigma_0 \cup \Sigma \cup \Sigma_c$-clauses there exists a set G' of Σ-flat ground unit $\Sigma_0 \cup \Sigma \cup \Sigma'_c$-clauses (where $\Sigma_c \subseteq \Sigma'_c$) of the form*

$$G' = C_d \wedge C_s \wedge C_\Sigma \wedge NC_{\Sigma'_c},$$

where C_d is a set of pure Σ_0-constraints, C_s is a set of (unit) Σ_s-clauses (if $\Sigma_2 \neq \emptyset$) and $C_\Sigma, NC_{\Sigma'_c}$ are (possibly empty) conjunctions of literals of the form:

C_Σ**:** *$(\neg)f(t_d) = t'$, where $f \in \Sigma_1 \cup \Sigma_2$, t_d is a $\Sigma_0 \cup \Sigma'_c$-term, t' a $\Sigma_{o(f)} \cup \Sigma'_c$-term;*
$(\neg)f(t_d) = g(t'_d)$, where $f, g \in \Sigma_2$, and t_d, t'_d are $\Sigma_0 \cup \Sigma'_c$-terms;
$NC_{\Sigma'_c}$**:** *$t_d \neq t'_d$, where t_d, t'_d are $\Sigma_0 \cup \Sigma'_c$-terms;*

such that G and G' are equisatisfiable w.r.t. $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathcal{K}$ for any set of clauses $\mathcal{K}$ axiomatizing the properties of the functions in Σ.

Proof: We can transform G into an equisatisfiable set of $\Sigma_1 \cup \Sigma_2$-flat clauses over a possible larger set Σ'_c of constants by replacing, in a bottom-up manner, the arguments of the $\Sigma_1 \cup \Sigma_2$-functions with new constants and adding their definitions to the positive part of G, and by also replacing every negative clause of sort d with a disequality between new constants using a similar renaming. We can then use the properties of the constructors for replacing every conjunct of the form $c(t_1, \dots, t_n) = d(s_1, \dots, s_n)$ with $\perp$ (false) if $c \neq d$ resp. with the conjunction $\bigwedge_{i=1}^n s_i = t_i$ if $c = d$, and any conjunct of the form $t = c(\dots, t, \dots)$ with $\perp$ (false). We can therefore assume without loss of generality that G only contains:

(i) positive clauses of the form:
- $a_i = a_j$ for $a_i, a_j \in \Sigma'_c$; $a = c(t_1, \dots, t_n)$, where $a \in \Sigma'_c$ does not occur in any t_i;
- $f(a) = t_d$, where $f \in \Sigma_1$, $a \in \Sigma'_c$, and t_d is a $\Sigma_0 \cup \Sigma \cup \Sigma'_c$-term of sort d;
- $f(a) = t_s$, where $f \in \Sigma_2$, $a \in \Sigma'_c$, and t_s is a term of sort s;
- $f(a) = g(b)$, where $f, g \in \Sigma_2$, $a, b \in \Sigma'_c$;

(ii) negative clauses of the following forms:
- $t \neq s$, where t, s are $\Sigma_0 \cup \Sigma'_c$-terms;
- $f(a) \neq t$, where $f \in \Sigma_1 \cup \Sigma_2$, $a \in \Sigma'_c$, and t is a term of sort $o(f)$;
- $f(a) \neq g(b)$, where $f, g \in \Sigma_2$, and $a, b \in \Sigma'_c$;

(iii) a set C_d of constraints over Σ_0-terms, and

(iv) a set C_s of constraints over Σ_s-terms.

After being brought in this form, G can be simplified further by replacing a_i with a_j whenever $a_i = a_j$ occurs as a conjunct in G and by replacing a by $c(t_1, \dots, t_n)$ whenever $a = c(t_1, \dots, t_n)$ occurs in G, and starting again the simplification procedure which uses the properties of the free constructors (the procedure terminates because of the acyclicity axiom). Using these additional transformations we can simplify G to an equisatisfiable set G' which only consists of unit clauses of the following forms:

- positive clauses of the form $f(t_d) = t$, where t_d is a term of sort d and t is a term of sort $o(f)$ (possibly containing variables in Σ'_c), and $f(a) = g(b)$, where $f, g \in \Sigma_2$, $a, b \in \Sigma'_c$;
- negative clauses of one of the following forms: $t \neq s$ where t, s are $\Sigma_0 \cup \Sigma'_c$-terms, $f(t_d) \neq t$, where t_d is a term of sort d and t is a term of sort $o(f)$ (possibly containing variables in Σ'_c); or clauses of the form $f(t_d) \neq g(s_d)$, where $f, g \in \Sigma_2$ and t_d, s_d are terms of sort d;
- a set C_d of pure Σ_0-constraints and
- a set C_s of pure Σ_s-constraints.

If $\mathcal{K} = \mathsf{Rec}_\Sigma$ then every term of the form $f(t)$, $f \in \Sigma$ is equivalent (w.r.t. $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathcal{K}$) to a term of the form $f(t')$, where t' either starts with a constructor $c \notin \Sigma_r(f)$, or is equal to some $a \in \Sigma_c$. If by making this simplification we introduce additional positive constraints between $\Sigma_0 \cup \Sigma_c$-terms we can eliminate them using the procedure mentioned at the beginning of the proof. □

Remark 12 *If $\mathcal{K}$=Rec_Σ we can further simplify any set of ground unit clauses as follows:*

- *By eagerly applying the recursive definitions as simplification rules we can ensure that, for every literal in C_Σ, the terms t_d (t'_d) either starts with a constructor $c \notin \Sigma_r(f)$ (resp. $c \notin \Sigma_r(f')$) or are equal to some $a \in \Sigma'_c$.*
- *If the definition of $f \in \Sigma$ is exhaustive (resp. quasi-exhaustive), we can ensure that the only occurrence of f in G' is at the root of a term, in terms of the form $f(a)$, where $a \in \Sigma_c$ (resp., if Rec_f is quasi-exhaustive, $a \in \Sigma_c \cup (\Sigma_0 \backslash \Sigma_r(f))$).*
- *We can ensure that each such $f(a)$, with $f \in \Sigma_1$, occurs in at most one positive clause by replacing any conjunction $f(a) = t_1 \wedge f(a) = t_2$ with $f(a) = t_1 \wedge t_1 = t_2$. $f(a) = t_1 \wedge f(a) \neq t_2$ can also be replaced with the (equisatisfiable) conjunction: $f(a) = t_1 \wedge t_1 \neq t_2$.*
- *We can also transform any set of unit ground clauses G containing $f(a)$ (with $f \in \Sigma_1$) only in negative literals $f(a) \neq t$ into an equisatisfiable set of unit ground clauses, by introducing a new constant c and replacing $f(a) \neq t$ with $f(a) = c \wedge c \neq t$.*

4.4 Locality results for extensions with recursively defined functions

We make the following assumptions:

Assumption 1: Either $\Sigma_1 = \emptyset$, or else $\Sigma_1 \neq \emptyset$ and Rec_{Σ_1} is quasi-exhaustive.
Assumption 2: G is a set of ground unit clauses with the property that any occurrence of a function symbol in Σ_1 is in positive unit clauses of G of the form $f(a) = t$, with $a \in \Sigma_c \cup (\Sigma_0 \backslash \Sigma_r(f))$, and G does not contain any equalities between $\Sigma_0 \cup \Sigma_c$-terms. (By Remark 12, we can assume w.l.o.g. that for all $f \in \Sigma_1$ and $a \in \Sigma_c \cup (\Sigma_0 \backslash \Sigma_r(f))$, $f(a)$ occurs in at most one positive unit clause of G of the form $f(a) = t$.)

Theorem 13 *If Assumption 1 holds, then:*

(1) $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathsf{Rec}_{\Sigma_2}$ *satisfies the* Ψ*-locality conditions as an extension of* $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s$ *for all sets* G *of clauses of the form obtained after the simplification described in Lemma 11.*

(2) If Rec_{Σ_1} *is quasi-exhaustive, then* $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathsf{Rec}_{\Sigma_1} \cup \mathsf{Rec}_{\Sigma_2}$ *satisfies the* Ψ*-locality conditions of an extension of* $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s$ *for every set* G *of unit clausesof the form obtained after the simplification described in Lemma 11 which satisfy the conditions in Assumption 2;*

where Ψ *associates with any set* T *of ground terms the smallest set which contains* T *and if* $f(c(t_1, \dots, t_n)) \in \Psi(T)$ *and* $c \in \Sigma_r(f)$ *then* $f(t_i) \in \Psi(T)$ *for* $i = 1, \dots, n$.

Similar results hold for extensions with Rec^g_{Σ} *(under similar assumptions) provided the guards* ϕ_i *in the recursive definitions of functions in* Σ_1 *are positive.*

Note: We can actually prove a variant of ELoc^{Ψ}, in which we can allow first-order Σ_s-constraints in $\mathsf{Rec}^{[g]}_{\Sigma}$ and in G.

Proof: Let $P = (P_d, P_s, \{f_P\}_{P \in \Sigma}, \{a_P\}_{a \in \Sigma_c})$ be a (partial) model of $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathsf{Rec}_{\Sigma} \cup G$ such that P_d is a total model of $\mathsf{AbsFree}_{\Sigma_0}$, P_s is a total model of $\mathcal{T}_s$, and for every $f \in \Sigma$, f_P is partially defined and satisfies Rec_{Σ}. We assume w.l.o.g. that all terms in $\Psi(G)$ are defined in P and only them.

Note that since G is in the canonical form obtained after the simplifications described in Lemma 11, no non-trivial equality between $\Sigma_0 \cup \Sigma'_c$-terms properly containing Σ'_c terms can be inferred from G. In order to prove this, note first that in the normal form obtained after the simplification in Lemma 11 does not contain any equalities between $\Sigma_0 \cup \Sigma'_c$-terms properly containing Σ'_c terms, and that any equality between such terms can be reduced to an equality of the form $a_i = a_j$ where $a_i, a_j \in \Sigma'_c$ and $a = c(t_1, \dots, t_n)$. The proof is by induction on the number of steps needed to infer an equality of the form $a_i = a_j$ or $a = c(t_1, \dots, t_n)$. Clearly, there are no proofs in one step. Any proof in one step would use transitivity of equality and a positive equality involving a term $f(t)$ with $f \in \Sigma_1$ containing a f symbol, i.e. would be of the form: $a_i = f(t) \wedge f(t) = a_j \rightarrow a_i = a_j$, resp. $a = f(t) \wedge f(t) = c(t_1, \dots, t_n) \rightarrow a = c(t_1, \dots, t_n)$. This is however prevented by the requirement on the form of the clauses we made in Assumption 2.

Note now that if no non-trivial equality between $\Sigma_0 \cup \Sigma'_c$-terms properly containing Σ'_c terms can be inferred from G, we can assume w.l.o.g. that in P no identities of the form $a_i = a_j$ or $a = c(t_1, \dots, t_n)$ hold, where $a_i, a_j, a \in \Sigma'_c$, c is a constructor, and $t_1, \dots, t_n$ are $\Sigma_0 \cup \Sigma_a$-terms. It follows that we can always choose a model P' of $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathsf{Rec}_{\Sigma}[\Psi(G)] \cup G$with the property that all subterms of G are defined in P' and if $f(c(t_1, \dots, t_n))$ is defined in P' then $f(c(t_1, \dots, t_n)) \in \Psi(G)$, and if $c \in \Sigma_r(f)$ then $f(t_1), \dots, f(t_n) \in \Psi(G)$ (i.e. they are defined in P). [5]

[5] Indeed, we can choose $P'_d = \{t_P \mid t \in T_{\Sigma_0}(\Sigma_c)\}$. Since P'_d is a Σ_0-subalgebra of P_d, it is also a model of $\mathsf{AbsFree}_{\Sigma_0}$. Therefore, P'_d is isomorphic to the free algebra

We define a total model $\overline{P} = (T_{\Sigma_0}(\Sigma_c), P_s, \{f_{\overline{P}}\}_{f\in\Sigma}, \{a_{\overline{P}}\}_{a\in\Sigma_c})$ of $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathsf{Rec}_\Sigma$ and G as follows. The support of $\overline{P}$ is the (absolutely) free algebra freely generated by Σ_c. Let $a_{\overline{P}} = a$ for every $a \in \Sigma_c$. Let $h : T_{\Sigma_0}(\Sigma_c) \to P_d$ be the unique Σ_0-homomorphism with the property that $h(a) = a_P$ for every $a \in \Sigma_A$. We define $f_{\overline{P}}$ on the layers of $T_{\Sigma_0}(\Sigma_c) = \bigcup_{i\geq 0} P_i$, where $P_0 = \Sigma_c$ and $P_{i+1} = \{c(t_1,\dots,t_n) \mid c \in \Sigma_0 \text{ and } t_i \in \bigcup_{0\leq j\leq i} P_j\}$. Let c_d, c_s be arbitrary but fixed elements in $T_{\Sigma_0}(\Sigma_c)$ resp. P_s.

Assume first that $f\in\Sigma_2$. Then $f_{\overline{P}}$ is defined on P_0 by:

$$f_{\overline{P}}(a) := \begin{cases} f_P(a_P) & \text{if } f_P(a_P) \text{ is defined} \\ c_s & \text{if } f_P(a_P) \text{ is not defined} \end{cases}$$

Assume that $f_{\overline{P}}$ is defined on $\bigcup_{j=0}^{i} P_i$. We extend it to P_{i+1} as follows:

$$f_{\overline{P}}(c(t_1,\dots,t_n)) := \begin{cases} f_P(c_P(h(t_1),\dots,h(t_n))) & \text{if } f_P(c_P(h(t_1),\dots,h(t_n))) \text{ defined} \\ g_f^c(f_{\overline{P}}(t_1),\dots,f_{\overline{P}}(t_n)) & \text{if } c \in \Sigma_r(f) \text{ and} \\ & f_P(c(h(t_1),\dots,h(t_n))) \text{ undefined} \\ c_s & \text{otherwise} \end{cases}$$

It is easy to see that $f_{\overline{P}}$ is well-defined. It is also clear, by definition, that if $f_P(h(t))$ is defined then $f_{\overline{P}}(t) = f_P(h(t))$. We prove that f satisfies the axioms in Rec_f. Let $t = c(t_1,\dots,t_n)$ with $c \in \Sigma_r(f)$.

- If $f_P(t_P) = f_P(h(t))$ is undefined then by definition $f_{\overline{P}}(c(t_1,\dots,t_n)) = g_f^c(f_{\overline{P}}(t_1),\dots,f_{\overline{P}}(t_n))$.
- If $f_P(t_P) = f_P(h(t))$ is defined then $f_P(c_P(h(t_1),\dots,h(t_n)))$ is defined, and (as $c \in \Sigma_r(f)$) $f_P(h(t_i))$ are defined for all i and – since P weakly satisfies Rec_f – $f_P(h(t)) = f_P(c_P(h(t_1),\dots,h(t_n))) = g_i^c(f_P(h(t_1)),\dots,f_P(h(t_n)))$. It follows that also in this case $f_{\overline{P}}(c(t_1,\dots,t_n)) = g_f^c(f_{\overline{P}}(t_1),\dots,f_{\overline{P}}(t_n))$.

The remarks above show that we can reformulate the definition for $f_{\overline{P}}$ as:

$$f_{\overline{P}}(c(t_1,\dots,t_n)) := \begin{cases} f_P(c_P(h(t_1),\dots,h(t_n))) & \text{if } f_P(c_P(h(t_1),\dots,h(t_n))) \text{ defined} \\ g_f^c(f_{\overline{P}}(t_1),\dots,f_{\overline{P}}(t_n)) & \text{if } c \in \Sigma_r(f) \\ c_s & \text{otherwise} \end{cases}$$

If $f \in \Sigma_1$, note that by Assumption 2, every occurrence of f in G is in a (unique) positive unit clase of the form $f(a) = t$. By the assumptions we made on P according to the form of the constraints in G, we can assume w.l.o.g. that $f_P(a_P)$ is defined iff $f(a)$ occurs in G.

We define $f_{\overline{P}}$ on P_0 as follows:

$$f_{\overline{P}}(a) := \begin{cases} t & \text{if } f_P(a_P) \text{ is defined (and } t \text{ is the unique term} \\ & \text{s.t. } f(a) = t \text{ occurs in } G) \\ c_d & \text{if } f_P(a_P) \text{ is not defined} \end{cases}$$

generated by a subset of Σ_c – in the absence of entailed constraints of the form $a = a'$ or $a = c(t_1,\dots,t_n)$ we can actually safely assume that P'_d is isomorphic to the free algebra generated by Σ_c.

(We use the fact that if $f_P(a_P)$ is defined then there exists a unique clause in G of the form $f(a) = t$. This is the term t we choose in the definition of $f_{\overline{P}}$.)

Assume that $f_{\overline{P}}$ is defined on $\bigcup_{j=0}^{i} P_i$. We extend $f_{\overline{P}}$ to P_{i+1} as follows:

$$f_{\overline{P}}(c(t_1, \dots, t_n)) := g_f^c(f_{\overline{P}}(t_1), \dots, f_{\overline{P}}(t_n))$$

if Rec_{Σ_0} is exhaustive (if it is quasi-exhaustive, we have to define in the first step all $f(c)$, where $c \in \Sigma_c \cup (\Sigma_1 \backslash \Sigma_r(f))$. We now prove that all $f_{\overline{P}}$ are well-defined. To prove that it is well-defined on P_0, assume that $f_P(a_P)$ is defined. We assumed that all occurrences of function symbols in Σ_1 in G were in clauses of the form $f(a) = t$, where $t \in T_{\Sigma_0}(\Sigma_c)$, and $f(a)$ occurs in a unique literal of this form. Thus, the term t is uniquely determined (and also that $f_P(a_P)$ is defined and equal to t_P). The fact that $f_{\overline{P}}$ is well-defined on $\bigcup_{i=1}^{n} P_i$ is immediate.

It is easy to see that, due to the way it is defined, $f_{\overline{P}}$ satisfies the axioms in Rec_f. Note that if $t \in T_{\Sigma_0}(\Sigma_c)$ and $f_P(t_P)$ is defined then $f(t) \in \Psi(G)$, so $t = a \in \Sigma_c$, and by definition $f_{\overline{P}}(t) = t'$ such that $t'_P = f_P(t_P)$.

We show that $\overline{P}$ is also a model of G. We assumed that G consists of Σ-flat ground unit $\Sigma_0 \cup \Sigma \cup \Sigma_c$-clauses of the form:

$$\bigwedge_{i=1}^{n} (\neg) f_i(t_d^i) = t^i \wedge \bigwedge_{j=1}^{m} (\neg) f_j(t_d^j) = f'_j(s_d^j) \wedge \bigwedge_{k=1}^{l} t_d^k \neq s_d^k \ \wedge \ C_d \ \wedge \ C_s,$$

where t_d^k, s_d^k, t_d are $\Sigma_0 \cup \Sigma_c$-terms; C_d is a set of pure Σ_0-constraints and C_s is a set of (unit) Σ_s-clauses (if $\Sigma_2 \neq \emptyset$).

Since G is true in P, all negative $\Sigma_0 \cup \Sigma_c$-clauses, as well as all pure Σ_s-formulae in C_s of G are true in P and hence also in $\overline{P}$. If $f(t) = s$ is a positive clause in G with s a Σ_s-term, then $f_P(t_P)$ is defined and equal to s_P, hence $f_{\overline{P}}(t) = f_P(t_P) = s_P$. Thus, $f(t) = s$ is true also in $\overline{P}$. Similarly for equalities $f(t) = g(s)$, where $a(f) = a(g) = d \rightarrow s$, and for any negative clause of the form $f(t) \neq s$, where $a(f) = d \rightarrow s$. Assume now that $f \in \Sigma_1$, and $f(t) = t'$ occurs in G. By Assumption 2, the clause is of the form $f(a) = t$. Then $f_P(a_P) = t_P$ and this t was used for defining $f_{\overline{P}}(a) = t$. In this case the clause is true in $\overline{P}$. [6]

In order to prove that similar results hold for extensions with guarded recursive definitions, we need to check that if f_P weakly satisfies $\mathsf{Rec}^g(f)$ then $f_{\overline{P}}$ can be constructed such that $\mathsf{Rec}^g(f)$ holds. We define $f_{\overline{P}}$ on P_0 as before; if $f \in \Sigma_2$ and $t_k \in \bigcup_{0 \leq j \leq i} P_i$ for $k = 1, \dots, n$ we define:

$$f_{\overline{P}}(c(t_1, \dots, t_n)) := \begin{cases} f_P(c_P(h(t_1), \dots, h(t_n))) & \text{if } f_P(c_P(h(t_1), \dots, h(t_n))) \text{ is defined} \\ g_i^c(f_{\overline{P}}(t_1), \dots, f_{\overline{P}}(t_n)) & \text{if } c \in \Sigma_r(f) \text{ and } \phi^i_{\overline{P}}(f_{\overline{P}}(t_1), \dots, f_{\overline{P}}(t_n)) \\ c_s & \text{otherwise} \end{cases}$$

[6] Note that also the truth of negative literals would be preserved in $\overline{P}$: If $f(t) \neq t'$ occurs in G then $f_P(t_P) \neq t'_P$. This means that $f_{\overline{P}}(t) = \overline{t}$ for some $\overline{t}$ such that $f_{\overline{P}}(t_P) = \overline{t}_P$. Then clearly $\overline{t} \neq t'$. Thus, the truth of negative unit clauses of sort d in G is preserved. However, by Assumption 2 no such occurrences exist.

The only point to be proved is that if $f_P(c_P(h(t_1), \dots, h(t_n)))$ is defined in P then the two definitions above agree. This is obvious for P_0. Assume that it holds for all $P_j,\ 0 \leq j \leq i$. We prove it for P_{i+1}. Assume that $c \in \Sigma_r(f)$ and $\phi^i_{\overline{P}}(f_{\overline{P}}(t_1), \dots, f_{\overline{P}}(t_n))$. Since the operations and relations in Σ_s are unchanged, and by the induction hypothesis, $f_{\overline{P}}(t_1){=}f_P(h(t_i))$, we know that (in P_s) ${\phi^i}_P(f_P(h(t_1)), \dots, f_P(h(p_1)))$ is true. Therefore, $f_P(c_P(h(t_1), \dots, h(t_n))) = {g^c_i}_P(f_P(h(t_1)), \dots, f_P(h(t_n))) = g^c_i(f_{\overline{P}}(t_1), \dots, f_{\overline{P}}(t_n))$.

If $f \in \Sigma_1$ we define $f(a) := t$ (the unique term such that $f(a) = t$ occurs in G, as previously) and if $t_k \in \bigcup_{0 \leq j \leq i} P_i$ we define:

$$f_{\overline{P}}(c(t_1, \dots, t_n)) := g^c_i(f_{\overline{P}}(t_1), \dots, f_{\overline{P}}(t_n)) \text{ if } \phi^i_{\overline{P}}(f_{\overline{P}}(t_1), \dots, f_{\overline{P}}(t_n)).$$

We want to prove that if $f_P(t_P)$ is defined in P then $h(f_{\overline{P}}(t)) = f_P(t_P)$. We proceed again by induction on the level of t. If $t \in \Sigma_c \cup (\Sigma_0 \backslash \Sigma_r(f))$ this is clear. Assume the property holds for all terms on the levels $P_0, \dots, P_i$. Let $t = c(t_1, \dots, t_n)$ be on level P_{i+1} such that $f_P(h(t))$ is defined. Then $f_P(h(t_i))$ is defined for all $i = 1, \dots, n$. Since Rec_f is quasi-exhaustive, in P $\phi^{i_0}(h(t_1), \dots, h(t_n))$ holds for some i_0, and $f_P(t_P) = f_P(c_P(h(t_1), \dots, h(t_n))) = g^{i_0}_f(f_P(h(t_1)), \dots, f_P(h(t_n)))$.

Assume now that $\phi^i_{\overline{P}}(f_{\overline{P}}(t_1), \dots, f_{\overline{P}}(t_n))$ is true in $\overline{P}$. Since the truth of positive sentences is preserved under homomorphisms, $\phi^i_P(h(f_{\overline{P}}(t_1)), \dots, h(f_{\overline{P}}(t_n)))$ is true, thus (using the induction hypothesis): $\phi^i_P(f_P(h(t_1)), \dots, f_P(h(t_n)))$ is true in P. It follows that $i = i_0$, so $h(f_{\overline{P}}(t)) = f_P(h(t))$ also for $t = c(t_1, \dots, t_n)$. □

ELoc$^{\Psi}$. Note that in the process of contructing the total model $\overline{P}$ from the partial model P the support of sort s of the model did not change. This means that all pure Σ_s formulae which were true in P remain true in $\overline{P}$. As shown in [14] this guarantee that the more general notion (ELoc^{Ψ}) of locality holds, i.e. we can allow the set G of clauses to contain *arbitrary* pure Σ_s-constraints, and we also can allow arbitrary Σ_s-formulae ϕ_i as guards in the definition $\mathsf{Rec}^g_{\Sigma_2}$.

4.5 ERec: locality results

The results in the previous section can be extended to recursive definitions of the form $\mathsf{ERec}^{[g]}_f$:

$$\begin{cases} f(k, x) = k_f(x) \\ f(c(x_1, \dots, x_n), x) = \begin{cases} g^{c,f}_1(f(x_1, x), \dots, f(x_n, x), x) & \text{if } \phi_1(f(x_1), \dots, f(x_n)) \\ \dots \\ g^{c,f}_k(f(x_1, x), \dots, f(x_n, x), x) & \text{if } \phi_k(f(x_1), \dots, f(x_n)) \end{cases} \end{cases}$$

where $k, c \in \Sigma_r(f)$, $a(k) = 0, a(c) = n$, $k_f(x)$ are $\Sigma_{o(f)}$-terms with free variable x, $g^{c,f}_i$ are functions expressible as $\Sigma_{o(f)}$-terms, and $\phi_i(x_1, \dots, x_n)$ are $\Sigma_{o(f)}$-formulae with free variables $x_1, \dots, x_n$, s.t. $\phi_i \wedge \phi_j \models_{\mathcal{T}_{o(f)}} \perp$ for $i \neq j$.

Definitions of type ERec_f are similar, but with no guards in the definition of $f(c(x_1, \dots, x_n))$. In what follows, $\mathsf{ERec}_{\Sigma} = \bigcup_{f \in \Sigma} \mathsf{ERec}_f$ and $\mathsf{ERec}^g_{\Sigma} = \bigcup_{f \in \Sigma} \mathsf{ERec}^g_f$.

Theorem 14 *If Assumption 1 holds, then:*

(1) $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathsf{ERec}_{\Sigma_2}$ *satisfies the* Ψ*-locality conditions as an extension of* $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s$ *for all sets* G *of clauses of the form obtained after the simplification described in Lemma 11.*

(2) If ERec_{Σ_1} *is quasi-exhaustive, then* $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathsf{Rec}_{\Sigma_1} \cup \mathsf{Rec}_{\Sigma_2}$ *satisfies the* Ψ*-locality conditions of an extension of* $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s$ *for every set* G *of unit clauses of the form obtained after the simplification described in Lemma 11 which satisfy the conditions in Assumption 2;*

where Ψ *associates with every set* T *of ground terms the smallest set containing* T *and such that if* $f(c(t_1, \dots, t_n), t) \in \Psi(T)$ *and* $c \in \Sigma_r(f)$ *then* $f(t_i, t) \in \Psi(T)$ *for all* i*.*

Similar results hold for extensions with ERec^g_Σ *(under similar assumptions) provided the guards* ϕ_i *in the recursive definitions of functions in* Σ_1 *are positive.*

Proof: We analyze the axioms of type ERec, by only pointing out the changes which need to be made. The process of constructing a total model $\overline{P}$ of $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathsf{ERec}^{[g]}_\Sigma$ and G from a weak partial model P of $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathsf{ERec}^{[g]}_\Sigma[\Psi(G)]$ and G is analogous to the one used in the proof of Theorem 13, and uses the layer structure of $T_{\Sigma_0}(\Sigma_c)$. For the sake of simplicity, below we only discuss the axioms without guards ERec_Σ.

Assume first that $f \in \Sigma_2$. We first define it on $P_0 \times T_{\Sigma_0}(\Sigma_c)$:

$$f_{\overline{P}}(a, t) := \begin{cases} f_P(a_P, t_P) & \text{if } f_P(a_P, t_P) \text{ is defined} \\ c_s & \text{if } f_P(a_P) \text{ is not defined} \end{cases}$$

Assume that $f_{\overline{P}}$ is defined on $\bigcup_{j=0}^i (P_i \times T_{\Sigma_0}(\Sigma_c))$. We extend it to $P_{i+1} \times T_{\Sigma_0}(\Sigma_c)$ as follows:

$$f_{\overline{P}}(c(t_1, \dots, t_n), t) := \begin{cases} f_P(c(h(t_1), \dots, h(t_n)), h(t)) & \text{if } f_P(c(h(t_1), \dots, h(t_n)), h(t)) \\ & \text{is defined} \\ g^c_f(f_{\overline{P}}(t_1), \dots, f_{\overline{P}}(t_n), t) & \text{if } c \in \Sigma_r(f) \\ c_s & \text{otherwise} \end{cases}$$

If $f \in \Sigma_1$ we define $f_{\overline{P}}$ on $P_0 \times T_{\Sigma_0}(\Sigma_c)$ by:

$$f_{\overline{P}}(a, s) := \begin{cases} t & \text{if } f_P(a_P, s_P) \text{ is defined and } t \text{ is the unique term s.t.} \\ & \quad f(a, s) = t \text{ occurs in } G \text{ (if such a term exists)} \\ c_d & \text{if } f_P(a_P, s_P) \text{ is not defined} \end{cases}$$

Assume that $f_{\overline{P}}$ is defined on $\bigcup_{j=0}^i (P_i \times T_{\Sigma_0}(\Sigma_c))$. We extend $f_{\overline{P}}$ to $P_{i+1} \times T_{\Sigma_0}(\Sigma_c)$ as follows:

$$f_{\overline{P}}(c(t_1, \dots, t_n)) := g^c_f(f_{\overline{P}}(t_1), \dots, f_{\overline{P}}(t_n))$$

if Rec_{Σ_1} is exhaustive (if it is quasi-exhaustive, we have to define in the first step all $f(c, t)$, where $c \in \Sigma_c \cup (\Sigma_1 \backslash \Sigma_r(f))$). It can be shown again that all $f_{\overline{P}}$ are well-defined and that $\overline{P}$ is the total model of G and $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathsf{ERec}_\Sigma$ $\square$

4.6 Example

We illustrate the hierarchical reasoning method for checking satisfiability of sets of ground clauses in extensions of $\mathsf{AbsFree}_{\Sigma_0}$ with recursively defined functions on the following example.

Example 3 *Let $\Sigma_0 = \{c_0, d_0, c\}$, where c is a binary constructor and c_0, d_0 are nullary. Consider the recursive definition $\mathsf{Rec}_{\mathsf{has}_{\mathsf{c_0}}}$ of the function $\mathsf{has}_{\mathsf{c_0}}$ in Example 2. We want to show that $\mathsf{AbsFree}_{\Sigma_0} \cup \mathsf{Bool} \cup \mathsf{Rec}_{\mathsf{has}_{\mathsf{c_0}}} \models G_1$ where*

$$G_1 = \forall \overline{x}(\mathsf{has}_{\mathsf{c_0}}(x){=}\mathsf{t} \wedge z_1{=}c(y_1, c(x_1, x)) \wedge z_1{=}c(y_2, y_3) \rightarrow \mathsf{has}_{\mathsf{c_0}}(y_3){=}\mathsf{t})$$

Step 1: Reduction to a satisfiability problem. *The problem of checking the validity of G_1 can alternatively expressed as the problem of checking the satisfiability w.r.t. $\mathsf{AbsFree}_{\Sigma_0} \cup \mathsf{Bool} \cup \mathsf{Rec}_{\mathsf{has}_{\mathsf{c_0}}}$ of*

$$G \; = \neg G_1 = (\mathsf{has}_{\mathsf{c_0}}(a){=}\mathsf{t} \wedge c_1{=}c(b_1, c(a_1, a)) \wedge c_1{=}c(b_2, b_3) \wedge \mathsf{has}_{\mathsf{c_0}}(b_3){=}\mathsf{f}),$$

where $\Sigma_c = \{a, a_1, b_1, b_2, b_3, c_1\}$ is a set of new constants obtained by skolemizing the existentially quantified variables in $\neg G_1$.

Step 2: Simplification. *We transform G as explained in Lemma 11 by inferring all equalities entailed by the equalities between constructor terms in G:*

- *From $c_1 = c(b_1, c(a_1, a)) \wedge c_1 = c(b_2, b_3)$ we infer: $b_1 = b_2 \wedge c(a_1, a) = b_3$;*
- *We replace everywhere b_2 with b_1 and b_3 with $c(a_1, a)$, and in what follows ignore the constants b_2, b_3.*

We obtain the equisatisfiable set of ground clauses:

$$G' = (\mathsf{has}_{\mathsf{c_0}}(a){=}\mathsf{t} \wedge \mathsf{has}_{\mathsf{c_0}}(c(a_1, a)){=}\mathsf{f}).$$

The set of unit clauses G' has the properties required for establishing the locality result in Theorem 13.

Step 3: Locality. *By the locality property we established in Theorem 13, the following are equivalent:*

(i) $\mathsf{AbsFree}_{\Sigma_0} \cup \mathsf{Bool} \cup \mathsf{Rec}_{\mathsf{has}_{\mathsf{c_0}}}) \cup G' \models \perp$
(ii) $\mathsf{AbsFree}_{\Sigma_0} \cup \mathsf{Bool}) \cup \mathsf{Rec}_{\mathsf{has}_{\mathsf{c_0}}}[\Psi(G')] \cup G' \models \perp$,
where $\Psi(G') = \{\mathsf{has}_{\mathsf{c_0}}(c(a_1, a)), \mathsf{has}_{\mathsf{c_0}}(a_1), \mathsf{has}_{\mathsf{c_0}}(a)\}$.

Step 4: Hierarchical reduction. *After purification we obtain:*

$\mathsf{Def}_{\mathsf{bool}}$	$G_0 \wedge \mathsf{Rec}_{\mathsf{has}_{\mathsf{c_0}}}[\Psi(G)]_0$
$\mathsf{has}_{\mathsf{c_0}}(a_1) = h_1 \wedge \mathsf{has}_{\mathsf{c_0}}(a) = h_2 \wedge \mathsf{has}_{\mathsf{c_0}}(c(a_1, a)) = h_3$	$h_2 = \mathsf{t} \wedge h_3 = \mathsf{f} \wedge h_3 = h_1 \sqcup h_2$

We immediately obtain a contradiction in Bool, without needing to consider Con_0 or a further reduction to a satisfiability test w.r.t. $\mathsf{AbsFree}_{\Sigma_0}$.

4.7 Combining recursive definitions with boundedness.

We analyze the locality of combinations of $\mathsf{Rec}_\Sigma^{[g]}$ with boundedness axioms, of the type:

$$\mathsf{Bounded}(f) \qquad \forall x(t_1 \leq f(x) \leq t_2)$$

Theorem 15 *Assume that $a(f) = d \to s$, t_1, t_2 are Σ_s-terms with $\mathcal{T}_s \models t_1 \leq t_2$, and all functions $g_i^{c,f}$ used in the definition of f have the property:*
$\forall x_1, \dots, x_n(\bigwedge_{i=1}^n t_1 \leq x_i \leq t_2 \to t_1 \leq g_i^{c,f}(x_1, \dots, x_n) \leq t_2)$, *where $n = a(c)$.*
If Assumption 1 holds then $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathsf{Rec}_f^{[g]} \cup \mathsf{Bounded}$ is a Ψ-local extension of $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s$, where Ψ is defined as in Theorem 13.

Proof: Analyzing the proof of Theorem 13, we note that with small changes in the construction of the total model $\overline{P}$ we can guarantee that the recursively defined functions satisfy axiom $\mathsf{Bounded}$, assuming that the partial functions satisfied the respective axioms on their domain of definition. For the sake of simplicity in what follows we will analyze definitions using Rec_Σ. We only need to consider functions in Σ_2.

The proof proceeds as the proof of Theorem 13, with the difference that in the definitions we choose the default definitions $c_s \in P_s$ such that they satisfy the constraint $t_1 \leq c_s \leq t_2$. Such a value exists because $\mathcal{T}_s \models t_1 \leq t_2$ and $\leq$ is reflexive. The definition on P_0 is:

$$f_{\overline{P}}(a) := \begin{cases} f_P(a_P) & \text{if } f_P(a_P) \text{ is defined} \\ c_s & \text{if } f_P(a_P) \text{ is not defined} \end{cases}$$

Obviously, with this definition $t_1 \leq f_{\overline{P}}(a) \leq t_2$ for all $a \in A$. Assume that $f_{\overline{P}}$ is defined on $\bigcup_{j=0}^i P_i$. We extend it to P_{i+1} as follows:

$$f_{\overline{P}}(c(t_1, \dots, t_n)) := \begin{cases} f_P(c(h(t_1), \dots, h(t_n))) & \text{if } f_P(c(h(t_1), \dots, h(t_n))) \text{ is defined} \\ g_f^c(f_{\overline{P}}(t_1), \dots, f_{\overline{P}}(t_n)) & \text{if } c \in \Sigma_r(f) \\ c_s & \text{otherwise} \end{cases}$$

Assume that the boundedness axioms hold on $P_0, \dots, P_i$. They always hold whenever $f_P(c(h(t_1), \dots, h(t_n)))$ is defined. Assume $c \in \Sigma_r(f)$. In this case $f_{\overline{P}}(c(t_1, \dots, t_n)) = g_f^c(f_{\overline{P}}(t_1), \dots, f_{\overline{P}}(t_n))$. Since $t_1 \leq f_{\overline{P}}(t_1) \leq t_2$ and g_f^c satisfies the condition

$$\forall \overline{x} \bigwedge_{i=1}^n t_1 \leq x_i \leq t_2 \to t_1 \leq g_f^c(x_1, \dots, x_n) \leq t_2,$$

it follows that $t_1 \leq g_f^c(f_{\overline{P}}(t_1), \dots, f_{\overline{P}}(t_n)) \leq t_2$. If $f_P(c(h(t_1), \dots, h(t_n)))$ is undefined and $c \not\in \Sigma_r(f)$ then $f_{\overline{P}}(c(t_1, \dots, t_n)) = c_s$ and $t_1 \leq c_s \leq t_2$.

Similar arguments can be used for definitions of type Rec_f^g if we require that g_i^c satisfy the condition:

$$\forall \overline{x} \quad \left(\phi_i(x_1, \ldots, x_n) \wedge \bigwedge_{i=1}^{n} t_1 \leq x_i \leq t_2 \right) \rightarrow t_1 \leq g_i^c(x_1, \ldots, x_n) \leq t_2.$$

We show that the induction step above can still be proved. Assume the boundedness axioms hold on $P_0, \ldots, P_i$. They also hold if $f_P(c_P(h(t_1), \ldots, h(t_n)))$ is defined. Assume $c \in \Sigma_r(f)$. If $\phi_{\overline{P}}(f_{\overline{P}}(t_1), \ldots, f_{\overline{P}}(t_n))$ is true then, by definition, $f_{\overline{P}}(c(t_1, \ldots, t_n)) = g_i^c(f_{\overline{P}}(t_1), \ldots, f_{\overline{P}}(t_n))$. Since $\phi_{P_s}(f_{\overline{P}}(t_1), \ldots, f_{\overline{P}}(t_n))$ is true and $t_1 \leq f_{\overline{P}}(t_i) \leq t_2$ it follows that $t_1 \leq g_i^c(f_{\overline{P}}(t_1), \ldots, f_{\overline{P}}(t_n)) \leq t_2$.

Similar arguments also hold for definitions of type $\mathsf{ERec}_f^{[g]}$. □

Example 4 *(1) We want to check whether* $\mathsf{AbsFree}_{\Sigma_0} \cup \mathbb{Z} \cup \mathsf{Rec}_{\mathsf{depth}}$ *entails*

$$G_1 = \forall x_1, x_2, x_3, x_4 \; (\mathsf{depth}(x_1) \leq \mathsf{depth}(x_2) \wedge \mathsf{depth}(x_4) \leq \mathsf{depth}(x_3) \wedge x_4 = c(x_2) \\ \rightarrow \mathsf{depth}(d(x_1, e(x_2, c'))) \leq \mathsf{depth}(e(x_4, x_3))),$$

where Σ_0 contains the constructors c' (nullary), c (unary), and d, e (binary).

Step 1: Reduction to a satisfiability test. *This problem can be reduced to testing the satisfiability of the following conjunction G of ground clauses containing the additional constants $\Sigma_c = \{a_1, a_2, a_3, a_4\}$ obtained from skolemization.*

$$G = \neg G_1 = (\mathsf{depth}(a_1) \leq \mathsf{depth}(a_2) \wedge \mathsf{depth}(a_4) \leq \mathsf{depth}(a_3) \wedge a_4 = c(a_2) \\ \wedge \mathsf{depth}(d(a_1, e(a_2, c'))) \not\leq \mathsf{depth}(e(a_4, a_3))).$$

Steps 2, 3: Simplification, flattening, locality. *By Ψ-locality, this can be reduced to testing the satisfiability of the following conjunction of ground clauses containing the additional constants*

$$\Sigma_c' = \{a_1, a_2, a_3, a_4, d_1, d_2, d_3, d_4, e_1, e_2, e_3, g_1, g_2, g_3, c_2', d_2'\}$$

(below we present the flattened and purified form), where $G = \neg G_1$:

Def_d	$\mathsf{Def}_{\mathsf{num}}$	G_{0d}	$G_{0\mathsf{num}}$	$\mathsf{Rec}_{\mathsf{depth}}[\Psi(G)]_0$
$d(a_1, e_2) = e_1$	$\mathsf{depth}(a_i) = d_i (i = 1-4)$	$a_4 = c_2'$	$d_1 \leq d_2$	$g_1 = 1 + \mathsf{max}\{d_1, g_2\}$
$e(a_2, c') = e_2$	$\mathsf{depth}(e_i) = g_i (i = 1, 2, 3)$		$d_4 \leq d_3$	$g_2 = 1 + \mathsf{max}\{d_2, 1\}$
$e(a_4, a_3) = e_3$	$\mathsf{depth}(c_2') = d_2'$		$g_1 \not\leq g_3$	$g_3 = 1 + \mathsf{max}\{d_4, d_3\}$
$c(a_2) = c_2'$				$d_2' = 1 + d_2$

Let Con_0 consist of all the instances of congruence axioms for c, d, e and depth. $G_0 \cup \mathsf{Rec}_{\mathsf{depth}}[\Psi(G)]_0 \cup \mathsf{Con}_0$ *is satisfiable in* $\mathsf{AbsFree}_{\Sigma_0} \cup \mathbb{Z}$. *A satisfying assignment is described below:*

- $d_1 = d_2 = 0$;
- $d_2' = d_4 = d_3 = 1$ *(d_2' and d_4 need to be equal due to Con_0 because $c_2' = a_4$; and $d_4 \leq d_3$).*

- $g_2 = 1 + \mathsf{max}\{0, 1\} = 2$,
- $g_1 = 1 + \mathsf{max}\{d_1, g_2\} = 3$ *and*
- $g_3 = 1 + \mathsf{max}\{d_4, d_3\} = 1 + d_4 = 2$.

Thus, $\mathsf{AbsFree}_{\Sigma_0} \cup \mathbb{Z} \cup \mathsf{Rec}_{\mathsf{depth}} \not\models G_1$.

(2) We now show that $\mathsf{AbsFree}_{\Sigma_0} \cup \mathbb{Z} \cup \mathsf{Rec}_{\mathsf{depth}} \cup \mathsf{Bounded}(\mathsf{depth}) \models G_1$, *where*

$$\mathsf{Bounded}(\mathsf{depth}) \qquad \forall x(\mathsf{depth}(x) \geq 1).$$

By Theorem 15, we only need to consider the instances of $\mathsf{Bounded}(\mathsf{depth})$ *containing terms in* $\mathsf{Def}_{\mathsf{num}}$*, i.e. the constraints:*

- $d_i \geq 1$ *for* $i \in \{1, \dots, 4\}$;
- $g_i \geq 1$ *for* $i \in \{1, \dots, 3\}$, *and*
- $d'_2 \geq 1$.

Con_0 *can be used to derive* $d_4 = d'_2$. *We obtain:*

- $g_1 = 1+\mathsf{max}\{d_1, g_2\} = 1+\mathsf{max}\{d_1, 1+\mathsf{max}\{d_2, 1\}\} = 1+\mathsf{max}\{d_1, 1+d_2\} = 2+d_2$
- $g_3 = 1+\mathsf{max}\{d_4, d_3\} = 1+d_3 \geq 1 + d_4 = 1 + d'_2 = 2 + d_2$.

which together with $g_1 \not\leq g_3$ *yields a contradiction.*

4.8 Restricting to term-generated algebras

The apparent paradox in the first part of Example 4 is due to the fact that the axiomatization of $\mathsf{AbsFree}_{\Sigma_0}$ makes it possible to consider models in which the constants in Σ_c are not interpreted as ground Σ_0-terms. We would like to consider only models for which the support A_d of sort d is the set $T_{\Sigma_0}(\emptyset)$ of ground Σ_0-terms (we will refer to them as *term generated models*)[7]. We will assume that the axiomatization of the recursive functions contains a family of constraints $\{C(a) \mid a \in \Sigma_c\}$ expressed in first order logic on the values the function needs to take on any element in Σ_c with the property:

$$(\mathbf{TG}) \quad C(a) \quad \text{iff} \quad \text{there exists } t \in T_{\Sigma_0}(\emptyset) \text{ such that for all } f \in \Sigma_2, f(a) = f(t).$$

Example 5 *Some examples are presented below:*

(1) Assume $\Sigma_2 = \{\mathsf{size}\}$ *(the size function over absolutely free algebras with set of constructors* $\{c_i \mid 1 \leq i \leq n\}$ *with arities* $a(c_i)$*). The following size constraints have the desired property (cf. also [19]):*

$$C(a) = \exists x_1, \dots, x_n(\mathsf{size}(a) = (\sum_{i=1}^{n} a(c_i) * x_i) + 1).$$

[7] For expressing this, we can use axiom IsC (cf. Theorem 10) or the axiom used in [19]:
$(\mathsf{IsConstr}) \quad \forall x \bigvee_{c \in \Sigma_0} \mathsf{Is}_c(x) \quad$ where $\quad \mathsf{Is}_c(x) = \exists x_1, \dots, x_n : x = c(x_1, \dots, x_n)$.

To prove this, note that for every term t, $\mathsf{size}(t) = (\sum_{i=1}^{n} a(c_i) * n(c_i, t) + 1)$, where $n(c_i, t)$ is the number of times c_i occurs in t. Thus, if there exists t such that $\mathsf{size}(t) = \mathsf{size}(a)$, then $C(a)$ is true. Conversely, if $C(a)$ is true $\mathsf{size}(a) = \mathsf{size}(t)$ for every term with x_i occurrences of the constructor c_i for $i = 1, ..., n$.

(2) Consider the depth function (with output sort int) over absolutely free algebras with set of constructors $\{c_i \mid 1 \leq i \leq n\}$. Then $C(a) := \mathsf{depth}(a) \geq 1$.

In what follows we will assume that $\Sigma_1 = \emptyset$.

Theorem 16 *Assume that for every $a \in \Sigma_c$, a set $C(a)$ of constraints satisfying condition* **(TG)** *exists. Then $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathsf{Rec}^{[g]}_{\Sigma_2} \cup \bigcup_{a\in\Sigma_c} C(a)$ satisfies the conditions of a Ψ-local extension of $\mathsf{AbsFree}_c \cup \mathcal{T}_s$ for all sets G of ground unit clauses satisfying the conditions mentioned in Theorem 13, where Ψ is defined as in Theorem 13.*

Note: As in Theorem 13, we can prove, in fact, ELoc^{Ψ}-locality. Hence, the possibility that $C(a)$ may be a first-order formula of sort s is not a problem.

Proof: Let P be a partial model of $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup (\mathsf{Rec}_{\Sigma_2} \cup \bigcup_{a\in\Sigma_c} C(a))[\Psi(G)]$ and of a set G of clauses, with the property that all terms in $\Psi(G)$ are defined, i.e. that if $f_P(c_P(p_1, \dots, p_n))$ is defined and $c \in \Sigma_r(f)$ then $f_P(p_1), \dots, f_P(p_n)$ are all defined in P. We construct a total model $\overline{P}$ of $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathsf{Rec}_{\Sigma_2} \cup \bigcup_{a\in\Sigma_c} C(a)$ and of G as explained in the proof of Theorem 13. We only need to make sure that the new model satisfies $\bigcup_{a\in\Sigma_c} C(a)$. The constraints obviously hold for all constants in Σ_c which occur in G. For all others this can be achieved by choosing, in the definition of every $f_{\overline{P}}$, $f_{\overline{P}}(a) := f(c_d)$ for every $f \in \Sigma_2$, where c_d is an arbitrary but fixed nullary constructor. □

In order to guarantee that we test satisfiability w.r.t. term generated models, in general we have to add, in addition to the constraints $C(a)$, for every function symbol $f \in \Sigma_2$, additional counting constraints describing, for every $x \in A_s$, the maximal number of distinct terms t in $T_{\Sigma_0}(\emptyset)$ with $f(t) = x$. If Σ_0 contains infinitely many nullary constructors the number of distinct terms t in $T_{\Sigma_0}(\emptyset)$ with $f(t) = x$ is infinite, so no counting constraints need to be imposed.

Counting constraints are important if Σ_0 contains only finitely many nullary constructors and if the set G of ground unit clauses we consider contains negative (unit) $\Sigma_0 \cup \Sigma_c$-clauses. For the sake of simplicity, we here only consider sets G of unit ground clauses which contain only negative (unit) clauses of sort s.

Lemma 17 *Assume that $\Sigma_1 = \emptyset$ and for every $a \in \Sigma_c$ there exists a set $C(a)$ of constraints such that condition* **(TG)** *holds. The following are equivalent for any set G of unit $\Sigma_0 \cup \Sigma_2 \cup \Sigma_c$-clauses in which all negative literals have all sort s.*

(1) There exists a term-generated model $A = (T_{\Sigma_0}(\emptyset), A_s, \{f_A\}_{f\in\Sigma_2}, \{a_A\}_{a\in\Sigma_c})$ of $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathsf{Rec}^{[g]}_{\Sigma_2}$ and G.

(2) There exists a model $F = (T_{\Sigma_0}(\Sigma_c), A_s, \{f_F\}_{f\in\Sigma_2}, \{a_F\}_{a\in\Sigma_c})$ of $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathsf{Rec}^{[g]}_{\Sigma_2} \cup \bigcup_{a\in\Sigma_c} C(a)$ and G, where for every $a \in \Sigma_c$, $a_F = a$.

(3) There exists a model $A = (A_d, A_s, \{f_A\}_{f \in \Sigma_2}, \{a_A\}_{a \in \Sigma_c})$ *of* $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathsf{Rec}^{[g]}_{\Sigma_2} \cup \bigcup_{a \in \Sigma_c} C(a)$ *and* G.

Proof: (1) $\Rightarrow$ (2): Let $A = (T_{\Sigma_0}(\emptyset), A_s, \{f_A\}_{f \in \Sigma_2}, \{a_A\}_{a \in \Sigma_c})$ be a model of $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathsf{Rec}_{\Sigma_1}$ as in (1). We define the model F as follows. Let $h : \Sigma_c \rightarrow T_{\Sigma_0}(\emptyset)$ be defined by $h(a) = t_a$, where $t_a \in T_{\Sigma_0}(\emptyset)$ is the term such that $a_A = t_a$. Let $\overline{h}$ be the unique extension of h to a Σ_0-homomorphism from $T_{\Sigma_0}(\Sigma_c)$ to $T_{\Sigma_0}(\emptyset)$. It can be proved by structural induction that for every $t \in T_{\Sigma_0}(\Sigma_c)$, $h(t) = t_A$ (where t_A is the evaluation of t in A, given the interpretation of the constants in Σ_c in A). We define, for every $f \in \Sigma_2$, and $t \in T_{\Sigma_0}(\Sigma_c)$:

$$f_F(t) = f_A(\overline{h}(t)) = f(t)_A.$$

It is easy to see that f_F satisfies Rec_{Σ_2}:

$$\begin{aligned} f_F(c(t_1, \dots, t_n)) &= f_A(\overline{h}(c(t_1, \dots, t_n))) = f_A(c_A(\overline{h}(t_1), \dots, \overline{h}(t_n))) \\ &= g^c_f(f_A(\overline{h}(t_1)), \dots, f_A(\overline{h}(t_n))) = g^c_f(f_F(t_1), \dots, f_F(t_n))). \end{aligned}$$

Note that if $f \in \Sigma_2$ and $t \in T_{\Sigma_0}(\emptyset)$ then $f_F(t) = f(t)_A$. For every $a \in \Sigma_c$, $f_F(a) = f_A(t_a) = f_F(t_a)$. It follows that F satisfies condition $C(a)$ for every $a \in \Sigma_c$. It remains to prove that F is a model of G. We analyze the unit clauses in G. The pure Σ_0-clauses and the pure Σ_s-clauses are obviously true. If $f(t_d) = t_s$ occurs in G, with t_d being a $\Sigma_0 \cup \Sigma_c$-term, and t_s a term of sort s, then $f(t_d)_F = f_F(t_d) = f_A(\overline{h}(t_d)) = f(t_d)_A = t_{sA} = t_{sF}$. Similar arguments can be used for clauses of the form $f(t_d) = g(s_d)$ and for negations thereof.

(2) $\Rightarrow$ (1): Let $F = (T_{\Sigma_0}(\Sigma_c), A_s, \{f_F\}_{f \in \Sigma_2}, \{a_F\}_{a \in \Sigma_c})$ be a model of $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathsf{Rec}_{\Sigma_2} \cup \bigcup_{a \in \Sigma_c} C(a)$ as in (2). Then for every $a \in \Sigma_c$ and every $f \in \Sigma_2$, $f_F(a) = f_F(t_a)$. It can be proved by structural induction that in this case for every $t \in T_{\Sigma_0}(\Sigma_c)$, $f_F(t) = f_F(\overline{h}(t))$, where (as before) $\overline{h} : T_{\Sigma_0}(\Sigma_c) \rightarrow T_{\Sigma_0}(\emptyset)$ is the unique homomorphism with the property that $\overline{h}(a) = t_a$. It is easy to see that for every term t_s of sort s, $\overline{h}(t) = t_F$.

Let $A = (T_{\Sigma_0}(\emptyset), A_s, \{f_A\}_{f \in \Sigma_2}, \{a_A\}_{a \in \Sigma_c})$, where for every $a \in \Sigma_c$, $a_A = t_a$, where the existence of the term t_a is guaranteed by $C(a)$, and f_A is the restriction of f_F to $T_{\Sigma_0}(\emptyset)$. It can be seen that if $t \in T_{\Sigma_c}(\Sigma_c)$ then $f_A(t_{dA}) = f_A(\overline{h}(t)) = f_F(\overline{h}(t))$, where $\overline{h}$ is defined as above.

It is easy to see that then f_A satisfies Rec_{Σ_2}. We prove that G is true in A. The pure Σ_0-clauses and the Σ_s-clauses are obviously true. If $f(t_d) = t_s$ occurs in G, where t_d is a $\Sigma_0 \cup \Sigma_c$-term, and t_s is a term of sort s then $f(t_d)_A = f_A(t_{dA}) = f_A(\overline{h}(t_d)) = f_F(\overline{h}(t_d)) = f_F(t_d) = t_{sF} = \overline{h}(t_s) = (t_s)_A$. Similar arguments can be used for clauses of the form $f(t_d) = g(s_d)$ and for negations thereof.

(2) $\Rightarrow$ (3) is obvious. We prove (3) $\Rightarrow$ (2). Let $A = (A_d, A_s, \{f_A\}_{f \in \Sigma_2}, \{a_A\}_{a \in \Sigma_c})$ be a model of $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathsf{Rec}_{\Sigma_2} \cup \bigcup_{a \in \Sigma_c} C(a)$. For every $a \in \Sigma_c$ let $t_a \in T_{\Sigma_0}(\emptyset)$ be the term with the property that $f_A(a_A) = f_A((t_a)_A)$ for all $f \in \Sigma_2$ (where the existence of the term t_a is guaranteed by $C(a)$). We can define a map

$\overline{h} : T_{\Sigma_0}(\Sigma_c) \rightarrow T_{\Sigma_c}(\emptyset)$ as before, and let $i : T_{\Sigma_c}(\emptyset) \rightarrow A_d$ be the usual evaluation map. We define F as follows: The support of sort s of F is A_s. For every $f \in \Sigma_2$ let $f_F(t) := f_A(i(\overline{h}(t)))$. As before, we can prove that F is a model of Rec_{Σ_2} and of $\bigcup_{a \in \Sigma_c} C(a)$, and – due to the form of G – also a model of G.

The proof remains the same if we replace Rec_{Σ_2} with $\mathsf{Rec}^g_{\Sigma_2}$. □

From Theorem 16 and Lemma 17 it follows that for every set G of ground unit clauses with the form in the statement of Theorem 13 in which all negative (unit) clauses consist of literals of sort s, testing whether there exists a term-generated model of $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s \cup \mathsf{Rec}^{[g]}_{\Sigma_2}$ and G can be done by computing $\mathsf{Rec}^{[g]}_{\Sigma_2}[\Psi(G)]$ and then reducing the problem hierarchically to a satisfiability test w.r.t. $\mathsf{AbsFree}_{\Sigma_0} \cup \mathcal{T}_s$.

Example 6 *Example 4 provides an example of a ground clause G for which:*

- $\mathsf{AbsFree}_{\Sigma_0} \cup \mathbb{Z} \cup \mathsf{Rec}_{\mathsf{depth}} \not\models G$, *and*
- $\mathsf{AbsFree}_{\Sigma_0} \cup \mathbb{Z} \cup \mathsf{Rec}_{\mathsf{depth}} \wedge \mathsf{Bounded}(\mathsf{depth}) \models G$.

Example 4(2) shows that $\mathsf{AbsFree}_{\Sigma_0} \cup \mathbb{Z} \cup \mathsf{Rec}_{\mathsf{depth}} \cup \bigcup_{a \in Const(G)} C(a) \models G$. *Therefore, by Lemma 17, G is true in every term-generated model of* $\mathsf{AbsFree}_{\Sigma_0} \cup \mathbb{Z} \cup \mathsf{Rec}_{\mathsf{depth}}$.

Similar results can be obtained if we relax the restriction on occurrences of negative clauses in G. If the set of nullary constructors in Σ_0 is infinite the extension is easy; otherwise we need to use equality completion and add counting constraints as done e.g. in [19] (assuming that there exist counting constraints expressible in first-order logic for the recursive definitions we consider). A general study of such aspects (including an analysis of possibilities of automatically finding counting constraints) is planned for future work.

5 More general data structures

We will now extend the results above to more general data structures. Consider a signature consisting of a set Σ_0 of constructors (including a set C of constants). Let E be an additional set of identities between Σ_0-terms.

Example 7 *Let $\Sigma_0 = \{c, c_0\}$, where c is a binary constructor and c_0 is a constant. We can impose that E includes one or more of the following equations:*

(A)	$c(c(x, y), z) = c(x, c(y, z))$	*(associativity)*
(C)	$c(x, y) = c(y, x)$	*(commutativity)*
(I)	$c(x, x) = x$	*(idempotence)*
(N)	$c(x, x) = c_0$	*(nilpotence)*

We consider many-sorted extensions of the theory defined by E with functions in $\Sigma = \Sigma_1 \cup \Sigma_2$, and sorts $S = \{d, s\}$, where the functions in Σ_1 have sort $d \rightarrow d$, those in Σ_2 have sort $d \rightarrow s$, and the functions in Σ satisfy additional axioms of the form Rec_Σ and ERec_Σ as defined in Section 4.[8] We therefore consider two-sorted theories of the form $E \cup \mathcal{T}_s \cup (\mathsf{E})\mathsf{Rec}_\Sigma$, where $\mathcal{T}_s$ is a theory of sort s. We make the following assumptions:

Assumption 3: We assume that:
- (a) The equations in E only contain constructors c with $c \in \bigcap_{f\in\Sigma} \Sigma_r(f)$.
- (b) For every $\forall \overline{x} \;\; t(\overline{x}) = s(\overline{x}) \in E$ and every $f \in \Sigma_1 \cup \Sigma_2$ let $t'(\overline{x})$ (resp. $s'(\overline{x})$) be the $\Sigma_{o(f)}$-term obtained by replacing every constructor $c \in \Sigma_0$ with the term-generated function[9] $g^{c,f}$. Then for every $f \in \Sigma_1$, $E \models \forall \overline{x} \;\; t'(\overline{x}) = s'(\overline{x})$, and for every $f \in \Sigma_2$, $\mathcal{T}_s \models \forall \overline{x} \;\; t'(\overline{x}) = s'(\overline{x})$.

Example 8 *Consider the extension of the theory of one binary associative and/or commutative function c with the size function defined as in Example 1(1). Then*

$$\mathsf{size}(c(x,y)) = g^c_{\mathsf{size}}(\mathsf{size}(x), \mathsf{size}(y)), \;\; \textit{where } g^c_{\mathsf{size}}(x,y) = 1 + x + y.$$

Note that g^c_{size} is associative and commutative, so Assumption 3 holds.

$$g^c_{\mathsf{size}}(g^c_{\mathsf{size}}(x,y),z) = 1{+}(1{+}x{+}y){+}z = 1{+}x{+}(1{+}y{+}z) = g^c_{\mathsf{size}}(x, g^c_{\mathsf{size}}(y,z));$$
$$g^c_{\mathsf{size}}(x,y) = 1{+}x{+}y = 1{+}y{+}x = g^c_{\mathsf{size}}(y,x).$$

Example 9 *Assume that Σ_0 only contains the binary constructor c satisfying a set E of axioms containing some of the axioms $\{(\mathbf{A}), (\mathbf{C}), (\mathbf{I})\}$ in Example 7. Let enc_k be a new function symbol (modeling encoding with key k) satisfying*

$$\mathsf{Rec}_{\mathsf{enc}} \qquad \mathsf{enc}_k(c(x,y)) = c(\mathsf{enc}_k(x), \mathsf{enc}_k(y)).$$

It is easy to see that $g^c_{\mathsf{enc}} = c$ and hence Assumption 3 is satisfied.

Lemma 18 *Under Assumption 3 the following holds. For every $f \in \Sigma$ and for all ground terms t, s of sort d, containing only constructors in $\Sigma_r(f)$ let $t'(\overline{x})$ (resp. $s'(\overline{x})$) be the $\Sigma_{o(f)}$-term obtained by replacing every constructor $c \in \Sigma_0$ with the term-generated function g^c_f. Under these conditions and with this notations, if $t \equiv_E s$ then if $f \in \Sigma_1$ then $E \models t' = s'$, and if $f \in \Sigma_2$ then $\mathcal{T}_s \models t' = s'$.*

Proof: We proceed by induction on the number of steps in the proof that $t \equiv_E s$. If the two terms are equal the property is obviously true. Assume now that one step is needed in the proof. We assume for the sake of simplicity that $f \in \Sigma_1$. The other case is similar. We distinguish two cases:

Case 1: There exists a substitution σ and $\forall \overline{x} \;\; u(\overline{x}) = v(\overline{x}) \in E$ such that $t = \sigma(u)$ and $s = \sigma(v)$. From Assumption 3(b), we know that $E \models \forall \overline{y} \; (u'(\overline{y}) = v'(\overline{y}))$

[8] We restrict to unguarded recursive definitions of type Rec_Σ and ERec_Σ to simplify the presentation. Similar results can be obtained for definitions of the type Rec^g_Σ and ERec^g_Σ, with minor changes in Assumption 3.

[9] $g^{c,f}$ is the function (expressible as a $\Sigma_{o(f)}$-term) from the definition $f(c(x_1, \dots, x_n)) = g^{c,f}(f(x_1), \dots, f(x_n))$ in Rec_f.

with the primed versions of terms defined as in Assumption 3(b). In particular, (again with the notations in Assumption 3(b)),

$$E \models u'((\sigma(x_1))', \dots, (\sigma(x_n))') = v'((\sigma(x_1))', \dots, (\sigma(x_n))'),$$

so $E \models t' = s'$.

Case 2: Assume that $t = C[t_1]$ and $s = C[s_1]$, where C is a context and $t_1 = s_1 \in E$ or $t_1 = s_1$ is an instance of an identity in E. Then by Case 1, $E \models t_1' = s_1'$. Since s, t contain only constructors in $\Sigma_r(f)$ and possibly constants, $E \models t' = (C[t_1])' = C'[t_1'] = C'[s_1'] = (C[s_1])' = s'$.

The arguments can now easily be extended to encompass deductions with any number of steps. □

Lemma 19 *Under Assumption 3 the following holds. For every $f \in \Sigma$ and for all ground terms t, s of sort d, containing only constructors in $\Sigma_r(f)$ and possibly constants in $\Sigma_0 \backslash \Sigma_r(f)$, let t' and s' be defined as in Lemma 18 with the difference that every constant c not in $\Sigma_r(f)$ is replaced with a (new) variable x_c. Under these assumptions and with these notations, if $f \in \Sigma_1$ then $E \models \forall x t'(x) = s'(x)$, and if $f \in \Sigma_2$ then $\mathcal{T}_s \models \forall x t'(x) = s'(x)$.*

Proof: We follow the arguments of Lemma 18, and proceed by induction on the number of steps in the proof that $t \equiv_E s$. If the two terms are equal the property is obviously true. Assume now that one step is needed in the proof. We assume for the sake of simplicity that $f \in \Sigma_1$. The other case is similar. As before we distinguish two cases. Since constants in $\Sigma_0 \backslash \Sigma_r(f)$ can only be introduced by applying substitutions to equations in E, we will only show how Case 1 of Lemma 18 can be adapted (the other cases and analogous).

Case 1: There exists a (ground) substitution σ and $\forall \overline{x} \;\; u(\overline{x}) = v(\overline{x}) \in E$ such that $t = \sigma(u)$ and $s = \sigma(v)$. Let $i : \Sigma_0 \backslash \Sigma_r(f) \rightarrow X$ be an injective map, and let $\sigma' : T_{\Sigma_r(f)}(\Sigma_0 \backslash \Sigma_r(f)) \rightarrow T_{\Sigma_r(f)}(X)$ be the unique homomorphism which extends i. [10] From Assumption 3(b), we know that $E \models \forall \overline{y} \; (u'(\overline{y}) = v'(\overline{y}))$. It follows therefore that $E \models \forall \overline{x} \; (u'(\overline{\sigma}"(y)(\overline{x}) = v'(\overline{\sigma}"(y)(\overline{x}))$.
Case 2, concerning applying rules of E in a context and the case of several steps can be proved as in Lemma 18. □

5.1 The problem

In what follows we assume that Assumption 3 holds, and that Rec_{Σ_1} is exhaustive. Note that in the presence of axioms such as associativity, the universal (Horn) theory of E itself may be undecidable. For the sake of simplicity, we here only consider a very simple class of proof tasks, namely the problem of checking whether

$$E \cup [\mathsf{E}]\mathsf{Rec}^{[\mathsf{g}]}{}_{\Sigma_1} \cup [\mathsf{E}]\mathsf{Rec}^{[\mathsf{g}]}{}_{\Sigma_2} \models G_1,$$

[10] $\sigma" = \sigma' \circ \sigma$ coincides with σ' except for the fact that in σ' all constants in $\Sigma_0 \backslash \Sigma_r(f)$ are replaced with variables.

where G_1 is a ground $\Sigma_0 \cup \Sigma_2$-clause of the form

$$\bigwedge_{i=1}^{n} f_i(t_i^d) = t_i^s \wedge \bigwedge_{j=1}^{m} f_j(t_j^d) = f_j'(t_j'^d) \rightarrow f(t_d) = t_s \tag{1}$$

where f_i, f_i', f are functions in Σ_2 (with output sort s different from d), $t_k^d, t_k'^d, t_d$ are ground Σ_0-terms, and $t_k^s, t_k'^s, t_s$ are Σ_s-terms.

Remark. Let G_1 be a clause of type 1 and let $G = \neg G_1$. If $f \in \Sigma_2$ and Rec_f is quasi-exhaustive G is equisatisfiable with a set of (unit) clauses in which every occurrence of f is in a term of the form $f(c)$, with $c \in \Sigma_0 \backslash \Sigma_r(f)$.

Theorem 20 *Assume that Rec_{Σ_1} is exhaustive, Rec_{Σ_2} is quasi-exhaustive and Assumption 3 holds. The following are equivalent for any set G of $\Sigma_0 \cup \Sigma$-clauses of form (1):*

(1) $E \cup \mathsf{Rec}_{\Sigma_1} \cup \mathcal{T}_s \cup \mathsf{Rec}_{\Sigma_2} \models G$.
(2) G is true in all models $A = (A_d, A_s, \{f_A\}_{f \in \Sigma})$ of $E \cup \mathsf{Rec}_{\Sigma_1} \cup \mathcal{T}_s \cup \mathsf{Rec}_{\Sigma_2}$.
(3) G is true in all models $F = (T_{\Sigma_0}(\emptyset)/{\equiv_E}, A_s, \{f_A\}_{f \in \Sigma})$ of $E \cup \mathcal{T}_s \cup \mathsf{Rec}_{\Sigma_1} \cup \mathsf{Rec}_{\Sigma_2}$.
(4) G is true in all weak partial models $F = (T_{\Sigma_0}(\emptyset)/{\equiv_E}, A_s, \{f_A\}_{f \in \Sigma})$ of $E \cup \mathcal{T}_s \cup (\mathsf{Rec}_{\Sigma_1} \cup \mathsf{Rec}_{\Sigma_1})[\Psi(G)]$ in which all terms in $\Psi(G)$ are defined.

Similar results can also be obtained for definitions of type $\mathsf{Rec}_{\Sigma}^{g}$ or $\mathsf{ERec}_{\Sigma}^{[g]}$.

Proof: (1) and (2) are equivalent by definition. (2) $\Rightarrow$ (3) and (4) $\Rightarrow$ (3) are obviously true. We prove that (3) implies (2) and that (3) implies (4).

(3)$\Rightarrow$ (2). Assume that there exists a model $A = (A_d, A_s, \{f_A\}_{f \in \Sigma})$ of $E \cup \mathsf{Rec}_{\Sigma} \cup \mathcal{T}_s$ which is not a model of G (i.e. it satisfies $\neg G$). Let $\overline{A} = (A_0, A_s, \{f_{A_0}\}_{f \in \Sigma})$, where A_0 is the Σ_0-substructure of A_d generated by the empty set, and for every $f \in \Sigma$, $f_{\overline{A}}$ is the reduct of f_A to A_0. The functions are well-defined:

- If $f \in \Sigma_1$ then for every $t \in A_0$, $f_A(t) \in A_0$ because Rec_{Σ_1} is exhaustive.
- If $f \in \Sigma_2$, then there are no problems if we define $f_{A_0}(t) := f_A(t)$.

Since A_0 is a subalgebra of A which contains all the terms in G, and the truth of universally quantified formulae is preserved under subalgebras, it follows that A_0 is a model of $E \cup \mathcal{T}_s \cup \mathsf{Rec}_{\Sigma_1} \cup \mathsf{Rec}_{\Sigma_2}$ and of $\neg G$. Let $h : T_{\Sigma}(\emptyset)/{\equiv_E} \rightarrow A_0$ be the canonical Σ-homomorphism; h is obviously onto. We now define a model $F = (T_{\Sigma}(\emptyset)/{\equiv_E}, A_s, \{f_A\}_{f \in \Sigma_1 \cup \Sigma_2})$ of $E \cup \mathcal{T}_s \cup \mathsf{Rec}_{\Sigma_1} \cup \mathsf{Rec}_{\Sigma_2}$ as follows: If $f \in \Sigma_1$ we define $f(t)$ as required by the rules in Rec_{Σ_1}, which we assumed to be exhaustive. If $f \in \Sigma_2$, we define $f_F([t]) = f_A(h([t]))$. It is easy to check that F with the operations defined this way is a model of $E \cup \mathcal{T}_s \cup \mathsf{Rec}_{\Sigma_1} \cup \mathsf{Rec}_{\Sigma_2}$. We show it is a model of $\neg G$. From the form of G, we know that no $f \in \Sigma_1$ occurs. From the definition of f_F is follows immediately that the truth of all equalities (disequalities) in F coincides with the truth in A.

(3) $\Rightarrow$ (4) Assume that there exists a weak partial model P of $\neg G$ and of $E \cup \mathcal{T}_s \cup (\mathsf{Rec}_{\Sigma_1} \cup \mathsf{Rec}_{\Sigma_1})[\Psi(G)]$, with totally defined $\Sigma_0 \cup \Sigma_s$-functions and partial $\Sigma_1 \cup \Sigma_2$-functions, in which all terms in $\Psi(G)$ are defined. Assume $P =$

$(T_\Sigma(\emptyset)/\equiv_E, A_s, \{f_A\}_{f\in\Sigma_1})$. We construct a total model $\overline{P}$ of $E \cup \mathcal{T}_s \cup \mathsf{Rec}_{\Sigma_1} \cup \mathsf{Rec}_{\Sigma_2}$ and $\neg G$. The model is constructed level-wise on the canonical terms in $T_\Sigma(\emptyset) = \bigcup_{i\geq 0} P_i$ (where $P_0 = \emptyset$ and $P_{i+1} = \{c(t_1, \dots, t_n) \mid c \in \Sigma_0 \text{ and } t_i \in \bigcup_{0\leq j\leq i} P_j\}$), as in the case of free constructors. If $f \in \Sigma_1 \cup \Sigma_2$, and $c \in \Sigma_r(f)$, we define:

$$f_{\overline{P}}([c(t_1, \dots, t_n)]) = g_f^c(f_{\overline{P}}([t_1]), \dots, f_{\overline{P}}([t_n])).$$

If $d \notin \Sigma_r(f)$ (and hence $f \in \Sigma_2$, and d is a constant constructor) then we define:

$$f_{\overline{P}}([d]) = \begin{cases} f_P([d]) & \text{if } f_P([d]) \text{ defined} \\ c_s & \text{otherwise,} \end{cases}$$

where c_s is an arbitrary (but fixed) element of the support of sort s of $\overline{P}$.

We first prove that the functions are well-defined. Let $t, s \in T_{\Sigma_0}(\emptyset)$ be such that $t = c(t_1, \dots, t_n)$, $s = d(s_1, \dots, s_m)$ and $t \equiv_E s$, i.e. $[c(t_1, \dots, t_n)] = [d(s_1, \dots, s_m)]$. If $f \in \Sigma$ and t, s contain only symbols in $\Sigma_r(f)$ then by Lemma 18 we know that $\mathcal{T}_{o(f)} \models t' = s'$, where $t'(\overline{x})$ (resp. $s'(\overline{x})$) are the $\Sigma_{o(f)}$-term obtained by replacing every constructor $c \in \Sigma_0$ with the term-generated function g_f^c, and $\mathcal{T}_d = E$. For $f \in \Sigma_1$ we have:

$$f_{\overline{P}}([t]) = f_{\overline{P}}([c(t_1, \dots, t_n)]) = [t']$$
$$f_{\overline{P}}([s]) = f_{\overline{P}}([d(s_1, \dots, s_m)]) = [s']$$

Similar for $f \in \Sigma_2$. We now analyze the situation when $f \in \Sigma_2$ and $t \equiv_E s$ and t, s may contain constants in $\Sigma_0 \backslash \Sigma_r(f)$. By Lemma 19, in this case $\mathcal{T}_{o(f)} \models \forall x t'(x) = s'(x)$, where t' and s' are defined as before, with the difference that every constant c not in $\Sigma_r(f)$ is replaced with a (new) variable x_c. Then

$$f_{\overline{P}}([t]) = f_{\overline{P}}([c(t_1, \dots, t_n)]) = t'(f(c_1), \dots, f(c_n))$$
$$f_{\overline{P}}([s]) = f_{\overline{P}}([d(s_1, \dots, s_m)]) = s'(f(c_1), \dots, f(c_n))$$

where $c_1, \dots, c_n$ are the only constants in $\Sigma_0 \backslash \Sigma_r(f)$ occurring in s, t. We know that $\mathcal{T}_s \models \forall \overline{x}(t'(\overline{x}) = s'(\overline{x}))$, hence in particular

$$\mathcal{T}_s \models t'(f(c_1), \dots, f(c_n)) = s'(f(c_1), \dots, f(c_n)).$$

This shows that for every $f \in \Sigma_1 \cup \Sigma_2$, $f_{\overline{P}}$ is well-defined. The fact that the axioms in Rec_Σ are satisfied is clear, by the way the functions are defined. □

Note: We can impose boundedness conditions on the recursively defined functions without losing locality (as for absolutely free constructors). [11]

[11] We can also consider axioms which link the values of functions $f_2 \in \Sigma_2$ and $f_1 \in \Sigma_1$ on the constants, such as e.g. "$f_2(f_1(c)){=}t_s$" if we consider clauses G in which if $f_1(c){=}t$ occurs then $t{=}c'$, where c' is a constant constructor not in $\Sigma_r(f_2)$. In the case of Σ_1-functions defined by ERec we can consider additional axioms of the form: $\phi(f_2(x)) {\rightarrow} f_2(f_1(c, x)){=}t'_s$, where t'_s is a ground term of sort s either containing f_2 (and of the form $f_2(c')$) or a pure Σ_s-term.

If Rec_{Σ_1} is exhaustive, the results can be extended to the more general problem of checking the satisfiability of sets of clauses of the form:

$$\bigwedge_{k=1}^{l} g_k(c_k) = t_k^d \wedge \bigwedge_{i=1}^{n} f_i(t_i^d) = t_i^s \wedge \bigwedge_{j=1}^{m} f_j(t_j^d) = f_j'(t_j'^d) \rightarrow f(t_d) = t_s$$

where $g_k \in \Sigma_1$, $c_k \in \Sigma_0 \backslash \Sigma_r(g_k)$, f_i, f_i', f are functions in Σ_2 (with output sort s different from d), $t_k^d, t_k'^d, t_d$ are ground Σ_0-terms, and $t_k^s, t_k'^s, t_s$ are Σ_s-terms.

6 An example inspired from cryptography

In this section we illustrate the ideas on an example inspired by the treatment of a Dolev-Yao security protocol considered in [4] (cf. also Examples 7 and 9). Let $\Sigma_0 = \{c\} \cup C$, where c is a binary constructor, and let enc be a binary function. We analyze the following situations:

(1) c satisfies a set E of axioms and enc is a free binary function. By Theorem 9, the extension of E with the free function enc is a local extension of E.

(2) c is an absolutely free constructor, and enc satisfies the recursive definition:

$$(\mathsf{ERec}_{\mathsf{enc}}) \quad \forall x, y, z \ \ \mathsf{enc}(c(x, y), z) = c(\mathsf{enc}(x, z), \mathsf{enc}(y, z)).$$

By Theorem 13, the extension $\mathsf{AbsFree}_c \subseteq \mathsf{AbsFree}_c \cup \mathsf{ERec}_{\mathsf{enc}}$ satisfies the Ψ-locality condition for all clauses satisfying Assumption 2 (with Ψ as in Theorem 13).

(3) If c is associative (resp. commutative) and enc satisfies axiom $\mathsf{ERec}_{\mathsf{enc}}$ then Assumption 3 is satisfied, so, by Theorem 20, $E \cup \mathsf{ERec}_{\mathsf{enc}}$ satisfies the condition of a Ψ-local extension of E for all clauses of type (1).

Formalizing the intruder deduction problem. We now formalize the version of the deduction system of the Dolev and Yao protocol given in [4]. Let E be the set of identities which specify the properties of the constructors in Σ_0. We use the following chain of successive theory extensions:

$$E \subseteq E \cup \mathsf{ERec}_{\mathsf{enc}} \subseteq E \cup \mathsf{ERec}_{\mathsf{enc}} \cup \mathsf{Bool} \cup \mathsf{Rec}^g_{\mathsf{known}},$$

where known has sort $d \rightarrow \mathsf{bool}$ and $\mathsf{Rec}^g_{\mathsf{known}}$ consists of the following axioms:

$$\begin{array}{lr} \forall x, y & \mathsf{known}(c(x, y)) = \mathsf{known}(x) \sqcap \mathsf{known}(y) \\ \forall x, y & \mathsf{known}(y) = \mathsf{t} \rightarrow \mathsf{known}(\mathsf{enc}(x, y)) = \mathsf{known}(x) \end{array}$$

Intruder deduction problem. The general statement of the intruder deduction problem is: "Given a finite set T of messages and a message m, is it possible to retrieve m from T?".

Encoding the intruder deduction problem. The finite set of known messages, $T = \{t_1, \dots, t_n\}$, where t_i are ground $\Sigma_0 \cup \{\mathsf{enc}\}$-terms, is encoded as $\bigwedge_{i=1}^n \mathsf{known}(t_i) = \mathsf{t}$. With this encoding, the intruder deduction problem becomes:
"Test whether $E \cup \mathsf{Rec}_{\mathsf{enc}} \cup \mathsf{Bool} \cup \mathsf{Rec}_{\mathsf{known}} \models \bigwedge_{i=1}^n \mathsf{known}(t_i) = \mathsf{t} \rightarrow \mathsf{known}(m) = \mathsf{t}$."

Example 10 *We illustrate the hierarchical reasoning method we propose on the following example: Assume that $E = \{(\mathbf{C})\}$ and the intruder knows the messages $c(a,b)$ and $\mathsf{enc}(c(c(e,f),e), c(b,a))$. We check if he can retrieve $c(f,e)$, i.e. if*

$$G : (\mathsf{known}(c(a,b))=\mathsf{t}) \wedge (\mathsf{known}(\mathsf{enc}(c(c(e,f),e), c(b,a)))=\mathsf{t}) \wedge (\mathsf{known}(c(f,e))=\mathsf{f})$$

is unsatisfiable w.r.t. $E \cup \mathsf{Bool} \cup \mathsf{ERec}_{\mathsf{enc}} \cup \mathsf{Rec}^g_{\mathsf{known}}$. By Theorem 20, we know that $E \cup \mathsf{Rec}_{\mathsf{enc}} \cup \mathsf{Bool} \cup \mathsf{Rec}_{\mathsf{known}} \wedge G' \models \perp$ iff $(E \cup \mathsf{Rec}_{\mathsf{enc}}) \cup \mathsf{Bool} \cup \mathsf{Rec}_{\mathsf{known}}[\Psi(G)] \wedge G \models \perp$. The reduction is illustrated below:

$\mathsf{Def}_{\mathsf{bool}}$		$G'_0 \wedge$	$\mathsf{Rec}_{\mathsf{known}}[\Psi(G')]_0$
$k_1 = \mathsf{known}(a)$	$k_5 = \mathsf{known}(\mathsf{enc}(c(c(e,f),e), c(b,a)))$	$k_6 = k_1 \sqcap k_2$	$k_6 = \mathsf{t}$
$k_2 = \mathsf{known}(b)$	$k_6 = \mathsf{known}(c(a,b))$	$k_7 = k_2 \sqcap k_1$	$k_5 = \mathsf{t}$
$k_3 = \mathsf{known}(e)$	$k_7 = \mathsf{known}(c(b,a))$	$k_{10} = k_4 \sqcap k_3$	$k_{10} = \mathsf{f}$
$k_4 = \mathsf{known}(f)$	$k_8 = \mathsf{known}(c(c(e,f),e))$	$k_9 = k_3 \sqcap k_4$	$k_8 = k_9 \sqcap k_3$
$k_9 = \mathsf{known}(c(e,f))$	$k_{10} = \mathsf{known}(c(f,e))$	$k_7 = \mathsf{t} \rightarrow k_5 = k_8$	

(We ignored Con_0.) The contradiction in Bool can be detected immediately.

7 Conclusion

We showed that many extensions with recursive definitions (which can be seen as generalized homomorphism properties) satisfy locality conditions. This allows us to reduce the task of reasoning about the class of recursive functions we consider to reasoning in the underlying theory of data structures (possibly combined with the theories attached to the co-domains of the additional functions). We illustrated the ideas on several examples (including one inspired from cryptography). The main advantage of the method we use consists in the fact that it has the potential of completely separating the task of reasoning about the recursive definitions from the task of reasoning about the underlying data structures. We believe that these ideas will make the automatic verification of certain properties of recursive programs or of cryptographic protocols much easier, and we plan to make a detailed study of applications to cryptography in future work. An implementation of the method for hierarchical reasoning in local theory extensions is available at `www.mpi-inf.mpg.de/~ihlemann/software/index.html` (cf. also [12]). In various test runs it turned out to be extremely efficient, and can be used as a decision procedure for local theory extensions. We plan to extend the program to handle the theory extensions considered in this paper; we expect that this will not pose any problems. There are other classes of bridging functions – such as, for instance, cardinality functions for finite sets and measure functions for subsets of $\mathbb{R}$ (for instance intervals) – which turn out to satisfy similar locality properties. We plan to present such phenomena in a separate paper.

Acknowledgments. Many thanks to the referees for their helpful comments.

This work was partly supported by the German Research Council (DFG) as part of the Transregional Collaborative Research Center "Automatic Verification and Analysis of Complex Systems" (SFB/TR 14 AVACS, `www.avacs.org`).

References

1. A. Armando, S. Ranise, and M. Rusinowitch. A rewriting approach to satisfiability procedures. *Information and Computation*, 183(2):140–164, 2003.
2. C. Barrett, I. Shikanian, and C. Tinelli. An abstract decision procedure for satisfiability in the theory of inductive data types. *Journal on Satisfiability, Boolean Modeling and Computation*, 3:1-17, 2007.
3. M.P. Bonacina and M. Echenim. Rewrite-based decision procedures. *Electronic Notes in Theoretical Computer Science*, 174(11):27-45, 2007.
4. H. Comon-Lundh, R. Treinen. Easy intruder deductions. In *Verification: Theory and Practice. LNCS 2772*, pages 225-242, Springer 2003.
5. H. Comon-Lundh. Challenges in the automated verification of security protocols. In *Automated Reasoning, 4th International Joint Conference, (IJCAR 2008)*, LNCS 5195, pages 396-409, Springer 2008.
6. S. Delaune. Easy intruder deduction problems with homomorphisms. *Information Processing Letters* 97(6), pages 213-218, 2006.
7. H. Ganzinger. Relating semantic and proof-theoretic concepts for polynomial time decidability of uniform word problems. In *16th Annual IEEE Symposium on Logic in Computer Science*, Boston, MA, USA, 2001, pages 81–90. IEEE Computer Society, Los Alamitos, CA, USA.
8. H. Ganzinger, V. Sofronie-Stokkermans, and U. Waldmann. Modular proof systems for partial functions with Evans equality. *Information and Computation*, 204(10):1453–1492, 2006.
9. R. Givan and D. McAllester. New results on local inference relations. In *Principles of Knowledge Representation and reasoning: Proceedings of the Third International Conference (KR'92)*, 1992, pages 403–412. Morgan Kaufmann Press.
10. R. Givan and D.A. McAllester. Polynomial-time computation via local inference relations. *ACM Transactions on Computational Logic*, 3(4):521–541, 2002.
11. C. Ihlemann, S. Jacobs, and V. Sofronie-Stokkermans. On local reasoning in verification. In *Proc. TACAS 2008, LNCS 4963*, pages 265-281, 2008.
12. C. Ihlemann and V. Sofronie-Stokkermans. System description. H-PiLOT. In *In Automated Deduction (CADE-22), LNAI 5663,* pages 131-139, Springer 2009.
13. D.C. Oppen. Reasoning about recursively defined data structures. *Journal of the ACM*, 27(3): 403-411, 1980.
14. V. Sofronie-Stokkermans. Hierarchic reasoning in local theory extensions. In *20th Int. Conf. on Automated Deduction (CADE-20), LNAI 3632*, pages 219–234. Springer, 2005.
15. V. Sofronie-Stokkermans. Hierarchical and modular reasoning in complex theories: The case of local theory extensions. In *Proc. 6th Int. Symp. Frontiers of Combining Systems (FroCos 2007), LNCS 4720*, pp. 47–71. Springer, 2007. Invited paper.
16. V. Sofronie-Stokkermans and C. Ihlemann. Automated reasoning in some local extensions of ordered structures. *J. of Multiple-Valued Logics and Soft Computing* 13(4–6):397–414, 2007.
17. V. Sofronie-Stokkermans. Efficient hierarchical reasoning about functions over numerical domains. In *Proc. KI 2008: Advances in Artificial Intelligence, LNAI 5243,* pages 135-143, Springer, 2008.
18. V. Sofronie-Stokkermans. Locality results for certain extensions of theories with bridging functions. In *In Automated Deduction (CADE-22), LNAI 5663,* pages 67–83, Springer, 2009.
19. T. Zhang, H. Sipma, Z. Manna. Decision procedures for term algebras with integer constraints. *Information and Computation* 204(10): 1526-1574, 2006.