On Combining Theories with Shared Set Operations

Thomas Wies, Ruzica Piskac, and Viktor Kuncak

EPFL School of Computer and Communication Sciences, Switzerland

Abstract. Motivated by applications in software verification, we explore automated reasoning about the non-disjoint combination of theories of infinitely many finite structures, where the theories share set variables and set operations. We prove a combination theorem and apply it to show the decidability of the satisfiability problem for a class of formulas obtained by applying propositional connectives to formulas belonging to: 1) Boolean Algebra with Presburger Arithmetic (with quantifiers over sets and integers), 2) weak monadic second-order logic over trees (with monadic second-order quantifiers), 3) two-variable logic with counting quantifiers (ranging over elements), 4) the Bernays-Schönfinkel-Ramsey class of first-order logic with equality (with $\exists^*\forall^*$ quantifier prefix), and 5) the quantifier-free logic of multisets with cardinality constraints.

1 Introduction

Constraint solvers based on satisfiability modulo theories (SMT) [4, 8, 13] are a key enabling technique in software and hardware verification systems [2, 3]. The range of problems amenable to such approaches depends on the expressive power of the logics supported by the SMT solvers. Current SMT solvers implement the combination of quantifier-free stably infinite theories with disjoint signatures, in essence following the approach pioneered by Nelson and Oppen [27]. Such solvers serve as decision procedures for quantifier-free formulas, typically containing uninterpreted function symbols, linear arithmetic, and bit vectors. The limited expressiveness of SMT prover logics translates into a limited class of properties that automated verification tools can handle.

To support a broader set of applications, this paper considers decision procedures for the combination of *possibly quantified* formulas in *non-disjoint* theories. The idea of combining rich theories within an expressive language has been explored in interactive provers [5, 7, 26, 28]. Such integration efforts are very useful, but do not result in complete decision procedures for the combined logics. The study of completeness for non-disjoint combination is relatively recent [37, 40] and provides foundations for the general problem. Under certain conditions, such as local finiteness, decidability results have been obtained even for non-disjoint theories [14]. Our paper considers a case of combination of non-disjoint theories sharing operations on *sets of uninterpreted elements*, a case that was not considered before. The theories that we consider have the property that the tuples of cardinalities of Venn regions over shared set variables in the models of a formula are a semi-linear set (i.e., expressible in Presburger arithmetic).

Reasoning about combinations of decidable logics. The idea of deciding a combination of logics is to check the satisfiability of a conjunction of formulas $A \wedge B$ by using one decision procedure, D_A, for A, and another decision procedure, D_B, for B. To obtain a complete decision procedure, D_A and D_B must communicate to ensure that a model found by D_A and a model found by D_B can be merged into a model for $A \wedge B$.

Reduction-based decision procedure. We follow a reduction approach to decision procedures. The first decision procedure, D_A, computes a *projection*, S_A, of A onto *shared* set variables, which are free in both A and B. This projection is semantically equivalent to existentially quantifying over predicates and variables that are free in A but not in B; it is the strongest consequence of A expressible only using the shared set variables. D_B similarly computes the projection S_B of B. This reduces the satisfiability of $A \wedge B$ to satisfiability of the formula $S_A \wedge S_B$, which contains only set variables.

A logic for shared constraints on sets. A key parameter of our combination approach is the logic of sets used to express the projections S_A and S_B. A suitable logic depends on the logics of formulas A and B. Inspired by verification of linked data structures, we consider as the logics for A, B the following: weak monadic second-order logic of two successors WS2S [36], two-variable logic with counting C^2 [16,29,34], the Bernays-Schönfinkel-Ramsey class of first-order logic [6], BAPA [22], and quantifier-free logics of multisets [31,32]. Remarkably, the smallest logic needed to express the projection formulas in these logics has the expressive power of Boolean Algebra with Presburger Arithmetic (BAPA), described in [23] and in Fig. 4. We show that the decision procedures for these four logics can be naturally extended to a reduction to BAPA that captures precisely the constraints on set variables. The existence of these reductions, along with quantifier elimination [22] and NP membership of the quantifier-free fragment [23], make BAPA an appealing reduction target for expressive logics.

Contribution summary. We present a technique for showing decidability of theories that share sets of elements. Furthermore, we show that the logics

1. Boolean Algebra with Presburger Arithmetic [9,22,23],
2. weak monadic second-order logic of two successors WS2S [36],
3. two-variable logic with counting C^2 [34],
4. Bernays-Schönfinkel-Ramsey class [6], and
5. quantifier-free multisets with cardinality constraints [31,32]

all meet the conditions of our combination technique. Consequently, we obtain the decidability of quantifier-free combination of formulas in these logics.[1]

```
class Node {Node left, right ; Object data;}
class Tree {
    private static Node root;
    private static int size ; /*:
    private static specvar nodes :: objset ;
    vardefs "nodes=={x. (root,x) ∈ {(x,y). left x = y ∨ right x = y}*}";
    private static specvar content :: objset ;
    vardefs "content=={x. ∃ n. n ≠ null ∧ n ∈ nodes ∧ data n = x} " */

    private void insertAt (Node p, Object e) /*:
      requires "tree [ left , right ] ∧ nodes ⊆ Object.alloc ∧ size = card content ∧
                e ∉ content ∧ e ≠ null ∧ p ∈ nodes ∧ p ≠ null ∧ left p = null"
      modifies nodes,content, left , right ,data, size
      ensures " size = card content" */
    {
        Node tmp = new Node();
        tmp.data = e;
        p. left = tmp;
        size = size + 1;
    }
}
```

Fig. 1. Fragment of insertion into a tree

2 Example: Verifying a Code Fragment

Our example shows a verification condition formula generated when verifying an unbounded linked data structure. The formula belongs to our new decidable class obtained by combining several decidable logics.

Specification and verification in Jahob. Fig. 1 shows a fragment of Java code for insertion into a binary search tree, factored out into a separate `insertAt` method. The search tree has fields (`left`, `right`) that form a tree, and field `data`, which is not necessarily an injective function (an element may be stored multiple times in the tree). The `insertAt` method is meant to be invoked when the insertion procedure has found a node `p` that has no left child. It inserts the given object `e` into a fresh node `tmp` that becomes the new left child of `p`. In addition to Java statements, the example in Fig. 1 contains preconditions and postconditions, written in the notation of the Jahob verification system [21, 39, 41]. The *vardefs* notation introduces two sets: 1) the set of auxiliary objects *nodes*, denoting the `Node` objects stored in the binary tree, and 2) the set *content* denoting the useful content of the tree. To verify such examples in the previously reported approach [41], the user of the system had to manually provide the definitions of auxiliary sets, and to manually introduce certain lemmas describing changes to these sets. Our decidability result means that there is no need to manually introduce these lemmas.

Decidability of the verification condition. Fig. 2 shows the verification condition formula for a method (`insertAt`) that inserts a node into a linked list.

[1] An earlier version of some of these results is in [24].

```
tree [ left , right ]  ∧ left p = null  ∧ p ∈ nodes ∧
nodes={x. (root,x) ∈ {(x,y). left  x = y| right  x = y}^∗} ∧
content={x. ∃ n. n ≠ null ∧ n ∈ nodes ∧ data n = x} ∧
e ∉ content ∧ nodes ⊆ alloc ∧
tmp ∉ alloc ∧ left tmp = null ∧ right tmp = null ∧
data tmp = null ∧ (∀ y. data y ≠ tmp) ∧
nodes1={x. (root,x) ∈ {(x,y). ( left (p:=tmp)) x = y) |  right  x = y} ∧
content1={x. ∃ n. n ≠ null ∧ n ∈ nodes1 ∧ (data(tmp:=e)) n = x} →
             card  content1  = card content  + 1
```

Fig. 2. Verification condition for Fig. 1

```
SHARED SETS: nodes, nodes1, content, content1, {e}, {tmp}

WS2S FRAGMENT:
 tree [ left , right ]  ∧ left p = null  ∧ p ∈ nodes ∧ left tmp = null ∧ right tmp = null ∧
 nodes={x. (root,x) ∈ {(x,y). left  x = y| right  x = y}^∗} ∧
 nodes1={x. (root,x) ∈ {(x,y). ( left (p:=tmp)) x = y) |  right  x = y}
CONSEQUENCE: nodes1=nodes ∪ {tmp}

C2 FRAGMENT:
  data tmp = null ∧ (∀ y. data y ≠ tmp) ∧ tmp ∉ alloc ∧ nodes ⊆ alloc ∧
  content={x. ∃ n. n ≠ null ∧ n ∈ nodes ∧ data n = x} ∧
  content1={x. ∃ n. n ≠ null ∧ n ∈ nodes1 ∧ (data(tmp:=e)) n = x}
CONSEQUENCE: nodes1 ≠ nodes ∪ {tmp} ∨ content1 = content ∪ {e}

BAPA FRAGMENT: e ∉ content ∧ card content1 ≠ card content + 1
CONSEQUENCE: e ∉ content ∧ card content1 ≠ card content + 1
```

Fig. 3. Negation of Fig. 2, and consequences on shared sets

The validity of this formula implies that invoking a method in a state satisfying the precondition results in a state that satisfies the postcondition of `insertAt`. The formula contains the transitive closure operator, quantifiers, set comprehensions, and the cardinality operator. Nevertheless, there is a (syntactically defined) decidable class of formulas that contains the verification condition in Fig. 2. This decidable class is a set-sharing combination of three decidable logics, and can be decided using the method we present in this paper.

To understand the method for proving the formula in Fig. 2, consider the problem of showing the unsatisfiability of the negation of the formula. Fig. 3 shows the conjuncts of the negation, grouped according to three decidable logics to which the conjuncts belong: 1) weak monadic second-order logic of two successors WS2S [36], 2) two-variable logic with counting C^2 [34], and 3) Boolean Algebra with Presburger Arithmetic (BAPA) [9,22,23]. For the formula in each of the fragments, Fig. 3 also shows a consequence formula that contains only shared sets and statements about their cardinalities. (We represent elements as singleton sets, so we admit formulas sharing elements as well.)

A decision procedure. Note that the conjunction of the consequences of three formula fragments is an unsatisfiable formula. This shows that the original verification condition is valid. In general, our decidability result shows that the decision procedures of logics such as WS2S and C^2 can be naturally extended to compute strongest consequences of formulas involving given shared sets. These consequences are all expressed in BAPA, which is decidable. In summary, the following is a decision procedure for satisfiability of combined formulas: 1) split the formula into fragments (belonging to WS2S, C^2, or BAPA); 2) for each fragment compute its strongest BAPA consequence; 3) check the satisfiability of the conjunction of consequences.

3 Syntax and Semantics of Formulas

Higer-order logic. We present our problem in a fragment of classical higher-order logic [1, Chapter 5] with a particular set of types, which we call sorts. We assume that formulas are well-formed according to sorts of variables and logical symbols. Each variable and each logical symbol have an associated sort. The primitive sorts we consider are 1) bool, interpreted as the two-element set $\{\mathsf{true}, \mathsf{false}\}$ of booleans; 2) int, interpreted as the set of integers $\mathbb{Z}$; and 3) obj, interpreted as a non-empty set of elements. The only sort constructors is the binary function space constructor '$\rightarrow$'. We represent a function mapping elements of sorts $s_1, \ldots, s_n$ into an element of sort s_0 as a term of sort $s_1 \times \ldots \times s_n \rightarrow s_0$ where $s_1 \times s_2 \times \ldots \times s_n \rightarrow s_0$ is a shorthand for $s_1 \rightarrow (s_2 \rightarrow \ldots (s_n \rightarrow s_0))$. When $s_1, \ldots, s_n$ are all the same sort s, we abbreviate $s_1 \times \ldots \times s_n \rightarrow s_0$ as $s^n \rightarrow s_0$. We represent a relation between elements of sorts $s_1, \ldots, s_n$ as a function $s_1 \times \ldots \times s_n \rightarrow \mathsf{bool}$. We use set as an abbreviation for the sort $\mathsf{obj} \rightarrow \mathsf{bool}$. We call variables of sort set *set variables*. The equality symbol applies only to terms of the same sort. We assume to have a distinct equality symbol for each sort of interest, but we use the same notation to denote all of them. Propositional operations connect terms of sort bool. We write $\forall x{:}s.F$ to denote a universally quantified formula where the quantified variable has sort s (analogously for $\exists x{:}s.F$ and $\exists x{:}s^K.F$ for counting quantifiers of Section 5.3). We denote by $\mathsf{FV}(F)$ the set of all free variables that occur free in F. We write $\mathsf{FV}_s(F)$ for the free variables of sort s. Note that the variables can be higher-order (we will see, however, that the shared variables are of sort set). A *theory* is simply a set of formulas, possibly with free variables.

Structures. A structure α specifies a finite set, which is also the meaning of obj, and we denote it $\alpha(\mathsf{obj})$.[2] When α is understood we use $[\![X]\!]$ to denote $\alpha(X)$, where X denotes a sort, a term, a formula, or a set of formulas. If S is a set of formulas then $\alpha(S) = \mathsf{true}$ means $\alpha(F) = \mathsf{true}$ for each $F \in X$. In every structure we let $[\![\mathsf{bool}]\!] = \{\mathsf{false}, \mathsf{true}\}$. Instead of $\alpha(F) = \mathsf{true}$ we often write simply $\alpha(F)$. We interpret terms of the sort $s_1 \times \ldots \times s_n \rightarrow s_0$ as total functions $[\![s_1]\!] \times \ldots [\![s_n]\!] \rightarrow [\![s_0]\!]$. For a set A, we identify a function $f : A \rightarrow \{\mathsf{false}, \mathsf{true}\}$ with the subset $\{x \in A \mid f(x) = \mathsf{true}\}$. We thus interpret variables of the sort

[2] We focus on the case of finite $\alpha(\mathsf{obj})$ primarily for simplicity; we believe the extension to the case where domains are either finite or countable is possible and can be done using results from [22, Section 8.1], [34, Section 5], [36].

$\mathsf{obj}^n \to \mathsf{bool}$ as subsets of $[\![\mathsf{obj}]\!]^n$. If s is a sort then $\alpha(s)$ depends only on $\alpha(\mathsf{obj})$ and we denote it also by $[\![s]\!]$. We interpret propositional operations $\wedge, \vee, \neg$ as usual in classical logic. A quantified variable of sort s ranges over all elements of $[\![s]\!]$. (Thus, as in standard model of HOL [1, Section 54], quantification over variables of sort $s_1 \to s_2$ is quantification over all total functions $[\![s_1]\!] \to [\![s_2]\!]$.)

3.1 Boolean Algebra with Presburger Arithmetic

$$\begin{aligned}
F &::= A \mid F_1 \wedge F_2 \mid F_1 \vee F_2 \mid \neg F \mid \forall x{:}s.F \mid \exists x{:}s.F \\
s &::= \mathsf{int} \mid \mathsf{obj} \mid \mathsf{set} \\
A &::= B_1 = B_2 \mid B_1 \subseteq B_2 \mid T_1 = T_2 \mid T_1 < T_2 \mid K \,\mathsf{dvd}\, T \\
B &::= x \mid \emptyset \mid \mathsf{Univ} \mid \{x\} \mid B_1 \cup B_2 \mid B_1 \cap B_2 \mid B^c \\
T &::= x \mid K \mid \mathsf{CardUniv} \mid T_1 + T_2 \mid K \cdot T \mid \mathsf{card}(B) \\
K &::= \ldots -2 \mid -1 \mid 0 \mid 1 \mid 2 \ldots
\end{aligned}$$

Fig. 4. Boolean Algebra with Presburger Arithmetic (BAPA)

It will be convenient to enrich the language of our formulas with operations on integers, sets, and cardinality operations. These operations could be given by a theory or defined in HOL, but we choose to simply treat them as built-in logical symbols, whose meaning must be respected by all structures α we consider. Fig. 4 shows the syntax of Boolean Algebra with Presburger Arithmetic (BAPA) [9,22]. The following are the sorts of symbols appearing in BAPA formulas: $\subseteq : \mathsf{set}^2 \to \mathsf{bool}$, $< : \mathsf{int}^2 \to \mathsf{bool}$, $\mathsf{dvd}_K : \mathsf{int} \to \mathsf{bool}$ for each integer constant K (with $\mathsf{dvd}_K(t)$ denoted by $K \,\mathsf{dvd}\, t$), $\emptyset, \mathsf{Univ} : \mathsf{set}$, $\mathsf{singleton} : \mathsf{obj} \to \mathsf{set}$ (with $\mathsf{singleton}(x)$ denoted as $\{x\}$), $\cap, \cup : \mathsf{set}^2 \to \mathsf{set}$, $\mathsf{complement} : \mathsf{set} \to \mathsf{set}$ (with $\mathsf{complement}(A)$ denoted by A^c), $K : \mathsf{int}$ for each integer constant K, $\mathsf{CardUniv} : \mathsf{int}$, $+ : \mathsf{int}^2 \to \mathsf{int}$, $\mathsf{mul}_K : \mathsf{int} \to \mathsf{int}$ for each integer constant K (with $\mathsf{mul}_K(t)$ denoted by $K \cdot t$), and $\mathsf{card} : \mathsf{set} \to \mathsf{int}$.

We sketch the meaning of the less common among the symbols in Fig. 4. Univ denotes the universal set, that is, $[\![\mathsf{Univ}]\!] = [\![\mathsf{obj}]\!]$. $\mathsf{card}(A)$ denotes the cardinality of the set A. $\mathsf{CardUniv}$ is interpreted as $\mathsf{card}(\mathsf{Univ})$. The formula $K \,\mathsf{dvd}\, t$ denotes that the integer constant K divides the integer t. We note that the condition $x \in A$ can be written in this language as $\{x\} \subseteq A$. Note that BAPA properly extends the first-order theory of Boolean Algebras over finite structures, which in turn subsumes the first-order logic with unary predicates and no function symbols, because e.g. $\exists x{:}\mathsf{obj}.F(x)$ can be written as $\exists X{:}\mathsf{set}.\, \mathsf{card}(X){=}1 \wedge F'(X)$ where in F' e.g. $P(x)$ is replaced by $X \subseteq P$.

BAPA-definable relations between sets. A *semilinear set* is a finite union of *linear sets*. A linear set is a set of the form $\{\boldsymbol{a} + k_1\boldsymbol{b}_1 + \ldots + k_n\boldsymbol{b}_n \mid k_1, \ldots, k_n \in \{0, 1, 2 \ldots\}\}$ where $\boldsymbol{a}, \boldsymbol{b}_1, \ldots, \boldsymbol{b}_n \in \mathbb{Z}^M$. We represent a linear set by its generating vectors $\boldsymbol{a}, \boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$, and a semilinear set by the finite set of representations of its linear sets. It was shown in [15] that a set of integer vectors $S \subseteq \mathbb{Z}^M$ is a solution set of a Presburger arithmetic formula P i.e. $S = \{(v_1, \ldots, v_n).P\}$ iff S is a semilinear set. We then have the following characterization of relationships between sets expressible in BAPA, which follows from [22].

Lemma 1 (BAPA-expressible means Venn-cardinality-semilinear). *Given a finite set U and a relation $\rho \subseteq (2^U)^p$ the following are equivalent:*

1. *there exists a BAPA formula F whose free variables are $A_1, \ldots, A_p$, and have the sort* set*, such that $\rho = \{(s_1, \ldots, s_p) \mid \{A_1 \mapsto s_1, \ldots, A_p \mapsto s_p\}(F)\}$;*
2. *the following subset of $\mathbb{Z}^M$ for $M = 2^p$ is semilinear:* $\{(|s_1^c \cap s_2^c \cap \ldots \cap s_p^c|, |s_1 \cap s_2^c \cap \ldots \cap s_p^c|, \ldots, |s_1 \cap s_2 \cap \ldots \cap s_p|) \mid (s_1, \ldots, s_p) \in \rho\}$.

Structures of interest in this paper. In the rest of this paper we consider structures that interpret the BAPA symbols as defined above. Because the meaning of BAPA-specific symbols is fixed, a structure α that interprets a set of formulas is determined by a finite set $\alpha(\mathsf{obj})$ as well as the values $\alpha(x)$ for each variable x free in the set of formulas. Let $\{\mathsf{obj} \mapsto u, x_1 \mapsto v_1, \ldots, x_n \mapsto v_n\}$ denote the structure α with domain u that interprets each variable x_i as v_i.

4 Combination by Reduction to BAPA

The Satisfiability Problem. We are interested in an algorithm to determine whether there exists a structure $\alpha \in \mathcal{M}$ in which the following formula is true

$$B(F_1, \ldots, F_n) \tag{1}$$

where

1. $F_1, \ldots, F_n$ are formulas with $\mathsf{FV}(F_i) \subseteq \{A_1, \ldots, A_p, x_1, \ldots, x_q\}$.
2. $V_S = \{A_1, \ldots, A_p\}$ are variables of sort set, whereas $x_1, \ldots, x_q$ are the remaining variables.[3]
3. Each formula F_i belongs to a given class of formulas, $\mathcal{F}_i$. For each $\mathcal{F}_i$ we assume that there is a corresponding theory $\mathcal{T}_i \subseteq \mathcal{F}_i$.
4. $B(F_1, \ldots, F_n)$ denotes a formula built from $F_1, \ldots, F_n$ using the propositional operations $\wedge, \vee$. [4]
5. As the set of structures $\mathcal{M}$ we consider all structures α of interest (with finite $[\![\mathsf{obj}]\!]$, interpreting BAPA symbols in the standard way) for which $\alpha(\cup_{i=1}^n \mathcal{T}_i)$.
6. (Set Sharing Condition) If $i \neq j$, then $\mathsf{FV}(\{F_i\} \cup \mathcal{T}_i) \cap \mathsf{FV}(\{F_j\} \cup \mathcal{T}_j) \subseteq V_S$.

Note that, as a special case, if we embed a class of first-order formulas into our framework, we obtain a framework that supports sharing unary predicates, but not e.g. binary predicates.

[3] For notational simplicity we do not consider variables of sort obj because they can be represented as singleton sets, of sort set.

[4] The absence of negation is usually not a loss of generality because most $\mathcal{F}_i$ are closed under negation so B is the negation-normal form of a quantifier-free combination.

Combination Theorem. The formula B in (1) is satisfiable iff one of the disjuncts in its disjunctive normal form is satisfiable. Consider a disjunct $F_1 \wedge \ldots \wedge F_m$ for $m \leq n$. By definition of the satisfiability problem (1), $F_1 \wedge \ldots \wedge F_m$ is satisfiable iff there exists a structure α such that for each $1 \leq i \leq m$, for each $G \in \{F_i\} \cup \mathcal{T}_i$, we have $\alpha(G) = \mathsf{true}$. Let each variable x_i have some sort s_i (such as $\mathsf{obj}^2 \to \mathsf{bool}$). Then the satisfiability of $F_1 \wedge \ldots \wedge F_m$ is equivalent to the following condition:

$$\begin{array}{l}\exists \text{ finite set } u.\ \exists a_1, \ldots, a_p \subseteq u.\ \exists v_1 \in [\![s_1]\!]^u. \ldots\ \exists v_q \in [\![s_q]\!]^u.\ \bigwedge_{i=1}^m \\ \{\mathsf{obj} \to u, A_1 \mapsto a_1, \ldots, A_p \mapsto a_p, x_1 \mapsto v_1, \ldots, x_q \mapsto v_q\}(\{F_i\} \cup \mathcal{T}_i)\end{array} \tag{2}$$

By the set sharing condition, each of the variables $x_1, \ldots, x_q$ appears only in one conjunct and can be moved inwards from the top level to this conjunct. Using x_{ij} to denote the j-th variable in the i-th conjunct we obtain the condition

$$\exists \text{ finite set } u.\ \exists a_1, \ldots, a_p \subseteq u.\ \bigwedge_{i=1}^m C_i(u, a_1, \ldots, a_p) \tag{3}$$

where $C_i(u, a_1, \ldots, a_p)$ is

$$\begin{array}{l}\exists v_{i1}. \ldots \exists v_{iw_i}. \\ \{\mathsf{obj} \to u, A_1 \mapsto a_1, \ldots, A_p \mapsto a_p, x_{i1} \mapsto v_{i1}, \ldots, x_{iw_i} \mapsto v_{iw_i}\}(\{F_i\} \cup \mathcal{T}_i)\end{array}$$

The idea of our combination method is to simplify each condition $C_i(u, a_1, \ldots, a_p)$ into the truth value of a BAPA formula. If this is possible, we say that there exists a BAPA reduction.

Definition 2 (BAPA Reduction). *If $\mathcal{F}_i$ is a set of formulas and $\mathcal{T}_i \subseteq \mathcal{F}_i$ a theory, we call a function $\rho : \mathcal{F}_i \to \mathcal{F}_{\mathsf{BAPA}}$ a BAPA reduction for $(\mathcal{F}_i, \mathcal{T}_i)$ iff for every formula $F_i \in \mathcal{F}_i$ and for all finite u and $a_1, \ldots, a_p \subseteq u$, the condition*

$$\begin{array}{l}\exists v_{i1} \ldots \exists v_{iw_i}. \\ \{\mathsf{obj} \to u, A_1 \mapsto a_1, \ldots, A_p \mapsto a_p, x_{i1} \mapsto v_{i1}, \ldots, x_{iw_i} \mapsto v_{iw_i}\}(\{F_i\} \cup \mathcal{T}_i)\end{array}$$

is equivalent to the condition $\{\mathsf{obj} \to u, A_1 \mapsto a_1, \ldots, A_p \mapsto a_p\}(\rho(F_i))$.

A computable BAPA reduction is a BAPA reduction which is computable as a function on formula syntax trees.

Theorem 3. *Suppose that for every $1 \leq i \leq n$ for $(\mathcal{F}_i, \mathcal{T}_i)$ there exists a computable BAPA reduction ρ_i. Then the problem (1) in Section 4 is decidable.*

Specifically, to check satisfiability of $B(F_1, \ldots, F_n)$, compute $B(\rho_1(F_1), \ldots, \rho_n(F_n))$ and then check its satisfiability using a BAPA decision procedure [22, 23].

5 BAPA Reductions

5.1 Monadic Second-Order Logic of Finite Trees

Figure 5 shows the syntax of (our presentation of) monadic second-order logic of finite trees (FT), a variant of weak monadic second-order logic of two successors

$$
\begin{aligned}
F &::= P \mid F_1 \wedge F_2 \mid F_1 \vee F_2 \mid \neg F \mid \forall x{:}s.F \mid \exists x{:}s.F \\
s &::= \mathsf{obj} \mid \mathsf{set} \\
P &::= B_1 = B_2 \mid B_1 \subseteq B_2 \mid r(x,y) \\
r &::= \mathsf{succ}_L \mid \mathsf{succ}_R \\
B &::= x \mid \epsilon \mid \emptyset \mid \mathsf{Univ} \mid \{x\} \mid B_1 \cup B_2 \mid B_1 \cap B_2 \mid B^c
\end{aligned}
$$

Fig. 5. Monadic Second-Order Logic of Finite Trees (FT)

(WS2S) [19,36]. The following are the sorts of variables specific to FT formulas: $\mathsf{succ}_L, \mathsf{succ}_R : \mathsf{obj}^2 \to \mathsf{bool}$.

We interpret the sort obj over finite, prefix-closed sets of binary strings. More precisely, we use $\{1,2\}$ as the binary alphabet, and we let $[\![\mathsf{obj}]\!] \subset \{1,2\}^*$ such that

$$\forall w \in \{1,2\}^*.\ (w1 \in [\![\mathsf{obj}]\!] \vee w2 \in [\![\mathsf{obj}]\!]) \to w \in [\![\mathsf{obj}]\!]$$

In each model, $[\![\mathsf{set}]\!]$ is the set of all subsets of $[\![\mathsf{obj}]\!]$. We let $[\![\epsilon]\!]$ be the empty string which we also denote by ϵ. We define

$$[\![\mathsf{succ}_L]\!] = \{(w, w1) \mid w1 \in [\![\mathsf{obj}]\!]\} \quad \text{and} \quad [\![\mathsf{succ}_R]\!] = \{(w, w2) \mid w2 \in [\![\mathsf{obj}]\!]\}$$

The remaining constants and operations on sets are interpreted as in BAPA.

Let $\mathcal{F}_{\mathsf{FT}}$ be the set of all formulas in Figure 5. Let $\mathcal{M}_{\mathsf{FT}}$ be the set of all (finite) structures described above. We define $\mathcal{T}_{\mathsf{FT}}$ as the set of all formulas $F \in \mathcal{F}_{\mathsf{FT}}$ such that F is true in all structures from $\mathcal{M}_{\mathsf{FT}}$.

The models of the theory $\mathcal{T}_{\mathsf{FT}}$ correspond up to isomorphism with the interpretations in $\mathcal{M}_{\mathsf{FT}}$.

Lemma 4. *If α is a structure such that $\alpha(\mathcal{T}_{\mathsf{FT}})$ then α is isomorphic to some structure in $\mathcal{M}_{\mathsf{FT}}$.*

Note that any FT formula $F(x)$ with a free variable x of sort obj can be transformed into the equisatisfiable formula $\exists x : \mathsf{obj}. y = \{x\} \wedge F(x)$ where y is a fresh variable of sort set. For conciseness of presentation, in the rest of this section we only consider FT formulas F with $\mathsf{FV}_{\mathsf{obj}}(F) = \emptyset$.

Finite tree automata. In the following, we recall the connection between FT formulas and finite tree automata. Let Σ be a finite ranked alphabet. We call symbols of rank 0 constant symbols and a symbol of rank $k > 0$ a k-ary function symbol. We denote by $\mathsf{Terms}(\Sigma)$ the set of all terms over Σ. We associate a position $p \in \{1, \dots, r_{\max}\}^*$ with each subterm in a term t where $r_{\max}$ is the maximal rank of all symbols in Σ. We denote by $t[p]$ the topmost symbol of the subterm at position p. For instance, consider the term $t = f(g(a,b,c),a)$ then we have $t[\epsilon] = f$ and $t[13] = c$.

A finite (deterministic bottom-up) tree automaton A for alphabet Σ is a tuple (Q, Q_f, ι) where Q is a finite set of states, $Q_f \subseteq Q$ is a set of final states, and ι is a function that associates with each constant symbol $c \in \Sigma$ a state $\iota(c) \in Q$ and with each k-ary function symbol $f \in \Sigma$ a function $\iota(f) : Q^k \to Q$.

We homomorphically extend ι from symbols in Σ to Σ-terms. We say that A accepts a term $t \in \mathsf{Terms}(\Sigma)$ if $\iota(t) \in Q_f$. The language $\mathcal{L}(A)$ accepted by A is the set of all Σ-terms accepted by A.

Let F be an FT formula and let $\mathsf{SV}(F)$ be the set $\mathsf{SV}(F) = \mathsf{FV}(F) \cup \{\mathsf{Univ}\}$. We denote by Σ_F the alphabet consisting of the constant symbol $\bot$ and all binary function symbols f_ν where ν is a function $\nu : \mathsf{SV}(F) \to \{0,1\}$. We inductively associate a Σ_F-term $t_{\alpha,w}$ with every structure $\alpha \in \mathcal{M}_{\mathsf{FT}}$ and string $w \in \{1,2\}^*$ as follows:

$$t_{\alpha,w} = \begin{cases} f_{\nu_{\alpha,w}}(t_{\alpha,w1}, t_{\alpha,w2}) & \text{if } w \in \alpha(\mathsf{obj}) \\ \bot & \text{otherwise} \end{cases}$$

such that for all $x \in \mathsf{SV}(F)$, $\nu_{\alpha,w}(x) = 1$ iff $w \in \alpha(x)$. The language $\mathcal{L}(F) \subseteq \mathsf{Terms}(\Sigma_F)$ of F is then defined by $\mathcal{L}(F) = \{\, t_{\alpha,\epsilon} \mid \alpha \in \mathcal{M}_{\mathsf{FT}} \wedge \alpha(F) \,\}$.

The following theorem states the connection between the structures satisfying FT formulas and the languages accepted by finite tree automata[5].

Theorem 5 (Thatcher and Wright [36]). *For every FT formula F there exists a finite tree automaton A_F over alphabet Σ_F such that $\mathcal{L}(F) = \mathcal{L}(A_F)$ and A_F can be effectively constructed from F.*

Parikh image. We recall Parikh's commutative image [30]. The Parikh image for an alphabet Σ is the function $\mathsf{Parikh} : \Sigma^* \to \Sigma \to \mathbb{N}$ such that for any word $w \in \Sigma^*$ and symbol $\sigma \in \Sigma$, $\mathsf{Parikh}(w)(\sigma)$ is the number of occurrences of σ in w. The Parikh image is extended pointwise from words to sets of words: $\mathsf{Parikh}(W) = \{\, \mathsf{Parikh}(w) \mid w \in W \,\}$. In the following, we implicitly identify $\mathsf{Parikh}(W)$ with the set of integer vectors $\{\, (\chi(\sigma_1), \dots, \chi(\sigma_n)) \mid \chi \in \mathsf{Parikh}(W) \,\}$ where we assume some fixed order on the symbols $\sigma_1, \dots, \sigma_n$ in Σ.

Theorem 6 (Parikh [30]). *Let G be a context-free grammar and $\mathcal{L}(G)$ the language generated from G then the Parikh image of $\mathcal{L}(G)$ is a semilinear set and its finite representation is effectively computable from G.*

We generalize the Parikh image from words to terms as expected: the Parikh image for a ranked alphabet Σ is the function $\mathsf{Parikh} : \mathsf{Terms}(\Sigma) \to \Sigma \to \mathbb{N}$ such that for all $t \in \mathsf{Terms}(\Sigma)$ and $\sigma \in \Sigma$, $\mathsf{Parikh}(t)(\sigma)$ is the number of positions p in t such that $t[p] = \sigma$. Again we extend this function pointwise from terms to sets of terms.

Lemma 7. *Let A be a finite tree automaton over alphabet Σ. Then the Parikh image of $\mathcal{L}(A)$ is a semilinear set and its finite representation is effectively computable from A.*

[5] The theorem was originally stated for WS2S where the universe of all structures is fixed to the infinite binary tree $\{1,2\}^*$ and where all set variables range over finite subsets of $\{1,2\}^*$. It carries over to finite trees in a straightforward manner.

5.2 BAPA Reduction for Monadic Second-Order Logic of Finite Trees

In the following, we prove the existence of a computable BAPA reduction for the theory of monadic second-order logic of finite trees.

Let F be an FT formula and let Σ_F^2 be the set of all binary function symbols in Σ_F, i.e., $\Sigma_F^2 \stackrel{\text{def}}{=} \Sigma_F \setminus \{\bot\}$. We associate with each $\sigma_\nu \in \Sigma_F^2$ the *Venn region* $\mathsf{vr}(\sigma_\nu)$, which is given by a set-algebraic expression over $\mathsf{SV}(F)$: let $\mathsf{SV}(F) = \{x_1, \ldots, x_n\}$ then

$$\mathsf{vr}(\sigma_\nu) \stackrel{\text{def}}{=} x_1^{\nu(x_1)} \cap \cdots \cap x_n^{\nu(x_n)} \ .$$

Hereby x_i^0 denotes x_i^c and x_i^1 denotes x_i. Let $\alpha \in \mathcal{M}_{\mathsf{FT}}$ be a model of F. Then the term $t_{\alpha,\epsilon}$ encodes for each $w \in \alpha(\mathsf{obj})$ the Venn region to which w belongs in α, namely $\mathsf{vr}(t_{\alpha,\epsilon}[w])$. Thus, the Parikh image $\mathsf{Parikh}(t_{\alpha,\epsilon})$ encodes the cardinality of each Venn region over $\mathsf{SV}(F)$ in α.

Lemma 8. *Let F be an FT formula then*

$$\mathsf{Parikh}(\mathcal{L}(F))|_{\Sigma_F^2} = \{\, \{\, \sigma \mapsto |\alpha(\mathsf{vr}(\sigma))| \mid \sigma \in \Sigma_F^2 \,\} \mid \alpha \in \mathcal{M}_{\mathsf{FT}} \land \alpha(F) \,\} \ .$$

According to Theorem 5 we can construct a finite tree automaton A_F over Σ_F such that $\mathcal{L}(F) = \mathcal{L}(A_F)$. From Lemma 7 follows that $\mathsf{Parikh}(\mathcal{L}(F))$ is a semilinear set whose finite representation in terms of base and step vectors is effectively computable from A_F. From this finite representation we can construct a Presburger arithmetic formula ϕ_F over free integer variables $\{\, x_\sigma \mid \sigma \in \Sigma_F \,\}$ whose set of solutions is the Parikh image of $\mathcal{L}(F)$, i.e.

$$\mathsf{Parikh}(\mathcal{L}(F)) = \{\, \{\, \sigma \mapsto k_\sigma \mid \sigma \in \Sigma_F \,\} \mid \{\, x_\sigma \mapsto k_\sigma \mid \sigma \in \Sigma \,\}(\phi_F) \,\} \tag{4}$$

Using the above construction of the Presburger arithmetic formula ϕ_F for a given FT formula F, we define the function $\rho_{\mathsf{FT}} : \mathcal{F}_{\mathsf{FT}} \to \mathcal{F}_{\mathsf{BAPA}}$ as follows:

$$\rho_{\mathsf{FT}}(F) \stackrel{\text{def}}{=} \exists \boldsymbol{x_\sigma}.\, \phi_F \land \bigwedge_{\sigma \in \Sigma_F^2} \mathsf{card}(\mathsf{vr}(\sigma)) = x_\sigma$$

where $\boldsymbol{x}_\sigma$ are the free integer variables of ϕ_F.

Theorem 9. *The function ρ_{FT} is a BAPA reduction for $(\mathcal{F}_{\mathsf{FT}}, \mathcal{T}_{\mathsf{FT}})$.*

5.3 Two-Variable Logic with Counting

Figure 6 shows the syntax of (our presentation of) two-variable logic with counting (denoted C^2) [33]. As usual in C^2, we require that every sub-formula of a formula has at most two free variables. In the atomic formula $r(x_1, x_2)$, variables x_1, x_2 are of sort obj and r is a relation variable of sort $\mathsf{obj}^2 \to \mathsf{bool}$. The formula $\{x\} \subseteq A$ replaces $A(x)$ in predicate-logic notation, and has the expected meaning, with the variable x is of sort obj and A of sort set. The interpretation

$$\begin{aligned} F &::= P \mid F_1 \wedge F_2 \mid F_1 \vee F_2 \mid \neg F \mid \exists^K x{:}\mathsf{obj}.F \\ P &::= x_1 = x_2 \mid \{x\} \subseteq A \mid r(x_1, x_2) \end{aligned}$$

Fig. 6. Two-Variable Logic with Counting (C^2)

of the counting quantifier $\exists^K x{:}\mathsf{obj}.F$ for a positive constant K is that there exist at least K distinct elements x for which the formula F holds.

Let $\mathcal{F}_{C2}$ be the set of all formulas in Figure 6. Let $\mathcal{M}_{C2}$ be the set of structures that interpret formulas in $\mathcal{F}_{C2}$. We define $\mathcal{T}_{C2}$ as the set of all formulas $F \in \mathcal{F}_{C2}$ such that F is true in all structures from $\mathcal{M}_{C2}$. Modulo our minor variation in syntax and terminology (using relation and set variables instead of predicate symbols), $\mathcal{T}_{C2}$ corresponds to the standard set of valid C^2 formulas over finite structures [33].

5.4 BAPA Reduction for Two-Variable Logic with Counting

We next build on the results in [34] to define a BAPA reduction for C^2. We fix set variables $A_1, \ldots, A_p$ and relation variables $r_1, \ldots, r_q$. Throughout this section, let $\Sigma_A = \{A_1, \ldots, A_p\}$, $\Sigma_R = \{r_1, \ldots, r_q\}$, and $\Sigma_0 = \Sigma_A \cup \Sigma_R$. We call $\Sigma_A, \Sigma_R, \Sigma_0$ signatures because they correspond to the notion of signature in the traditional first-order logic formulation of C^2.

Model theoretic types. Define the model-theoretic notion of n-type $\pi_\Sigma(x_1, \ldots, x_n)$ in the signature Σ as the maximal consistent set of non-equality literals in Σ whose obj-sort variables are included in $\{x_1, \ldots, x_n\}$. [6] Given a structure α such that $\alpha(x_1), \ldots, \alpha(x_n)$ are all distinct, α *induces* an n-type

$$\mathsf{ityp}^{\alpha,\Sigma}(x_1, \ldots, x_n) = \{L \mid \alpha(L) \wedge \mathsf{FV}(L) \subseteq \{x_1, \ldots, x_n\},\ L \text{ is } \Sigma\text{-literal without '='}\}$$

We also define the set of n-tuples for which a type π holds in a structure α:

$$S^\alpha(\pi(x_1, \ldots, x_n)) = \{(e_1, \ldots, e_n) \in \alpha(\mathsf{obj})^n \mid \alpha(x_1 := e_1, \ldots, x_n := e_n)(\pi)\}$$

If $\Sigma \subseteq \Sigma'$ and π' is an n-type in signature Σ', by $\pi'|_\Sigma$ we denote the subset of π containing precisely those literals from π whose sets and relations belong to Σ. The family of sets $\{S^\alpha(\pi') \mid \pi'|_\Sigma = \pi\}$ is a partition of $S^\alpha(\pi')$. We will be particularly interested in 1-types. We identify a 1-type $\pi(x)$ in the signature Σ_A with the corresponding Venn region

$$\bigcap\{A_i \mid (\{x\} \subseteq A_i) \in \pi(x)\} \cap \bigcap\{A_i^c \mid (\neg(\{x\} \subseteq A_i)) \in \pi(x)\}.$$

If $\pi_1, \ldots, \pi_m$ is the sequence of all 1-types in the signature Σ and α is a structure, let $I^\alpha(\Sigma) = (|S^\alpha(\pi_1)|, \ldots, |S^\alpha(\pi_m)|)$. If $\mathcal{M}$ is a set of structures let $I^{\mathcal{M}}(\Sigma) = \{I^\alpha(\Sigma) \mid \alpha \in \mathcal{M}\}$.

[6] For example, if Σ has one relation variable r, and two set variables A_1, A_2, then each 2-type with free variables x, y contains, for each of the atomic formulas with variables x, y (i.e. $\{x\} \subseteq A_1$, $\{y\} \subseteq A_1$, $\{x\} \subseteq A_2$, $\{y\} \subseteq A_2$, $r(x,x)$, $r(y,y)$, $r(x,y)$, $r(y,x)$), either the formula or its negation.

Observation 10 *If π is a 1-type in Σ and π' a 1-type in Σ' for $\Sigma \subseteq \Sigma'$, then*

$$|I^\alpha(\pi)| = \sum_{\pi'|_\Sigma = \pi} |I^\alpha(\pi')|$$

Making structures differentiated, chromatic, sparse preserves 1-types. Let ϕ be a C^2 formula with signature Σ_0 of relation symbols. By Scott normal form transformation [34, Lemma 1] it is possible to introduce fresh set variables and compute another C^2 formula ϕ^* in an extended signature $\Sigma^* \supseteq \Sigma_0$, and compute a constant C_ϕ such that, for all sets u with $|u| \geq C_\phi$: 1) if α_0 is a Σ_0 interpretation with domain u such that $\alpha_0(\phi)$, then there exists its Σ^* extension $\alpha^* \supseteq \alpha_0$ such that $\alpha^*(\phi^*)$, and 2) if α^* is a Σ^* interpretation with domain u such that $\alpha^*(\phi^*)$, then for its restriction $\alpha_0 = \alpha^*|_\Sigma$ we have $\alpha_0(\phi)$. By introducing further fresh set- and relation- symbols, [34, lemmas 2 and 3] shows that we can extend the signature from Σ^* to Σ such that each model α^* in Σ^* extends to a model α in Σ, where α satisfies some further conditions of interest: α is *chromatic* and *differentiated*. [34, Lemma 10] then shows that it is possible to transform a model of a formula into a so-called X-*sparse* model for an appropriately computed integer constant X. What is important for us is the following.

Observation 11 *The transformations that start from α_0 with $\alpha_0(\phi)$, and that produce a chromatic, differentiated, X-sparse structure α with $\alpha(\phi)$, have the property that, for structures of size C_ϕ or more,*

1. *the domain remains the same: $\alpha_0(\mathsf{obj}) = \alpha(\mathsf{obj})$,*
2. *the induced 1-types in the signature Σ_0 remain the same: for each 1-type π in signature Σ_0, $S^{\alpha_0}(\pi) = S^\alpha(\pi)$.*

Star types. [34, Definition 9] introduces a star-type $(\pi, \boldsymbol{v})$ (denoted by letter σ) as a description of a local neighborhood of a domain element, containing its induced 1-type π as well as an integer vector $\boldsymbol{v} \subseteq \mathbb{Z}^N$ that counts 2-types in which the element participates, where N is a function of the signature Σ. A star type thus gives a more precise description of the properties of a domain element than a 1-type. Without repeating the definition of star type [34, Definition 9], we note that we can similarly define the set $S^\alpha((\pi, \boldsymbol{v}))$ of elements that realize a given star type $(\pi, \boldsymbol{v})$. Moreover, for a given 1-type π, the family of the non-empty among the sets $S^\alpha((\pi, \boldsymbol{v}))$ partitions the set $S^\alpha(\pi)$.

Frames. The notion of Y-*bounded chromatic frame* [34, Definition 11] can be thought of as a representation of a disjunct in a normal form for the formula ϕ^*. It summarizes the properties of elements in the structure and specifies (among others), the list of possible star types $\sigma_1, \ldots, \sigma_N$ whose integer vectors $\boldsymbol{v}$ are bounded by Y. For a given ϕ^*, it is possible to effectively compute the set of C_ϕ-bounded frames $\mathcal{F}$ such that $\mathcal{F} \models \phi^*$ holds. The '$\models$' in $\mathcal{F} \models \phi^*$ is a certain syntactic relation defined in [34, Definition 13].

For each frame $\mathcal{F}$ with star-types $\sigma_1, \ldots, \sigma_N$, [34, Definition 14] introduces an effectively computable Presburger arithmetic formula $P_\mathcal{F}$ with N free variables.

We write $P_{\mathcal{F}}(w_1, \ldots, w_N)$ if $P_{\mathcal{F}}$ is true when these variables take the values $w_1, \ldots, w_N$. The following statement is similar to the main [34, Theorem 1], and can be directly recovered from its proof and the proofs of the underlying [34, lemmas 12,13,14].

Theorem 12. *Given a formula ϕ^*, and the corresponding integer constant C_ϕ, there exists a computable constant X such that if $N \leq X$, if $\sigma_1, \ldots, \sigma_N$ is a sequence of star types in Σ whose integer vectors are bounded by C_ϕ, and $w_1, \ldots, w_N$ are integers, then the following are equivalent:*

1. *There exists a chromatic differentiated structure α such that $\alpha(\phi^*)$, $w_i{=}|S^\alpha(\sigma_i)|$ for $1 \leq i \leq N$, and $\alpha(\mathsf{obj}) = \bigcup_{i=1}^{N} S^\alpha(\sigma_i)$.*
2. *There exists a chromatic frame $\mathcal{F}$ with star types $\sigma_1, \ldots, \sigma_N$, such that $\mathcal{F} \models \phi^*$ and $P_{\mathcal{F}}(w_1, \ldots, w_N)$.*

We are now ready to describe our BAPA reduction. Fix $V_1, \ldots, V_M$ to be the list of all 1-types in signature Σ_A; let $s_1, \ldots, s_M$ be variables corresponding to their counts. By the transformation of models into chromatic, differentiated, X-sparse ones, the observations 11, 10, and Theorem 12, we obtain

Corollary 13. *If $\mathcal{M} = \{\alpha \mid \alpha(\phi^*)\}$, then there is a computable constant X such that $I^{\mathcal{M}}(\Sigma_A) = \{(s_1, \ldots, s_M) \mid F_{\phi^*}(s_1, \ldots, s_M)\}$ where $F_{\phi^*}(s_1, \ldots, s_M)$ is the following Presburger arithmetic formula*

$$\bigvee_{N,\sigma_1,\ldots,\sigma_N,\mathcal{F}} \exists w_1, \ldots, w_N.\ P_{\mathcal{F}}(w_1, \ldots, w_N) \wedge \bigwedge_{j=1}^{M} s_j = \sum \{w_i \mid V_j = (\pi_i|_{\Sigma_A})\}$$

where N ranges over $\{0, 1, \ldots, X\}$, $\sigma_1, \ldots, \sigma_N$ range over sequences of C_ϕ-bounded star types, and where $\mathcal{F}$ ranges over the C_ϕ-bounded frames with star types $\sigma_1, \ldots, \sigma_N$ such that $\mathcal{F} \models \phi^*$.

By adjusting for the small structures to take into account Scott normal form transformation, we further obtain

Corollary 14. *If $\mathcal{M} = \{\alpha \mid \alpha(\phi)\}$, then $I^{\mathcal{M}}(\Sigma_A) = \{(s_1, \ldots, s_M) \mid G_\phi(s_1, \ldots, s_M)\}$ where $G_\phi(s_1, \ldots, s_M)$ is the Presburger arithmetic formula*

$$\begin{array}{l} \sum_{i=1}^{M} s_i \geq C_\phi \wedge F_{\phi^*}(s_1, \ldots, s_M) \quad \bigvee \\ \bigvee \{ \bigwedge_{i=1}^{M} s_i = d_i \mid \exists \alpha.\ |\alpha(\mathsf{obj})| < C_\phi \ \wedge (d_1, \ldots, d_M) \in I^\alpha(\Sigma_A)\} \end{array}$$

Theorem 15. *The following is a BAPA reduction for C^2 over finite models to variables Σ_A: given a two-variable logic formula ϕ, compute the BAPA formula $\exists s_1, \ldots, s_M.\ G_\phi(s_1, \ldots, s_M) \ \wedge \bigwedge_{i=1}^{M} \mathsf{card}(V_i) = s_i$.*

5.5 Bernays-Schönfinkel-Ramsey Fragment of First-Order Logic

Figure 7 shows the syntax of (our presentation of) the Bernays-Schönfinkel-Ramsey fragment of first-order logic with equality [6], often called effectively propositional logic (EPR). The interpretation of atomic formulas is analogous as for C^2 in previous section. Quantification is restricted to variables of sort obj and must obey the usual restriction of $\exists^*\forall^*$-prenex form that characterizes the Bernays-Schönfinkel-Ramsey class.

$$\begin{aligned}
F &::= \exists z_1{:}\mathsf{obj}.\ldots\exists z_n{:}\mathsf{obj}.\,\forall y_1{:}\mathsf{obj}.\ldots\forall y_m{:}\mathsf{obj}.\,B \\
B &::= P \mid B_1 \wedge B_2 \mid B_1 \vee B_2 \mid \neg B \\
P &::= x_1 = x_2 \mid \{x\} \subseteq A \mid r(x_1,\ldots,x_k)
\end{aligned}$$

Fig. 7. Bernays-Schönfinkel-Ramsey Fragment of First-Order Logic

5.6 BAPA Reduction for Bernays-Schönfinkel-Ramsey Fragment

Our BAPA reduction for the Bernays-Schönfinkel-Ramsey fragment (EPR) is in fact a reduction from EPR formulas to *unary* EPR formulas, in which all free variables have the sort set. To convert a unary EPR formula into BAPA, treat first-order variables as singleton sets and apply quantifier elimination for BAPA [22].

Theorem 16 (BAPA Reduction for EPR). *Let ϕ be a quantifier-free formula whose free variables are: 1) $A_1,\ldots,A_p$, of sort* set*, 2) $r_1,\ldots,r_q$, each r_i of sorts $\mathsf{obj}^{K(i)} \to \mathsf{bool}$ for some $K(i) \geq 2$, 3) $z_1,\ldots,z_n,y_1,\ldots,y_m$, of sort* obj*. Then*

$$\exists r_1,\ldots r_q.\ \exists z_1,\ldots,z_n.\ \forall y_1,\ldots,y_m.\ \phi$$

is equivalent to an effectively computable BAPA formula.

The proof of Theorem 16 builds on and generalizes, for finite models, the results on the spectra of EPR formulas [10, 11, 35]. We here provide some intuition. The key insight [35] is that, when a domain of a model of an EPR formula has sufficiently many elements, then the model contains an induced submodel S of m nodes such that for every $0 \leq k < m$ elements $e_1,\ldots,e_k$ outside S the m-type induced by $e_1,\ldots,e_k$ and any $m-k$ elements in S is the same. Then an element of S can be replicated to create a model with more elements, without changing the set of all m-types in the model and thus without changing the truth value of the formula. Moreover, every sufficiently large model of the EPR formula that has a submodel S with more than m such symmetric elements can be shrunk to a model by whose expansion it can be generated. This allows us to enumerate a finite (even if very large) number of characteristic models whose expansion generates all models. The expansion of a characteristic model increases by one the number of elements of some existing 1-type, so the cardinalities of Venn regions of models are a semilinear set whose base vectors are given by characteristic models and whose step vectors are given by the 1-types being replicated.

5.7 Quantifier-free Mutlisets with Cardinality Constraints

Figure 8 shows the syntax of quantifier-free multiset constraints with cardinality operators [31] in our setting. Multiset expressions have sort $\mathsf{obj} \to \mathsf{int}$, which we abbreviate by $\mathsf{multiset}$ in the following. Formulas are built from set expressions over set variables of sort set, multiset expressions over multiset variables of sort $\mathsf{multiset}$ and inner and outer linear arithmetic formulas. Formally, we have distinct variables for operations on sets and multisets, e.g., we have a variable $\cup_s : \mathsf{set}^2 \to \mathsf{set}$ and a variable $\cup_m : \mathsf{multiset}^2 \to \mathsf{multiset}$, but we use the same symbol $\cup$ for both of them.

top-level formulas:

$F ::= A \mid F \wedge F \mid \neg F$

$A ::= S{=}S \mid S \subseteq S \mid M{=}M \mid M \subseteq M \mid \forall e : \mathsf{obj}.\mathsf{F}^{\mathsf{in}} \mid \mathsf{A}^{\mathsf{out}}$

outer linear arithmetic formulas:

$\mathsf{F}^{\mathsf{out}} ::= \mathsf{A}^{\mathsf{out}} \mid \mathsf{F}^{\mathsf{out}} \wedge \mathsf{F}^{\mathsf{out}} \mid \neg \mathsf{F}^{\mathsf{out}}$

$\mathsf{A}^{\mathsf{out}} ::= \mathsf{t}^{\mathsf{out}} \leq \mathsf{t}^{\mathsf{out}} \mid \mathsf{t}^{\mathsf{out}}{=}\mathsf{t}^{\mathsf{out}} \mid (\mathsf{t}^{\mathsf{out}}, \ldots, \mathsf{t}^{\mathsf{out}}){=}\sum(\mathsf{t}^{\mathsf{in}}, \ldots, \mathsf{t}^{\mathsf{in}})$

$\mathsf{t}^{\mathsf{out}} ::= k \mid \mathsf{card}(S) \mid \mathsf{card}(M) \mid K \mid \mathsf{t}^{\mathsf{out}} + \mathsf{t}^{\mathsf{out}} \mid K \cdot \mathsf{t}^{\mathsf{out}} \mid \mathsf{ite}(\mathsf{F}^{\mathsf{out}}, \mathsf{t}^{\mathsf{out}}, \mathsf{t}^{\mathsf{out}})$

inner linear arithmetic formulas:

$\mathsf{F}^{\mathsf{in}} ::= \mathsf{A}^{\mathsf{in}} \mid \mathsf{F}^{\mathsf{in}} \wedge \mathsf{F}^{\mathsf{in}} \mid \neg \mathsf{F}^{\mathsf{in}}$

$\mathsf{A}^{\mathsf{in}} ::= \mathsf{t}^{\mathsf{in}} \leq \mathsf{t}^{\mathsf{in}} \mid \mathsf{t}^{\mathsf{in}}{=}\mathsf{t}^{\mathsf{in}}$

$\mathsf{t}^{\mathsf{in}} ::= m(e) \mid K \mid \mathsf{t}^{\mathsf{in}} + \mathsf{t}^{\mathsf{in}} \mid K \cdot \mathsf{t}^{\mathsf{in}} \mid \mathsf{ite}(\mathsf{F}^{\mathsf{in}}, \mathsf{t}^{\mathsf{in}}, \mathsf{t}^{\mathsf{in}})$

set expressions:

$S ::= s \mid \emptyset \mid S \cap S \mid S \cup S \mid S^c \mid \mathsf{setof}(M)$

multiset expressions:

$M ::= m \mid \emptyset \mid M \cap M \mid M \cup M \mid M \uplus M \mid M \setminus M \mid M \setminus\!\setminus M \mid \mathsf{multisetof}(S)$

terminals:

s - set variables; m - multiset variables; e - index variable (fixed)

k - integer variable; K - integer constant

Fig. 8. Quantifier-Free Multiset Constraints with Cardinality Operators

We restrict ourself to structures α that interpret multiset variables as functions from $[\![\mathsf{obj}]\!]$ to the nonnegative integers. Set and arithmetic operations are interpreted as in BAPA. Multiset operations are interpreted as expected, in particular, '$\uplus$' denotes additive union, '$\setminus$' denotes multiset difference, and '$\setminus\!\setminus$' denotes set difference. The variables $\mathsf{multisetof}$ and setof are interpreted as functions that convert between multisets and sets, e.g., $[\![\mathsf{setof}]\!]$ maps a multiset $M : [\![\mathsf{obj}]\!] \to \mathbb{N}$ to the set $\{\, e \mid M(e) > 0 \,\}$. The variable ite is interpreted as the conditional choice function, i.e., $[\![\mathsf{ite}(F, t_1, t_2)]\!]$ denotes $[\![t_1]\!]$ if $[\![F]\!] = \mathsf{true}$ and $[\![t_2]\!]$ otherwise. Finally, the atom $(u_1, \ldots, u_n) {=} \sum(t_1, \ldots, t_n)$ denotes true iff for all $i \in [1, n]$

$$\alpha(u_i) = \sum_{o \in [\![obj]\!]} \alpha[e := o](t_i)$$

Let $\mathcal{F}_{\mathsf{MS}}$ be the set of all formulas defined in Figure 8 and let $\mathcal{M}_{\mathsf{MS}}$ be the set of all structures interpreting formulas in $\mathcal{F}_{\mathsf{MS}}$ as described above. We define the theory of quantifier-free multisets with cardinality constraints $\mathcal{T}_{\mathsf{MS}}$ as the set of all formulas $F \in \mathcal{F}_{\mathsf{MS}}$ such that $\alpha(F)$ is true for all structures α in $\mathcal{M}_{\mathsf{MS}}$.

5.8 BAPA Reduction for Quantifier-free Multiset Constraints

The satisfiability of the quantifier-free fragment of multisets with cardinality operators is decidable [31]. There is, in fact, also a BAPA reduction from a quantifier-free multiset formula over multiset and set variables to a BAPA formula ranging only over the set variables.

Let $F \in \mathcal{F}_{\mathsf{MS}}$ be a multiset constraint containing set variables $A_1, \ldots, A_p$ and multiset variables $M_1, \ldots, M_q$. To obtain a BAPA reduction, we apply the decision procedure described in [31] to the formula

$$F_1 \overset{\text{def}}{=} F \wedge \bigwedge_{i=1}^{w} \mathsf{card}(V_i){=}k_i$$

where $k_1, \ldots, k_w$ are fresh integer variables and $V_1, \ldots, V_w$ are the Venn regions over the set variables $A_1, \ldots, A_p$. Before applying the decision procedure we convert F_1 to a formula F_2 that only ranges over multiset variables. This is done by replacing every set operation in F by the corresponding multiset operation, replacing every set variable A_i by a fresh multiset variable M_{A_i}, and conjoining the formula

$$\forall e : \mathsf{obj}. \bigwedge_{i=1}^{p} (M_{A_i}(e) = 0 \vee M_{A_i}(e) = 1) \ .$$

The decision procedure constructs a Presburger arithmetic formula P with $\{k_1, \ldots, k_w\} \subseteq \mathsf{FV}(P)$. From the proofs of [31, Theorems 1, 2, and Lemma 3] follows that

$$\{\, \alpha|_{\{k_1,\ldots,k_w\}} \mid \alpha(F_2) \,\} = \{\, \alpha|_{\{k_1,\ldots,k_w\}} \mid \alpha(P) \,\}$$

If $x_1, \ldots, x_n$ are the variables in P other than $k_1, \ldots, k_w$ then the result of the BAPA reduction is the formula

$$P_F \overset{\text{def}}{=} \exists k_1 : \mathsf{int}. \ldots \exists k_w : \mathsf{int}. \ (\bigwedge_{i=1}^{w} \mathsf{card}(V_i){=}k_i) \wedge (\exists x_1{:}\mathsf{int}. \ldots \exists x_n{:}\mathsf{int}.\ P) \qquad (5)$$

Theorem 17. *The function mapping a formula $F \in \mathcal{F}_{\mathsf{MS}}$ to the BAPA formula P_F is a BAPA reduction for $(\mathcal{F}_{\mathsf{MS}}, \mathcal{T}_{\mathsf{MS}})$.*

The proof of Theorem 17 uses Equation 5 following a similar argumentation than in the proof of Theorem 9.

6 Further Related Work

There are combination results for the disjoint combinations of non-stably infinite theories [10, 11, 20, 38]. These results are based on the observation that such combinations are possible whenever one can decide for each component theory whether a model of a specific cardinality exists. Our combination result takes into account not only the cardinality of the models, i.e. the interpretation of the universal set, but cardinalities of Venn regions over the interpretations of arbitrary shared set variables. It is a natural generalization of the disjoint case restricted to theories that share the theory of finite sets, thus, leading to a non-disjoint combination of non-stably infinite theories.

Ghilardi [14] proposes a model-theoretic condition for decidability of the non-disjoint combination of theories based on quantifier elimination and local finiteness of the shared theory. Note that BAPA is not locally finite and that, in general, we need the full expressive power of BAPA to compute the projections on the shared set variables. For instance, consider the C^2 formula

$$(\forall x.\exists^{=1} y.r(x,y)) \land (\forall x.\exists^{=1} y.r(y,x)) \land (\forall y.\ y \in B \leftrightarrow (\exists x.x \in A \land r(x,y)))$$

where r is a binary relation variable establishing the bijection between A and B. This constraint expresses $|A| = |B|$ without imposing any additional constraint on A and B. Similar examples can be given for weak monadic second-order logic of finite trees.

The reduction approach to combination of decision procedures has previously been applied in the simpler scenario of reduction to propositional logic [25]. Like propositional logic, quantifier-Free BAPA is NP-complete, so it presents an appealing alternative for combination of theories that share sets.

Gabbay and Ohlbach [12] present a procedure, called SCAN, for second-order quantifier elimination. However, [12] gives no characterization of when SCAN terminates. We were therefore unable to use SCAN to derive any BAPA reductions.

The general combination of weak monadic second-order logics with linear cardinality constraints has been proven undecidable by Klaedtke and Rueß [17, 18]. They introduce the notion of Parikh automata to identify decidable fragments of this logic which inspired our BAPA reduction of MSOL of finite trees. Our combined logic is incomparable to the decidable fragments identified by Klaedtke and Rueß because it supports non-tree structures as well. However, by applying projection to C^2 and the Bernays-Schönfinkel-Ramsey class, we can combine our logic with [17, 18], obtaining an even more expressive decidable logic.

7 Conclusion

Many verification techniques rely on decision procedures to achieve a high degree of automation. The class of properties that such techniques are able to verify is therefore limited by the expressive power of the logics supported by the underlying decision procedures. We have presented a combination result for logics that share operations on sets. This result yields an expressive decidable logic that is useful for software verification. We therefore believe that we made an important step in increasing the class of properties that are amenable to automated verification.

References

1. P. B. Andrews. *An Introduction to Mathematical Logic and Type Theory: To Truth Through Proof.* Springer (Kluwer), 2nd edition, 2002.
2. T. Ball, A. Podelski, and S. K. Rajamani. Relative completeness of abstraction refinement for software model checking. In *TACAS'02*, volume 2280 of *LNCS*, 2002.
3. M. Barnett, R. DeLine, M. Fähndrich, K. R. M. Leino, and W. Schulte. Verification of object-oriented programs with invariants. *Journal of Object Technology*, 3(6):27–56, 2004.
4. C. Barrett and C. Tinelli. CVC3. In *CAV*, volume 4590 of *LNCS*, 2007.
5. D. Basin and S. Friedrich. Combining WS1S and HOL. In *FroCoS*, 1998.
6. E. Börger, E. Grädel, and Y. Gurevich. *The Classical Decision Problem.* Springer-Verlag, 1997.
7. R. S. Boyer and J. S. Moore. Integrating decision procedures into heuristic theorem provers: A case study of linear arithmetic. In *Machine Intelligence*, volume 11. Oxford University Press, 1988.
8. L. de Moura and N. Bjørner. Z3: An efficient SMT solver. In *TACAS*, pages 337–340, 2008.
9. S. Feferman and R. L. Vaught. The first order properties of products of algebraic systems. *Fundamenta Mathematicae*, 47:57–103, 1959.
10. P. Fontaine. Combinations of theories and the bernays-schönfinkel-ramsey class. In *VERIFY*, 2007.
11. P. Fontaine. Combinations of decidable fragments of first-order logic. In *FroCoS*, 2009.
12. D. M. Gabbay and H. J. Ohlbach. Quantifier elimination in second-order predicate logic. In B. Nebel, C. Rich, and W. Swartout, editors, *Principles of Knowledge Representation and Reasoning.* Morgan-Kaufmann, 1992.
13. Y. Ge, C. Barrett, and C. Tinelli. Solving quantified verification conditions using satisfiability modulo theories. In *CADE*, 2007.
14. S. Ghilardi. Model theoretic methods in combined constraint satisfiability. *Journal of Automated Reasoning*, 33(3-4):221–249, 2005.
15. S. Ginsburg and E. Spanier. Semigroups, Pressburger formulas and languages. *Pacific Journal of Mathematics*, 16(2):285–296, 1966.
16. E. Grädel, M. Otto, and E. Rosen. Two-variable logic with counting is decidable. In *LICS*, 1997.
17. F. Klaedtke and H. Rueß. Parikh automata and monadic second-order logics with linear cardinality constraints. Technical Report 177, Institute of Computer Science at Freiburg University, 2002.
18. F. Klaedtke and H. Rueß. Monadic second-order logics with cardinalities. In *ICALP*, volume 2719 of *LNCS*, 2003.
19. N. Klarlund and A. Møller. *MONA Version 1.4 User Manual.* BRICS Notes Series NS-01-1, Department of Computer Science, University of Aarhus, January 2001.

20. S. Krstic, A. Goel, J. Grundy, and C. Tinelli. Combined satisfiability modulo parametric theories. In *TACAS*, volume 4424 of *LNCS*, pages 602–617, 2007.
21. V. Kuncak. *Modular Data Structure Verification.* PhD thesis, EECS Department, Massachusetts Institute of Technology, February 2007.
22. V. Kuncak, H. H. Nguyen, and M. Rinard. Deciding Boolean Algebra with Presburger Arithmetic. *J. of Automated Reasoning*, 2006.
23. V. Kuncak and M. Rinard. Towards efficient satisfiability checking for Boolean Algebra with Presburger Arithmetic. In *CADE-21*, 2007.
24. V. Kuncak and T. Wies. On set-driven combination of logics and verifiers. Technical Report LARA-REPORT-2009-001, EPFL, February 2009.
25. S. K. Lahiri and S. A. Seshia. The UCLID decision procedure. In *CAV'04*, 2004.
26. S. McLaughlin, C. Barrett, and Y. Ge. Cooperating theorem provers: A case study combining HOL-Light and CVC Lite. In *PDPAR*, volume 144(2) of *ENTCS*, 2006.
27. G. Nelson and D. C. Oppen. Simplification by cooperating decision procedures. *ACM TOPLAS*, 1(2):245–257, 1979.
28. S. Owre, J. M. Rushby, and N. Shankar. PVS: A prototype verification system. In D. Kapur, editor, *11th CADE*, volume 607 of *LNAI*, pages 748–752, jun 1992.
29. L. Pacholski, W. Szwast, and L. Tendera. Complexity results for first-order two-variable logic with counting. *SIAM J. on Computing*, 29(4):1083–1117, 2000.
30. R. J. Parikh. On context-free languages. *J. ACM*, 13(4):570–581, 1966.
31. R. Piskac and V. Kuncak. Decision procedures for multisets with cardinality constraints. In *VMCAI*, number 4905 in LNCS, 2008.
32. R. Piskac and V. Kuncak. Linear arithmetic with stars. In *CAV*, 2008.
33. I. Pratt-Hartmann. Complexity of the two-variable fragment with (binary-coded) counting quantifiers. *CoRR*, cs.LO/0411031, 2004.
34. I. Pratt-Hartmann. Complexity of the two-variable fragment with counting quantifiers. *Journal of Logic, Language and Information*, 14(3):369–395, 2005.
35. F. P. Ramsey. On a problem of formal logic. *Proc. London Math. Soc.*, s2-30:264–286, 1930. doi:10.1112/plms/s2-30.1.264.
36. J. W. Thatcher and J. B. Wright. Generalized finite automata theory with an application to a decision problem of second-order logic. *Mathematical Systems Theory*, 2(1):57–81, 1968.
37. C. Tinelli and C. Ringeissen. Unions of non-disjoint theories and combinations of satisfiability procedures. *Theoretical Computer Science*, 290(1):291–353, Jan. 2003.
38. C. Tinelli and C. Zarba. Combining nonstably infinite theories. *Journal of Automated Reasoning*, 34(3), 2005.
39. T. Wies. *Symbolic Shape Analysis.* PhD thesis, University of Freiburg, 2009.
40. C. G. Zarba. A tableau calculus for combining non-disjoint theories. In *TABLEAUX '02: Proc. Int. Conf. Automated Reasoning with Analytic Tableaux and Related Methods*, pages 315–329, 2002.
41. K. Zee, V. Kuncak, and M. Rinard. Full functional verification of linked data structures. In *ACM Conf. Programming Language Design and Implementation (PLDI)*, 2008.

A Additional Proofs

A.1 Proof of Lemma 4

Lemma 4. *If α is a structure such that $\alpha(\mathcal{T}_{\mathsf{FT}})$ then α is isomorphic to some structure in $\mathcal{M}_{\mathsf{FT}}$.*

Proof. First, define FT formulas $\mathit{Succ}(x,y)$ and $\mathit{Reach}(x,y)$ as follows, where x, y are variables of sort obj:

$$\begin{aligned}\mathit{Succ}(x,y) &= (\mathsf{succ}_L(x,y) \vee \mathsf{succ}_R(x,y))\\ \mathit{Reach}(x,y) &= (\forall S\!:\!\mathsf{set}.\, x \in S \wedge (\forall u,v\!:\!\mathsf{obj}.\, u \in S \wedge \mathit{Succ}(u,v) \rightarrow v \in S) \rightarrow y \in S)\end{aligned}$$

Note that in any structure α, Succ denotes the union of the relations succ_L and succ_R and Reach denotes the reflexive transitive closure of Succ. The following FT formulas are true in all structures $\mathcal{M}_{\mathsf{FT}}$:

$$\begin{aligned}\mathit{OneRoot} &= (\forall x : \mathsf{obj}.(\forall y : \mathsf{obj}.\neg \mathit{Succ}(y,x)) \rightarrow x = \epsilon)\\ \mathit{Acyclic} &= (\forall x : \mathsf{obj}, y : \mathsf{obj}.\mathit{Succ}(x,y) \rightarrow \neg \mathit{Reach}(y,x))\\ \mathit{NoShared} &= (\forall x : \mathsf{obj}, y : \mathsf{obj}, z : \mathsf{obj}.(\mathit{Succ}(x,z) \wedge \mathit{Succ}(y,z) \rightarrow x = y) \wedge\\ &\ (\forall x : \mathsf{obj}, y : \mathsf{obj}.\neg \mathsf{succ}_L(x,y) \vee \neg \mathsf{succ}_R(x,y))\end{aligned}$$

Now, assume $\alpha(\mathcal{T}_{\mathsf{FT}})$. Let $r = \alpha(\epsilon)$ and let further $[\![\mathit{Succ}]\!]$ be the relation denoted by $\mathit{Succ}(x,y)$ in α and $[\![\mathit{Reach}]\!]$ the relation denoted by $\mathit{Reach}(x,y)$. Furthermore, let $h : \alpha(\mathsf{obj}) \times \{1,2\}^*$ be the smallest relation satisfying:

- $(r, \epsilon) \in h$
- for all $o_1, o_2 \in \alpha(\mathsf{obj})$, $(o_1, w) \in h$ and $(o_1, o_2) \in \alpha(\mathsf{succ}_L)$ implies $(o_2, w1) \in h$
- for all $o_1, o_2 \in \alpha(\mathsf{obj})$, $(o_1, w) \in h$ and $(o_1, o_2) \in \alpha(\mathsf{succ}_R)$ implies $(o_2, w2) \in h$

Let H_1 be the projection of h to its first component, i.e.

$$H_1 \stackrel{\text{def}}{=} \{\, o_1 \mid \exists o_2.(o_1, o_2) \in h \,\} \quad .$$

Likewise, let H_2 be the projection of h to its second component. Note that for all $o \in H_1$ we have $(r, o) \in [\![\mathit{Reach}]\!]$.

We claim that h is a bijection between $\alpha(\mathsf{obj})$ and H_2. First, assume that there exists $o_0 \in \alpha(\mathsf{obj}) \setminus H_1$. Then $(r, o_0) \notin [\![\mathit{Reach}]\!]$. Assume there is some o_r, such that $(r, o_r) \notin [\![\mathit{Reach}]\!]$, $(o_r, o_0) \in [\![\mathit{Reach}]\!]$ and for all $o_1 \in \alpha(\mathsf{obj})$, $(o_1, o_r) \notin [\![\mathit{Succ}]\!]$. Thus, in particular $o_r \neq r$. This contradicts the fact that $\mathit{OneRoot}$ is true in α. Thus, assume there is no such o_r. Then there exists an infinite chain $o_0, o_1, o_2, \ldots$ of elements in $\alpha(\mathsf{obj})$ such that for all $i \in \mathbb{N}$, $(o_{i+1}, o_i) \in [\![\mathit{Succ}]\!]$. It follows for all $i, j \in \mathbb{N}$ with $i < j$ that $(o_j, o_i) \in [\![\mathit{Reach}]\!]$. Since $\alpha(\mathsf{obj})$ is finite, there exist $i, j \in \mathbb{N}$ such that $i < j$ and $o_1 = o_j$. Thus, $(o_i, o_{j-1}) \in [\![\mathit{Succ}]\!]$ and $(o_{j-1}, o_i) \in [\![\mathit{Reach}]\!]$. This contradicts the fact that $\mathit{Acyclic}$ is true in α. We conclude $H_1 = \alpha(\mathsf{obj})$.

For proving that h is functional, assume that there is $o \in \alpha(\mathsf{obj})$ and distinct $w_1, w_2 \in H_2$ such that $(o, w_1) \in h$ and $(o, w_2) \in h$. Then there exist distinct o_1 and o_2 such that $(o_1, o) \in [\![\mathit{Succ}]\!]$ and $(o_2, o) \in [\![\mathit{Succ}]\!]$. This contradicts the fact that $\mathit{NoShared}$ is true in α. Thus, h is functional. By similar reasoning we can prove that h is injective.

Since h is a bijection between $\alpha(\mathsf{obj})$ and H_2, it follows that H_2 is finite. By construction, H_2 is also prefix-closed. Let α' be the structure in $\mathcal{M}_{\mathsf{FT}}$ that is determined by H_2. Again by construction, h is isomomorphic with respect to the interpretations of ϵ, succ_L, and succ_R in α and α', which proves the lemma.

A.2 Proof of Lemma 7

Lemma 7. *Let A be a finite tree automaton over alphabet Σ. Then the Parikh image of $\mathcal{L}(A)$ is a semilinear set and its finite representation is effectively computable from A.*

Proof. Let $A = (Q, Q_f, \iota)$ be a tree automaton over ranked alphabet Σ. Consider the context-free grammar $G = (N, T, R, S)$ where S is a fresh start symbol disjoint from Σ and Q, $N = Q$, $T = \Sigma$, and R is the smallest set containing the production rules:

- $S \to q$: if $q \in Q_f$,
- $q \to c$: if c is a constant symbol in Σ and $\iota(c) = q$,
- $q \to f q_1 \dots q_k$: if f is a k-ary function symbol in Σ and $\iota(f)(q_1, \dots, q_k) = q$.

Then G generates all words in Σ^* that result from a pre-order traversal of some Σ-term accepted by A, i.e., $\mathsf{Parikh}(\mathcal{L}(G)) = \mathsf{Parikh}(\mathcal{L}(A))$. Then the lemma follows from Theorem 6.

A.3 Proof of Theorem 9

Theorem 9. *The function ρ_{FT} is a BAPA reduction for $(\mathcal{F}_{\mathsf{FT}}, \mathcal{T}_{\mathsf{FT}})$.*

Proof. Let F be an FT formula. For proving the left-to-right direction assume that α is a structure such that $\alpha(\mathcal{T}_{\mathsf{FT}} \cup \{F\})$. From Lemma 4 follows that there exists some structure $\alpha' \in \mathcal{M}_{\mathsf{FT}}$ such that α' is isomorphic to α. Thus, α' is a model of F. According to Lemma 8 there exists $k \in \mathbb{N}$ such that

$$(\{\, \sigma \mapsto |\alpha'(\mathsf{vr}(\sigma))| \mid \sigma \in \Sigma_F^2 \,\} \cup \{\bot \mapsto k\}) \in \mathsf{Parikh}(\mathcal{L}(F))$$

From Equation 4 and the definition of ρ_{FT} follows $\alpha'|_{\mathsf{SV}(F)}(\rho_{\mathsf{FT}})$. Since α and α' agree up to isomorphism on the interpretations of obj and the free set variables of F, we conclude $\alpha|_{\mathsf{SV}(F)}(\rho_{\mathsf{FT}})$.

For the right-to-left direction assume that α is a structure such that $\alpha(\rho_{\mathsf{FT}})$ holds. We need to find interpretations for succ_L and succ_R that extend α to a model of $\mathcal{T}_{\mathsf{FT}} \cup \{F\}$. From the definition of ρ_{FT} and Equation 4 follows that there exists $k \in \mathbb{N}$ such that

$$(\{\, \sigma \mapsto |\alpha(\mathsf{vr}(\sigma))| \mid \sigma \in \Sigma_F^2 \,\} \cup \{\bot \mapsto k\}) \in \mathsf{Parikh}(\mathcal{L}(F)) \ .$$

From Lemma 8 follows that there exists $\alpha' \in \mathcal{M}_{\mathsf{FT}}$ such that $\alpha'(F)$ and for all $\sigma \in \Sigma_F^2$, $|\alpha(\mathsf{vr}(\sigma))| = |\alpha'(\mathsf{vr}(\sigma))|$. Since α and α' agree on the cardinalities of all Venn regions over $\mathsf{SV}(F)$, there exists a bijection $h : \alpha(\mathsf{obj}) \to \alpha'(\mathsf{obj})$ which is isomorphic with respect to the interpretation of all $x \in \mathsf{SV}(F)$ in α and α'. Choose one such isomorphism h. Let $s_L, s_R : \alpha(\mathsf{obj})^2 \to \{\mathsf{true}, \mathsf{false}\}$ be such that for all $o_1, o_2 \in \alpha(\mathsf{obj})$, $(o_1, o_2) \in s_{L,R}$ iff $(h(o_1), h(o_2)) \in \alpha'(\mathsf{succ}_{L,R})$. Then $(\alpha \cup \{\mathsf{succ}_L \mapsto s_L, \mathsf{succ}_R \mapsto s_R, \epsilon \mapsto h(\epsilon)\})$ is a model of $\mathcal{T}_{\mathsf{FT}} \cup \{F\}$.