On Decision Procedures for Ordered Collections

LARA-REPORT-2010-001

Ruzica Piskac, Philippe Suter, and Viktor Kuncak[⋆]

`firstname.lastname@epfl.ch`
EPFL School of Computer and Communication Sciences, Lausanne, Switzerland

Abstract. We describe a decision procedure for a logic that supports 1) finite collections of elements (sets or multisets), 2) the cardinality operator, 3) a total order relation on elements, and 4) min and max operators on entire collections. Among the applications of this logic are 1) reasoning about the externally observable behavior of data structures such as random access priority queues, 2) specifying witness functions for synthesis problem of set algebra, and 3) reasoning about constraints on orderings arising in termination proofs.

1 Introduction and Background

In many cases it is useful to check satisfiability of formulas involving collections such as sets and multisets. Such formulas arise in a variety of tasks, from software verification to interactive theorem proving. In addition to operators that combine collections into new ones (such as union or intersection), these formulas often involve the cardinality operator computing the number of elements in collections. Several decision procedures for sets and multisets with cardinality operator have been described recently [KNR06, KR07, PK08c, PK08a, PK08b]. Among these results is the NP-completeness of the theory of sets and multisets with the cardinality operator [PK08c]. In addition to their use in verification, these decision procedures can be used to synthesize code from specifications [SPMK09]. The applicability of these decision procedures can be increased by combining them with other decision procedures and theorem provers [SDK10,YKP10,WPK09,Kun07]. The existing decision procedures for collections with cardinality bounds do not consider operations that couple collection operations with the operations on the elements. Full second-order theory of sets of totally ordered elements without cardinality operator is known to be decidable [Rab69, She75]. Quantifier-free fragments of multisets have also been considered [Zar02]. However, these results do not support the cardinality operator. A weak form of cardinality operator can be supported using the Feferman-Vaught theorem [FV59] (see e.g. [Lug05]).

In this paper we start from the NP-completeness result for sets and multisets with cardinality operator [PK08c] and extend it to collections of totally ordered elements. Given a collection variable C, our formulas support computing not only

⋆ This research was supported in part by the Swiss National Science Foundation Grant #120433 “Precise and Scalable Analyses for Reliable Software”.

the number of elements $|C|$, but also the minimum $\min(C)$ and the maximum $\max(C)$ of all elements of C. The key challenge is to avoid NEXPTIME hardness as when adding e.g. relations to QFBAPA [YKP10]. As in the previous work [PK08a, PK08c] our constraints support all operations definable using integer linear arithmetic formulas (applied point-wise to the characteristic functions of collections). This includes, in particular, set and multiset algebra operations $\cap, \cup, \setminus, \subseteq$ as well as multiset disjoint union $\uplus$, and the $\mathsf{setof}(M)$ operator that computes the *set* of those elements that occur at least once in the multiset M. By combining set and multiset operations with min and max we can define the function $\mathsf{take}(k, C)$ that computes the least k elements from the collection C, where k is an integer variable. More generally, one can compute $\mathsf{lrange}(i, j, C)$ or $\mathsf{rrange}(C, j, i)$ as the collection of elements from position i to position j in the ordered collection, counting either from the minimum element or from the maximum element. A special case of this definable operation is extracting the i^{th} smallest or i^{th} biggest element of a sorted collection.

There is a number of areas in which we believe our constraints to be useful.

1. Our constraints can be used to model programs that manipulate data structures. Whereas previous decision procedures supported unsorted sets and multisets, our result allows us to additionally consider ordered sets and multisets. The presence of order means that we can define operations such as extracting the least element of a multiset, which gives us complete algebraic laws for the external behavior of priority queues and sorted lists or trees [CLRS01]. Our language supports not only operations of insertion and removal but also merging and comparison of collections, as well as selecting sub-collections or indexing elements.
2. We can define in our language a natural relationship $A \prec B$ on sets, denoting $\forall x \in A. \forall y \in B.\ x < y$, simply by $\max(A) < \min(B)$. This relationship is useful in specifying e.g. invariants of binary search trees.
3. Given a well-founded total order of elements, the standard multiset ordering on sets of elements is expressible in our language. Our decision procedure can therefore be used to check certain termination conditions for programs and relations.
4. Using sets defined over well-founded total orders, we are able to remove non-determinism in program synthesis. In [SPMK09] we have developed a synthesizer that works for arbitrary QFBAPA formulas. The synthesizer invokes a quantifier elimination procedure and uses the test terms from quantifier elimination as the synthesized program. These test terms involve choosing k elements from a Venn region, where the value k is computed in the synthesized program. Despite many good closure properties of QFBAPA, we found no natural way to introduce such test terms as part of QFBAPA itself. With the addition of ordering, functions such as $\mathsf{take}(k, C)$ suffice to specify all test terms. The presence of ordering in the specification language means that the user of synthesis can write specifications that have a unique solution.

Our result is formulated as a BAPA reduction and can thus be combined with other logics using the non-disjoint combination framework of [WPK09]. In the rest of this paper, we give an NP algorithm for quantifier-free formulas of multisets and sets of ordered elements with min, max, and cardinality operators on collections. We also show how to apply this result to the above-mentioned problems of interest.

2 Examples

2.1 Tree Find Example

As an example of the application of our decision procedure to program verification, consider the program of Figure 1.

```
object BSTSet {
  sealed abstract class Tree
  private case class Leaf() extends Tree
  private case class Node(left: Tree, value: Int, right: Tree) extends Tree {
    @invariant(content(this.left).max < this.value
          && this.value < content(this.right).min)
  }

  def content(t: Tree): Set[Int] = t match {
    case Leaf() ⇒ ∅
    case Node(l,e,r) ⇒ content(l) ∪ Set(e) ∪ content(r)
  }

  def find(e: Int, t: Tree): Boolean = (t match {
    case Leaf() ⇒ false
    case t @ Node(l,v,r) ⇒
      if (e < v) find(e, t.left)
      else if (e == v) true
      else find(e, t.right)
  }) ensuring (res ⇒ res ⇔ e ∈ content(t) )
}
```

Fig. 1. Looking up an element in a binary search tree.

Verifying the property specified for find requires taking into account the invariant on the sortedness of trees. The difficult case is showing that if the procedure does *not* find an element, then the element is indeed not in the tree. The proof uses the fact that, for each node with value v, all elements L in the left subtree are less than v, and all elements in the right subtree are larger than v. We can express this condition as $\max(L) < v < \min(R)$. By applying standard techniques to reduce functional programs to formulas, we obtain the following verification conditions for find:

$$\begin{aligned} &(\max(L) < e < \min(R) \land e < v) \rightarrow (e \in (L \cup \{v\} \cup R) \leftrightarrow e \in L) \\ \land\ &(\max(L) < e < \min(R) \land e = v) \rightarrow (e \in (L \cup \{v\} \cup R) \leftrightarrow e = v) \\ \land\ &(\max(L) < e < \min(R) \land e > v) \rightarrow (e \in (L \cup \{v\} \cup R) \leftrightarrow e \in R) \end{aligned}$$

These formulas belong to our decidable class, and can be handled using the decision procedure that we present in the sequel.

Here content computes a set or a multiset of elements contained in a given tree. Decidable classes for such functions over algebraic data types are presented in [SDK10]. When instead of this functional code we have an imperative structure, the corresponding content functions can often be defined using monadic second-order logic over trees and its combination with logics supporting set and multiset images [WPK09, YKP10].

2.2 Multiset Ordering for Termination

When proving that a program terminates, tools and programmers often use multiset orderings [DM79]. We define them in more details in Section 5.2. A multiset ordering is an extension of an order on the base set over which multisets are defined. If this order is total, then the multiset ordering is also total [DM79]. Termination is proved with the help of a termination function τ which assigns to every program state a multiset. Let M be the multiset value of a given state. To show that the program terminates, it suffices to show that for each possible successor state, the multiset value associated to it is smaller in the multiset ordering than M.

We illustrate the use of multiset orderings for program termination proofs. . Consider the following program which computes $\gcd(x, y)$:

```
while x ≠ y do
  if x > y then
    x := x − y;
  else
    y := y − x;
```

We define a termination function τ as $\tau(x, y) = \{x, y\}$. In this example we consider only the case when $x > y$. The case when $y > x$ is handled analogously. Let M be the value of a multiset associated with the current program state. We can define M_{n}, the multiset value in the next program state as follows:

$$M_{\mathsf{n}} = (M \setminus \{x\}) \uplus \{x - y\}$$

To guarantee termination on the $x > y$ branch, we need to show that $(M \setminus \{x\}) \uplus \{x - y\} \prec^{\mathsf{m}} M$ is a valid formula, i.e. that

$$x > y \land (M \preceq^{\mathsf{m}} (M \setminus \{x\}) \uplus \{x - y\})$$

is unsatisfiable. We consider the two cases

$$\begin{aligned} & x > y \wedge (M = (M \setminus \{x\}) \uplus \{x - y\}) \\ \text{and} \quad & x > y \wedge (M \prec^{\mathsf{m}} (M \setminus \{x\}) \uplus \{x - y\}) \end{aligned}$$

Let us start with the first formula. If a multiset did not change after adding and removing an element, it must have been the same element which was added and removed. Using this observation, the formula reduces to $x > y \wedge x = x - y$ which is clearly unsatisfiable in $\mathbb{N}_+$.

The second formula contains the $\prec^{\mathsf{m}}$ operator which the logic that we study ($\mathsf{QFMAPA}^{\prec}$) does not permit. However, in Section 5.2 we show how it can be rewritten to use only the min and max operators. Using this rewriting, we obtain the formula

$$\begin{aligned} & x > y \\ \wedge\ & X \neq \emptyset \\ \wedge\ & X \subseteq (M \setminus \{x\}) \uplus \{x - y\} \\ \wedge\ & M = (((M \setminus \{x\}) \uplus \{x - y\}) \setminus X) \uplus Y \\ \wedge\ & \max(Y) \prec \max(X) \end{aligned}$$

This is a $\mathsf{QFMAPA}^{\prec}$ formula defined over natural numbers and we can apply the algorithm defined in Section 4 to prove that it is unsatisfiable.

2.3 Using Ordered Sets in Program Synthesis

Synthesizing software from given specifications [MW80] should increase the productivity of a programmer and the chances of obtaining error-free software that entirely corresponds to its specification. The concept of ordering immediately yields a much larger number of definable functions, such as take, lrange, rrange. These functions are sufficient to provide witnesses for Skolem functions of quantified formulas of BAPA. Consider, for example, the formula

$$\forall S. \forall k. \exists A. \exists B.\ (|S| = 2k \rightarrow S = A \cup B \wedge A \cap B = \emptyset \wedge |A| = |B|)$$

This formula has a witness function $f(S, k)$ computing the sets (A, B)

$$f(S, k) = (\mathsf{take}(k/2, S), S \setminus \mathsf{take}(k/2, S))$$

where $k/2$ denotes integer division by 2, which is definable in integer linear arithmetic. Using ordering on sets, we can define such a computable witness function for every valid BAPA formula with a $\forall^\star \exists^\star$ prefix. We have used such witness functions as an output of a synthesis procedure for BAPA [SPMK09]. Without an ordering on elements it was not clear how to specify a particular subset of a set of a given size.

3 Sets of Densely Ordered Elements

In this section, we introduce QFBAPA$^{\prec}$, a logic to reason about sets of totally ordered elements, and a strict extension of QFBAPA [KR07]. We show that the problem of establishing the satisfiability of a formula in QFBAPA$^{\prec}$ is NP-complete.

$$\begin{aligned}
F &::= A \mid F \wedge F \mid F \vee F \mid \neg F \\
A &::= A_S \mid A_T \mid A_E \\
A_S &::= S = S \mid S \subset S \mid S \subseteq S \mid S \prec S \\
S &::= B \mid S \cup S \mid S \cap S \mid S \setminus S \mid \mathcal{U} \mid \emptyset \\
A_T &::= T = T \mid T < T \mid T \leq T \\
T &::= k \mid C \mid T + T \mid C * T \mid |S| \\
C &::= \ldots \mid -2 \mid -1 \mid 0 \mid 1 \mid 2 \mid \ldots \\
A_E &::= E = E \mid E \prec E \mid E \preceq E \mid E \in S \\
E &::= e \mid \text{constants of } \mathbb{E} \mid \bot \mid \top \mid \min(S) \mid \max(S)
\end{aligned}$$

Fig. 2. Syntax of QFBAPA$^{\prec}$. In the text we generally use S, T and E to denote set, integer and element terms respectively. We use A, B, C for set variables, i, j, k for integer variables and e for element variables. Note that QFBAPA$^{\prec}$ without the rules A_E, E and the partial ordering $\prec$ on sets corresponds to QFBAPA.

The complete syntax of QFBAPA$^{\prec}$ is presented in Figure 2. QFBAPA$^{\prec}$ has three sorts: sets, integers and set elements. Integers range over $\mathbb{Z}$, set elements over a finite unbounded domain $\mathbb{E}$ and sets over $2^{\mathbb{E}}$. We are interested in the satisfiability over a family of models containing a model for each finite set $\mathbb{E}$. We further assume that $\prec$ is a dense, total order on $\mathbb{E}$. (See Section 4 for the case of the non-dense order on integers.) To make the min and max functions total, we extend the domain $\mathbb{E}$ with the special values $\bot$ and $\top$, which have the following properties:

- $\min(S) = \top \leftrightarrow S = \emptyset$
- $\max(S) = \bot \leftrightarrow S = \emptyset$
- $\forall e \in \mathbb{E} \,.\, \bot \preceq e$
- $\forall e \in \mathbb{E} \,.\, e \preceq \top$
- $\forall A \in 2^{\mathbb{E}} \,.\, \min(A) \neq \bot$
- $\forall A \in 2^{\mathbb{E}} \,.\, \max(A) \neq \top$

Note that these rules are consistent with the interpretation of min and max over set terms. For instance, the identity $\min(A \cup B) = \min(\{\min(A), \min(B)\})$ should always hold. When $B = \emptyset$, we have $\min(A \cup \emptyset) = \min(\{\min(A), \min(\emptyset)\})$, so it is consistent that $\min(\emptyset)$ is greater than any possible value for $\min(A)$.

The total order $\prec$ on elements of sets induces a partial order on sets.

Definition 1. *We define* $A \prec B$ *to hold iff* $\max(A) \prec \min(B)$.

3.1 A Decision Procedure for $\mathsf{QFBAPA}^{\prec}$

We now describe all important steps of our decision procedure for $\mathsf{QFBAPA}^{\prec}$. The algorithm presented here applies to a conjunction F of literals. Formulas of arbitrary boolean structure can be handled by first rewriting them into disjunctive normal form for instance, or by using the DPLL(T) approach [GHN+04].

Rewritings. We start by applying the following rewritings:

$$\begin{aligned} E \in S &\leadsto e_{\mathsf{f}} = E \wedge |A_{\mathsf{f}}| = 1 \wedge \min(A_{\mathsf{f}}) = e_{\mathsf{f}} \wedge \max(A_{\mathsf{f}}) = e_{\mathsf{f}} \wedge A_{\mathsf{f}} \subseteq S \\ &\quad \text{where } e_{\mathsf{f}} \text{ and } A_{\mathsf{f}} \text{ are fresh} \\ S_1 \prec S_2 &\leadsto \max(S_1) \prec \min(S_2) \end{aligned}$$

Purification. We introduce a fresh set variable A_{f} for each non-variable set term T that appears as parameter of the min or max function. When doing this we add to our formula the constraint that $A_{\mathsf{f}} = T$. After this step, we can assume that min and max are only applied to set variables. We then separate our formula F into two formulas, F_{QFBAPA} and $F_{\prec}$ such that $F \equiv F_{\mathsf{QFBAPA}} \wedge F_{\prec}$, and such that $F_{\prec}$ contains only atoms over set element terms. In particular, we observe that all literals in $F_{\prec}$ are of one of the forms $E_1 = E_2$, $E_1 \prec E_2$ or $E_1 \preceq E_2$, where E_1 and E_2 can be constants, element variables or min and max applied to a set variable. We also observe that F_{QFBAPA} is a formula in QFBAPA (and therefore makes no mention of sets elements).

Guessing the empty sets. For each set variable A, we guess whether A is empty or not. If yes, we add the constraint $|A| = 0$ to F_{QFBAPA} and the constraint $\min(A) = \top \wedge \max(A) = \bot$ to $F_{\prec}$. If not, we add the constraint $|A| \geq 1$ to F_{QFBAPA} and the constraint $\min(A) \preceq \max(A)$ to $F_{\prec}$. Note that for each set variable A, the terms $\min(A)$ and $\max(A)$ now appear in $F_{\prec}$.

Guessing an ordering on elements. We consider the set $\boldsymbol{E}$ of all terms appearing in $F_{\prec}$ (including constants). We guess an arrangement into (equality) equivalence classes of all terms in $\boldsymbol{E}$ that does not violate the formula $F_{\prec}$. If no such arrangement can be found, we conclude that the original formula F is unsatisfiable. We then guess an ordering between the equivalence classes, consistent with the $\prec$ and $\preceq$ constraints of $F_{\prec}$. If we cannot find such an ordering, we also conclude that F is unsatisfiable.

Segmentation of the domain. We number the equivalence classes from the previous point in increasing order from 1 to n. For each of them, we create a fresh set variable C_i. We constrain each of these sets to be a singleton by adding to F_{QFBAPA} the constraints $|C_i| = 1$ for $1 \leq i \leq n$. We create $n-1$ more fresh set variables C_{n+1} to C_{2n-1}. We force all the fresh set variables to represent disjoint sets by adding to F_{QFBAPA} the constraints $|C_i \cap C_j| = 0$ for $1 \leq i < j \leq n$. In the following we interpret the fresh sets as points (C_1 to C_n) and intervals (C_{n+1} to C_{2n-1}) on the total order on $\mathbb{E}$ (see Figure 3). Using this interpretation,

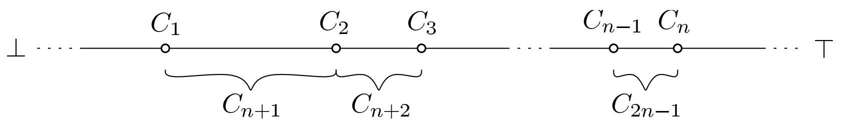

Fig. 3. Segmentation of the domain $\mathbb{E}$.

we express each non-fresh set variable A using the point and interval sets. For each non-empty set A we introduce $2n-1$ fresh variables A_i representing the intersection of A with each point or interval set. It is then clear that A can be expressed as the (disjoint) union of all variables A_i:

$$\bigwedge_{1 \leq i < 2n} A_i = A \cap C_i \land A = \bigcup_{1 \leq i < 2n} A_i$$

If we guessed that $\min(A)$ and $\max(A)$ were in the p^{th} and q^{th} equivalence classes respectively (with the possibility that $p = q$), we add the following to F_{QFBAPA}:

$$\bigwedge_{1 \leq i < p} A_i = \emptyset \land A_{n+i} = \emptyset \land |A_p| = 1 \land |A_q| = 1 \land \bigwedge_{q < i \leq n} A_i = \emptyset \land A_{n+i} = \emptyset$$

Solution of the QFBAPA constraints. As a final step, we use the decision procedure for QFBAPA [KR07] on F_{QFBAPA}. Our original formula is satisfiable if and only if F_{QFBAPA} is satisfiable.

Theorem 1. *Our decision procedure for* $\mathsf{QFBAPA}^{\prec}$ *is sound and complete.*

Proof. **Soundness.** We first show that each of our reasoning steps results in a logically sound conclusion. The rewritings are correct by definition. The purification process introduces fresh variables that are constrained to be equal to the term they represent, so any model for the formula without the fresh variables can trivially be extended to the original variables. It is sound to non-deterministically guess which sets are empty, and the constraints that we add are immediate consequences of the guesses. Failing to guess equivalence classes implies that at least two elements are constrained to be equal and strictly ordered at the same time, which means that the ordering constraints are unsatisfiable. Similarly, a failure to establish a total ordering on the equivalence classes implies that the strict ordering constraints form at least one cycle and are thus unsatisfiable. It remains to show that when we introduce the fresh variables C_i and A_i and the constraints on them, we do not exclude any solution for the existing variables. To show this, consider a valuation for all non-fresh variables, and consider the ordering of equivalence classes generated by the values of min and max applied to the values of non-fresh set variables. For each variable A, define $A_i = A \cap \{x \mid n_p < x < n_q\}$ for $i > n$ and $A_i = A \cap S_i$ where S_i is the appropriate singleton containing n_q (here n_p and n_q correspond to max and min of sets). Note that A_i are all finite sets because each A is finite. Then define C_i as the union of A_i over all variables

i. Note that $A_i = A \cap C_i$ and $C_i \subseteq \{x \mid n_p < x < n_q\}$ for $i > n$ and $C_i \subseteq S_i$, otherwise. Thus all C_i sets are also disjoint.

Completeness. To show completeness we need to show that we can build a model for the original formula from a model for our formula F_{QFBAPA}. The model for F_{QFBAPA} will contain the cardinality of each set, and all we need to do is populate them with elements from $\mathbb{E}$. We start by populating the singleton sets C_i, for $1 \leq i \leq n$: if the i^{th} equivalence class contains a constant from $\mathbb{E}$, we set it to be the element of C_i. We do this for all classes with a known constant. It is then always possible, because the order is dense, to pick a value that does not contradict the ordering for each of the other classes. We then proceed to populate the sets C_{n+i} for $1 \leq i < n$. We know the cardinality of C_{n+i} from the model of F_{QFBAPA}, and we want to pick only values that are between the value in C_i and the value in C_{i+1}. Again, because the order is dense, this is always possible. This outlines the model generation procedure for the sets C_1 to C_{2n-1}. The construction for all the other sets follows then from their cardinalities and from the constraints we added when we segmented the domain.

Theorem 2. *The satisfiability problem for* $\mathsf{QFBAPA}^{\prec}$ *is NP-complete.*

Proof. Since $\mathsf{QFBAPA}^{\prec}$ is a strict extension of QFBAPA and the satisfiability problem for QFBAPA is NP-complete, we can conclude that it is NP-hard for $\mathsf{QFBAPA}^{\prec}$. It remains to show that the complexity does not increase with the extension. To this end, since the algorithm is essentially a reduction to QFBAPA, it is sufficient to see that we only add polynomially many constraints to the QFBAPA formula and that we only guess polynomially many variables.

4 Sets of Integer Elements

In this section we consider sets of integers instead of a set of elements from a dense order. The high-level idea of the decision procedure is similar to the one in Section 3. However, note that for a non-empty finite set of integers A the following always holds:

$$|A| \leq \max(A) - \min(A) + 1 \tag{1}$$

We thus obtain an additional upper bound on cardinalities of sets, which was not present in the case of dense orders. It is therefore natural for collections of integers to permit integer variables to participate both in cardinality constraints and in $\min, \max$ constraints. We obtain the syntax in Figure 4. Note that, by conjoining a formula with literals $0 \leq \min(A)$ for every set variable A, we can model sets of non-negative integers with the natural well-founded total order.

Note that using these constructs we can write the following property of a set C:

$$\min(C) = p \wedge \max(C) = q \wedge |C| = q - p + 1,$$

which ensures that C is equal to the interval $\{x \mid p \leq x \leq q\}$.

The steps of the decision procedure are similar to those for dense orders. The steps **Rewriting** and **Purification** are analogous. Because the sort of collection elements is identical to the sort of integers used for cardinalities, the result of these two phases is a conjunction of a QFBAPA formula and constraints of the form $\min(A) = i$ or $\max(A) = j$. The steps **Guessing the empty sets** and **Guessing an ordering of elements** are identical. The ordering is expressed using the ordering relation on integers. The step **Segmentation of the domain** differs only in adding the constraints

$$|C_{n+i}| \leq \min(C_{i+1}) - \min(C_i) - 1 \tag{2}$$

in addition to all other constraints added. Finally, **Invoking** QFBAPA is similar but now all generated constraints are given to QFBAPA, including the ordering constraints on variables. Note that the presence of $\bot$ and $\top$ does not significantly affect the decision procedure for Presburger arithmetic or QFBAPA [KNR06, FV59].

The questions of soundness, completeness, and NP membership are natural generalizations of the corresponding properties for dense orders. Even though the constraints here are more tightly coupled, the description of model constructions are simpler because models are sets of integers.

For soundness, the key step is defining an extension of a model to the model of set variables C_i and A_i. In this case, for $i \leq n$ the single element of C_i is given by the corresponding integer variable. The value of C_{n+i} is simply the set of integers $\{x \mid \min(C_i) < x < \min(C_{i+1})\}$, which is a finite set. The value of A_i is simply $A \cap C_i$.

For completeness, the condition (2) ensures that A_i has no more elements than C_i, which makes the construction of the model straightforward.

$$\begin{aligned}
F &::= A \mid F \wedge F \mid F \vee F \mid \neg F \\
A &::= A_S \mid A_T \\
A_S &::= S = S \mid S \subset S \mid S \subseteq S \mid S \prec S \\
S &::= B \mid S \cup S \mid S \cap S \mid S \setminus S \mid \mathcal{U} \mid \emptyset \\
A_T &::= T = T \mid T < T \mid T \leq T \mid T \in S \\
T &::= k \mid C \mid T + T \mid C * T \mid |S| \mid \min(S) \mid \max(S) \\
C &::= \bot \mid \ldots \mid -2 \mid -1 \mid 0 \mid 1 \mid 2 \mid \ldots \mid \top
\end{aligned}$$

Fig. 4. QFBAPA$^{\mathbb{Z}}$: algebra of sets of integers with cardinality, max, and min.

5 Multisets Defined over an Ordered Set

Quantifier-free multiset formulas can be seen as a natural extension of QFBAPA formulas. While in sets the same element cannot be repeated, in multisets duplicate elements can occur. Given a finite domain $\mathbb{E}$, we define a multiset M as a function counting how many times an element from $\mathbb{E}$ appears in M. Formally, a multiset is a function from $\mathbb{E}$ to $\mathbb{N}$. Checking satisfiability of a multiset formula means to define a base set $\mathbb{E}$ and a function from $\mathbb{E}$ to $\mathbb{N}$ for each multiset variable such that the formula evaluates to `true` under the standard semantics. Decidability and complexity of the logic that includes reasoning about multisets and integers was studied in [Zar02]. Adding cardinality constraints to multisets and combining that with Presburger arithmetic was further investigated in [PK08a, PK08c]. All these approaches are compatible with the above definition, as there are no additional constraint on the base set $\mathbb{E}$.

In this paper we adopt the logic presented in [PK08a], but add the additional constraint that the base set must be totally ordered. This allows us to express various properties, including the minimal or the maximal element of a multiset. For example, we can express the constraint $\min(M_1) \in M_2$. We call this logic $\mathsf{QFMAPA}^{\prec}$, and its syntax is shown in Figure 5.

As the domain of elements, we start by considering dense orders, for simplicity. The results also applies to multisets over integers, in which case they extend $\mathsf{QFBAPA}^{\mathbb{Z}}$ rather than $\mathsf{QFBAPA}^{\prec}$. Note that, even when the domain of elements is $\mathbb{Z}$, we do not allow the use of integer variables denoting cardinalities or min, max within the inner integer arithmetic formulas (used to build Σ or $\forall e.F$ formulas).

5.1 Decidability and Complexity of $\mathsf{QFMAPA}^{\prec}$

Decidability and complexity of $\mathsf{QFMAPA}^{\prec}$ strongly relies on the decidability and complexity of $\mathsf{QFBAPA}^{\prec}$, as we show in Theorem 3. In the proof, the operator $\mathsf{setof}()$ plays a major role. It takes a multiset and creates a set from it. As an illustration, $\mathsf{setof}(\{a, a, a, b, b\}) = \{a, b\}$. Formally, it is defined as $\mathsf{setof}(M) = \{e \mid M(e) > 0\}$.

Theorem 3. *Checking the satisfiability of a formula F in $\mathsf{QFMAPA}^{\prec}$, is NP-complete.*

Proof. We prove this using a similar technique as in the case of $\mathsf{QFBAPA}^{\prec}$. However, we cannot simply apply the proof used in establishing the NP-completeness of $\mathsf{QFBAPA}^{\prec}$, as min and max are here defined on multisets. Defining a multiset as a union of disjoint regions does not apply, since this would not capture the repeated elements. However, we notice that the properties $\min(M) = \min(\mathsf{setof}(M))$ and $\max(M) = \max(\mathsf{setof}(M))$ always hold. Using this observation, for each multiset expression M that occurs as a parameter of the min or max function, we introduce a fresh multiset variable S_{f}. We flatten the formula using the following two rules:

top-level formulas:

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

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

outer linear arithmetic formulas:

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

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

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

inner linear arithmetic formulas:

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

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

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

multiset expressions:

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

expressions about ordered elements:

$E ::= x \mid \min(M) \mid \max(M)$

terminals:

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

k - integer variable; C - integer constant

Fig. 5. Quantifier-free multiset constraints with the cardinality operator over a totally ordered set as a domain ($\mathsf{QFMAPA}^{\prec}$).

– $C[\min(M)] \rightsquigarrow C[\min(S_{\mathsf{f}})] \wedge S_{\mathsf{f}} = \mathsf{setof}(M)$
– $C[\max(M)] \rightsquigarrow C[\max(S_{\mathsf{f}})] \wedge S_{\mathsf{f}} = \mathsf{setof}(M)$

(Here $C[\ \ ]$ denotes a rewriting context.)

Although the newly introduced variables are multisets variables, they are actually sets, since they never contain repeating elements. This means that the min and max operators are now applied only to sets and we can apply the same procedure as before.

We separate the initial formula into two parts: one about pure QFMAPA constraints and the other about ordering constraints. Note that $S_{\mathsf{f}} = \mathsf{setof}(M)$ is also a QFMAPA constraint. Now we repeat the same procedure as in the case of $\mathsf{QFBAPA}^{\prec}$. We denote by F_{BC} the newly derived QFBAPA constraints. Every QFBAPA formula can be easily transformed into a QFMAPA formula. We do this by adding the constraint stating that S is a set for each set variable S: $\forall e.(S(e) = 0 \vee S(e) = 1)$. The translated constraints from F_{BC} are conjoined with the original QFMAPA constraints.

We thus obtain a new formula F_M which is entirely in QFMAPA. The increase in size is at most polynomial. Checking satisfiability of QFMAPA formulas is an NP-complete problem as it was shown in [PK08c]. Therefore, checking satisfiability of F_M is in NP. This concludes our proof that checking satisfiability of $\mathsf{QFMAPA}^{\prec}$ formulas is an NP-complete problem.

The soundness and completeness of the procedure follow from these properties for $\mathsf{QFBAPA}^{\prec}$. For soundness, we note that all reasoning steps we performed in the $\mathsf{QFBAPA}^{\prec}$ reduction are valid within $\mathsf{QFMAPA}^{\prec}$ as well, For completeness, the proof of completeness for $\mathsf{QFBAPA}^{\prec}$ shows that the solution for sets can be extended to an ordering on the elements that satisfies the total order. The solution of the formula on multisets is also a solution of the formula on sets. Moreover, thanks to the use of the $\mathsf{setof}()$ transformation, all ordering constraints on the multiset formula apply to sets. This in particular implies that the elements can be ordered using the same method as for $\mathsf{QFBAPA}^{\prec}$ to make the $\mathsf{QFMAPA}^{\prec}$ formula true.

5.2 Multiset Ordering

Let $\mathcal{M}(\mathbb{E})$ be the class of all finite multisets defined over a totally ordered set $(\mathbb{E}, \preceq)$. Our goal is to define an order on $\mathcal{M}(\mathbb{E})$. The following is the standard definition given in [DM79]:

Definition 2. *Given two multisets N and M defined over a total order $(\mathbb{E}, \preceq)$, $N \prec^{\mathsf{m}} M$ if and only if there exist two multisets X and Y such that $X \neq \emptyset$ and $X \subseteq M$ and $N = (M \setminus X) \uplus Y$ and $\forall y \in Y. \exists x \in X. y \prec x$.*

One can visualize the definition of $N \prec^{\mathsf{m}} M$ as follows: let Z be a subset of the intersection of the multisets M and N. The multiset X is defined as a multiset of all the elements of M that are not contained in the intersection: $X = M \setminus Z$. Similarly, the multiset Y contains all elements of N that are not in the intersection. The multisets X and Y are disjoint.

Lemma 1. *Let X and Y be disjoint multisets and $X \neq \emptyset$. The formula $\forall y \in Y. \exists x \in X. y \prec x$ is equivalent to $\max(Y) \prec \max(X)$.*

Proof. Every element of Y is bounded by $\max(Y)$: $\forall y \in Y. y \preceq \max(Y)$. Since $\max(Y) \in Y$, there exists an element $E \in X$ such that $\max(Y) \prec E$. Taking this element to be $\max(X)$ we are sure that it is a greater element of every element of Y. The other direction is similar.

Lemma 2. *It is possible to extend a $\mathsf{QFMAPA}^{\prec}$ solver in such a way that it can also reason about multiset orderings by rewriting every multiset ordering subformula $N \prec^{\mathsf{m}} M$ into an equivalent $\mathsf{QFMAPA}^{\prec}$ formula*

$$X \neq \emptyset \wedge X \subseteq M \wedge N = (M \setminus X) \uplus Y \wedge \max(Y) \prec \max(X)$$

Apart from their use for proving termination of programs, multiset orderings are also used in a model construction based on resolution proofs [BG01]. We therefore expect that automation of formal correctness proofs of such model construction would benefit from a decision procedure for multiset orderings.

6 Conclusions

We had previously identified a number of uses for constraints on sets and cardinality bounds and established their optimal complexity. In this paper we generalized these results to the case of a total ordering relation on collection elements. We have shown that this step beyond uninterpreted elements provides important expressive power while preserving the key complexity properties of the decision problems. We expect the resulting decision procedures to be even more useful in data structure verification, synthesis, and termination proofs. Moreover, these new results naturally fit into the framework of combining theories by sharing sets.

References

BG01. Leo Bachmair and Harald Ganzinger. Resolution theorem proving. In *Handbook of Automated Reasoning*, pages 19–99. MIT Press, 2001.

CLRS01. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Cliff Stein. *Introduction to Algorithms (Second Edition)*. MIT Press and McGraw-Hill, 2001.

DM79. Nachum Dershowitz and Zohar Manna. Proving termination with multiset orderings. *Commun. ACM*, 22(8):465–476, 1979.

FV59. S. Feferman and R. L. Vaught. The first order properties of products of algebraic systems. *Fundamenta Mathematicae*, 47:57–103, 1959.

GHN$^+$04. Harald Ganzinger, George Hagen, Robert Nieuwenhuis, Albert Oliveras, and Cesare Tinelli. DPLL(T): Fast decision procedures. In R. Alur and D. Peled, editors, *16th Conf. Computer Aided Verification*, volume 3114 of *LNCS*, pages 175–188. Springer, 2004.

KNR06. Viktor Kuncak, Hai Huu Nguyen, and Martin Rinard. Deciding Boolean Algebra with Presburger Arithmetic. *J. of Automated Reasoning*, 2006. http://dx.doi.org/10.1007/s10817-006-9042-1.

KR07. Viktor Kuncak and Martin Rinard. Towards efficient satisfiability checking for Boolean Algebra with Presburger Arithmetic. In *CADE-21*, 2007.

Kun07. Viktor Kuncak. *Modular Data Structure Verification*. PhD thesis, EECS Department, Massachusetts Institute of Technology, February 2007.

Lug05. Denis Lugiez. Multitree automata that count. *Theor. Comput. Sci.*, 333(1-2):225–263, 2005.

MW80. Zohar Manna and Richard Waldinger. A deductive approach to program synthesis. *ACM Trans. Program. Lang. Syst.*, 2(1):90–121, 1980.

PK08a. Ruzica Piskac and Viktor Kuncak. Decision procedures for multisets with cardinality constraints. In *VMCAI*, number 4905 in LNCS, 2008.

PK08b. Ruzica Piskac and Viktor Kuncak. Fractional collections with cardinality bounds. In *Computer Science Logic (CSL)*, 2008.

PK08c. Ruzica Piskac and Viktor Kuncak. Linear arithmetic with stars. In *CAV*, 2008.

Rab69. Michael O. Rabin. Decidability of second-order theories and automata on infinite trees. *Trans. Amer. Math. Soc.*, 141:1–35, 1969.

SDK10. Philippe Suter, Mirco Dotta, and Viktor Kuncak. Decision procedures for algebraic data types with abstractions. In *37th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages (POPL)*, 2010.

She75. Saharon Shelah. The monadic theory of order. *Tha Annals of Mathematics of Mathematics*, 102(3):379–419, Nov 1975.

SPMK09. Philippe Suter, Ruzica Piskac, Mikael Mayer, and Viktor Kuncak. On complete functional synthesis. Technical Report LARA-REPORT-2009-006, EPFL, 2009.

WPK09. Thomas Wies, Ruzica Piskac, and Viktor Kuncak. Combining theories with shared set operations. In *FroCoS: Frontiers in Combining Systems*, 2009.

YKP10. Kuat Yessenov, Viktor Kuncak, and Ruzica Piskac. Collections, cardinalities, and relations. In *Verification, Model Checking, and Abstract Interpretation (VMCAI)*, 2010.

Zar02. Calogero G. Zarba. Combining multisets with integers. In *CADE-18*, 2002.