

# The Calculus of Computation

Zohar Manna  
Spring 2009

## Chapter 11: Arrays

## Arrays I: Quantifier-free Fragment of $T_A$

Signature:

$$\Sigma_A : \{ \cdot[\cdot], \cdot\langle \cdot \rangle, = \}$$

where

- ▶  $a[i]$  binary function – read array  $a$  at index  $i$  (“read( $a, i$ )”)
- ▶  $a\langle i \rangle v$  ternary function – write value  $v$  to index  $i$  of array  $a$  (“write( $a, i, v$ )”)

Axioms

1. the axioms of (reflexivity), (symmetry), and (transitivity) of  $T_E$
2.  $\forall a, i, j. i = j \rightarrow a[i] = a[j]$  (array congruence)
3.  $\forall a, v, i, j. i = j \rightarrow a\langle i \rangle v [j] = v$  (read-over-write 1)
4.  $\forall a, v, i, j. i \neq j \rightarrow a\langle i \rangle v [j] = a[j]$  (read-over-write 2)

## Infinite Domain

We add an axiom schema to  $T_A$  that forbids interpretations with finite arrays.

For each positive natural number  $n$ , the following is an axiom:

$$\forall x_1, \dots, x_n. \exists y. \bigwedge_{i=1}^n y \neq x_i$$

## Equality in $T_A$

Note: = is only defined for array elements:

$$a[i] = e \rightarrow a\langle i \rangle e = a$$

not  $T_A$ -valid, but

$$a[i] = e \rightarrow \forall j. a\langle i \rangle e [j] = a[j],$$

is  $T_A$ -valid.

Also

$$a = b \rightarrow a[i] = b[i]$$

is not  $T_A$ -valid: We only axiomatized a restricted congruence.

$T_A$  is undecidable

Quantifier-free fragment of  $T_A$  is decidable

## Example: Quantifier-free fragment (QFF) of $T_A$

Is

$$a[i] = e_1 \wedge e_1 \neq e_2 \rightarrow a(i \triangleleft e_2)[i] \neq a[i]$$

$T_A$ -valid?

Alternatively, is

$$a[i] = e_1 \wedge e_1 \neq e_2 \wedge a(i \triangleleft e_2)[i] = a[i]$$

$T_A$ -unsatisfiable?

## Decision Procedure for $T_A$

Given quantifier-free conjunctive  $\Sigma_A$ -formula  $F$ .

To decide the  $T_A$ -satisfiability of  $F$ :

### Step 1

If  $F$  does not contain any write terms  $a(i \triangleleft v)$ , then

1. associate array variables  $a$  with fresh function symbol  $f_a$ , and replace read terms  $a[i]$  with  $f_a(i)$ ;
2. decide the  $T_E$ -satisfiability of the resulting formula.

## Decision Procedure for $T_A$

### Step 2

Select some read-over-write term  $a(i \triangleleft v)[j]$  (note that  $a$  may itself be a write term) and split on two cases:

1. According to (read-over-write 1), replace

$$F[a(i \triangleleft v)[j]] \quad \text{with} \quad F_1: F[v] \wedge i = j,$$

and recurse on  $F_1$ . If  $F_1$  is found to be  $T_A$ -satisfiable, return satisfiable.

2. According to (read-over-write 2), replace

$$F[a(i \triangleleft v)[j]] \quad \text{with} \quad F_2: F[a[j]] \wedge i \neq j,$$

and recurse on  $F_2$ . If  $F_2$  is found to be  $T_A$ -satisfiable, return satisfiable.

If both  $F_1$  and  $F_2$  are found to be  $T_A$ -unsatisfiable, return unsatisfiable.

## Example

Consider  $\Sigma_A$ -formula

$$F: i_1 = j \wedge i_1 \neq i_2 \wedge a[j] = v_1 \wedge a(i_1 \triangleleft v_1)(i_2 \triangleleft v_2)[j] \neq a[j].$$

$F$  contains a write term,

$$a(i_1 \triangleleft v_1)(i_2 \triangleleft v_2)[j] \neq a[j].$$

According to (read-over-write 1), assume  $i_2 = j$  and recurse on

$$F_1: i_2 = j \wedge i_1 = j \wedge i_1 \neq i_2 \wedge a[j] = v_1 \wedge v_2 \neq a[j].$$

$F_1$  does not contain any write terms, so rewrite it to

$$F'_1: \underbrace{i_2 = j \wedge i_1 = j \wedge i_1 \neq i_2}_{\text{contradiction}} \wedge f_a(j) = v_1 \wedge v_2 \neq f_a(j).$$

Contradiction —  $F'_1$  is  $T_E$ -unsatisfiable.

Returning, we try the second case:  
 according to (read-over-write 2), assume  $i_2 \neq j$  and recurse on

$$F_2: i_2 \neq j \wedge i_1 = j \wedge i_1 \neq i_2 \wedge a[j] = v_1 \wedge a(i_1 \triangleleft v_1)[j] \neq a[j].$$

$F_2$  contains a write term. According to (read-over-write 1),  
assume  $i_1 = j$  and recurse on

$$F_3: i_1 = j \wedge i_2 \neq j \wedge i_1 = j \wedge i_1 \neq i_2 \wedge \underbrace{a[j] = v_1 \wedge v_1 \neq a[j]}.$$

Contradiction. Thus, according to (read-over-write 2),  
assume  $i_1 \neq j$  and recurse on

$$F_4: i_1 \neq j \wedge i_2 \neq j \wedge i_1 = j \wedge i_1 \neq i_2 \wedge \underbrace{a[j] = v_1 \wedge a[j] \neq a[j]}.$$

Contradiction: all branches have been tried, and thus  $F$  is  
 $T_A$ -unsatisfiable.

Question: Suppose instead that  $F$  does not contain the literal  
 $i_1 \neq i_2$ . Is this new formula  $T_A$ -satisfiable?

## Decision Procedure for Arrays

The quantifier free fragment of  $T_A$  is *decidable*.  
 However *too weak* to express important properties:

- ▶ Containment:  $\forall i. \ell \leq i \leq u \implies a[i] \neq e$
- ▶ Sortedness:  $\forall i, j. \ell \leq i \leq j \leq u \implies a[i] \leq a[j]$
- ▶ Partitioning:  $\forall i, j. \ell_1 \leq i \leq u_1 \wedge \ell_2 \leq j \leq u_2 \implies a[i] \leq a[j]$

The general theory of arrays  $T_A$  with quantifier is *not decidable*.

Is there a decidable fragment of  $T_A$  that contains the above  
 formulae?

## Example

We want to prove validity for a formula, such as:

$$(\forall i. a[i] \neq e) \wedge e \neq f \rightarrow (\forall i. a(j \triangleleft f)[i] \neq e).$$

Equivalently show unsatisfiability of

$$(\forall i. a[i] \neq e) \wedge e \neq f \wedge (\exists i. a(j \triangleleft f)[i] = e).$$

or the equisatisfiable formula

$$(\forall i. a[i] \neq e) \wedge e \neq f \wedge a(j \triangleleft f)[i] = e.$$

We need to handle a universal quantifier.

## Arrays II: Array Property Fragment of $T_A$

Decidable fragment of  $T_A$  that includes  $\forall$  quantifiers

### Array property

$\Sigma_A$ -formula of form

$$\forall \vec{i}. \alpha[\vec{i}] \rightarrow \beta[\vec{i}],$$

where  $\vec{i}$  is a list of variables.

- ▶ *index guard*  $\alpha[\vec{i}]$ :

$$\begin{aligned} \text{iguard} &\rightarrow \text{iguard} \wedge \text{iguard} \mid \text{iguard} \vee \text{iguard} \mid \text{atom} \\ \text{atom} &\rightarrow \text{var} = \text{var} \mid \text{evar} \neq \text{var} \mid \text{var} \neq \text{evar} \mid \top \\ \text{var} &\rightarrow \text{evar} \mid \text{uvar} \end{aligned}$$

where *uvar* is any universally quantified index variable,  
 and *evar* is any unquantified free variable.

## Arrays II: Array Property Fragment of $T_A$ (cont)

### ▶ value constraint $\beta[\bar{i}]$ :

Any qff, but a universally quantified index can occur only in a read  $a[\bar{i}]$ , where  $a$  is an array term.

### Array property Fragment:

Boolean combinations of quantifier-free  $\Sigma_A$ -formulae and array properties

Note:  $a[b[k]]$  for unquantified variable  $k$  is okay, but  $a[b[i]]$  for universally quantified variable  $i$  is forbidden. Cannot replace it by

$$\forall i, j. \dots b[i] = j \wedge a[j] \dots$$

In  $\beta$ , the universally quantified variable  $j$  may occur in  $a[j]$  but not in  $b[i] = j$ .

## Example: Array Property Fragment

Is this formula in the array property fragment?

$$F : \forall i. i \neq a[k] \rightarrow a[i] = a[k]$$

The antecedent is not a legal index guard since  $a[k]$  is not a variable (neither a *uvar* nor an *evar*); however, by simple manipulation

$$F' : v = a[k] \wedge \forall i. i \neq v \rightarrow a[i] = a[k]$$

Here,  $i \neq v$  is a legal index guard, and  $a[i] = a[k]$  is a legal value constraint.  $F$  and  $F'$  are equisatisfiable.

However, no manipulation works for:

$$G : \forall i. i \neq a[i] \rightarrow a[i] = a[k].$$

Thus,  $G$  is not in the array property fragment.

## Array property fragment and extensionality

Array property fragment allows expressing equality between arrays (*extensionality*): two arrays are equal precisely when their corresponding elements are equal.

For given formula

$$F : \dots \wedge a = b \wedge \dots$$

with array terms  $a$  and  $b$ , rewrite  $F$  as

$$F' : \dots \wedge (\forall i. \top \rightarrow a[i] = b[i]) \wedge \dots$$

$F$  and  $F'$  are equisatisfiable.

## Decision Procedure for Array Property Fragment

Basic Idea: Replace universal quantification  $\forall i. F[i]$  by finite conjunction  $F[t_1] \wedge \dots \wedge F[t_n]$ .

We call  $t_1, \dots, t_n$  the index terms and they depend on the formula.

## Example

Consider

$$F : a(i \triangleleft v) = a \wedge a[i] \neq v ,$$

which expands to

$$F' : \forall j. a(i \triangleleft v)[j] = a[j] \wedge a[i] \neq v .$$

Intuitively, to determine that  $F'$  is  $T_A$ -unsatisfiable requires merely examining index  $i$ :

$$F'' : \left( \bigwedge_{j \in \{i\}} a(i \triangleleft v)[j] = a[j] \right) \wedge a[i] \neq v ,$$

or simply

$$a(i \triangleleft v)[i] = a[i] \wedge a[i] \neq v .$$

Simplifying,

$$v = a[i] \wedge a[i] \neq v ,$$

it is clear that this formula, and thus  $F$ , is  $T_A$ -unsatisfiable.



Page 17 of 55

Steps 4-6 accomplish the reduction of universal quantification to finite conjunction.

Main idea: select a set of symbolic index terms on which to instantiate all universal quantifiers. The set is sufficient for correctness.

### Step 4

From the output  $F_3$  of Step 3, construct the **index set**  $\mathcal{I}$ :

$$\begin{aligned} \mathcal{I} = & \cup \{t : \exists [i] \in F_3 \text{ such that } t \text{ is not a universally quantified variable}\} \\ & \cup \{t : t \text{ occurs as an evar in the parsing of index guards}\} \\ & \cup \{\lambda\} \end{aligned}$$

This index set is the finite set of "symbolic indices" that need to be examined. It includes

- ▶ all terms  $t$  that occur in some read  $a[t]$  anywhere in  $F_3$  (unless it is a universally quantified variable); e.g.,  $k$  in  $a[k]$ .
- ▶ all terms  $t$  (unquantified variable) that are compared to a universally quantified variable in some index guard  $F[i]$ ; e.g.,  $k$  in  $i = k$ .
- ▶  $\lambda$  is a fresh constant that represents all other index positions that are not explicitly in  $\mathcal{I}$ .



Page 19 of 55

## The Algorithm

Given array property formula  $F$ , decide its  $T_A$ -satisfiability by the following steps:

### Step 1

Put  $F$  in NNF.

### Step 2

Apply the following rule exhaustively to remove writes:

$$\frac{G[a(i \triangleleft v)]}{G[a'] \wedge a'[i] = v \wedge (\forall j. j \neq i \rightarrow a[j] = a'[j])} \text{ for fresh } a' \text{ (write)}$$

After an application of the rule, the resulting formula contains at least one fewer write terms than the given formula.

### Step 3

Apply the following rule exhaustively to remove existential quantification:

$$\frac{F[\exists \bar{i}. G[\bar{i}]]}{F[G[\bar{j}]]} \text{ for fresh } \bar{j} \text{ (exists)}$$

Existential quantification can arise during Step 1 if the given formula has a negated array property.



Page 18 of 55

### Step 5 (Key step)

Apply the following rule exhaustively to remove universal quantification:

$$H \left[ \frac{H[\forall \bar{i}. \alpha[\bar{i}] \rightarrow \beta[\bar{i}]]}{\bigwedge_{\bar{i} \in \mathcal{I}^n} (\alpha[\bar{i}] \rightarrow \beta[\bar{i}])} \right] \text{ (forall)}$$

where  $n$  is the size of the list of quantified variables  $\bar{i}$ .

### Step 6

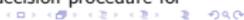
From the output  $F_5$  of Step 5, construct

$$F_6 : F_5 \wedge \bigwedge_{t \in \mathcal{I} \setminus \{\lambda\}} \lambda \neq t .$$

The new conjuncts assert that the variable  $\lambda$  introduced in Step 4 is indeed unique.

### Step 7

Decide the  $T_A$ -satisfiability of  $F_6$  using the decision procedure for the quantifier-free fragment.



Page 20 of 55

$$F : a = b(i \triangleleft v) \wedge a[i] \neq v$$

In array property fragment:

$$(\forall j. a[j] = b(i \triangleleft v)[j]) \wedge a[i] \neq v$$

Eliminate write:

$$\begin{aligned} & (\forall j. a[j] = b'[j]) \\ \wedge & a[i] \neq v \\ \wedge & b'[i] = v \\ \wedge & (\forall j. j \neq i \rightarrow b'[j] = b[j]) \end{aligned}$$

Index set:

$$\mathcal{I} : \{i, \lambda\}$$

QF formula:

$$\begin{aligned} & a[i] = b'[i] \wedge a[\lambda] = b'[\lambda] \\ \wedge & a[i] \neq v \wedge b'[i] = v \\ \wedge & (i \neq i \rightarrow b'[i] = b[i]) \wedge (\lambda \neq i \rightarrow b'[\lambda] = b[\lambda]) \\ \wedge & \lambda \neq i \end{aligned}$$

Simplified:

$$\begin{aligned} & \boxed{a[i] = b'[i]} \wedge a[\lambda] = b'[\lambda] \\ \wedge & \boxed{a[i] \neq v} \wedge \boxed{b'[i] = v} \\ \wedge & b'[\lambda] = b[\lambda] \\ \wedge & \lambda \neq i \end{aligned}$$

Contradiction. So  $F$  is unsatisfiable.

## Example

Is this  $T_A^-$ -formula (arrays with extensionality) valid?

$$F : (\forall i. i \neq k \rightarrow a[i] = b[i]) \wedge b[k] = v \rightarrow a(k \triangleleft v) = b$$

Check unsatisfiability of  $T_A^-$ -formula:

$$\neg((\forall i. i \neq k \rightarrow a[i] = b[i]) \wedge b[k] = v \rightarrow (\forall i. a(k \triangleleft v)[i] = b[i]))$$

**Step 1:** NNF

$$F_1 : (\forall i. i \neq k \rightarrow a[i] = b[i]) \wedge b[k] = v \wedge (\exists i. a(k \triangleleft v)[i] \neq b[i])$$

**Step 2:** Remove array writes

$$\begin{aligned} F_2 : & (\forall i. i \neq k \rightarrow a[i] = b[i]) \wedge b[k] = v \wedge (\exists i. a'[i] \neq b[i]) \\ & \wedge a'[k] = v \wedge (\forall i. i \neq k \rightarrow a'[i] = a[i]) \end{aligned}$$

## Example (cont)

**Step 3:** Remove existential quantifier

$$\begin{aligned} F_3 : & (\forall i. i \neq k \rightarrow a[i] = b[i]) \wedge b[k] = v \wedge a'[i] \neq b[i] \\ & \wedge a'[k] = v \wedge (\forall i. i \neq k \rightarrow a'[i] = a[i]) \end{aligned}$$

## Example (cont)

**Step 4:** Compute index set  $\mathcal{I} = \{\lambda, k, j\}$

**Step 5+6:** Replace universal quantifier:

$$\begin{aligned}
F_6 : & (\lambda \neq k \rightarrow a[\lambda] = b[\lambda]) \\
& \wedge (k \neq k \rightarrow a[k] = b[k]) \\
& \wedge (j \neq k \rightarrow a[j] = b[j]) \\
& \wedge b[k] = v \wedge a'[j] \neq b[j] \wedge a'[k] = v \\
& \wedge (\lambda \neq k \rightarrow a'[\lambda] = a[\lambda]) \\
& \wedge (k \neq k \rightarrow a'[k] = a[k]) \\
& \wedge (j \neq k \rightarrow a'[j] = a[j]) \\
& \wedge \lambda \neq k \wedge \lambda \neq j
\end{aligned}$$

Case distinction on  $j = k$  (4th line) and  $j \neq k$  (3rd line, 4th line, and 7th line) proves unsatisfiability of  $F_6$ .

Therefore  $F$  is valid.

## The importance of $\lambda$

Is this formula satisfiable?

$$F : (\forall i. i \neq j \rightarrow a[i] = b[i]) \wedge (\forall i. i \neq k \rightarrow a[i] \neq b[i])$$

The algorithm produces (for  $\{\lambda, j, k\}$ ):

$$\begin{aligned}
F_6 : & \lambda \neq j \rightarrow a[\lambda] = b[\lambda] \\
& \wedge j \neq j \rightarrow a[j] = b[j] \\
& \wedge k \neq j \rightarrow a[k] = b[k] \\
& \wedge \lambda \neq k \rightarrow a[\lambda] \neq b[\lambda] \\
& \wedge j \neq k \rightarrow a[j] \neq b[j] \\
& \wedge k \neq k \rightarrow a[k] \neq b[k] \\
& \wedge \lambda \neq j \wedge \lambda \neq k
\end{aligned}$$

The 1st, 4th and last lines give a contradiction!  $F$  is unsatisfiable.

## The importance of $\lambda$ (cont)

Without  $\lambda$  we had the formula:

$$\begin{aligned}
F'_6 : & j \neq j \rightarrow a[j] = b[j] \\
& \wedge k \neq j \rightarrow a[k] = b[k] \\
& \wedge j \neq k \rightarrow a[j] \neq b[j] \\
& \wedge k \neq k \rightarrow a[k] \neq b[k]
\end{aligned}$$

which simplifies to:

$$j \neq k \rightarrow a[k] = b[k] \wedge a[j] \neq b[j].$$

This formula  $F$  is satisfiable!

## Example

Consider array property formula

$$\begin{aligned}
F : & a[\ell < v][k] = b[k] \wedge b[k] \neq v \wedge a[k] = v \\
& \wedge \underbrace{(\forall i. i \neq \ell \rightarrow a[i] = b[i])}_{\text{array property}}
\end{aligned}$$

By Step 2, rewrite  $F$  as

$$\begin{aligned}
F_2 : & a'[k] = b[k] \wedge b[k] \neq v \wedge a[k] = v \wedge (\forall i. i \neq \ell \rightarrow a[i] = b[i]) \\
& \wedge a'[\ell] = v \wedge (\forall j. j \neq \ell \rightarrow a[j] = a'[j])
\end{aligned}$$

$F_2$  does not contain any existential quantifiers. Its index set is

$$\mathcal{I} = \{\lambda, k, \ell\}.$$

## Example (cont)

Thus, by Step 5, replace universal quantification (and step 6):

$$a'[k] = b[k] \wedge b[k] \neq v \wedge a[k] = v \wedge \bigwedge_{i \in \mathcal{I}} (i \neq \ell \rightarrow a[i] = b[i])$$

$$F_6: \wedge a'[\ell] = v \wedge \bigwedge_{j \in \mathcal{I}} (j \neq \ell \rightarrow a[j] = a'[j]) \\ \wedge \lambda \neq k \wedge \lambda \neq \ell$$

Expanding produces

$$a'[k] = b[k] \wedge b[k] \neq v \wedge a[k] = v$$

$$\wedge (\lambda \neq \ell \rightarrow a[\lambda] = b[\lambda])$$

$$\wedge (k \neq \ell \rightarrow a[k] = b[k])$$

$$F'_6: \wedge (\ell \neq \ell \rightarrow a[\ell] = b[\ell])$$

$$\wedge a'[\ell] = v$$

$$\wedge (\lambda \neq \ell \rightarrow a[\lambda] = a'[\lambda])$$

$$\wedge (k \neq \ell \rightarrow a[k] = a'[k]) \wedge (\ell \neq \ell \rightarrow a[\ell] = a'[\ell])$$

$$\wedge \lambda \neq k \wedge \lambda \neq \ell$$

Page 29 of 55

## Correctness of Decision Procedure

### Theorem

Consider a  $\Sigma_A$ -formula  $F$  from the array property fragment of  $T_A$ . The output  $F_6$  of Step 6 of the algorithm is  $T_A$ -equisatisfiable to  $F$ .

This also works when extending the Logic with an arbitrary theory  $T$  with signature  $\Sigma$  for the elements:

### Theorem

Consider a  $\Sigma_A \cup \Sigma$ -formula  $F$  from the array property fragment of  $T_A \cup T$ . The output  $F_6$  of Step 6 of the algorithm is  $T_A \cup T$ -equisatisfiable to  $F$ .

Page 31 of 55

## Example (cont)

Simplifying,

$$a'[k] = b[k] \wedge b[k] \neq v \wedge a[k] = v$$

$$\wedge a[\lambda] = b[\lambda] \wedge (k \neq \ell \rightarrow a[k] = b[k])$$

$$F''_6: \wedge a'[\ell] = v$$

$$\wedge a[\lambda] = a'[\lambda] \wedge (k \neq \ell \rightarrow a[k] = a'[k])$$

$$\wedge \lambda \neq k \wedge \lambda \neq \ell$$

There are two cases to consider.

- ▶ If  $k = \ell$ , then  $a'[\ell] = v$  (3rd line) and  $a'[k] = b[k]$  (1st line) imply  $b[k] = v$ , yet  $b[k] \neq v$ .
- ▶ If  $k \neq \ell$ , then  $a[k] = v$  (1st line) and  $a[k] = b[k]$  (2nd line) imply  $b[k] = v$ , but again  $b[k] \neq v$ .

Hence,  $F''_6$  is  $T_A$ -unsatisfiable, indicating that  $F$  is  $T_A$ -unsatisfiable.

Page 30 of 55

## Nelson-Open Combination Method

**Given:**

- ▶ Theories  $T_1, \dots, T_k$  that share only = (and are stably infinite)
- ▶ Decision procedures  $P_1, \dots, P_k$
- ▶ Quantifier-free ( $\Sigma_1 \cup \dots \cup \Sigma_k$ )-formula  $F$

**Decide** if  $F$  is  $(T_1 \cup \dots \cup T_k)$ -satisfiable using  $P_1, \dots, P_k$ .

Think about arrays in context of Nelson-Openpen.

Page 32 of 55

- ▶ 1962: John McCarthy formalizes arrays as first-order theory  $T_A$ .
- ▶ 1969: James King describes and implements DP for QFF of  $T_A$ .
- ▶ 1979: Nelson & Oppen describe combination method for QF theories sharing  $=$ .
- ▶ 1980s: Suzuki, Jefferson; Jaffar; Mateti describe DPs for QFF of theories of arrays with predicates for sorted, partitioned, etc.
- ▶ 1997: Levitt describes DP for QFF of extensional theory of arrays in thesis.
- ▶ 2001: Stump, Barrett, Dill, Levitt describe DP for QFF of extensional theory of arrays.
- ▶ 2006: Bradley, Manna, Sipma describe DP for array property fragment of  $T_A$ ,  $T_A^Z$ .

Signature:

$$\Sigma_A^Z : \Sigma_A \cup \Sigma_Z = \{a[i], a(i < v), =, 0, 1, +, \leq\}$$

$\leq$  enables reasoning about subarrays and properties such as whether the subarray is sorted or partitioned.

Axioms of  $T_A^Z$ : both axioms of  $T_A$  and  $T_Z$

## Array Property Fragment of $T_A^Z$

Array property:  $\Sigma_A^Z$ -formula of the form

$$\forall \vec{i}. \alpha[\vec{i}] \rightarrow \beta[\vec{i}],$$

where  $\vec{i}$  is a list of integer variables.

- ▶  $\alpha[\vec{i}]$  *index guard*:

iguard  $\rightarrow$  iguard  $\wedge$  iguard | iguard  $\vee$  iguard | atom

atom  $\rightarrow$  expr  $\leq$  expr | expr = expr

expr  $\rightarrow$  uvar | pexpr

pexpr  $\rightarrow$  pexpr'

pexpr'  $\rightarrow$   $\mathbb{Z}$  |  $\mathbb{Z} \cdot$  evar | pexpr' + pexpr'

where *uvar* is any universally quantified integer variable, and *evar* is any unquantified free integer variable.

Note: Why both pexpr and pexpr'? E.g., in  $i \leq 3k + j$ , the expression  $3k + j$  is pexpr, but not  $k$  or  $j$ .

## Array Property Fragment of $T_A^Z$ (cont)

- ▶ *value constraint*  $\beta[\vec{i}]$ :

Any qff, but a universally quantified index can occur only in a read  $a[i]$ , where *a* is an array term.

Array property Fragment (APF):

Boolean combinations of quantifier-free  $\Sigma_A^Z$ -formulae and array properties

Note:  $a[b[k]]$  for unquantified variable *k* is okay, but  $a[b[i]]$  for universally quantified variable *i* is forbidden.

## Application: array property fragments

- ▶ Array equality  $a = b$  in  $T_A$ :

$$\forall i. a[i] = b[i]$$

- ▶ Bounded array equality  $\text{beq}(a, b, \ell, u)$  in  $T_A^Z$ :

$$\forall i. \ell \leq i \leq u \rightarrow a[i] = b[i]$$

- ▶ Universal properties  $F[x]$  in  $T_A$ :

$$\forall i. F[a[i]]$$

- ▶ Bounded universal properties  $F[x]$  in  $T_A^Z$ :

$$\forall i. \ell \leq i \leq u \rightarrow F[a[i]]$$

- ▶ Bounded sorted arrays  $\text{sorted}(a, \ell, u)$  in  $T_A^Z$  or  $T_A^Z \cup T_Q$ :

$$\forall i, j. \ell \leq i \leq j \leq u \rightarrow a[i] \leq a[j]$$

- ▶ Partitioned arrays  $\text{partitioned}(a, \ell_1, u_1, \ell_2, u_2)$  in  $T_A^Z$  or  $T_A^Z \cup T_Q$ :

$$\forall i, j. \ell_1 \leq i \leq u_1 < \ell_2 \leq j \leq u_2 \rightarrow a[i] \leq a[j]$$

Page 37 of 55

## The Decision Procedure (Step 1–2)

The idea again is to reduce universal quantification to finite conjunction.

Given  $F$  from the array property fragment of  $T_A^Z$ , decide its  $T_A^Z$ -satisfiability as follows:

### Step 1

Put  $F$  in NNF.

### Step 2

Apply the following rule exhaustively to remove writes:

$$\frac{G[a[i < e]]}{G[a'] \wedge a'[i] = e \wedge (\forall j. j \neq i \rightarrow a[j] = a'[j])} \text{ for fresh } a' \text{ (write)}$$

To meet the syntactic requirements on an index guard, rewrite the third conjunct as

$$\forall j. j \leq i - 1 \vee i + 1 \leq j \rightarrow a[j] = a'[j].$$

Page 38 of 55

## The Decision Procedure (Step 3–4)

### Step 3

Apply the following rule exhaustively to remove existential quantification:

$$\frac{F[\exists \vec{i}. G[\vec{i}]]}{F[G[\vec{j}]]} \text{ for fresh } \vec{j} \text{ (exists)}$$

Existential quantification can arise during Step 1 if the given formula has a negated array property.

### Step 4

From the output of Step 3,  $F_3$ , construct the index set  $\mathcal{I}$ :

$$\mathcal{I} = \{t : \cdot[t] \in F_3 \text{ such that } t \text{ is not a universally quantified variable}\} \\ \cup \{t : t \text{ occurs as a } \underline{\text{pexpr}} \text{ in the parsing of index guards}\}$$

If  $\mathcal{I} = \emptyset$ , then let  $\mathcal{I} = \{0\}$ . The index set contains all relevant

symbolic indices that occur in  $F_3$ . Note: no  $\lambda!$

Page 39 of 55

## The Decision Procedure (Step 5–6)

### Step 5

Apply the following rule exhaustively to remove universal quantification:

$$\frac{H[\forall \vec{i}. F[\vec{i}] \rightarrow G[\vec{i}]]}{H \left[ \bigwedge_{\vec{i} \in \mathcal{I}^n} (F[\vec{i}] \rightarrow G[\vec{i}]) \right]} \text{ (forall)}$$

$n$  is the size of the block of universal quantifiers over  $\vec{i}$ .

### Step 6

$F_5$  is quantifier-free in the combination theory  $T_A \cup T_Z$ . Decide the  $(T_A \cup T_Z)$ -satisfiability of the resulting formula.

Page 40 of 55

## Example

$\Sigma_A^Z$ -formula:

$$F: (\forall i. \ell \leq i \leq u \rightarrow a[i] = b[i]) \\ \wedge \neg(\forall i. \ell \leq i \leq u+1 \rightarrow a[u+1] < b[u+1])[i] = b[i])$$

In NNF, we have

$$F_1: (\forall i. \ell \leq i \leq u \rightarrow a[i] = b[i]) \\ \wedge (\exists i. \ell \leq i \leq u+1 \wedge a[u+1] < b[u+1])[i] \neq b[i])$$

Step 2 produces

$$F_2: (\forall i. \ell \leq i \leq u \rightarrow a[i] = b[i]) \\ \wedge (\exists i. \ell \leq i \leq u+1 \wedge a'[i] \neq b[i]) \\ \wedge a'[u+1] = b[u+1] \\ \wedge (\forall j. j \leq u \vee u+2 \leq j \rightarrow a[j] = a'[j])$$

Page 41 of 55

Step 5 rewrites universal quantification to finite conjunction over this set:

$$F_5: \bigwedge_{i \in \mathcal{I}} (\ell \leq i \leq u \rightarrow a[i] = b[i]) \\ \wedge \ell \leq k \leq u+1 \wedge a'[k] \neq b[k] \\ \wedge a'[u+1] = b[u+1] \\ \wedge \bigwedge_{j \in \mathcal{I}} (j \leq u \vee u+2 \leq j \rightarrow a[j] = a'[j])$$

Expanding the conjunctions according to the index set  $\mathcal{I}$  and simplifying according to trivially true or false antecedents (e.g.,  $\ell \leq u+1 \leq u$  simplifies to  $\perp$ , while  $u \leq u \vee u+2 \leq u$  simplifies to  $\top$ ) produces:

Page 43 of 55

Step 3 removes the existential quantifier by introducing a fresh constant  $k$ :

$$F_3: (\forall i. \ell \leq i \leq u \rightarrow a[i] = b[i]) \\ \wedge \ell \leq k \leq u+1 \wedge a'[k] \neq b[k] \\ \wedge a'[u+1] = b[u+1] \\ \wedge (\forall j. j \leq u \vee u+2 \leq j \rightarrow a[j] = a'[j])$$

The index set is

$$\mathcal{I} = \{k, u+1\} \cup \{\ell, u, u+2\},$$

which includes the read indices  $k$  and  $u+1$  and the terms  $\ell$ ,  $u$ , and  $u+2$  that occur as pexprs in the index guards.

Page 42 of 55

$$F'_5: (\ell \leq k \leq u \rightarrow a[k] = b[k]) \quad (1) \\ \wedge (\ell \leq u \rightarrow a[\ell] = b[\ell] \wedge a[u] = b[u]) \quad (2) \\ \wedge \ell \leq k \leq u+1 \quad (3) \\ \wedge a'[k] \neq b[k] \quad (4) \\ \wedge a'[u+1] = b[u+1] \quad (5) \\ \wedge (k \leq u \vee u+2 \leq k \rightarrow a[k] = a'[k]) \quad (6) \\ \wedge (\ell \leq u \vee u+2 \leq \ell \rightarrow a[\ell] = a'[\ell]) \quad (7) \\ \wedge a[u] = a'[u] \wedge a[u+2] = a'[u+2] \quad (8)$$

$(T_A \cup T_Z)$ -unsatisfiability of this quantifier-free  $(\Sigma_A \cup \Sigma_Z)$ -formula can be decided using the techniques of Combination of Theories.

Informally,  $\ell \leq k \leq u+1$  (3)

- ▶ If  $k \in [\ell, u]$  then  $a[k] = b[k]$  (1). Since  $k \leq u$  then  $a[k] = a'[k]$  (6), contradicting  $a'[k] \neq b[k]$  (4).
- ▶ if  $k = u+1$ ,  $a'[k] \neq b[k] = b[u+1] = a'[u+1] = a'[k]$  by (4) and (5), a contradiction.

Hence,  $F$  is  $T_A^Z$ -unsatisfiable.

Page 44 of 55

## Theorem

Consider a  $\Sigma_A^Z \cup \Sigma$ -formula  $F$  from the array property fragment of  $T_A^Z \cup T$ .

The output  $F_5$  of Step 5 of the algorithm is  $T_A^Z \cup T$ -equisatisfiable to  $F$ .

$$\text{sorted}(a, \ell, u) : \forall i, j. \ell \leq i \leq j \leq u \rightarrow a[i] \leq a[j]$$

Is

$$\text{sorted}(a(0 \triangleleft 0) \langle 5 \triangleleft 1 \rangle, 0, 5) \wedge \text{sorted}(a(0 \triangleleft 10) \langle 5 \triangleleft 11 \rangle, 0, 5)$$

$T_A^Z$ -satisfiable?

0	w	x	y	z	1
---	---	---	---	---	---

10	w	x	y	z	11
----	---	---	---	---	----

## Example

$$\text{sorted}(a(0 \triangleleft 0) \langle 5 \triangleleft 1 \rangle, 0, 5) \wedge \text{sorted}(a(0 \triangleleft 10) \langle 5 \triangleleft 11 \rangle, 0, 5)$$

Index set:  $\{-1, 0, 1, 4, 5, 6\}$

- ▶  $\{0, 5\}$  from  $0 \leq i \leq j \leq 5$
- ▶  $\{-1, 1\}$  from  $\cdot \langle 0 \triangleleft \cdot \rangle$
- ▶  $\{4, 6\}$  from  $\cdot \langle 5 \triangleleft \cdot \rangle$

Contradiction:

$$a[0] \leq a[1] \leq a[5] \quad \wedge \quad a[0] \leq a[1] \leq a[5]$$

$$0 \leq a[1] \leq 1 \quad \wedge \quad 10 \leq a[1] \leq 11$$

Need 1 or 4 in index set.

## Undecidable Extensions

- ▶ Extra quantifier alternation (e.g.,  $\forall i \exists j. \dots$ )
- ▶ Nested reads:  $a[a[i]]$
- ▶ No separation:  $\forall i. F[a[i], i]$  (e.g.,  $a[i] = i$ )
- ▶ Arithmetic:  $a[i + 1]$  when  $i$  is universal
- ▶ Strict comparison:  $i < j$  when  $i, j$  are universal
- ▶ Permutation predicate (even weak permutation)

## Theory of Sets

Consider a theory  $T_{\text{set}}$  of sets with signature

$$\Sigma_{\text{set}} : \{\in, \subseteq, =, \subset, \cap, \cup, \setminus\},$$

where symbols are intended as follows:

- ▶  $e \in s$ :  $e$  is a member of  $s$ ;
- ▶  $s \subseteq t$ :  $s$  is a subset of  $t$ ;
- ▶  $s = t$ :  $s$  and  $t$  are equal;
- ▶  $s \subset t$ :  $s$  is a *strict* subset of  $t$ ;
- ▶  $s \cap t$  is the intersection of  $s$  and  $t$ ;
- ▶  $s \cup t$  is the union of  $s$  and  $t$ ;
- ▶  $s \setminus t$ , the set difference of  $s$  and  $t$ , is the set that includes all elements of  $s$  that are not members of  $t$ .

## Theory of Sets (cont)

Let us encode an arbitrary  $\Sigma_{\text{set}}$ -formula as a  $\Sigma_E$ -formula (or a  $\Sigma_A$ -formula). To do so, simply consider the atoms:

- ▶  $e \in s$ : let  $s(\cdot)$  be a unary predicate; then replace

$$e \in s \text{ by } s(e)$$

- ▶  $s \subseteq t$ :  $\forall e. e \in s \rightarrow e \in t$ , or in other words,  $\forall e. s(e) \rightarrow t(e)$ ;
- ▶  $s = t$ :  $\forall e. s(e) \leftrightarrow t(e)$ ;
- ▶  $s \subset t$ :  $s \subseteq t \wedge s \neq t$ ;
- ▶  $u = s \cap t$ :  $\forall e. u(e) \leftrightarrow s(e) \wedge t(e)$ ;
- ▶  $u = s \cup t$ :  $\forall e. u(e) \leftrightarrow s(e) \vee t(e)$ ;
- ▶  $u = s \setminus t$ :  $\forall e. u(e) \leftrightarrow s(e) \wedge \neg t(e)$ .

## Theory of Sets (cont)

Atoms with complex terms can be written more simply via “flattening” (as in the Nelson-Oppen procedure); for example, write

$$s \cap (t \cap u) \text{ as } s \cap w \wedge w = t \cap u.$$

Then the encodability of an arbitrary  $\Sigma_{\text{set}}$ -formula into a  $\Sigma_E$ -formula (or a  $\Sigma_A$ -formula) follows by structural induction.

### Claim

Satisfiability of the quantifier-free fragment of  $T_{\text{set}}$  is decidable:

- ▶ simply apply the decision procedure for  $T_E$  (or  $T_A$ ) to the new formula.

## Theory of Multisets

Consider a theory  $T_{\text{mset}}$  of multisets with signature

$$\Sigma_{\text{mset}} : \{C, \leq, =, <, \uplus, \cap, -\}.$$

Multisets can have multiple occurrences of elements.

For example:  $\{1, 3, 5\}$  is a set and  $\{1, 1, 3, 5, 5, 5\}$  is a multiset.

The symbols are intended as follows:

- ▶  $C(s, e)$ : the number of occurrences (the “count”) of  $e$  in  $s$ ;
- ▶  $s \leq t$ : the count of each element of  $s$  is bounded by its count in  $t$ ;
- ▶  $s = t$ : element counts are the same in  $s$  and  $t$ ;
- ▶  $s < t$ : the count of each element of  $s$  is bounded by its count in  $t$ , and some element has a lower count;
- ▶  $s \uplus t$  is the multiset union, whose counts are the element-wise sums of counts in  $s$  and  $t$ ;

## Theory of Multisets (cont)

- ▶  $s \cap t$  is the multiset intersection, whose counts are the element-wise minima of counts in  $s$  and  $t$ ;
- ▶  $s - t$  is the multiset difference, whose counts are the element-wise maxima of 0 and the difference of counts in  $s$  and  $t$ .

Let us encode an arbitrary  $\Sigma_{\text{mset}}$ -formula as a  $(\Sigma_E \cup \Sigma_Z)$ -formula (or a  $(\Sigma_A \cup \Sigma_Z)$ -formula). A multiset is modeled by an uninterpreted function whose range is the nonnegative integers.

## Theory of Multisets (cont)

Now consider the atoms:

- ▶  $C(s, e)$ : let  $s$  be a unary function whose range is  $\mathbb{N}$ ; then replace

$C(s, e)$  by  $s(e)$

and conjoin  $\forall e. s(e) \geq 0$  to the formula;

- ▶  $s \leq t$ :  $\forall e. s(e) \leq t(e)$ ;
- ▶  $s = t$ :  $\forall e. s(e) = t(e)$ ;
- ▶  $s < t$ :  $s \leq t \wedge s \neq t$ ;
- ▶  $u = s \uplus t$ :  $\forall e. u(e) = s(e) + t(e)$ ;
- ▶  $u = s \cap t$ :

$$\forall e. (s(e) < t(e) \wedge u(e) = s(e)) \vee (s(e) \geq t(e) \wedge u(e) = t(e)) ;$$

## Theory of Multisets (cont)

- ▶  $u = s - t$ :

$$\forall e. (s(e) < t(e) \wedge u(e) = 0) \vee (s(e) \geq t(e) \wedge u(e) = s(e) - t(e)) .$$

As before, encodability follows by structural induction.