Arithmetic Integration of Decision Procedures (Special University Ph.D. Oral Examination)

Ting Zhang

Advisor: Prof. Zohar Manna

Stanford University

December 7, 2005

Introduction

What is a Decision Procedure?

Introduction
- Decision Procedure
- Why Do We Need New Decision Procedures?
- Combination?
- Combination of Theories
- Limitation
- What are Common Combinations?
- Our Approach
- Our Contribution (1)
- Our Contribution (2)
- Publication (1)
- Publication (2)
- Outline

PART I. Term Algebras with Integers

PART II. Queues with Integers

PART III. Knuth-Bendix Order

PART IV. Conclusion and Future Work

Thank You!

An algorithm that checks whether a formula is valid in a given decidable theory.

Always terminates with either a positive or a negative answer.

Relieve users from tedious interaction with theorem prover.

Why Do We Need New Decision Procedures?

Decision procedures exist for specific theories

- Arithmetic: integers, reals, . . . ,
- Data types: lists, queues, arrays, sets, multisets, . . . ,
- Algebraic structures: linear dense orders . . . ,

But

- programming languages involve multiple theories.
- verification conditions do not belong to a single theory.

☞ We need to reason about *mixed* constraints from multiple theories.

What is Combining Decision Procedure?

Σ_1-theory T_1

$\boxed{P_1}$ for T_1-satisfiability

Σ_2-theory T_2

$\boxed{P_2}$ for T_2-satisfiability

?

$\boxed{P}$ for $(T_1 \cup T_2)$-satisfiability

Combination of Theories

General Framework:

Nelson-Oppen Combination Method [NO79]

Recent Advances:

- Non-disjoint Signature.

  Tinelli and Ringeissen [TR03]
- Model-theoretic.

  Ghilardi [Ghi05]
- Proof-theoretic.

  Zarba [Zar02]

  Armando, Ranise and Rusinowitch [ARR01]

Limitation

- All existing combination techniques impose severe restrictions on the theories to be combined.
- None of the techniques is applicable to multi-sorted theories with functions connecting the different sorts.

☞ Logic theories are *fragile*.

- Nelson-Oppen combination should be viewed as exceptional.
- Why should modular combinations always exist?
- Concentrate on concrete problems instead of looking for grand scheme.

What are Common Combinations?

- Integration of recursive data structures with integer arithmetic
  - Term algebras (tree-like objects) + integers
  - Queues (linear objects)+ integers
- Why? To automatically decide the validity of verification conditions arising in the analysis of any property involving data structures and size.

  Examples:
  - buffer overflows
  - array out of bounds
  - memory overflow
  - . . .

Our Approach

Exploit the algebraic properties of constituent theories.

- For quantifier-free combinations:

  Extract exact integer constraints **induced** by constraints of data types.

- For quantified combinations:

  Reduce quantifiers on data types to quantifiers on integers.

☞ Reduce theories of data domain to the theory of integer domain.

Our Contribution (1)

Decision procedures for the combination of data structures with integer constraints.

- Essential for practical program verification.
- Can express memory safety properties.

Main approach:

Exploit the algebraic properties of constituent theories.

Main challenge:

Integer constraints must be precise (equisatisfiable with the data constraints).

Our Contribution (2)

Proof of decidability of the first-order theory of Knuth-Bendix orders

- Long-standing open problem (RTA problem #99).
- Important result for term rewriting.
- Many partial solutions:
  - Quantifier-free theory [KV00, KV01]
  - Unary quantified theory [KV02]
- Same approach applicable to very different problem.

Publication (1)

Decision procedures for term algebras with integer constraints:

T. Zhang, H.B. Sipma, and Z. Manna, *Decision Procedures for Recursive Data Structures with Integer Constraints.* In Proc. 2nd International Joint Conference on Automated Reasoning (IJCAR) July 2004, LNCS, vol. 3097, pp. 152–167 (Best Paper Award, accepted for publication in *Information and Computation*).

T. Zhang, H.B. Sipma and Z. Manna, *Term Algebras with Length Function and Bounded Quantifier Alternation.* In Proc. of the 17th International Conference on Theorem Proving in Higher Order Logics (TPHOLs 2004), LNCS, vol. 3223, pp. 321-336.

(journal version in preparation)

Publication (2)

Decision procedures for queues with integer constraints:

T. Zhang, H.B. Sipma and Z. Manna,
Decision Procedures for Queues with Integer Constraints.
In Proc. Foundations of Software Technology and Theoretical Computer Science (FSTTCS), Dec 2005, LNCS, vol. 3821, pp. 225–237.

Decision procedures for Knuth-Bendix orders:

T. Zhang, H.B. Sipma, Z. Manna,
The Decidability of the First-order Theory of Knuth-Bendix Order.
In Proc. Conference on Automated Deduction (CADE) July 2005, LNCS, vol. 3632, pp. 131–148.

(journal version in preparation)

Outline

I. Term Algebras with Integers

II. Queues with Integers

III. Knuth-Bendix Orders

IV. Conclusions and Future Work

PART I. Term Algebras with Integers

Previous Work on Term Algebras

- Quantifier-free theory.

  Nelson and Oppen [NO80]; Oppen [Opp80];

  Downey, Sethi and Tarjan [DST80]

- Quantified theory.

  Malcev [Mal71]

- Extensions.
  - Infinite and rational trees: Maher [Mah88];
  - Tree with membership: Comon and Delor [CD94];
  - Feature trees: Backofen [Bac95];
  - Term power: Kuncak and Rinard [KR03b].

Term Algebras

A term algebra $\mathrm{TA} : \langle \mathbb{T}; \mathcal{C}, \mathcal{A}, \mathcal{S}, \mathcal{T} \rangle$ consists of

- $\mathbb{T}$: The term domain.
- $\mathcal{C}$: A finite set of constructors: α, β, γ, ...
- $\mathcal{A}$: A finite set of constants: $a, b, c, \ldots$. Require $\mathcal{A} \subseteq \mathcal{C}$.
- $\mathcal{S}$: A finite set of selectors. $\alpha = (\mathrm{s}_1^\alpha, \ldots, \mathrm{s}_k^\alpha)$.
- $\mathcal{T}$: A finite set of testers. Is_α for $\alpha \in \mathcal{C}$.

- $\mathbb{T}$ is generated exclusively using $\mathcal{C}$.
- Each element of TA is uniquely generated.

Example: LISP lists

- Signature:

$$\langle \mathrm{list}; \{\mathrm{cons}, \mathrm{nil}\}; \{\mathrm{nil}\}; \{\mathrm{car}, \mathrm{cdr}\}; \{\mathrm{Is}_{\mathrm{nil}}, \mathrm{Is}_{\mathrm{cons}}\}\rangle$$

- Axioms:

$$\begin{aligned}
\mathrm{Is}_{\mathrm{nil}}(x) &\leftrightarrow \neg \mathrm{Is}_{\mathrm{cons}}(x), \\
x &= \mathrm{car}(\mathrm{cons}(x, y)), \\
y &= \mathrm{cdr}(\mathrm{cons}(x, y)), \\
\mathrm{Is}_{\mathrm{nil}}(x) &\leftrightarrow \{\mathrm{car}, \mathrm{cdr}\}^{+}(x) = x, \\
\mathrm{Is}_{\mathrm{cons}}(x) &\leftrightarrow \mathrm{cons}(\mathrm{car}(x), \mathrm{cdr}(x)) = x.
\end{aligned}$$

Term Algebras with Integers

Presburger arithmetic (PA): $\mathcal{L}_{\mathbb{Z}}$, PA.

Two-sorted language $\Sigma = \Sigma_{\mathbb{T}} \cup \Sigma_{\mathbb{Z}} \cup \{|\cdot|\}$:

1. $\Sigma_{\mathbb{T}}$: signature of term algebras.
2. $\Sigma_{\mathbb{Z}}$: signature of Presburger arithmetic.
3. $|\cdot| : \mathbb{T} \to \mathbb{N}$, the length function such that

$$|t| = \begin{cases} 1 & \textit{if} \quad t \textit{ is a constant,} \\ \sum_{i=1}^{k} |t_i| & \textit{if} \quad t \equiv \alpha(t_1, \dots, t_k). \end{cases}$$

The Problem

The presence of $\Phi_{\mathbb{Z}}$ restricts solutions to $\Phi_{\mathbb{T}}$.

$$x \neq \mathrm{cons}(\mathrm{cons}(\mathrm{nil}, \mathrm{nil}), \mathrm{nil}) \ \wedge \ x \neq \mathrm{cons}(\mathrm{nil}, \mathrm{cons}(\mathrm{nil}, \mathrm{nil}))$$

is unsatisfiable with $|x| = 5$.

> There are “hidden” constraints on data structure length that may contradict the integer constraints.

Length Constraint Completion (LCC)

A formula $\Phi_\Delta(\bar{\mathbf{X}})$ is an *LCC* for $\Phi_{\mathbb{T}}(\bar{\mathbf{X}}) \wedge \Phi_{\mathbb{Z}}(\bar{\mathbf{X}})$, if the following formulae are valid:

$$\Phi_{\mathbb{T}}(\bar{\mathbf{X}}) \wedge \Phi_{\mathbb{Z}}(\bar{\mathbf{X}}) \rightarrow (\exists \bar{\mathbf{z}} : \mathbb{Z})\Big(\Phi_\Delta(\bar{\mathbf{z}}) \wedge |\bar{\mathbf{X}}| = \bar{\mathbf{z}}\Big),$$

$$\Phi_\Delta(\bar{\mathbf{z}}) \rightarrow (\exists \bar{\mathbf{X}} : \mathbb{T})\Big(\Phi_{\mathbb{T}}(\bar{\mathbf{X}}) \wedge \Phi_{\mathbb{Z}}(\bar{\mathbf{X}}) \wedge |\bar{\mathbf{X}}| = \bar{\mathbf{z}}\Big).$$

Informally,

$$\Phi_{\mathbb{T}}(\bar{\mathbf{X}}) \wedge \Phi_{\mathbb{Z}}(\bar{\mathbf{X}}) \;\;\text{``}\leftrightarrow\text{''}\;\; \Phi_\Delta(\bar{\mathbf{X}})$$

☞ $\Phi_\Delta(\bar{\mathbf{X}})$ fully characterizes $\Phi_{\mathbb{T}}(\bar{\mathbf{X}}) \wedge \Phi_{\mathbb{Z}}(\bar{\mathbf{X}})$.

☞ We reduce the combined constraint to the integer domain!

LCC (2)

Let $\Phi_{\Delta+}$ be the formula that (when in place of Φ_Δ) satisfies

$$\Phi_{\mathbb{T}}(\bar{\mathbf{X}}) \wedge \Phi_{\mathbb{Z}}(\bar{\mathbf{X}}) \rightarrow (\exists \bar{\mathbf{z}} : \mathbb{Z})\Big(\Phi_\Delta(\bar{\mathbf{z}}) \wedge |\bar{\mathbf{X}}| = \bar{\mathbf{z}}\Big).$$

$\Phi_{\Delta+}$ is *sound*:

$|\cdot|$ maps a satisfying $\sigma_{\mathbb{T}}$ in $\mathbb{T}$ to a satisfying $\sigma_{\mathbb{Z}}$ in PA.

Let $\Phi_{\Delta-}$ be the formula that (when in place of Φ_Δ) satisfies

$$\Phi_\Delta(\bar{\mathbf{z}}) \rightarrow (\exists \bar{\mathbf{X}} : \mathbb{T})\Big(\Phi_{\mathbb{T}}(\bar{\mathbf{X}}) \wedge \Phi_{\mathbb{Z}}(\bar{\mathbf{X}}) \wedge |\bar{\mathbf{X}}| = \bar{\mathbf{z}}\Big)$$

$\Phi_{\Delta-}$ is *complete*:

any satisfying $\sigma_{\mathbb{Z}}$ in PA is an image under $|\cdot|$ of a satisfying $\sigma_{\mathbb{T}}$ in $\mathbb{T}$.

LCC (3)

Identify constraints with the corresponding solution set.

$\Phi_{\Delta+}$ is an *over-approximation* of Φ_Δ:

$$\Phi_\Delta \subseteq \Phi_{\Delta+}$$

$\Phi_{\Delta-}$ is an *under-approximation* of Φ_Δ:

$$\Phi_{\Delta-} \subseteq \Phi_\Delta$$

Φ_Δ is *unique* up to equivalence:

$$\Phi_{\Delta'} \subseteq \Phi_\Delta \subseteq \Phi_{\Delta'}$$

Example: LCC

$$\Phi_{\mathbb{T}} : \quad x \neq \mathrm{cons}(\mathrm{nil}, \mathrm{nil}) \ \wedge \ y \neq \mathrm{cons}(\mathrm{nil}, \mathrm{nil}) \ \wedge \ x \neq y$$

$$\Phi_{\mathbb{Z}} : \quad |x| = |y|$$

$$\Phi_{\Delta+} : \quad 2 \nmid |x| \ \wedge \ |x| = |y|$$

$$\Phi_{\Delta-} : \quad |x| > 5 \ \wedge \ 2 \nmid |x| \ \wedge \ |x| = |y|$$

$$\Phi_{\Delta} : \quad |x| > 3 \ \wedge \ 2 \nmid |x| \ \wedge \ |x| = |y|$$

Main Theorem

Given $\Phi_{\mathbb{T}} \wedge \Phi_{\mathbb{Z}}$.

Let Φ_Δ be an LCC for $\Phi_{\mathbb{T}} \wedge \Phi_{\mathbb{Z}}$. Then

$$\mathrm{TA}_{\mathbb{Z}} \models_{\exists} \Phi_{\mathbb{T}} \wedge \Phi_{\mathbb{Z}} \quad \Leftrightarrow \quad \mathrm{TA} \models_{\exists} \Phi_{\mathbb{T}} \;\;\&\;\; \mathrm{PA} \models_{\exists} \Phi_{\Delta}.$$

Decision Problem $\mapsto$ Computation of LCC.

Generic Decision Procedure

Input: $\Phi_{\mathbb{T}} \wedge \Phi_{\mathbb{Z}}$.

1. Return *FAIL* if $\mathrm{TA} \not\models_{\exists} \Phi_{\mathbb{T}}$.
2. For each partition $\Phi_{\mathbb{T}}^{(i)} \wedge \Phi_{\mathbb{Z}}^{(i)}$ of $\Phi_{\mathbb{T}} \wedge \Phi_{\mathbb{Z}}$:
   (a) Compute an LCC $\Phi_{\Delta}^{(i)}$ for $\Phi_{\mathbb{T}}^{(i)} / \Phi_{\mathbb{Z}}^{(i)}$.
   (b) Return *SUCCESS* if $\mathrm{PA} \models_{\exists} \Phi_{\Delta}^{(i)}$.
3. Return *FAIL*.

How to compute LCC?

Computing the LCC

- **Infinite constant domain:**
  - create DAG representation of the formula.

    Oppen's algorithm [Opp80]
  - extract size constraints from the DAG.
- **Finite constant domain:**
  - create DAG representation of the formula.
  - extract size constraints from the DAG.
  - **add counting constraints to express bounded number of distinct terms of given length.**
  - **need to know which terms are of equal length: equality completion.**

LCC for Infinite Constant Domain

Input:

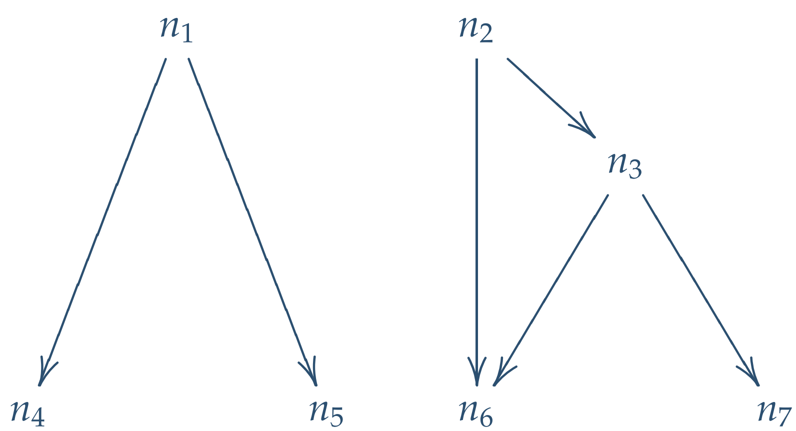
1. $\Phi_{\mathbb{T}} \wedge \Phi_{\mathbb{Z}}$.
2. $G_{\mathbb{T}}$: the DAG of $\Phi_{\mathbb{T}}$,
3. $R\!\downharpoonleft\!\upharpoonright$: the equivalence relation on $G_{\mathbb{T}}$.

Initially set $\Phi_{\Delta} = \Phi_{\mathbb{Z}}$. For each term t add the following to Φ_{Δ}.

- $|t| = 1$, if t is a constant;
- $|t| = |s|$, if $(t, s) \in R\!\downharpoonleft\!\upharpoonright$.
- $\mathrm{Tree}(t)$ if t is an untyped leaf vertex.
- $\mathrm{Node}^{\alpha}(t, \bar{\mathbf{t}}_{\alpha})$ if t is an α-typed vertex with children $\bar{\mathbf{t}}_{\alpha}$.
- $\mathrm{Tree}^{\alpha}(t)$ if t is an α-typed leaf vertex.

Example: LCC for Infinite Constant Domain (

$$\mathrm{Is}_{\mathrm{cons}}(y) \wedge x = \mathrm{cons}(\mathrm{car}(y), y) \ \wedge \ |\mathrm{cons}(\mathrm{car}(y), y)| \ < \ 2|\mathrm{car}(x)|.$$



$$\begin{array}{lcl} n_1 & : & x \\ n_2 & : & \mathrm{cons}(\mathrm{car}(y), y) \\ n_3 & : & y \\ n_4 & : & \mathrm{car}(x) \\ n_5 & : & \mathrm{cdr}(x) \\ n_6 & : & \mathrm{car}(y) \\ n_7 & : & \mathrm{cdr}(y) \end{array}$$

Equivalence relation:

$$\{\{n_1, n_2\}, \{n_3, n_5\}, \{n_4, n_6\}, \{n_7\}\}.$$

$$\begin{tikzcd}
n_1,n_2 \arrow[dd] \arrow[dr] & & \\
& n_3,n_5 \arrow[dl] \arrow[dr] & \\
n_4,n_6 & & n_7
\end{tikzcd}$$

$$\begin{aligned}
n_1, n_2 &\quad : \quad \{x, \mathrm{cons}(\mathrm{car}(y), y)\} \\
n_3, n_5 &\quad : \quad \{y, \mathrm{cdr}(x)\} \\
n_4, n_6 &\quad : \quad \{\mathrm{car}(x), \mathrm{car}(y)\} \\
n_7 &\quad : \quad \mathrm{cdr}(y)
\end{aligned}$$

Example: LCC for Infinite Constant Domain (

Induced length constraints:

$$|\text{car}(x)| \geq 1 \ \wedge \ |\text{cdr}(x)| \geq 1 \ \wedge \ |\text{car}(y)| \geq 1 \ \wedge \ |\text{cdr}(y)| \geq 1,$$
$$|x| = |\text{cons}(\text{car}(y), y)| \ \wedge \ |\text{car}(x)| = |\text{car}(y)| \ \wedge \ |\text{cdr}(x)| = |y|,$$

$$|x| = |\text{car}(x)| + |\text{cdr}(x)| \ \wedge \ |y| = |\text{car}(y)| + |\text{cdr}(y)| \ \wedge$$
$$|\text{cons}(\text{car}(y), y)| = |\text{car}(y)| + |y|$$

which imply $|\text{cons}(\text{car}(y), y)| \geq 2|\text{car}(x)| + 1$.

☞ $\text{Is}_{\text{cons}}(y) \wedge x = \text{cons}(\text{car}(y), y) \ \wedge \ |\text{cons}(\text{car}(y), y)| \ < \ 2|\text{car}(x)|.$

is unsatisfiable.

LCC for Finite Constant Domain

With finite constant domain we have more "hidden" constraints.

- there are only a bounded number of distinct terms of a given length.
- need to add counting constraint $\mathrm{CNT}^{\alpha}_{k,n}(x)$ that says that

  there are at least $n{+}1$ different α-terms of length x in the structure having k constants.

- $\mathrm{CNT}^{\alpha}_{k,n}(x)$ is expressible in Presburger arithmetic.
- need to know which terms are of equal length: equality completion.

Equality Completion

Φ is called *equality complete* if for any u, v in Φ,

- exactly one of $u = v$ and $u \neq v$, and
- exactly one of $|u| = |v|$ and $|u| \neq |v|$ are in Φ.

We say that $x_1, \ldots, x_n$ is in a cluster if

$x_1, \ldots, x_n$ have the same length but pairwise unequal.

Equality Completion puts terms into stratified clusters.

Example: Equality Completion

$$x \neq z \;\wedge\; y \neq \mathrm{cons}(x, z)$$

can be made equality complete by adding

$$|y| = |\mathrm{cons}(x, z)| \;\wedge\; |x| = |z|.$$

Picture this:

$$\begin{tikzcd}
y \arrow[rr, dotted, dash] & & \mathrm{cons}(x,z) \arrow[dl] \arrow[dr] & \\
& x \arrow[rr, dotted, dash] & & z
\end{tikzcd}$$

LCC for Finite Constant Domain

Input:

1. $\Phi_{\mathbb{T}} \wedge \Phi_{\mathbb{Z}}$ (equality complete).
2. $G_{\mathbb{T}}$: the DAG of $\Phi_{\mathbb{T}}$,
3. $R\downharpoonleft\upharpoonright$: the equivalence relation on $G_{\mathbb{T}}$.

Initially set $\Phi_{\Delta} = \Phi_{\mathbb{Z}}$. For each term t add the following to Φ_{Δ}.

- $|t| = 1$, if t is a constant;
- $|t| = |s|$, if $(t, s) \in R\downharpoonleft\upharpoonright$.
- $\text{Tree}(t)$ if t is an untyped leaf vertex.
- $\text{Node}(t, t_1, \ldots, t_k)$ if t is a node with children $t_1, \ldots, t_k$.
- $\text{Tree}^{\alpha}(t)$ if t is an α-typed leaf vertex.
- $\text{CNT}^{\alpha}_{1,n}(|t|)$ *if there exist* $t_1, \ldots, t_n$ *s.t.* $t, t_1, \ldots t_n$ *are in the same cluster.*

Example: LCC for Finite Constant Domain

$$\Phi : \quad x \neq \mathrm{cons}(\mathrm{nil}, \mathrm{nil}) \ \wedge\ |x| = 3.$$

implies that x and $\mathrm{cons}(\mathrm{nil}, \mathrm{nil}))$ are in the same cluster.

Then Φ_Δ contains

$$\mathrm{CNT}^{\mathrm{cons}}_{1,2}(|x|) : \quad |x| \nmid 2 \ \wedge\ |x| > 3.$$

So Φ is unsatisfiable.

Quantifier Elimination for $\mathrm{Th}(\mathrm{TA}_{\mathbb{Z}})$

1. Blockwise Elimination. Remove a block of quantifiers in one step.

$$(\exists x_1, \ldots, \exists x_n)\Phi(x_1, \ldots, x_n, y_1, \ldots, y_m) \quad \mapsto \quad \Phi'(y_1, \ldots, y_m)$$

2. Almost Optimal Complexity. One exponential for each quantifier alternation.

   (Term algebras itself are non-elementary.)

$$\left. 2^{2^{\cdot^{\cdot^{\cdot^{n}}}}} \right\} O(n)$$

PART II. Queues with Integers

Previous Work on Queues

- Quantifier-free theory with subsequence relations.

  Bjørner [Bjø98]

- Quantified theory.

  Rybina and Voronkov [RV00] [RV03]

- With prefix relation.

  Benedikt, Libkin, Schwentick and Segoufin [BLSS01]

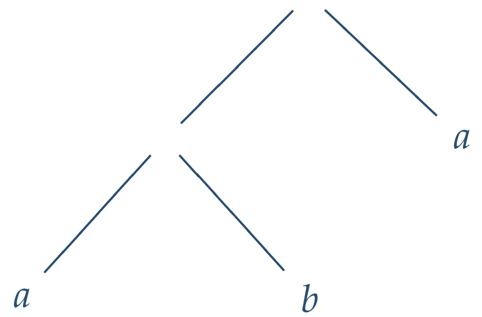
- WS1S with cardinality constraints.

  Klaedtke and Ruess [KR03a]

Difference Between Term Algebras and Queu

A term is constructed uniquely. For example,

$\mathrm{cons}(\mathrm{cons}(a, b), a)$:



A queue can be constructed in many ways. For example,

$$aba : \quad \begin{array}{ccccccc} (& (& a &) & b &) & a \\ a & (& b & (& a &) &) \\ (& a & (& b &) &) & a \\ a & (& (& b &) & a &) \end{array}$$

Queues (1)

$\mathfrak{Q}: \ \langle \mathcal{Q}; \mathcal{A}, \mathcal{C}, \mathcal{S} \rangle$:

1. $\mathcal{A}$: Constants: a, b, c, . . .
2. $\mathcal{Q}$: Sequences of constants. $\epsilon_{\mathcal{Q}}$: the empty queue.
3. $\mathcal{C}$: Constructors:

Left Insertion $\mathrm{la} : \mathcal{A} \times \mathcal{Q} \to \mathcal{Q}$

Right Insertion $\mathrm{ra} : \mathcal{A} \times \mathcal{Q} \to \mathcal{Q}$, s.t.

$$\mathrm{la}(a, \epsilon_{\mathcal{Q}}) = \mathrm{ra}(a, \epsilon_{\mathcal{Q}}) = \langle a \rangle,$$
$$\mathrm{la}(a, \langle s_1, \ldots, s_n \rangle) = \langle a, s_1, \ldots, s_n \rangle,$$
$$\mathrm{ra}(a, \langle s_1, \ldots, s_n \rangle) = \langle s_1, \ldots, s_n, a \rangle.$$

Queues (2)

4 $\mathcal{S}$: Selectors:

Left Head $\mathrm{lh} : \mathcal{Q} \to \mathcal{A}$, *Left Tail* $\mathrm{lt} : \mathcal{Q} \to \mathcal{Q}$,

Right Head $\mathrm{rh} : \mathcal{Q} \to \mathcal{A}$, *Right Tail* $\mathrm{rt} : \mathcal{Q} \to \mathcal{Q}$, s.t.

$$
\begin{aligned}
\mathrm{lh}(\langle s_1, \ldots, s_n \rangle) &= s_1, \\
\mathrm{lt}(\langle s_1, \ldots, s_n \rangle) &= \langle s_2, \ldots, s_n \rangle, \\
\mathrm{rh}(\langle s_1, \ldots, s_n \rangle) &= s_n, \\
\mathrm{rt}(\langle s_1, \ldots, s_n \rangle) &= \langle s_1, \ldots, s_{n-1} \rangle.
\end{aligned}
$$

Convention: use *concatenation operator* $\circ$.

$a \circ X \circ b$ *stands for* $\mathrm{ra}(b, \mathrm{la}(a, X))$ *or* $\mathrm{la}(a, \mathrm{ra}(b, X))$.

Input: $\Phi \equiv \mathcal{E} \cup \mathcal{D}$.

1. Normalize Φ to $\Phi' : \mathcal{E}' \cup \mathcal{D}'$.
2. Return *FAIL*, if inconsistency is discovered;

   Return *SUCCESS*.

Normal Form in $\mathfrak{Q}$

Let $X \in \mathrm{orb}(\alpha, k)$ denote that X is of the form $\alpha^*\alpha[1..k]$.

A queue constraint Φ_Q is in *normal form* if

- all equalities are in triangular form,
- for each X there exists at most one literal $X \in \mathrm{orb}(\alpha, k)$,
- if $X \in \mathrm{orb}(\alpha, k)$ occurs, then no $X \notin \mathrm{orb}(\alpha', k')$ occurs, and
- disequalities are in the form $\alpha X \neq Y\beta$ for $X \not\equiv Y$.

Queues with Integers

Presburger arithmetic (PA): $\mathscr{L}_{\mathbb{Z}}$, PA.

Two-sorted language $\Sigma = \Sigma_Q \cup \Sigma_{\mathbb{Z}} \cup \{|\cdot|\}$:

1. Σ_Q: signature of queues.
2. $\Sigma_{\mathbb{Z}}$: signature of Presburger arithmetic.
3. $|\cdot| \; : \; \mathfrak{Q} \to \mathbb{N}$, the length function.

Problem I

The presence of $\Phi_{\mathbb{Z}}$ restricts solutions to Φ_{Q}.

Example: Suppose $\mathcal{A} = \{a, b\}$. Then

$$\Phi_Q : \quad Xba \neq abY \ \wedge \ Xab \neq baY \ \wedge \ Xaa \neq baY \ \wedge \ Xab \neq aaY$$

is not satisfiable with $\Phi_{\mathbb{Z}} : |X| = |Y| = 1$.

Problem I

The presence of $\Phi_{\mathbb{Z}}$ restricts solutions to Φ_Q.

Example: Suppose $\mathcal{A} = \{a, b\}$. Then

$$\Phi_Q: \quad Xba \neq abY \;\wedge\; Xab \neq baY \;\wedge\; Xaa \neq baY \;\wedge\; Xab \neq aaY$$

is not satisfiable with $\Phi_{\mathbb{Z}} : |X| = |Y| = 1$.

$$\boxed{\text{Computing LCC.}}$$

Example:

$\Phi_{\mathbb{Z}}$	Φ_{Δ}
$\lvert X\rvert = \lvert Y\rvert$	$\lvert X\rvert \neq 1 \;\wedge\; \lvert X\rvert = \lvert Y\rvert$

Problem I

The presence of $\Phi_{\mathbb{Z}}$ restricts solutions to Φ_Q.

Example: Suppose $\mathcal{A} = \{a, b\}$. Then

$$\Phi_Q : \quad Xba \neq abY \ \wedge \ Xab \neq baY \ \wedge \ Xaa \neq baY \ \wedge \ Xab \neq aaY$$

is not satisfiable with $\Phi_{\mathbb{Z}} : |X| = |Y| = 1$.

$$\boxed{\text{Computing LCC.}}$$

Example:

$\Phi_{\mathbb{Z}}$	Φ_{Δ}
$\lvert X\rvert = \lvert Y\rvert$	$\lvert X\rvert \neq 1 \ \wedge \ \lvert X\rvert = \lvert Y\rvert$

But more work needs to be done here: new normalization.

Problem II

We cannot partition terms into *stratified clusters* and construct a satisfying assignment inductively.

Example: Consider

$$X \neq Y \;\wedge\; aX \neq Yb \;\wedge\; Xa \neq bY$$

Infinitely many assignments of the form

$$X = (ba)^n b, \qquad\qquad Y = a(ba)^n$$

satisfy $X \neq Y$, but neither $aX \neq Yb$ nor $Xa \neq bY$.

Problem II

We cannot partition terms into *stratified clusters* and construct a satisfying assignment inductively.

Example: Consider

$$X \neq Y \ \wedge \ aX \neq Yb \ \wedge \ Xa \neq bY$$

Infinitely many assignments of the form

$$X = (ba)^n b, \qquad\qquad Y = a(ba)^n$$

satisfy $X \neq Y$, but neither $aX \neq Yb$ nor $Xa \neq bY$.

Find a *cut length*!

Cut Length

1. Φ_Q can be satisfied by sufficiently long queues.
2. There exists a *cut length* δ such that for any solution $(l_i)_n$ for $\Phi_{\Delta+}$ with $l_i \geq \delta$ is realizable.
3. But δ is not the smallest $max\{(\mu_i)_n\}$ such that

$$\mathfrak{Q}_{\mathbb{Z}} \models_{\exists} \Phi_Q \ \wedge \ \bigwedge_{i=1}^{n} |X_i| = \mu_i$$

Example: $\{X := \epsilon_Q, Y := \epsilon_Q\}$ is a solution for

$$Xba \neq abY \ \wedge \ Xab \neq baY \ \wedge \ Xaa \neq baY \ \wedge \ Xab \neq aaY$$

while there is no solution σ such that $|\sigma(X)| = |\sigma(Y)| = 1$.

Computation of Cut Length

PRE_Φ : the set of all words α s.t. αX or α is a proper term in Φ_Q.

d_Φ: the shortest strongly primitive word d such that

$$(\forall \alpha \in \mathrm{PRE}_\Phi)\ d \notin \mathrm{orb}(\alpha).$$

L_d: the length of d_Φ.

L_c: the smallest number of letters to create a unique identifying word, called a *color*, for each queue variable in Φ_Q.

L_t: $L_c + L_d$.

We claim that $L_t \geq \delta$.

Example

Consider

$$Xba \neq abY \ \wedge \ Xab \neq baY \wedge Xaa \neq baY \ \wedge \ Xab \neq aaY$$

Then

1. $\mathrm{PRE}_\Phi = \{ab, ba, aa\}$.
2. $L_d = 3$; $d_\Phi = aab$.
3. $L_c = 1$; Φ_Q includes two queue variables.

So $L_t = L_c + L_d = 4$.

Computation of LCC for Queues

Input: $\Phi_Q \wedge \Phi_{\mathbb{Z}}$ in normal form of $\mathfrak{Q}_{\mathbb{Z}}$.

Initially set $\Phi_\Delta = \Phi_{\mathbb{Z}}$. Add to Φ_Δ:

- $|t_1| = |t_2|$, if $t_1 \neq t_2$ or $t_1 = t_2$;
- $|X| + |\alpha| = |\alpha X| = |X\alpha|$, if αX or $X\alpha$ occurs;
- $|X| \equiv k (\bmod |\alpha|)$, if $X \in \mathrm{orb}(\alpha, k)$.
- $|X| = i$ (for some $i < L_t$) or $|X| \geq L_t$ for each X in Φ_Q.

Normal Form in $\mathfrak{Q}_{\mathbb{Z}}$

Φ_Q is in *normal form* in $\mathfrak{Q}_{\mathbb{Z}}$ if

1. Φ_Q is in normal form in $\mathcal{Q}$;
2. Φ_Q is equality complete;
3. if $\alpha X \neq Y\beta$ occurs with either $X \in \mathrm{orb}(\alpha', k)$ or $Y \in \mathrm{orb}(\beta', l)$, then $\alpha \equiv \epsilon_Q$;
4. $\alpha X \neq Y\beta$ does not occur with both $X \in \mathrm{orb}(\alpha', k)$ and $Y \in \mathrm{orb}(\beta', l)$.

New Normalization. To deal with parameters $\bar{Y}$.

Blockwise Elimination. Remove a block of quantifiers in one step.

$$(\exists x_1, \ldots, \exists x_n)\Phi(x_1, \ldots, x_n, y_1, \ldots, y_m) \quad \mapsto \quad \Phi'(y_1, \ldots, y_m)$$

PART III. Knuth-Bendix Order

Motivation

- **Termination Proofs.** To rank program states:

$$\langle x = 3, y = 2\rangle > \langle x = 3, y = 1\rangle$$

- **Ordered Resolution.** To restrict the search space:

$$\frac{A \vee C \qquad \neg A' \vee C'}{(C \vee C')\sigma} \qquad \begin{array}{c} \sigma = \mathbf{mgu}(A, A') \\ \forall B \in C\sigma \vee C'\sigma\,(A\sigma \not< B\sigma) \end{array}$$

- **Ordered Rewriting.** To orient commutative equations:

$$L \to R \qquad (L\sigma > R\sigma)$$

How to decide satisfiability of order constraints?

Background: Previous Work (1)

Two types of widely used orderings:

	Syntactic Nature	Hybrid Nature
	LPO	KBO
syntactic precedence	√	√
lexicographical ordering	√	√
numerical ordering		√

Background: Previous Work (2)

Decidability Status:

	LPO	KBO
QFT	√ [Com90] [Nie93]	√ [KV00] [KV01]
UQT	√ [NR00]	√ [KV02]
GQT	× [Tre92, CT97]	?

QFT: Quantifier-free Theory.

UQT: Unary Quantified Theory.

GQT: General Quantified Theory.

Background: Previous Work (2)

Decidability Status:

	LPO	KBO
QFT	√ [Com90] [Nie93]	√ [KV00] [KV01]
UQT	√ [NR00]	√ [KV02]
GQT	× [Tre92, CT97]	√ [ZSM05]

QFT: Quantifier-free Theory.

UQT: Unary Quantified Theory.

GQT: General Quantified Theory.

Knuth-Bendix Order (1)

A Knuth-Bendix order (KBO) $\prec^{\mathrm{kb}}$ is parametrically defined with

- $\mathsf{W} : \mathrm{TA} \to \mathbb{N}$: a weight function satisfying

$$\mathsf{W}(\alpha(t_1, \ldots, t_k)) = \mathsf{W}(\alpha) + \sum_{i=1}^{k} \mathsf{W}(t_i).$$

- $\prec^{\Sigma}$: a linear (precedence) order on C such that

$$\alpha_1 \succ^{\Sigma} \alpha_2 \succ^{\Sigma} \ldots \succ^{\Sigma} \alpha_{|C|}.$$

Knuth-Bendix Order (2)

For $u, v \in \mathrm{TA}$, $u \prec^{\mathrm{kb}} v$ if one of the following holds:

- $\mathrm{W}(u) < \mathrm{W}(v)$.
- $\mathrm{W}(u) = \mathrm{W}(v)$ and $\mathrm{type}(u) \prec^{\Sigma} \mathrm{type}(v)$.
- $\mathrm{W}(u) = \mathrm{W}(v)$, $u \equiv \alpha(u_1, \ldots, u_k)$, $v \equiv \alpha(v_1, \ldots, v_k)$, and

$$\exists i \big[1 \le i \le k \wedge u_i \prec^{\mathrm{kb}} v_i \wedge \forall j (1 \le j < i \rightarrow u_j = v_j) \big].$$

Quantifier Elimination

- Suffices to eliminate $\exists$-quantifiers from primitive formulas

$$\exists \bar{\mathbf{x}} \Big[A_1(\bar{\mathbf{x}}) \ \wedge \ \ldots \ \wedge \ A_n(\bar{\mathbf{x}}) \Big],$$

  where $A_i(\bar{\mathbf{x}})$ are literals.

- Suffices to assume $A_i \not\equiv x = t$ if $x \notin t$, because

$$\exists x \Big[x = t \ \wedge \ \varphi(x, \bar{\mathbf{y}}) \Big] \leftrightarrow \ \varphi(t, \bar{\mathbf{y}}).$$

Main Idea: Depth Reduction

Eliminating $\exists x$ from $(\exists x)\varphi(x, \bar{\mathbf{y}})$ is straightforward if

$$depth_{\varphi}(x) = 0.$$

Such $\varphi(x, \bar{\mathbf{y}})$ is said to be *solved in x*.

$\Big(depth_{\varphi}(x)$: the length of the longest selector sequence in front of x in φ.$\Big)$

Solved Form

- $\varphi(x, \bar{\mathbf{y}})$ is solved in x if it is in the form

$$\bigwedge_{i \leq m} u_i \prec^{\mathrm{kb}} x \ \wedge \ \bigwedge_{j \leq n} x \prec^{\mathrm{kb}} v_j \ \wedge \ \varphi'(\bar{\mathbf{y}}),$$

where x does not appear in u_i, v_i and φ'.

- If $\varphi(x, \bar{\mathbf{y}})$ is solved in x, then $(\exists x)\ \varphi(x, \bar{\mathbf{y}})$ simplifies to

$$\bigwedge_{i \leq m, j \leq n} u_i \prec_2^{\mathrm{kb}} v_j \ \wedge \ \varphi'(\bar{\mathbf{y}})$$

where $x \prec_n^{\mathrm{kb}} y$, called gap order, states there is an increasing chain from x to y of length at least n.

Depth Reduction: Case 1

Case 1: All occurrences of x have depth greater than 0.

In this case, $\exists x \varphi(x, \bar{\mathbf{y}})$ goes to

$$\exists x_1, \ldots, \exists x_k \varphi'(x_1, \ldots, x_k, \bar{\mathbf{y}}),$$

where

$$\varphi'(x_1, \ldots, x_k, \bar{\mathbf{y}}) \equiv \varphi(x, \bar{\mathbf{y}}) \left[\mathsf{s}_i^{\alpha}(x) \leftarrow x_i \right].$$

Case 1:Example

$$(\exists x)\Big[\,\mathrm{car}(x) \prec^{\mathrm{kb}} \mathrm{cdr}(x)\,\Big]$$

$$\Rightarrow\ (\exists x_1)(\exists x_2)(\exists x)\Big[\,x_1 = \mathrm{car}(x)\ \wedge\ x_2 = \mathrm{cdr}(x)\ \wedge\ \mathrm{car}(x) \prec^{\mathrm{kb}} \mathrm{cdr}(x)\,\Big]$$

(decompose x)

$$\Rightarrow\ (\exists x_1)(\exists x_2)(\exists x)\Big[\,x_1 = \mathrm{car}(x)\ \wedge\ x_2 = \mathrm{cdr}(x)\ \wedge\ x_1 \prec^{\mathrm{kb}} x_2\,\Big]$$

(substitution)

$$\Rightarrow\ (\exists x_1)(\exists x_2)\Big[\,x_1 \prec^{\mathrm{kb}} x_2\,\Big]$$

(remove x)

Depth Reduction: Case 2

Case 2: Some x have depth 0 and some do not.

- Decompose 0-depth occurrences of x in terms of

$$\mathsf{s}_1^{\alpha}(x), \ldots, \mathsf{s}_k^{\alpha}(x).$$

- This amounts to expressing $x \prec_n^{\mathrm{kb}} t$ and $t \prec_n^{\mathrm{kb}} x$ using

$$\mathsf{s}_1^{\alpha}(x), \ldots, \mathsf{s}_k^{\alpha}(x).$$

- Then apply the reduction as in Case 1!

Case 2: Example

$$(\exists x)\Big[\,\mathrm{car}(x) \prec^{\mathrm{kb}} y \ \wedge\ y \prec^{\mathrm{kb}} x\,\Big]$$

$$\Rightarrow \quad (\exists x_1)(\exists x_2)(\exists x)\Big[\,x_1 = \mathrm{car}(x) \ \wedge\ x_2 = \mathrm{cdr}(x) \ \wedge\ \mathrm{car}(x) \prec^{\mathrm{kb}} y \ \wedge\ y \prec^{\mathrm{kb}} x\,\Big]$$

(decompose x)

$$\Rightarrow \quad (\exists x_1)(\exists x_2)(\exists x)\Big[\,x_1 = \mathrm{car}(x) \ \wedge\ x_2 = \mathrm{cdr}(x) \ \wedge\ \mathrm{car}(x) \prec^{\mathrm{kb}} y \ \wedge\ \mathrm{car}(y) = \mathrm{car}(x) \ \wedge\ \mathrm{cdr}(y) \prec^{\mathrm{kb}} \mathrm{cdr}(x)\,\Big]$$

(decompose $y \prec^{\mathrm{kb}} x$; reduce to case 1)

Case 2:Example (Cont'd)

$$\Rightarrow \quad (\exists x_1)(\exists x_2)(\exists x)\Big[\, x_1 = \mathrm{car}(x) \ \wedge\ x_2 = \mathrm{cdr}(x) \ \wedge x_1 <^{\mathrm{kb}} y \ \wedge\ \mathrm{car}(y) = x_1 \ \wedge\ \mathrm{cdr}(y) <^{\mathrm{kb}} x_2 \,\Big]$$

(substitution)

$$\Rightarrow \quad (\exists x_1)(\exists x_2)\Big[\, x_1 <^{\mathrm{kb}} y \ \wedge\ \mathrm{car}(y) = x_1 \ \wedge\ \mathrm{cdr}(y) <^{\mathrm{kb}} x_2 \,\Big]$$

(remove x)

Input: $(\exists \bar{\mathbf{x}})\ \varphi(\bar{\mathbf{x}}, \bar{\mathbf{y}})$.

While $\bar{\mathbf{x}} \neq \emptyset$.

- While $(\forall x \in \bar{\mathbf{x}})\, depth_{\varphi}(x) > 0$.

  Depth Reduction.

  - VARIABLE SELECTION.
  - DECOMPOSITION.
  - SIMPLIFICATION.

  Done.

- While $(\exists x \in \bar{\mathbf{x}})\, depth_{\varphi}(x) = 0$.

  Elimination.

  Done.

Done.

Variable Selection

Select a variable $x \in \bar{\mathbf{x}}$ such that $\mathrm{s}_i^\alpha(x)$ appears in $\varphi(\bar{\mathbf{x}}, \bar{\mathbf{y}})$.

☞ The variable selection is done in depth-first manner.

☞ I.e., choose variables generated in the previous round.

Decomposition

Rewrite $(\exists \bar{\mathbf{x}})\ \varphi(\bar{\mathbf{x}}, \bar{\mathbf{y}})$ to

$$\exists x_1 \ldots \exists x_k \exists \bar{\mathbf{x}} \left[\mathrm{Is}_\alpha(x) \wedge \bigwedge_{1 \le i \le k} \mathrm{s}_i^\alpha(x) = x_i \wedge \varphi(\bar{\mathbf{x}}, \bar{\mathbf{y}}) \right].$$

Simplification

Apply the following rules to each occurrence of x.

1. Replace $x \prec_n^\sharp t$ (or $t \prec_n^\sharp x$) by a quantifier-free formula
$$\varphi'(\mathsf{s}_1^\alpha(x), \ldots, \mathsf{s}_k^\alpha(x), \mathsf{s}_1^\alpha(t), \ldots, \mathsf{s}_k^\alpha(t)).$$
2. Replace $\mathsf{s}_i^\alpha(x)$ in $\varphi(\bar{\mathbf{x}}, \bar{\mathbf{y}})$ by x_i $(1 \leq i \leq k)$.

Denote the result of this simplification by

$$\exists x_1 \ldots \exists x_k \exists (\bar{\mathbf{x}} \setminus x) \left[\varphi'(\bar{\mathbf{x}} \setminus x, x_1, \ldots, x_k, \bar{\mathbf{y}}) \right].$$

Elimination

- We have

$$\exists x \Big[\bigwedge_{i \leq m} u_i \prec^{\mathrm{kb}} x \ \wedge \ \bigwedge_{j \leq n} x \prec^{\mathrm{kb}} v_j \ \wedge \ \varphi'(\bar{\mathbf{y}}) \Big],$$

where x appears none of u_i, v_j and φ'.

- Guessing a gap order completion, we rewrite it to

$$u_{i'} \prec_2^{\mathrm{kb}} v_{j'} \ \wedge \ \varphi'(\bar{\mathbf{y}})$$

$$\wedge \text{ “}u_{i'} \textit{ is the greatest of } \{u_i \,|\, i \leq m\}\text{”}$$

$$\wedge \text{ “}v_{j'} \textit{ is the smallest of } \{v_j \,|\, j \leq n\}\text{”}.$$

Technical Challenges (1)

1. Decompose $\prec^{\mathrm{kb}}$ into three disjoint suborders $\prec^{\mathsf{w}}$, $\prec^{\mathsf{p}}$ and $\prec^{\mathsf{l}}$.
2. Extend $\prec^{\mathsf{w}}$, $\prec^{\mathsf{p}}$ and $\prec^{\mathsf{l}}$ to $\prec^{\mathsf{w}}_n$, $\prec^{\mathsf{p}}_n$ and $\prec^{\mathsf{l}}_n$, respectively.
3. Add Presburger arithmetic explicitly to represent weight.
4. Define counting constraints to count terms of certain weight.
5. Define boundary functions to delineate gap orders.

$$0^{\mathsf{w}}(n), \quad 0^{\mathsf{p}}(n,p), \quad 1^{\mathsf{w}}(n), \quad 1^{\mathsf{p}}(n,p).$$

6. Extend all aforementioned notions to tuples of terms.

Technical Challenges (2)

- Elimination of Complex Terms.

$$\mathrm{car}(0^{\mathsf{w}}_{((\mathrm{car}(x))^{\mathsf{w}})}).$$

- Elimination of Integer Quantifiers.

$$(\exists z : \mathbb{Z})\Big[\ \mathrm{car}(0^{\mathsf{w}}_{(z)}) \prec^{\mathrm{kb}} \mathrm{cdr}(0^{\mathsf{w}}_{(z)})\ \Big].$$

- Elimination of Equalities.

$$\exists x\Big[\ x = 0^{\mathsf{w}}_{((\mathrm{car}(x))^{\mathsf{w}})}\ \wedge\ \mathrm{car}(x) \prec^{\mathsf{p}}_{4} \mathrm{cdr}(x)\ \Big].$$

- Elimination of Negations.

$$\neg\Big(\mathrm{car}(x) \prec^{\mathsf{w}}_{3} \mathrm{cdr}(x)\Big).$$

- TERMINATION!

PART IV. Conclusion and Future Work

Conclusion

- Decision procedures for the combination of data structures with integer constraints
  - Express memory safety property.
  - Essential for practical program verification.
- Proof of decidability of the first-order theory of Knuth-Bendix orders.
  - Long-standing open problem (RTA problem #99).
  - Important result for term rewriting.

Exploit algebraic properties of concrete domains.

Future Work (1)

- Implementation and experimentation.
- More expressive languages.
  - Term algebras with subterm relation
  - Queues with subsequence relations, namely, prefix $\preceq_p$, subqueue $\preceq$ and suffix $\preceq_s$

Future Work (1)

- Implementation and experimentation.
- More expressive languages.
  - Term algebras with subterm relation
  - Queues with subsequence relations, namely, prefix $\preceq_p$, subqueue $\preceq$ and suffix $\preceq_s$

    With our decision procedures for

    $$\mathfrak{Q}_{\mathbb{Z}} + \preceq_p + \preceq \quad \textit{and} \quad \mathfrak{Q}_{\mathbb{Z}} + \preceq_s + \preceq,$$

    the next step is $\boxed{\mathfrak{Q}_{\mathbb{Z}} + \preceq_p + \preceq_s}$!

Future Work (2)

$\mathfrak{Q}_{\mathbb{Z}}+\preceq_p+\preceq_s$ is a very expressive theory.

1. Equivalent to the theory of concatenation with integers. (Open problem since 80's, Büchi and Senger [BS88])

$$uv^2 = vuv \ \wedge \ |u| < |v|$$

2. Interpret the theory of arrays.

$$q[i] = a \leftrightarrow \exists p\,(pa \preceq_p q \ \wedge \ |pa| = i)$$

3. Interpret Presburger arithmetic with divisibility predicate.

$$x = y + 2 \ \wedge \ y \mid x$$

4. Augmentable to theory of unbounded bit-vectors.

$$u \oplus v = w \wedge uv = ww$$

Thank You!

[ARR01] Alessandro Armando, Silvio Ranise, and Michaël Rusinowitch. Uniform derivation of decision procedures by superposition. *Lecture Notes in Computer Science*, 2142:513–527, 2001.

[Bac95] Rolf Backofen. A complete axiomatization of a theory with feature and arity constraints. *Journal of Logical Programming*, 24(1&2):37–71, 1995.

[Bjø98] Nikolaj S. Bjørner. *Integrating Decision Procedures for Temporal Verification*. PhD thesis, Computer Science Department, Stanford University, November 1998.

[BLSS01] Michael Benedikt, Leonid Libkin, Thomas Schwentick, and Luc Segoufin. A model-theoretic approach to regular string relations. In *Proc. 16th IEEE Symp. Logic in Comp. Sci.*, pages 431–440. IEEE Computer Society Press, 2001.

[BS88] J. Richard Büchi and Steen Senger. Definability in the existential theory of concatenation and undecidable extensions of this theory. *Zeitschr. f. math. Logik und Grundlagen d. Math.*, 34:337–342, 1988.

[CD94] Hubert Comon and Catherine Delor. Equational formulae with membership constraints. *Information and Computation*, 112(2):167–216, 1994.

[Com90] Hubert Comon. Solving symbolic ordering constraints. *International Journal of Foundations of Computer Science*, 1(4):387–411, 1990.

[CT97] Hubert Comon and Ralf Treinen. The first-order theory of lexicographic path orderings is undecidable. *Theoretical Computer Science*, 176(1-2):67–87, 1997.

[DST80] J. Downey, R. Sethi, and R. E. Tarjan. Variations of the common subexpression problem. *J. ACM*, 27:758–771, 1980.

[Ghi05] Silvio Ghilardi. Model-theoretic methods in combined constraint satisfiability. *Journal of Automated Reasoning*, 33(3-4):221–249, 2005.

[KR03a] Felix Klaedtke and Harald Rueß. Monadic second-order logics with cardinalities. In Jos C. M. Baeten, Jan Karel Lenstra, Joachim Parrow, and Gerhard J. Woeginger, editors, *30th International Colloquium on Automata, Languages and Programming, ICALP'2003*, volume 2719 of *Lecture Notes in Computer Science*. Springer-Verlag, 2003.

[KR03b] Viktor Kuncak and Martin Rinard. The structural subtyping of non-recursive types is decidable. In *Proc. 18th IEEE Symp. Logic in Comp. Sci.*, pages 96–107. IEEE Computer Society Press, 2003.

[KV00] Konstantin Korovin and Andrei Voronkov. A decision procedure for the existential theory of term algebras with the Knuth-Bendix ordering. In *Proc. 15th IEEE Symp. Logic in Comp. Sci.*, pages 291 – 302, 2000.

[KV01] Konstantin Korovin and Andrei Voronkov. Knuth-Bendix constraint solving is NP-complete. In *Proceedings of*

28th International Colloquium on Automata, Languages and Programming (ICALP'01), volume 2076 of *Lecture Notes in Computer Science*, pages 979–992. Springer-Verlag, 2001.

[KV02] Konstantin Korovin and Andrei Voronkov. The decidability of the first-order theory of the Knuth-Bendix order in the case of unary signatures. In *Proceedings of the 22th Conference on Foundations of Software Technology and Theoretical Computer Science, (FSTTCS'02)*, volume 2556 of *Lecture Notes in Computer Science*, pages 230–240. Springer-Verlag, 2002.

[Mah88] M. J. Maher. Complete axiomatizations of the algebras of finite, rational and infinite tree. In *Proceedings of the Third Annual Symposium on Logic in Computer Science*, pages 348–357. IEEE Computer Society Press, 1988.

[Mal71] A. I. Mal'cev. Axiomatizable classes of locally free algebras of various types. In *The*

Metamathematics of Algebraic Systems, Collected Papers, chapter 23, pages 262–281. North Holland, 1971.

[Nie93] Robert Nieuwenhuis. Simple LPO constraint solving methods. *Information Processing Letters*, 47(2):65–69, 1993.

[NO79] Greg Nelson and Derek C. Oppen. Simplification by cooperating decision procedures. *ACM Trans. Prog. Lang. Sys.*, 1(2):245–257, October 1979.

[NO80] Greg Nelson and Derek C. Oppen. Fast decision procedures based on congruence closure. *J. ACM*, 27(2):356–364, April 1980.

[NR00] Paliath Narendran and Michael Rusinowitch. The theory of total unary RPO is decidable. In *CL 2000*, volume 1861 of *Lecture Notes in Artificial Intelligence*, pages 660–672. Springer-Verlag, 2000.

[Opp80] Derek C. Oppen. Reasoning about recursively defined data structures. *J. ACM*, 27(3), July 1980.

[RV00] Tatiana Rybina and Andrei Voronkov. A decision procedure for term algebras with queues. In *Proc. 15th IEEE Symp. Logic in Comp. Sci.*, pages 279 – 290, 2000.

[RV03] Tatiana Rybina and Andrei Voronkov. Upper bounds for a theory of queues. In *Proceedings of 30th International Colloquium on Automata, Languages and Programming (ICALP'03)*, volume 2719 of *LNCS*, pages 714–724. Springer-Verlag, 2003.

[TR03] Cesare Tinelli and Christophe Ringeissen. Unions of non-disjoint theories and combinations of satisfiability procedures. *Theoretical Computer Science*, 290(1):291–353, January 2003.

[Tre92] Ralf Treinen. A new method for undecidability proofs of first order theories. *Journal of Symbolic Computation*, 14:437–457, 1992.

[Zar02] Calogero G. Zarba. A tableau calculus for combining non-disjoint theories. volume 2381 of *Lecture Notes in Artificial Intelligence*, pages 315–329. Springer, 2002.

[ZSM05] Ting Zhang, Henny B. Sipma, and Zohar Manna. The decidability of the first-order theory of term algebras with Knuth-Bendix order. In Robert Nieuwenhuis, editor, *the* 20^{th} *International Conference on Automated Deduction (CADE'05)*, volume 3632 of *LNCS*, pages 131–148. Springer-Verlag, 2005.