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?

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

\[ \varphi \rightarrow \text{decision procedure} \rightarrow \begin{cases} \text{satisfiable} \\
\text{unsatisfiable} \end{cases} \]

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?

\[ \Sigma_1\text{-theory } T_1 \quad \Sigma_2\text{-theory } T_2 \]

\[ P_1 \quad \text{for } T_1\text{-satisfiability} \quad P_2 \quad \text{for } T_2\text{-satisfiability} \]

? \quad \text{for } (T_1 \cup T_2)\text{-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 \textit{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.
Decision procedures for term algebras with integer constraints:


Decision procedures for queues with integer constraints:
T. Zhang, H.B. Sipma and Z. Manna,
*Decision Procedures for Queues with Integer Constraints.*

Decision procedures for Knuth-Bendix orders:
T. Zhang, H.B. Sipma, Z. Manna,
*The Decidability of the First-order Theory of Knuth-Bendix Order.*
(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].
A term algebra $TA: \langle T; C, A, S, T \rangle$ consists of

- $T$: The term domain.
- $C$: A finite set of constructors: $\alpha, \beta, \gamma, \ldots$.
- $A$: A finite set of constants: $a, b, c, \ldots$. Require $A \subseteq C$.
- $S$: A finite set of selectors. $\alpha = (s_1^{\alpha}, \ldots, s_k^{\alpha})$.
- $T$: A finite set of testers. $I_{s_{\alpha}}$ for $\alpha \in C$.

- $T$ is generated **exclusively** using $C$.

- Each element of $TA$ is **uniquely** generated.
Example: LISP lists

- **Signature:**

\[ \langle \text{list}; \{\text{cons, nil}\}; \{\text{nil}\}; \{\text{car, cdr}\}; \{\text{Is}_{\text{nil}}, \text{Is}_{\text{cons}}\} \rangle \]

- **Axioms:**

\[ \text{Is}_{\text{nil}}(x) \leftrightarrow \neg \text{Is}_{\text{cons}}(x), \]
\[ x = \text{car}(\text{cons}(x, y)), \]
\[ y = \text{cdr}(\text{cons}(x, y)), \]
\[ \text{Is}_{\text{nil}}(x) \leftrightarrow \{\text{car, cdr}\}^+(x) = x, \]
\[ \text{Is}_{\text{cons}}(x) \leftrightarrow \text{cons}(\text{car}(x), \text{cdr}(x)) = x. \]
Term Algebras with Integers

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

Two-sorted language $\Sigma = \Sigma_T \cup \Sigma_Z \cup \{| \cdot |\}$:

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

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

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

$$x \neq \text{cons}(	ext{cons}(	ext{nil}, \text{nil}), \text{nil}) \land x \neq \text{cons}(\text{nil}, \text{cons}(	ext{nil}, \text{nil}))$$

is unsatisfiable with $|x| = 5$.

There are “hidden” constraints on data structure length that may contradict the integer constraints.
A formula $\Phi_\Delta(\bar{X})$ is an LCC for $\Phi_T(\bar{X}) \land \Phi_Z(\bar{X})$, if the following formulae are valid:

$$\Phi_T(\bar{X}) \land \Phi_Z(\bar{X}) \rightarrow (\exists \bar{z} : \mathbb{Z}) \left( \Phi_\Delta(\bar{z}) \land |\bar{X}| = \bar{z} \right),$$

$$\Phi_\Delta(\bar{z}) \rightarrow (\exists \bar{X} : T) \left( \Phi_T(\bar{X}) \land \Phi_Z(\bar{X}) \land |\bar{X}| = \bar{z} \right).$$

Informally,

$$\Phi_T(\bar{X}) \land \Phi_Z(\bar{X}) \; \leftrightarrow \; \Phi_\Delta(\bar{X})$$

$\Phi_\Delta(\bar{X})$ fully characterizes $\Phi_T(\bar{X}) \land \Phi_Z(\bar{X})$.

We reduce the combined constraint to the integer domain!
LCC (2)

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

$$\Phi_T(\bar{X}) \land \Phi_Z(\bar{X}) \rightarrow (\exists \bar{z} : \mathbb{Z}) \left( \Phi_\Delta(\bar{z}) \land |\bar{X}| = \bar{z} \right).$$

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

$| \cdot |$ maps a satisfying $\sigma_T$ in $T$ to a satisfying $\sigma_Z$ in PA.

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

$$\Phi_\Delta(\bar{z}) \rightarrow (\exists X : T) \left( \Phi_T(\bar{X}) \land \Phi_Z(\bar{X}) \land |\bar{X}| = \bar{z} \right)$$

$\Phi_{\Delta^-}$ is **complete**: any satisfying $\sigma_Z$ in PA is an image under $| \cdot |$ of a satisfying $\sigma_T$ in $T$. 

---

**PART I. Term Algebras with Integers**

- Previous Work on Term Algebras
- Term Algebras
- Example: LISP lists
- Term Algebras-Integers
- The Problem
- LCC

**LCC (2)**

- LCC (3)
- Example
- Main Theorem
- Generic Decision Procedure
- Computing the LCC
- LCC for Infi nite $\mathcal{A}$
- Example: LCC for Infi nite $\mathcal{A}$ (1)
- Example: LCC for Infi nite $\mathcal{A}$ (2)
- Example: LCC for Infi nite $\mathcal{A}$ (3)
- LCC for Finite Constant Domain
- Equality Completion
- Example: Equality Completion
- LCC for Finite $\mathcal{A}$
- Example: LCC for Finite $\mathcal{A}$
- Quantifier Elimination
Identify constraints with the corresponding solution set.

\( \Phi_{\Delta^+} \) is an \textit{over-approximation} of \( \Phi_{\Delta} \):

\[
\Phi_{\Delta} \subseteq \Phi_{\Delta^+}
\]

\( \Phi_{\Delta^-} \) is an \textit{under-approximation} of \( \Phi_{\Delta} \):

\[
\Phi_{\Delta^-} \subseteq \Phi_{\Delta}
\]

\( \Phi_{\Delta} \) is \textit{unique} up to equivalence:

\[
\Phi_{\Delta'} \subseteq \Phi_{\Delta} \subseteq \Phi_{\Delta'}
\]
Example: LCC

\( \Phi_T : \quad x \neq \text{cons}(\text{nil}, \text{nil}) \land y \neq \text{cons}(\text{nil}, \text{nil}) \land x \neq y \)

\( \Phi_Z : \quad |x| = |y| \)

\( \Phi_{\Delta^+} : \quad 2 \nmid |x| \land |x| = |y| \)

\( \Phi_{\Delta^-} : \quad |x| > 5 \land 2 \nmid |x| \land |x| = |y| \)

\( \Phi_\Delta : \quad |x| > 3 \land 2 \nmid |x| \land |x| = |y| \)
Main Theorem

Given $\Phi_T \land \Phi_Z$.

Let $\Phi_\Delta$ be an LCC for $\Phi_T \land \Phi_Z$. Then

$$TA_Z \models \Phi_T \land \Phi_Z \iff TA \models \Phi_T \land \Phi_\Delta.$$
Generic Decision Procedure

Input: $\Phi_T \land \Phi_Z$.

1. Return $FAIL$ if $TA \not\models \exists \Phi_T$.

2. For each partition $\Phi_T^{(i)} \land \Phi_Z^{(i)}$ of $\Phi_T \land \Phi_Z$:
   
   (a) Compute an LCC $\Phi_{\Delta}^{(i)}$ for $\Phi_T^{(i)}/\Phi_Z^{(i)}$.
   
   (b) Return $SUCCESS$ if $PA \models \exists \Phi_{\Delta}^{(i)}$.

3. Return $FAIL$.

How to compute LCC?
Computing the LCC

- **Infinite constant domain:**
  - create DAG representation of the formula.
  - Oppen’s algorithm \([\text{Opp80}]\)
  - 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. $\Phi_T \land \Phi_Z$.
2. $G_T$: the DAG of $\Phi_T$,
3. $R\upharpoonright$: the equivalence relation on $G_T$.

Initially set $\Phi_\Delta = \Phi_Z$. For each term $t$ add the following to $\Phi_\Delta$.

- $|t| = 1$, if $t$ is a constant;
- $|t| = |s|$, if $(t, s) \in R\upharpoonright$.
- $\text{Tree}(t)$ if $t$ is an untyped leaf vertex.
- $\text{Node}^\alpha(t, \mathbf{t}_\alpha)$ if $t$ is an $\alpha$-typed vertex with children $\mathbf{t}_\alpha$.
- $\text{Tree}^\alpha(t)$ if $t$ is an $\alpha$-typed leaf vertex.
Example: LCC for Infinite Constant Domain (1)

\[ \text{Is}_{\text{cons}}(y) \land x = \text{cons}(\text{car}(y), y) \land |\text{cons}(\text{car}(y), y)| < 2|\text{car}(x)|. \]

\[ n_1 : x \\
 n_2 : \text{cons}(\text{car}(y), y) \\
 n_3 : y \\
 n_4 : \text{car}(x) \\
 n_5 : \text{cdr}(x) \\
 n_6 : \text{car}(y) \\
 n_7 : \text{cdr}(y) \]
Example: LCC for Infinite Constant Domain

Equivalence relation:

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

\begin{align*}
n_1, n_2 & : \{x, \text{cons}(\text{car}(y), y)\} \\
n_3, n_5 & : \{y, \text{cdr}(x)\} \\
n_4, n_6 & : \{\text{car}(x), \text{car}(y)\} \\
n_7 & : \text{cdr}(y)
\end{align*}
Example: LCC for Infinite Constant Domain (3)

Induced length constraints:

\[ |\text{car}(x)| \geq 1 \land |\text{cdr}(x)| \geq 1 \land |\text{car}(y)| \geq 1 \land |\text{cdr}(y)| \geq 1, \]

\[ |x| = |\text{cons}(\text{car}(y), y)| \land |\text{car}(x)| = |\text{car}(y)| \land |\text{cdr}(x)| = |y|, \]

\[ |x| = |\text{car}(x)| + |\text{cdr}(x)| \land |y| = |\text{car}(y)| + |\text{cdr}(y)| \land \]

\[ |\text{cons}(\text{car}(y), y)| = |\text{car}(y)| + |y| \]

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

\[ \text{Is}_{\text{cons}}(y) \land x = \text{cons}(\text{car}(y), y) \land |\text{cons}(\text{car}(y), y)| < 2|\text{car}(x)|. \]

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 $\text{CNT}_{k,n}^\alpha(x)$ that says that there are at least $n+1$ different $\alpha$-terms of length $x$ in the structure having $k$ constants.

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

\( \Phi \) is called \textit{equality complete} if for any \( u, v \) in \( \Phi \),

- exactly one of \( u = v \) and \( u \neq v \), and
- exactly one of \( |u| = |v| \) and \( |u| \neq |v| \) are in \( \Phi \).

We say that \( x_1, \ldots, x_n \) is in a \textbf{cluster} if

\[ x_1, \ldots, x_n \text{ have the same length but pairwise unequal.} \]

Equality Completion puts terms into \textbf{stratified} clusters.
**Example: Equality Completion**

\[ x \neq z \land y \neq \text{cons}(x, z) \]

can be made equality complete by adding

\[ |y| = |\text{cons}(x, z)| \land |x| = |z|. \]

**Picture this:**

\[
\begin{align*}
|y| &= |\text{cons}(x, z)| \\
|x| &= |z|
\end{align*}
\]
**LCC for Finite Constant Domain**

**Input:**
1. $\Phi_T \land \Phi_Z$ (equality complete).
2. $G_T$: the DAG of $\Phi_T$,
3. $R\upharpoonright$: the equivalence relation on $G_T$.

Initially set $\Phi_\Delta = \Phi_Z$. For each term $t$ add the following to $\Phi_\Delta$.

- $|t| = 1$, if $t$ is a constant;
- $|t| = |s|$, if $(t, s) \in R\upharpoonright$.
- Tree$(t)$ if $t$ is an untyped leaf vertex.
- Node$(t, t_1, \ldots, t_k)$ if $t$ is a node with children $t_1, \ldots, t_k$.
- Tree$^\alpha(t)$ if $t$ is an $\alpha$-typed leaf vertex.
- $\text{Cnt}_{1,n}^\alpha(|t|)$ 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

Φ : x ≠ cons(nil, nil) ∧ |x| = 3.

implies that x and cons(nil, nil)) are in the same cluster.

Then Φ contains

CNT_{1,2}^{cons(|x|)} : |x| ⊤ 2 ∧ |x| > 3.

So Φ is unsatisfiable.
Quantifier Elimination for $\text{Th}(\text{TA}_\mathbb{Z})$

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

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

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

   (Term algebras itself are non-elementary.)

   $$(\mathcal{O}(n))^{2^{\cdots^{2^n}}}, n \geq 2$$
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]
A term is constructed **uniquely**. For example,

$$\text{cons(cons}(a, b), a)$$

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

$$aba : \begin{align*}
\ ( \ ( \ a \ ) \ b \ ) & \ a \\
\ a \ ( \ b \ ( \ a \ ) \ ) \\
\ a \ ( \ ( \ b \ ) \ a \ ) 
\end{align*}$$
Queues (1)

\[ Q : \langle Q, \mathcal{A}, C, \mathcal{S} \rangle : \]

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

**Left Insertion** \( \text{la} : \mathcal{A} \times Q \rightarrow Q \)

**Right Insertion** \( \text{ra} : \mathcal{A} \times Q \rightarrow Q \), s.t.

\[
\begin{align*}
\text{la}(a, \epsilon_Q) &= \text{ra}(a, \epsilon_Q) = \langle a \rangle, \\
\text{la}(a, \langle s_1, \ldots, s_n \rangle) &= \langle a, s_1, \ldots, s_n \rangle, \\
\text{ra}(a, \langle s_1, \ldots, s_n \rangle) &= \langle s_1, \ldots, s_n, a \rangle.
\end{align*}
\]
4 \( S \): Selectors:

- **Left Head** \( lh : Q \rightarrow A \), **Left Tail** \( lt : Q \rightarrow Q \),
- **Right Head** \( rh : Q \rightarrow A \), **Right Tail** \( rt : Q \rightarrow Q \), s.t.

\[
\begin{align*}
lh(\langle s_1, \ldots, s_n \rangle) &= s_1, \\
lt(\langle s_1, \ldots, s_n \rangle) &= \langle s_2, \ldots, s_n \rangle, \\
rh(\langle s_1, \ldots, s_n \rangle) &= s_n, \\
rt(\langle s_1, \ldots, s_n \rangle) &= \langle s_1, \ldots, s_{n-1} \rangle.
\end{align*}
\]

**Convention:** use *concatenation operator* \( \circ \).

\[ a \circ X \circ b \quad \text{stands for} \quad ra(b, la(a, X)) \quad \text{or} \quad la(a, ra(b, X)). \]
Input: $\Phi \equiv E \cup D$.

1. Normalize $\Phi$ to $\Phi' : E' \cup D'$.

2. Return $FAIL$, if inconsistency is discovered;

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

A queue constraint $\Phi_Q$ is in **normal form** if

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

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

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

1. \( \Sigma_Q \): signature of queues.
2. \( \Sigma_\mathbb{Z} \): signature of Presburger arithmetic.
3. \( | \cdot | : \mathcal{Q} \to \mathbb{N} \), the length function.
Problem I

The presence of $\Phi_Z$ restricts solutions to $\Phi_Q$.

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

$$\Phi_Q : \ Xba \neq abY \land Xab \neq baY \land Xaa \neq baY \land Xab \neq aaY$$

is not satisfiable with $\Phi_Z : |X| = |Y| = 1$. 
Problem I

The presence of $\Phi_Z$ restricts solutions to $\Phi_Q$.

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

$$\Phi_Q : Xba \neq abY \land Xab \neq baY \land Xaa \neq baY \land Xab \neq aaY$$

is not satisfiable with $\Phi_Z : |X| = |Y| = 1$.

**Computing LCC.**

**Example:**

$$\Phi_Z : |X| = |Y| \quad \Phi_\Delta : |X| \neq 1 \land |X| = |Y|$$
Problem I

The presence of $\Phi_Z$ restricts solutions to $\Phi_Q$.

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

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

is not satisfiable with $\Phi_Z : |X| = |Y| = 1$.

Computing LCC.

**Example:**

$$\Phi_Z \quad \quad \Phi_\Delta$$

$$|X| = |Y| \quad \quad |X| \neq 1 \land |X| = |Y|$$

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

We cannot partition terms into \textit{stratified clusters} and construct a satisfying assignment inductively.

\textbf{Example:} Consider

\[ X \neq Y \land aX \neq Yb \land Xa \neq bY \]

Infinitely many assignments of the form

\[ X = (ba)^nb, \quad Y = a(ba)^n \]

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

\[ X \neq Y \land aX \neq Yb \land Xa \neq bY \]

Infinitely many assignments of the form

\[ X = (ba)^n b, \quad Y = a(ba)^n \]

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

Find a *cut length*!
Cut Length

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

$$\mathcal{Q}_Z \models \exists \Phi_Q \wedge \bigwedge_{i=1}^{n} |X_i| = \mu_i$$

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 $\sigma$ such that $|\sigma(X)| = |\sigma(Y)| = 1.$
Computation of Cut Length

\[ \text{PRE}_\Phi : \text{the set of all words } \alpha \text{ s.t. } \alpha X \text{ or } \alpha \text{ is a proper term in } \Phi_Q. \]

\[ d_\Phi : \text{the shortest strongly primitive word } d \text{ such that } (\forall \alpha \in \text{PRE}_\Phi) \ d \not\in \text{orb}(\alpha). \]

\[ L_d : \text{the length of } d_\Phi. \]

\[ L_c : \text{the smallest number of letters to create a unique identifying word, called a } \textcolor{red}{\text{color}}, \text{ for each queue variable in } \Phi_Q. \]

\[ L_t : L_c + L_d. \]

We claim that \( L_t \geq \delta. \)
Example

Consider

\[ Xba \neq abY \land Xab \neq baY \land Xaa \neq baY \land Xab \neq aaY \]

Then

1. \( \text{PRE}_\Phi = \{ab, ba, aa\} \).
2. \( L_d = 3; \ d_\Phi = aab \).
3. \( L_c = 1; \ \Phi_Q \) includes two queue variables.

So \( L_t = L_c + L_d = 4 \).
Computation of LCC for Queues

Input: $\Phi_Q \land \Phi_Z$ in normal form of $\mathbb{Q}_Z$.

Initially set $\Phi_\Delta = \Phi_Z$. Add to $\Phi_\Delta$:

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

Thank You!
\( \Phi_Q \) is in *normal form* in \( \mathbb{Q}_\mathbb{Z} \) if

1. \( \Phi_Q \) is in normal form in \( \mathbb{Q} \);
2. \( \Phi_Q \) is equality complete;
3. if \( \alpha X \neq Y \beta \) occurs with either \( X \in \text{orb}(\alpha', k) \) or \( Y \in \text{orb}(\beta', l) \), then \( \alpha \equiv \epsilon_Q \);
4. \( \alpha X \neq Y \beta \) does not occur with both \( X \in \text{orb}(\alpha', k) \) and \( Y \in \text{orb}(\beta', l) \).
New Normalization. To deal with parameters $\bar{Y}$.

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

$$\left( \exists x_1, \ldots, \exists x_n \right) \Phi(x_1, \ldots, x_n, y_1, \ldots, y_m) \iff \Phi'(y_1, \ldots, y_m)$$
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:

\[
\frac{A \lor C}{(C \lor C')\sigma} \quad \frac{\neg A' \lor C'}{\sigma = \text{mgu}(A, A')} \\
\forall B \in C \sigma \lor C' \sigma (A \sigma \not< B \sigma)
\]

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

\[ L \rightarrow R \quad (L \sigma > R \sigma) \]

How to decide satisfiability of order constraints?
### Background: Previous Work (1)

Two types of widely used orderings:

<table>
<thead>
<tr>
<th>Syntactic Nature</th>
<th>Hybrid Nature</th>
</tr>
</thead>
<tbody>
<tr>
<td>syntactic precedence</td>
<td>√</td>
</tr>
<tr>
<td>lexicographical ordering</td>
<td>√</td>
</tr>
<tr>
<td>numerical ordering</td>
<td></td>
</tr>
</tbody>
</table>
**Background: Previous Work (2)**

Decidability Status:

<table>
<thead>
<tr>
<th></th>
<th>LPO</th>
<th>KBO</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>QFT</strong></td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>[Com90] [Nie93]</td>
<td>[KV00] [KV01]</td>
</tr>
<tr>
<td><strong>UQT</strong></td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>[NR00]</td>
<td></td>
</tr>
<tr>
<td><strong>GQT</strong></td>
<td>✗</td>
<td>?</td>
</tr>
<tr>
<td></td>
<td>[Tre92, CT97]</td>
<td></td>
</tr>
</tbody>
</table>

QFT: Quantifier-free Theory.

UQT: Unary Quantified Theory.

GQT: General Quantified Theory.
### Background: Previous Work (2)

#### Decidability Status:

<table>
<thead>
<tr>
<th>Theory</th>
<th>LPO</th>
<th>KBO</th>
</tr>
</thead>
<tbody>
<tr>
<td>QFT</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>[Com90]</td>
<td>[Nie93]</td>
<td>[KV00]</td>
</tr>
<tr>
<td>UQT</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>[NR00]</td>
<td></td>
<td>[KV02]</td>
</tr>
<tr>
<td>GQT</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>[Tre92]</td>
<td>[CT97]</td>
<td>[ZSM05]</td>
</tr>
</tbody>
</table>

**QFT**: Quantifier-free Theory.

**UQT**: Unary Quantified Theory.

**GQT**: General Quantified Theory.
Knuth-Bendix Order (1)

A Knuth-Bendix order (KBO) \( \prec_{\text{kb}} \) is parametrically defined with

- \( W : \text{TA} \rightarrow \mathbb{N} \): a weight function satisfying

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

- \( \preceq : \) a linear (precedence) order on \( C \) such that

\[
\alpha_1 \preceq \alpha_2 \preceq \ldots \preceq \alpha_{|C|}.
\]
Knuth-Bendix Order (2)

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

1. $W(u) < W(v)$.
2. $W(u) = W(v)$ and $\text{type}(u) <^\Sigma \text{type}(v)$.
3. $W(u) = W(v)$, $u \equiv \alpha(u_1, \ldots, u_k)$, $v \equiv \alpha(v_1, \ldots, v_k)$, and

$$\exists i \left[ 1 \leq i \leq k \land u_i <^{kb} v_i \land \forall j(1 \leq j < i \rightarrow u_j = v_j) \right].$$
Quantifier Elimination

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

$$\exists \bar{x} \left[ A_1(\bar{x}) \land \ldots \land A_n(\bar{x}) \right],$$

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

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

$$\exists x \left[ x = t \land \varphi(x, \bar{y}) \right] \leftrightarrow \varphi(t, \bar{y}).$$
Main Idea: Depth Reduction

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

$$\text{depth}_\varphi(x) = 0.$$  

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

$$\left( \text{depth}_\varphi(x): \text{the length of the longest selector sequence in front of } x \text{ in } \varphi. \right)$$
Solved Form

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

\[
\bigwedge_{i \leq m} u_i <_{\text{kb}} x \land \bigwedge_{j \leq n} x <_{\text{kb}} v_j \land \varphi'(\bar{y}),
\]

where \( x \) does not appear in \( u_i, v_i \) and \( \varphi' \).

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

\[
\bigwedge_{i \leq m, j \leq n} u_i <_{\text{kb}} v_j \land \varphi'(\bar{y})
\]

where \( x <_{\text{kb}} y \), 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, $\exists x\varphi(x, \bar{y})$ goes to

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

where

$$\varphi'(x_1, \ldots, x_k, \bar{y}) \equiv \varphi(x, \bar{y}) \left[ s_i^\alpha(x) \leftarrow x_i \right].$$
Case 1: Example

\[(\exists x) \left[ \text{car}(x) <^{\text{kb}} \text{cdr}(x) \right] \]

\[\Rightarrow (\exists x_1)(\exists x_2)(\exists x) \left[ x_1 = \text{car}(x) \land x_2 = \text{cdr}(x) \land \text{car}(x) <^{\text{kb}} \text{cdr}(x) \right] \]

\((\text{decompose } x)\)

\[\Rightarrow (\exists x_1)(\exists x_2)(\exists x) \left[ x_1 = \text{car}(x) \land x_2 = \text{cdr}(x) \land x_1 <^{\text{kb}} x_2 \right] \]

\((\text{substitution})\)

\[\Rightarrow (\exists x_1)(\exists x_2) \left[ x_1 <^{\text{kb}} x_2 \right] \]

\((\text{remove } x)\)
**Depth Reduction: Case 2**

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

- Decompose 0-depth occurrences of $x$ in terms of $s_1^\alpha(x), \ldots, s_k^\alpha(x)$.

- This amounts to expressing $x <_{^\text{n}}^{^\text{kb}} t$ and $t <_{^\text{n}}^{^\text{kb}} x$ using $s_1^\alpha(x), \ldots, s_k^\alpha(x)$.

- Then apply the reduction as in Case 1!
Case 2: Example

\[(\exists x) \left[ \text{car}(x) <^\text{kb} y \land y <^\text{kb} x \right] \]

\[\Rightarrow (\exists x_1)(\exists x_2)(\exists x) \left[ x_1 = \text{car}(x) \land x_2 = \text{cdr}(x) \right. \]
\[\left. \land \text{car}(x) <^\text{kb} y \land y <^\text{kb} x \right] \]

\[(\text{decompose } x)\]

\[\Rightarrow (\exists x_1)(\exists x_2)(\exists x) \left[ x_1 = \text{car}(x) \land x_2 = \text{cdr}(x) \land \text{car}(x) <^\text{kb} y \right. \]
\[\left. \land \text{car}(y) = \text{car}(x) \land \text{cdr}(y) <^\text{kb} \text{cdr}(x) \right] \]

\[(\text{decompose } y <^\text{kb} x; \text{ reduce to case } 1)\]
Case 2: Example (Cont'd)

\[ (\exists x_1)(\exists x_2)(\exists x) \left[ x_1 = \text{car}(x) \land x_2 = \text{cdr}(x) \land x_1 <_{\text{kb}}^\text{y} \land \text{car}(y) = x_1 \land \text{cdr}(y) <_{\text{kb}}^x x_2 \right] \]

(substitution)

\[ \Rightarrow (\exists x_1)(\exists x_2) \left[ x_1 <_{\text{kb}}^\text{y} \land \text{car}(y) = x_1 \land \text{cdr}(y) <_{\text{kb}}^x x_2 \right] \]

(remove x)
Quantifier Elimination for Knuth-Bendix Order

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

While \(\bar{x} \neq \emptyset\).

- While \((\forall x \in \bar{x}) depth_\varphi(x) > 0\).
  
  Depth Reduction.
  
  - **VARIABLE SELECTION.**
  
  - **DECOMPOSITION.**
  
  - **SIMPLIFICATION.**

  Done.

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

  Elimination.

  Done.

Done.
Variable Selection

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

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

☞ i.e., choose variables generated in the previous round.
Decomposition

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

$$\exists x_1 \ldots \exists x_k \exists \bar{x} \left[ I_{\alpha}(x) \land \bigwedge_{1 \leq i \leq k} s^\alpha_i(x) = x_i \land \varphi(\bar{x}, \bar{y}) \right].$$
Simplification

Apply the following rules to each occurrence of $x$.

1. Replace $x <^n t$ (or $t <^n x$) by a quantifier-free formula

$$\varphi'(s_1^\alpha(x), \ldots, s_k^\alpha(x), s_1^\alpha(t), \ldots, s_k^\alpha(t)).$$

2. Replace $s_i^\alpha(x)$ in $\varphi(\bar{x}, \bar{y})$ by $x_i$ ($1 \leq i \leq k$).

Denote the result of this simplification by

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

- We have

\[ \exists x \left[ \bigwedge_{i \leq m} u_i \stackrel{\text{kb}}{<} x \land \bigwedge_{j \leq n} x \stackrel{\text{kb}}{<} v_j \land \varphi'(\bar{y}) \right], \]

where \( x \) appears none of \( u_i, v_j \) and \( \varphi' \).

- Guessing a gap order completion, we rewrite it to

\[ u_{i'} \stackrel{\text{kb}}{<} v_{j'} \land \varphi'(\bar{y}) \land \text{“} u_{i'} \text{ is the greatest of } \{ u_i \mid i \leq m \} \text{”} \]

\[ \land \text{“} v_{j'} \text{ is the smallest of } \{ v_j \mid j \leq n \} \text{”}. \]
Technical Challenges (1)

1. Decompose $<^{\text{kb}}$ into three disjoint suborders $<^{\text{w}}$, $<^{\text{p}}$ and $<^{\text{l}}$.

2. Extend $<^{\text{w}}$, $<^{\text{p}}$ and $<^{\text{l}}$ to $<_{n}^{\text{w}}$, $<_{n}^{\text{p}}$ and $<_{n}^{\text{l}}$, 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.

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

6. Extend all aforementioned notions to tuples of terms.
Technical Challenges (2)

■ Elimination of Complex Terms.

\[ \text{car}(0^w_{\text{((car)(x))w}}) \]

■ Elimination of Integer Quantifiers.

\[ (\exists z : \mathbb{Z}) [ \text{car}(0^w_z) <^{kb} \text{cdr}(0^w_z) ] \]

■ Elimination of Equalities.

\[ (\exists x) [ x = 0^w_{\text{((car)(x))w}} ) \wedge \text{car}(x) <^p_4 \text{cdr}(x) ] \]

■ Elimination of Negations.

\[ \neg (\text{car}(x) <^w_3 \text{cdr}(x)) \]

■ 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, $\text{prefix} \leq_p$, $\text{subqueue} \leq$ and $\text{suffix} \leq_s$
Future Work (1)

- Implementation and experimentation.

- More expressive languages.
  - Term algebras with subterm relation
  - Queues with subsequence relations, namely, prefix $\leq_p$, subqueue $\leq$ and suffix $\leq_s$

  With our decision procedures for

  $\mathbb{Q}_Z + \leq_p + \leq$ and $\mathbb{Q}_Z + \leq_s + \leq$,

  the next step is $\mathbb{Q}_Z + \leq_p + \leq_s$!
Future Work (2)

$\mathbb{Q}_Z + \leq_p + \leq_s$ is a very expressive theory.

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

   \[ uv^2 = vuv \land |u| < |v| \]

2. Interpret the theory of arrays.

   \[ q[i] = a \leftrightarrow \exists p \ (pa \leq_p q \land |pa| = i) \]

3. Interpret Presburger arithmetic with divisibility predicate.

   \[ x = y + 2 \land y \mid x \]


   \[ u \oplus v = w \land uv = ww \]
Thank You!


[BS88] J. Richard Büchi and Steen Senger. Definability in the existential theory of concatenation and undecidable extensions of this


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


A. I. Mal’cev. Axiomatizable classes of locally free algebras of various types. In *The


