Little Engines of Proof

N. Shankar, L. de Moura, H. Ruess, A. Tiwari

shankar@csl.sri.com

URL: http://www.csl.sri.com/~shankar/LEP.html

Computer Science Laboratory

SRI International

Menlo Park, CA

In Today's Lecture

I Equality over constants, no boolean structure:

II Equality over constants, with boolean structure

III Equality over ground terms, no boolean structure: Special Strategies

IV Equality over ground terms, with boolean structure

V Equality over ground terms, with MORE "special" function symbols

A Note About Equational Theories

E: set of equations (ground or nonground)

$T(\Sigma)$: set of all ground terms over Σ

$T(\Sigma)/\leftrightarrow_E^*$: equivalence classes modulo $\leftrightarrow_E^*$

Initial Model of E: $(\frac{T(\Sigma)}{\leftrightarrow_E^*}, I)$ s.t. $I(f([s_1], \ldots, [s_k]) = [f s_1 \ldots s_k]$

$E \models (s = t)$ iff $(\frac{T(\Sigma)}{\leftrightarrow_E^*}, I) \models s = t$ iff $E \vdash s = t$

Satisfiability procedure rules: enable to compute over the initial model

Completeness: The final state can be used to read off the initial model

III. Abstract Congruence Closure: Strategies

Some popular congruence closure algorithms are

- Downey-Sethi-Tarjan (DST)
- Nelson-Oppen (NO)
- Shostak (Sho)

DST and Sho can be described as specific strategies over the ACC inference rules

NO uses a slightly different deduction mechanism

III. Abstract Congruence Closure: Shostak

Dynamic congruence closure algorithm:

$[(\mathbf{Sim}^*; \mathbf{Ext}^?)^*; (\mathbf{Del} \cup \mathbf{Ori}); (\mathbf{Col}; \mathbf{Sup}^*)^*]^*$

General principle: Eager simplification

New equations can be added at any time

How to identify where $\mathbf{Col}$ and then $\mathbf{Sup}$ are applied? Use additional indexing mechanisms

Called use lists by Shostak

Shostak did not use the $n\log(n)$ trick

III. Shostak's Congruence Closure: Example

$a = fab,\ f(fab)b = b$	
$c_1 = fab,\ f(fab)b = b$	$a \to c_1$
$c_1 = fc_1b,\ f(fab)b = b$	$a \to c_1$
$c_1 = fc_1c_2,\ f(fab)b = b$	$a \to c_1, b \to c_2$
$c_1 = c_3,\ f(fab)b = b$	$a \to c_1, b \to c_2, fc_1c_2 \to c_3$
$c_1 \to c_3,\ f(fab)b = b$	$a \to c_1, b \to c_2, fc_1c_2 \to c_3$
$c_1 \to c_3,\ f(fab)b = b$	$a \to c_1, b \to c_2, fc_3c_2 \to c_3$
$\vdots$	$\vdots$

III. Shostak's Congruence Closure: Example

$c_1 \to c_3,\ f(fab)b = b$	$a \to c_1, b \to c_2, fc_3c_2 \to c_3$
$c_1 \to c_3,\ f(fc_3c_2)b = b$	$a \to c_1, b \to c_2, fc_3c_2 \to c_3$
$c_1 \to c_3,\ fc_3b = b$	$a \to c_1, b \to c_2, fc_3c_2 \to c_3$
$c_1 \to c_3,\ fc_3c_2 = c_2$	$a \to c_1, b \to c_2, fc_3c_2 \to c_3$
$c_1 \to c_3,\ c_3 = c_2$	$a \to c_1, b \to c_2, fc_3c_2 \to c_3$
$c_1 \to c_3,\ c_3 \to c_2$	$a \to c_1, b \to c_2, fc_3c_2 \to c_3$
$c_1 \to c_3,\ c_3 \to c_2$	$a \to c_1, b \to c_2, fc_2c_2 \to c_3$

Note: We needed only 3 constants, c_1, c_2, c_3

Note: Compose is not used.

III. Abstract Congruence Closure: DST

Downey, Sethi, and Tarjan's congruence closure algorithm:

$[(\mathbf{Col}; (\mathbf{Sup} \cup \{\epsilon\}))^*; (\mathbf{Sim}^*; (\mathbf{Del} \cup \mathbf{Ori}))^*]^*$

Offline algorithm, all input equations are preprocessed into DAG form

Data structures (signature-table) for effective application of these rules

First described the $n\log(n)$ trick

III. Downey-Sethi-Tarjan Congruence Closure: Example

$a = fab,\ f(fab)b = b$	
$c_1 = c_3, c_4 = c_2$	$a \to c_1, b \to c_2, fc_1c_2 \to c_3, fc_3c_2 \to c_4$
$c_1 \to c_3, c_4 = c_2$	$a \to c_1, b \to c_2, fc_1c_2 \to c_3, fc_3c_2 \to c_4$
$c_1 \to c_3, c_4 = c_2$	$a \to c_1, b \to c_2, fc_3c_2 \to c_3, fc_3c_2 \to c_4$
$c_1 \to c_3, c_4 = c_2, c_3 = c_4$	$a \to c_1, b \to c_2, fc_3c_2 \to c_4$
$c_1 \to c_3, c_4 \to c_2, c_3 = c_4$	$a \to c_1, b \to c_2, fc_3c_2 \to c_4$
$c_1 \to c_3, c_4 \to c_2, c_3 = c_2$	$a \to c_1, b \to c_2, fc_3c_2 \to c_4$
$c_1 \to c_3, c_4 \to c_2, c_3 \to c_2$	$a \to c_1, b \to c_2, fc_3c_2 \to c_4$
$c_1 \to c_3, c_4 \to c_2, c_3 \to c_2$	$a \to c_1, b \to c_2, fc_2c_2 \to c_4$

III. Abstract Congruence Closure: NO

$[(\mathbf{Sim}^*; (\mathbf{Ori} \cup \mathbf{Del}); \mathbf{NOSup}^*]^*$

NOSup: modified rule for superposing modulo C-rules

$$\text{NOSup} \quad \frac{fc_1 \ldots c_k \to c, fd_1 \ldots d_k \to d, \Gamma}{fc_1 \ldots c_k \to c, fd_1 \ldots d_k \to d, c = d, \Gamma} \quad \text{if } c_i \leftrightarrow^*_\Gamma d_i$$

Motivation: Original DAG not modified (cf. not changing clause database in Davis-Putnam)

Quadratic time

How to avoid redundant inferences?

III. Nelson-Oppen Congruence Closure: Example

$a = fab,\ f(fab)b = b$	
$c_1 = c_3, c_4 = c_2$	$a \to c_1, b \to c_2, fc_1c_2 \to c_3, fc_3c_2 \to c_4$
$c_1 \to c_3, c_4 = c_2$	$a \to c_1, b \to c_2, fc_1c_2 \to c_3, fc_3c_2 \to c_4$
$c_1 \to c_3, c_4 = c_2$	$a \to c_1, b \to c_2, fc_1c_2 \to c_3, fc_3c_2 \to c_4$
$c_1 \to c_3, c_4 = c_2, c_3 = c_4$	$a \to c_1, b \to c_2, fc_1c_2 \to c_3, fc_3c_2 \to c_4$
$c_1 \to c_3, c_4 \to c_2, c_3 = c_4$	$a \to c_1, b \to c_2, fc_1c_2 \to c_3, fc_3c_2 \to c_4$
$c_1 \to c_3, c_4 \to c_2, c_3 = c_2$	$a \to c_1, b \to c_2, fc_1c_2 \to c_3, fc_3c_2 \to c_4$
$c_1 \to c_3, c_4 \to c_2, c_3 \to c_2$	$a \to c_1, b \to c_2, fc_1c_2 \to c_3, fc_3c_2 \to c_4$

Note: Term DAG does not change

Note: Nontrivial NODed rule tested after each Orient

Note: Do not get a canonizer in the end

III. Abstract Congruence Closure: Exercises

Ex: Show that E induces finite equivalence classes iff corresponding congruence-closed DAG is acyclic

Ex: Design a linear time algorithm for computing congruence closure for sets E that induce finite equivalence classes

Ex: Develop a method to eliminate all the new constants from the final state so that the resulting rewrite system over the original signature is terminating and confluent?

Ex: Design a correct inference system for conjunctions of equations and disequations over ground terms containing a commutative function symbol

Ex: R is locally confluent if $\leftarrow_R \circ \to_R \subseteq \to^*_R \circ \leftarrow^*_R$. Prove that termination and local confluence implies confluence.

IV. Ground Equality: With Boolean Structure

$\phi : ((l_1 \vee l_2 \vee \cdots \vee l_k) \wedge (\cdots \vee \cdots) \wedge \cdots)$

Method 1: Convert ϕ to CNF + Congruence-Closure

Method 2: Transform to the "equality on constants" case

Ackerman Transformation:

Extend	$\dfrac{\phi[fc_1 \ldots c_k], D}{\phi[c], D \cup \{fc_1 \ldots c_k = c\}}$	if c is new
ElimD	$\dfrac{\phi, D \cup \{fc_1 \ldots c_k = c, fd_1 \ldots d_k = d\}}{\phi \wedge (c_1 = d_1 \wedge \cdots \wedge c_k = d_k \Rightarrow c = d), D \cup \{\ldots\}}$	
Terminate	$\dfrac{\phi, D}{\phi, \emptyset}$	if all ElimD inferences are redundant

Ex: Any FAIR derivation using above rules terminates.

IV. Ground Equality: With Boolean Structure

Exercises:

Ex: Show that there exists a finite set of sufficient interpretation for "Ground equations and disequations with boolean structure"

Ex: Translate to propositional SAT using the above result

Define: A clause is horn if it has atmost one positive literal

Ex: Show that if all clauses in ϕ are horn, then satisfiability of ϕ can be efficiently decided

Define: A clause is nhorn if it has atmost one negative literal

Ex: Show that if all clauses in ϕ are nhorn, then satisfiability of ϕ can be efficiently decided

IV. Ground Equality: With Boolean Structure

Recall II.Method4: Lifted Ordered-Transitive-Closure to Clauses

The same calculus works here too: Basic Superposition:

Superpose Right	$\dfrac{\underline{s \to t} \vee C,\ \underline{w[s] \to u} \vee D,\ \Gamma}{w[t] = u \vee C \vee D, \ldots}$	if $w = u \succ s = t$
Superpose Left	$\dfrac{\underline{s \to t} \vee C,\ w[s] \neq u \vee D,\ \Gamma}{w[t] \neq u \vee C \vee D, \ldots}$	if $w \succ u, w \neq u \succeq Max(D)$
EqResolution	$\dfrac{s \neq s \vee C, \Gamma}{C, \Gamma}$	if $s \neq s \succ Max(D)$
EqFactoring	$\dfrac{\underline{s \to t} \vee s \to u \vee C, \Gamma}{t \neq u \vee s \to u \vee C, \ldots}$	if $t \succeq u$

Note: $\perp$ is the empty clause

IV. Superposition Calculi: Remarks

Ordering: $\succ$ is a total reduction ordering on terms

Define: $Measure(s = t) = \{\{s\}, \{t\}\}$; $Measure(s \neq t) = \{\{s, t\}\}$

Literal Ordering: $L_1 \succ L_2$ iff $Measure(L_1) \succ^m Measure(L_2)$

Clause Ordering: multiset extension of the ordering on literals

Notation: $s \to t$ means $s = t$ and $s \succ t$

Notation: $\underline{s = t} \vee C$ means $s = t \succ C$

Ex: Show that the inference system is sound.

IV. Superposition Calculi: Completeness

Suppose clause set is unsatisfiable, but final state Γ is not $\bot$

Initial Model $M_0 = (T(\Sigma), I)$, I maps f to syntactic f

Clauses in Γ would be false in M_0. We will fix M_0

In each step: Current Model = M

1. Pick the least false clause $\underline{s \to t} \vee C$ in Γ,
2. Set $s = t$ in M to get new M

Claim: The final model M is a model for Γ

IV. Superposition Calculi: Completeness

Each successive model M: $\frac{T(\Sigma)}{\leftrightarrow_E^*}$ for some E

Monotonically adding equations in E (governed by $\succ$)

Imagine M is represented by a convergent rewrite system R s.t. $s \to t$ iff t is the equivalence class representative for s

IV. Superposition Calculi: Problem1

Problem: We can make smaller clauses false as we proceed.

Example: $\Gamma = \{b \neq c,\ a = c,\ a = b\}$, with $a \succ b \succ c$

- First you make $a = c$ in the model and then $a = b$
- But you have now contradicted a smaller clause $b \neq c$

Problem: We picked $\underline{s \to t} \vee C$ from Γ, but s was already assigned in a previous round

Solution: Γ is saturated under Superpose Right

Example: $a = c$ and $a = b$ means we also have $b = c$ in Γ

If $\underline{s \to t} \vee C$ is selected from Γ, then s is in its own equivalence class, $\therefore$ free to be assigned

IV. Superposition Calculi: Problem2

Problem: We may make a larger clause unsatisfiable

Example: $\Gamma = \{b = c, a = c, a \neq b\}$

- After making $b = c$ and $a = c$, we can't claim that $a \neq b$

Solution: Γ is saturated under Superpose Left

Example: $a = c$ and $a \neq b$ means we also have $c \neq b$ in Γ

When $\underline{s \to t} \vee C$ is selected from Γ, all facts about equivalence classes of terms smaller than s has been asserted

IV. Superposition Calculi: Problem3

The above informal argument can be formalized using

- a refined definition of iterative candidate model construction
- so that if there is a clause which is false in a candidate model
- then using BS we can get a smaller clause which is also false

When completeness proof is formalized this way, we also need EqFactoring

Example: $\Gamma = \{b = c, a = b \vee a = c, a \neq b \vee a \neq c\}$

Candidate model: $\{b = c\}$

Minimal counter example: $a = b \vee a = c$

Reduced counter example: $b \neq c \vee a = c$

Candidate model: $\{b = c, a = b\}$

Minimal counter example: $a \neq b \vee a \neq c$

Reduced "counter example": $b \neq b \vee a \neq c \vee a = c$

Modified Candidate model construction:
In each step: Current Model $= M$

1. Pick the least false clause $\underline{s \to t} \vee C$ in Γ,
2. C is a also false in $M \cup \{s = t\}$

IV. Superposition Calculi: Example

Candidate Model: In blue

Minimal Conflict Clause: In red

$$\begin{array}{c}
C1 : a \neq b \vee a \neq c,\ C2 : a = b \vee a = c,\ C3 : b = c \\ \hline
C1,\ C2,\ C4 : b \neq c \vee a = c,\ C3 \\ \hline
C1,\ C5 : c \neq b \vee a \neq c \vee b \neq c,\ C2,\ C4,\ C3 \\ \hline
C1,\ C5,\ C6 : c \neq b \vee c \neq c \vee b \neq c \vee b \neq c,\ C2,\ C4,\ C3 \\ \hline
C1,\ C5,\ C6 : c \neq c \vee c \neq c \vee c \neq c \vee c \neq c,\ C2,\ C4,\ C3 \\ \hline
C1,\ C5,\ C6 : \mathit{false},\ C2,\ C4,\ C3 \\ \hline
\perp
\end{array}$$

Ex: Derive $\perp$ without using EqFactoring, but allowing for tautologies.