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

Equality

Logical symbols: $\wedge$, $\vee$, $\neg$, $\Rightarrow$, $\Leftrightarrow$

Another important logical symbol: $=$

Equality over what? Terms.

Variables: $x, y, z, \ldots$

Constants: $a, b, c, \ldots$

Function symbols: $f, g, h, \ldots$

Example of terms: $a, f(a), g(a, f(b)), \ldots$

Terms: $s, t, u, \ldots$

Atomic formulas: $s = t$, $s \neq t$

Notation

Formulas: $a = b \wedge fa = c \Rightarrow fb = c$

Context: A term with holes

Example: $f(\_)$, $h(\_, f(\_), a)$

Notation: $C[\_]$, $C[\_, \_]$

Substitution: mapping from variables to terms

Example: $x \mapsto a, y \mapsto f(a), \ldots$

Notation: $t\sigma$ denotes term obtained by replacing x by $x\sigma$ in t

Example: $f(x)\sigma = f(a)$

The axioms of equality

Reflexivity, Symmetry, Transitivity, and closed under substitution and congruence.

Reflexivity $\dfrac{}{s = s}$	Symmetry $\dfrac{s = t}{t = s}$
Transitivity $\dfrac{s = t, t = u}{s = u}$	Substitution $\dfrac{s = t}{s\sigma = t\sigma}$
Congruence $\dfrac{s = t}{C[s] = C[t]}$	

Classes of Problems

0 Transitive relation over constants, no boolean structure: Atomic formulas are of the form $c > d$

$$\exists \vec{x}.\ (l_1 \wedge l_2 \wedge \ldots \wedge l_n \wedge \neg l'_1 \wedge \neg l'_2 \wedge \ldots \wedge \neg l'_m)$$

I Equality over constants, no boolean structure: Atomic formulas are of the form $x_i = x_j,\ x_i = a,\ a = b$

II Equality over constants, with boolean structure

$$(\ldots \vee \ldots) \wedge (\ldots \vee \ldots) \wedge \cdots$$

III Equality over ground terms, no boolean structure: Atomic formulas are of the form $s = t$

IV Equality over ground terms, with boolean structure

Satisfiability of Quantifier-free Formulas

Validity of

$$\forall \vec{x}.\ [l_1 \wedge l_2 \wedge \ldots \wedge l_n \Rightarrow (l'_1 \vee l'_2 \vee \ldots \vee l'_m)]$$

Satisfiability of

$$\exists \vec{x}.\ (l_1 \wedge l_2 \wedge \ldots \wedge l_n \wedge \neg l'_1 \wedge \neg l'_2 \wedge \ldots \wedge \neg l'_m)$$

No boolean complexity

No quantifiers (existentially quantified variables can be treated as constants)

0. Transitive Relation over Constants

Each l_i is either $c_i > c_j$ or $\neg(c_i > c_j)$

How to decide if

$$l_1 \wedge l_2 \wedge \ldots \wedge l_n$$

is satisfiable?

Configuration: Set of positive and negative literals

Goal: Derive $\perp$ if the conjunction is unsatisfiable

Transitive-closure procedure:

Transitivity	$\dfrac{c_1 > c_2, c_2 > c_3, \Gamma}{c_1 > c_2, c_2 > c_3, c_1 > c_3, \Gamma}$
Contradiction	$\dfrac{c_1 > c_2, \neg(c_1 > c_2), \Gamma}{\perp}$

0. Transitive Closure

Transitivity rule: computes closure under transitivity

Transitive closure computation is well studied problem in Algorithms

Warshall's algorithm applies Transitivity in a particular strategy

Boolean matrix multiplication yields yet another strategy

Optimization: Based on identifying strongly connected components, i.e., whenever $c > c$ is generated

Reflexive-Transitive Closure: Additionally,

Contradiction	$\dfrac{\neg(c > c), \Gamma}{\perp, \Gamma}$

I. Equality over Constants

No function symbols, only skolem constants

Each l_i is either $c_i = c_j$ or $c_i \neq c_j$

In other words, we just need to deal with reflexivity, symmetry, and transitivity

How to decide if

$$l_1 \wedge l_2 \wedge \ldots \wedge l_n$$

is satisfiable?

Configuration: Set of equations and disequations

Goal: Derive $\perp$ if the conjunction is unsatisfiable

I. Equality over Constants

Reflexive-Symmetric-Transitive-closure procedure:

Transitivity $\dfrac{c_1 = c_2, c_2 = c_3, \Gamma}{c_1 = c_2, c_2 = c_3, c_1 = c_3, \Gamma}$	
Contradiction $\dfrac{c_1 = c_2, c_1 \neq c_2, \Gamma}{\perp}$	
Contradiction $\dfrac{c \neq c, \Gamma}{\perp}$	

Equations are treated symmetrically when using these inference rules.

Time Complexity: $O(n^3)$

I. Efficient Variant: Ordered Transitive-closure

Let $\succ$ be an ordering on the constants.

Say, $c_1 \succ c_2 \succ \cdots \succ c_n$.

Notation: $c \to d$ means $c = d$ and $c \succ d$.

Orient $\dfrac{c = d, \Gamma}{c \to d, \Gamma}$ if $c \succ d$	Delete $\dfrac{c = c, \Gamma}{\Gamma}$
Simplify $\dfrac{c = d, c \to d', \Gamma}{d' = d, c \to d', \Gamma}$	Simplify $\dfrac{c \neq d, c \to d', \Gamma}{d' \neq d, c \to d', \Gamma}$
Collapse $\dfrac{c \to d', c \to d, \Gamma}{c \to d', d \to d', \Gamma}$ if $d \succ d'$	
Compose $\dfrac{c \to d, d \to d', \Gamma}{c \to d', d \to d', \Gamma}$	
Contradict $\dfrac{c \neq c, \Gamma}{\perp}$	

Efficient Variant: Properties

The relation $\to$ is terminating.

The inference rules Orient, Simplify, Collapse, Compose, and Contradict are terminating. Ex: Why?

Length of *any* derivation is bounded by $O(n^2)$. Ex: Why?

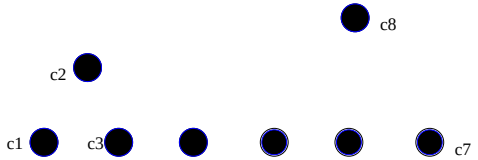
Union-Find data structure (with path compression) can be simulated using the above rules. Ex: How?

Technical point: Do not fix the ordering $\succ$ a priori. Discover it as the derivation proceeds (as in Union-Find.)

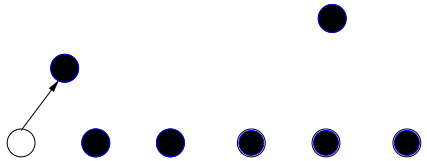
Therefore, the best strategy gives an $O(n\alpha(n))$ procedure; where α is the inverse Ackerman function.

I. Example

Blue circles denote the constants $\{c_1, c_2, \ldots, c_8\}$:

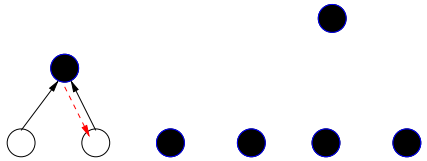


First we Orient the equation $c_1 = c_2$:

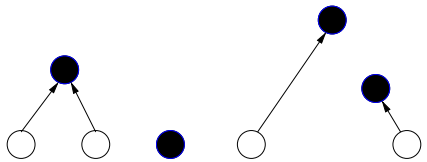


I. Example

Next, we Orient the equation $c_2 = c_3$:

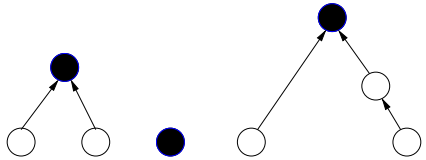


After Orienting $c_5 = c_8$ and $c_6 = c_7$:

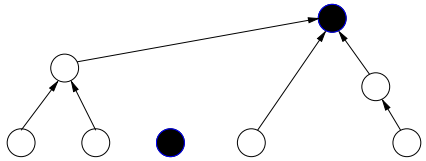


I. Example

The equation $c_7 = c_8$ is Oriented as:



And the equation $c_2 = c_8$ is oriented as:



Therefore, maximum depth is $O(\log(n))$.

I. Example

$c_1 = c_2, \quad c_2 = c_3, \quad c_5 = c_8, \quad c_6 = c_7, \quad c_6 = c_5, \quad c_3 = c_5$

$c_1 \rightarrow c_2,$

$c_3 \rightarrow c_2,$

$c_5 \rightarrow c_8, \quad c_6 \rightarrow c_7,$

$c_7 = c_8,$

$c_7 \rightarrow c_8,$

$c_2 = c_8$

$c_1 \rightarrow c_2, \quad c_3 \rightarrow c_2, \quad c_5 \rightarrow c_8, \quad c_6 \rightarrow c_7, \quad c_7 \rightarrow c_8, \quad c_2 \rightarrow c_8$

I. Observations

Let $E_0 \vdash E_1 \vdash \cdots \vdash R$ be a maximal derivation

- R is terminating
- Each c rewrites to a unique d
- Hence, R is a canonizer: it induces unique normal forms

The set of inference rules Orient, Collapse, Simplify, Compose, (Contradict) are more than just sound, complete, terminating.

Equal constants rewrite by R to the same canonical form.

II. Equality over constants: With Boolean Structure

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

Method 1: Convert ϕ to CNF + Union-Find

Method 2: Transform to propositional satisfiability + Use SAT solver

Mapping: α: atomic formulas $\mapsto$ boolean variables

Abstract	$\dfrac{\psi[c=d], D}{\psi[\alpha(c=d)], D \cup \{\alpha(c=d) \leftrightarrow (c=d)\}}$
Semantics	$\dfrac{\psi, D}{\psi \wedge (\alpha(c=d) \wedge \alpha(d=e) \Rightarrow \alpha(c=e)), D}$ if ...

Ex: Is Abstract+Semantics+SAT sound and complete?

II. Equality over constants: With Boolean Structure

Method 3: Finite Domain Instantiation

Different translation to propositional SAT + SAT solver.

ϕ is satisfiable iff it is satisfiable in an interpretation over the domain $\{1, 2, \ldots, n\}$.

This gives a propositional instance over $n\lceil \log n \rceil$ propositions

Ex: Reduce this number to $\sim \log(n!)$.

A set of interpretations $\mathcal{I}$ is sufficient for ϕ if whenever ϕ is satisfiable, it is satisfiable with an interpretation in $\mathcal{I}$

Example: $\mathcal{I} = \{I : I(c_i) \in \{1, 2, \ldots, i\}\}$ is sufficient for any ϕ

For a given ϕ, $|\mathcal{I}|$ can be substantially reduced with some preprocessing

II.3 Finite Domain instantation

Ignore the boolean structure of ϕ to obtain small domains that define $\mathcal{I}$

Constants: $c_1, c_2, \ldots, c_n$

Atomic formulas in ϕ: Af

Configuration: (Af, D), initially $D(c_i) = \{i\}$ for all c_i

PropEq	$\dfrac{Af, D}{Af, D[c_i := D(c_i) \cup \{j\}]}$ if $Af \vdash c_i = c_j$ and $j < i$

Ex: Upon termination, show that D can generate $\mathcal{I}$ which is sufficient for ϕ.

Ex: What is the worst case? Construct an example where this is exhibited.

Ex: Use disequality information to optimize D.

II. Generating Small Domains: Example

Let $\phi = \{(c_1 = c_2 \ \vee \ c_1 = c_3), (c_1 \neq c_2 \ \vee \ c_1 \neq c_3)\}$

Here $Af = \{c_1 = c_2, c_1 = c_3, c_1 \neq c_2, c_1 \neq c_3\}$

$$\frac{\{1\},\{2\},\{3\}}{\{1\},\{2,1\},\{3\}}$$
$$\frac{\{1\},\{2,1\},\{3,1\}}{}$$

We need to test satisfiability of ϕ over four possible interpretations

Suppose $I(c_1) = 1, I(c_2) = 1, I(c_3) = 3$

Interpretation I makes ϕ true, hence we conclude ϕ is satisfiable

II. Equality over constants: With Boolean Structure

Method 4: Lift Ordered-Transitive-Closure to Clauses

General Idea: Specialize calculi complete for equational logic

Superpose Right	$\dfrac{c = d \vee C,\ c = e \vee D,\ \Gamma}{d = e \vee C \vee D, \ldots}$ if c is maximal
Superpose Left	$\dfrac{c = d \vee C,\ c \neq e \vee D,\ \Gamma}{d \neq e \vee C \vee D, \ldots}$ if c is maximal
EqResolution	$\dfrac{c \neq c \vee C, \Gamma}{C, \Gamma}$
EqFactoring	$\dfrac{c = d \vee c = e \vee C, \Gamma}{d \neq e \vee c = e \vee C, \ldots}$ if c is maximal

More when we discuss general equational theorem proving

Ex: Show that the inference system is sound and complete.