Little Engines of Proof: Mid-Term Summary

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

Overview

Automated deduction is a form of scientific computing that infers the consequences of a collection of statements.

The field of automated deduction has a rich collection of fundamental ideas, algorithms, proof methods, implementation techniques, and applications in engineering and artificial intelligence.

This course explores the foundations of automated deduction through a series of algorithms and their correctness proofs.

Our emphasis is on simplicity and uniformity in the presentations of the algorithms and their proofs.

This simplicity also has an impact on the ease of implementation, integration, and optimization.

Logic

Logic consists of a trinity between language, meaning (semantics), and method (proof).

Language is used to define concepts and make statements about these concepts.

Meaning distinguishes the valid statements from refutable ones.

Method is used to effectively demonstrate the validity of a statement through valid syntactic inference steps.

A proof method is sound if all provable statements are valid, and complete if all valid statements are provable.

Example: Propositional Logic

Formulas: $\phi := P \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \phi_1 \wedge \phi_2 \mid \phi_1 \Rightarrow \phi_2$.

Semantics: Truth value of a formula is calculated with respect to an interpretation $\mathcal{M}$, an assignment of $\top$ or $\bot$ to the propositional atoms.

A formula is satisfiable if it evaluates to $\top$ under some interpretation, otherwise, it is unsatisfiable.

A formula is valid if it evaluates to $\top$ under every interpretation, and is invalid or refutable, otherwise.

A formula is valid iff its negation is unsatisfiable.

As examples, $p \vee \neg p$ is valid, $p \wedge \neg p$ is unsatisfiable, $p \vee \neg q$ is satisfiable and refutable.

Proof Methods

Propositional logic has a number of different proof methods given by Hilbert-style calculi, and Gentzen-style natural deduction and sequent calculi.

These logics consist of proof rules that assert the provability of a conclusion formula from those of some premise formulas.

A proof method is sound if whenever the premises to a proof rule are valid, then so are the conclusions.

A set of formulas is consistent if they cannot derive both a formula and its negation.

For example, $\{p \wedge \neg q, p, \neg q\}$ is consistent, but $\{p \wedge \neg q, p, q\}$ is not.

Completeness

Completeness shows that whenever a formula is valid, it is provable.

Alternately, if a formula is not provable, then its negation must be satisfiable.

Alternately, any consistent set of formulas is satisfiable.

A consistent set of formulas Γ can be extended to a complete and consistent set of formulas $\hat{\Gamma}$

The satisfying interpretation $\mathcal{M}_{\hat{\Gamma}}$ for $\hat{\Gamma}$ (and hence, Γ) can be read off from $\hat{\Gamma}$.

For example $\{p \wedge \neg q, p \vee q\}$ is consistent and can be extended to a complete set $\{p \wedge \neg q, p \vee q, p, \neg q, \ldots\}$.

The assignment $\{p \leftarrow \top, q \leftarrow \bot\}$ can be read off.

Completeness for Equality

We now add constants $c_0, c_1, \ldots$ to the logic and let propositional atoms include simple equalities $c_i = c_j$.

The interpretation $\mathcal{M}$ now includes a domain D over which the constants are interpreted so that $\mathcal{M}(c_i) \in D$.

The proof rules include reflexivity, transitivity, and symmetry for equality.

Completeness: A consistent set of formulas Γ can be completed to $\hat{\Gamma}$. The domain $D_{\hat{\Gamma}}$ can be taken as the equivalence class of constants in $\hat{\Gamma}$ so that $[c_i] = \{c_j | c_i = c_j \in \hat{\Gamma}\}$, and $\mathcal{M}_{\hat{\Gamma}}(c_i) = [c_i]$.

Check that $\mathcal{M}_{\hat{\Gamma}} \models \Gamma$.

Equality with Terms

We now add n-ary function symbols f to construct terms of the form $f(t_1, \ldots, t_n)$ for n-ary function f, and n terms $t_1, \ldots, t_n$.

The interpretation $\mathcal{M}(f)$ is a function from D^n to D.

The congruence rule is added to the set of proof rules.

Completeness: As before, a consistent set of formulas is completed to yield $\hat{\Gamma}$ and the domain $D_{\hat{\Gamma}}$ is the equivalence class of terms in $\hat{\Gamma}$, and $\mathcal{M}_{\hat{\Gamma}}(f)([t_1], \ldots, [t_n]) = [f(t_1, \ldots, t_n)]$.

Check that $\mathcal{M}_{\hat{\Gamma}} \models \Gamma$.

Inference Rules

Automated Deduction is just an operationalization of the completeness proof.

An inference system consists of inference states, and inference rules that transform a premise inference state into a conclusion one.

Inference rules preserve satisfiability.

An inference state is reducible if it can be the premise of an inference rule.

$\bot$ is a special irreducible inference state denoting a contradiction.

Any non-$\bot$ irreducible inference state must be satisfiable.

An inference state is a disjunction of configurations, where each configuration is a conjunction of formulas.

Semantic Tableaux

The inference rules for the Semantic Tableaux procedure are:

$\frac{A \wedge B, \Gamma}{A, B, \Gamma} \wedge +$	$\frac{\neg(A \wedge B), \Gamma}{\neg A, \Gamma \mid \neg B, \Gamma} \wedge -$
$\frac{\neg(A \vee B), \Gamma}{\neg A, \neg B, \Gamma} \vee -$	$\frac{(A \vee B), \Gamma}{A, \Gamma \mid B, \Gamma} \vee +$
$\frac{\neg(A \Rightarrow B), \Gamma}{A, \neg B, \Gamma} \Rightarrow -$	$\frac{(A \Rightarrow B), \Gamma}{\neg A, \Gamma \mid B, \Gamma} \Rightarrow +$
$\frac{\neg\neg A, \Gamma}{A, \Gamma} \neg$	$\frac{A, \neg A, \Gamma}{\bot} \bot$

Conclusion states are smaller than premise states, but equisatisfiable.

An irreducible non-$\bot$ state is a collection of literals with no clashes, and is therefore satisfiable.

Clausal (CNF) Satisfiability

A clause is a disjunction of literals (atoms or negated atoms). Let literal $\overline{l}$ be the negation of literal l.

A set of clauses Γ can be shown satisfiable by ordered resolution.

Assume an ordering on the literals in Γ.

Remove duplication literals $l \vee l \vee C \Longrightarrow l \vee C$.

Eliminate tautology clauses of the form $l \vee \overline{l} \vee C$.

$$\frac{\Gamma, l \vee C_1, \overline{l} \vee C_2}{\Gamma, l \vee C_1, \overline{l} \vee C_2, C_1 \vee C_2} \mathbf{Res}$$

The resolution rule **Res** is applicable only when l is maximal in $l \vee C_1$, and $\overline{l}$ is maximal in $\overline{l} \vee C_2$, and $C_1 \vee C_2$ is not a tautology.

Correctness

Each state has only a single configuration.

Check that the premise and conclusion configurations are equisatisfiable.

Check that the conclusion configuration is smaller than the premise.

Completeness: Given an irreducible non-$\bot$ configuration Γ, build a series of partial interpretations $\mathcal{M}_i$ as follows:

1. Let $\mathcal{M}_0 = \emptyset$
2. If $l \vee C$ is the minimal clause unassigned in $\mathcal{M}_i$ with maximal literal l, then if $\overline{l} \vee D$ occurs in Γ, then clearly $\mathcal{M}_i \models C \vee D$. If $\mathcal{M}_i \models C$, then let $\mathcal{M}_{i+1} = \mathcal{M}_i\{\overline{l} \leftarrow \top\}$, else $\mathcal{M}_{i+1} = \mathcal{M}_i\{l \leftarrow \top\}$.

If there are n distinct atoms in Γ, check that $\mathcal{M}_n \models \Gamma$.

Equality

Let E contain equality between constants.

Assume an order $\leq$ between constants.

Let V be an oriented, idempotent, functional equality set such that $V(V(c)) = V(c)$ and if $c = d \in V$, then $c \leq d$.

$V \triangleright V' = \{c = V'(d) | c = d \in V\}$ and $V \circ V' = (V \triangleright V') \cup V'$.

$\dfrac{c = d, E; V}{E; V}$ if $V(c) \equiv V(d)$
$\dfrac{c = d, E; V}{E; V \circ \{orient(V(c) = V(d))\}}$ if $V(c) \not\equiv V(d)$

To check if $E \vdash c = d$, transform $E; \emptyset$ to $\emptyset; V$ and check if $V(c) \equiv V(d)$.

Correctness

Check that

1. $c = d$ and $orient(c = d)$ are equisatisfiable.
2. $c = d; V$ and $V(c) = V(d); V$ are equisatisfiable.
3. V, V' and $V \circ V'$ are equisatisfiable.
4. If functional equality sets V and V' have disjoint domains are oriented, then $V \circ V'$ is an oriented functional equality set.
5. If V is an oriented functional equality set, $V \models c = d$ iff $V(c) = V(d)$.

Term Equality

We are given as input a set of equalities E of the form $s = t$ where s and t are terms built from uninterpreted constants and function symbols.

Inference state is a triple $E; U; V$.

V is as before. U is a functional equality set with equalities of the form $c = f(c_1, \ldots, c_n)$ with $U \triangleright V = U$.

If $S = U; V$, define $S[a]$ as

$$\begin{aligned} S[c] &= V(c) \\ S[f(t_1, \ldots, t_n)] &= V(c), \text{ where} \\ &\quad S[t_i] = t_i', \text{ for } 1 \leq i \leq n \\ &\quad c = f(t_1', \ldots, t_n') \in U \\ S[f(t_1, \ldots, t_n)] &= f(t_1', \ldots, t_n'), \text{ otherwise.} \end{aligned}$$

The Inference Steps

$(U; V) \circ \{c = d\} = (U \triangleright \{c' = d'\}); (V \circ \{c' = d'\})$, where $c' = d' \equiv orient(c = d)$.

$\dfrac{c = d, E; U; V}{E; U; V}$ if $V(c) \equiv V(d)$
$\dfrac{c = d, E; U; V}{E; (U; V) \circ \{V(c) = V(d)\}}$ if $V(c) \not\equiv V(d)$
$\dfrac{(s = t)\{f(c_1, \ldots, c_n)\}, E; U; V}{(s = t)\{c\}, E; U; V}$ if $c = f(c_1', \ldots, c_n') \in U, c_i' = V(c_i)$
$\dfrac{(s = t)\{f(c_1, \ldots, c_n)\}; U; V}{(s = t)\{c\}, E; U \cup \{c = f(c_1', \ldots, c_n')\}; V}$ if $c = f(c_1', \ldots, c_n') \notin U$, c fresh, $c_i' = V(c_i)$ for $1 \leq i \leq n$
$\dfrac{E; U; V}{E; (U; V) \circ \{c = d\}}$ if $U(c) \equiv U(d) for V(c) \not\equiv V(d)$

Correctness

To verify $E \models s = t$: Transform $E; \emptyset; \emptyset$ to irreducible $\emptyset; U; V$, and check if $S[s] \equiv S[t]$ for $S = U; V$.

Termination: Either $size(E)$ decreases, or the number of equivalence classes in V decreases.

Check that if $E; U; V \vdash E'; U'; V'$, then $E; U; V$ and $E'; U'; V'$ are equisatisfiable.

Completeness: Need to show that if $S = U; V$ is irreducible, then $S \models s = t$ iff $S[s] \equiv S[t]$.

Term model construction: Check that $\mathcal{M}_S(s) \equiv S[s]$.

$$\begin{aligned} \mathcal{M}_S(c) &= V(c) \\ \mathcal{M}_S(f)(a_1, \ldots, a_n) &= V(c), \text{ if } c = f(a_1, \ldots, a_n) \in U \\ \mathcal{M}_S(f)(a_1, \ldots, a_n) &= f(a_1, \ldots, a_n), \text{ otherwise.} \end{aligned}$$