Automated Theorem Proving in a Simple Meta-Logic for LF

Carsten Schürmann
joint work with Frank Pfenning

CADE-15, Lindau, Germany
July 8, 1998

1. Reasoning About Deductions
2. Leveraging the Logical Framework
3. The Meta-Logic $\mathcal{M}_2$ for LF
4. Experimental Results
5. Conclusion

Deductive Systems

- Define a language

$$A ::= P \mid A \supset A$$

- Define a judgment via axioms and inference rules

$$K_{A,B} \;:\; \vdash A \supset (B \supset A)$$

$$S_{A,B,C} \;:\; \vdash (A \supset (B \supset C)) \supset (A \supset B) \supset A \supset C$$

$$MP_{A,B} \;:\; \frac{\vdash A \supset B \qquad \vdash A}{\vdash B}$$

- Pervasive use of schematic judgments

Automated Deduction

- Traditional theorem proving: Searching For Deductions

  Does there exist a deduction H_A of $\vdash A \supset A$?

- Our goal: Reasoning About Deductions

  – Soundness and completeness of proof procedures

    For every D of $\vdash A$ there is an E of $\vdash^{-} A$.

  – Consistency and cut-elimination

    For every D of $\Gamma \Longrightarrow A$ and E of $\Gamma, A \Longrightarrow C$
    there exists an F of $\Gamma \Longrightarrow C$.

Example: Hilbert's Deduction Theorem

- Requires hypothetical judgment $A \vdash B$
- **Theorem:** For every D of $A \vdash B$ there exists an E of $\vdash A \supset B$.
- **Proof:** By induction on the structure of D.
- **Case:** $D = A \vdash A$. Then

$$MP\ (MP\ S\ K)\ K \text{ of } \vdash A \supset A \qquad \text{by rules}$$

- This case: pure resolution

Proof Continued

- **Case:** $D = \dfrac{\begin{array}{cc} \mathcal{D}_1 & \mathcal{D}_2 \\ A \vdash B \supset C & A \vdash B \end{array}}{A \vdash C}$.

  E_1 of $\vdash A \supset (B \supset C)$ by i.h. on $\mathcal{D}_1$
  E_2 of $\vdash A \supset B$ by i.h. on $\mathcal{D}_2$
  $MP\ (MP\ S\ E_1)\ E_2$ of $\vdash A \supset C$ by rules

- This case: induction hypotheses, resolution

- Two remaining cases: resolution

Reduction to Theorem Proving

- Encode judgments and derivations in a meta-logic

  $\mathsf{ded}(\ulcorner E \urcorner, \ulcorner A \urcorner)$ iff E is a derivation of $\vdash A$.
  $\mathsf{entails}(\ulcorner D \urcorner, \ulcorner A \urcorner, \ulcorner B \urcorner)$ iff D is a derivation of $A \vdash B$.

- Encode and prove the Deduction Theorem

  $$\forall A. \forall B. \forall D. \mathsf{entails}(D, A, B) \Rightarrow \exists E. \mathsf{ded}(E, \mathsf{imp}(A, B))$$

- Additional lemmas must be proven explicitly, which could be assumed statically.

- Increased search space: $\mathsf{ded}(\ulcorner E \urcorner, \ulcorner A \urcorner)$ must be proven explicitly, which could be checked statically.

Automating Meta-Theory

- Degree of automation depends critically on
  - directness of encoding
  - reasoning principles in meta-logic
- Quantifier-free inductive definitions
  [Shankar'87] Nqthm, [Matthews'94] FS_0, ...
- Inductive type theories
  [Basin & Constable'93] Nuprl, [McKinna & Pollack'93] Lego,
  [Barras & Werner'97] Coq, ...

Claim

- Automating the meta-theory of deductive systems is
  - different enough and
  - important enough

  to warrant investigation of special-purpose theorem proving techniques.
- Applications: language design, logic design, theorem proving verification, secure mobile code, certifying compilation, etc.
- Other special purpose theorem provers: geometry theorem proving, model checking, arithmetic, various constraint systems, etc.

Logical Frameworks

- Specialized meta-languages for direct representation of deductive systems
- LF [Elf, Twelf]
  - dependently typed λ-calculus
  - schematic judgments
  - hypothetical judgments
  - judgments as types
    $A ::= a\, M_1 \ldots M_n \mid \Pi x{:}A_1.A_2 \mid A_1 \to A_2$
  - deductions as objects
    $M ::= c \mid x \mid \lambda x{:}A.M \mid M_1\, M_2$
- Hereditary Harrop formulas (FOHH) [Isabelle, λProlog]

Hilbert Calculus in LF

- Propositions: $\ulcorner A \urcorner : o$

$$\begin{array}{lll} o & : & \mathsf{type} \\ \mathsf{imp} & : & o \to o \to o \end{array} \qquad \textit{using infix } \mathsf{imp}$$

- Deductions: $\ulcorner D \urcorner : \mathsf{hil}\, \ulcorner A \urcorner$ iff D deduction of $\vdash A$

$$\begin{array}{lll} \mathsf{hil} & : & o \to \mathsf{type} \\ K & : & \mathsf{hil}\,(A \;\mathsf{imp}\; (B \;\mathsf{imp}\; A)) \\ S & : & \mathsf{hil}\,((A \;\mathsf{imp}\; (B \;\mathsf{imp}\; C)) \;\mathsf{imp}\; (A \;\mathsf{imp}\; B) \;\mathsf{imp}\; A \;\mathsf{imp}\; C) \\ MP & : & \mathsf{hil}\,(A \;\mathsf{imp}\; B) \to \mathsf{hil}\, A \to \mathsf{hil}\, B \end{array}$$

Reasoning About Deductions in LF

- Hypothetical judgments:

$$\ulcorner E \urcorner : \mathsf{hil}\,\ulcorner A \urcorner \to \mathsf{hil}\,\ulcorner B \urcorner \text{ iff } E \text{ deduction of } A \vdash B$$

- Cannot prove properties of such functions *within* LF
- Add one more layer to reason *about* LF objects
- Meta-logic design criteria
  - Meta-theoretic proofs should be expressible
  - Embedding of the logical framework should be direct

$\mathcal{M}_2$, a Simple Meta-Logic for LF

- Formulas quantify over *closed* LF objects.

$$F ::= \forall x_1{:}A_1 \ldots \forall x_n{:}A_n.\exists y_1{:}B_1 \ldots \exists y_m{:}B_m.\top$$

- Judgment

$$\Gamma; \Delta \vdash_{\Sigma} P \in F$$

  where Σ is an LF signature,
  Γ is an LF context,
  Δ contains lemmas and "inductive assumptions",
  P is a (meta-)proof term,
  F is meta-logic formula

- No primitive predicates

- Expressive power stems from LF

Some Sequent Rules for $\mathcal{M}_2$

- Filling x:

$$\frac{\Gamma \vdash^{LF}_{\Sigma} M : A \qquad \Gamma; \Delta \vdash P \in [M/x]F}{\Gamma; \Delta \vdash \langle M, P \rangle \in \exists x{:}A.F} \exists\mathsf{R}$$

- Recursion:

$$\frac{\Gamma; (\Delta, \mathbf{x} \in F) \vdash_{\Sigma} P \in F}{\Gamma; \Delta \vdash_{\Sigma} (\mu\mathbf{x} \in F.P) \in F} \mathsf{fix}$$

where $\mu\mathbf{x} \in F.P$ terminates in $\mathbf{x}$.

- Splitting x:

$$\frac{\Gamma(x) = B \qquad \Gamma; \Delta \vdash_{\Sigma} \Omega \ \mathsf{covers}\ B \ \mathsf{in}\ F}{\Gamma; \Delta \vdash_{\Sigma} \mathsf{case}\ x \ \mathsf{of}\ \Omega \in F} \mathsf{case}$$

where Ω is a pattern matching expression.

Correctness

- **Theorem:**

  If $\cdot; \cdot \vdash_{\Sigma} P \in \forall x_1{:}A_1 \ldots \forall x_n{:}A_n . \exists y_1{:}B_1 \ldots \exists y_m{:}B_m . \top$ then
  for all *closed* LF objects $M_1{:}A_1, \ldots, M_n{:}A_n$
  there exist *closed* LF objects $N_1{:}B_1, \ldots, N_m{:}B_m$
  over signature Σ.

- **Proof sketch:** Define an operational semantics for proof term P in $\mathcal{M}_2$ and show

  1. type preservation,
  2. progress, and
  3. termination.

Example: Hilbert's Deduction Theorem

- **Theorem:**

$$\forall A{:}o.\forall B{:}o.\forall D{:}(\mathsf{hil}\ A \to \mathsf{hil}\ B).\exists E{:}\mathsf{hil}\,(A\ \ \mathsf{imp}\ \ B).\top$$

- Proof term: A total function mapping

objects of type $(\mathsf{hil}\ A \to \mathsf{hil}\ B)$
to
objects of type $\mathsf{hil}\,(A\ \mathsf{imp}\ B)$

Example: Hilbert's Deduction Theorem

- **Theorem:**

$$\forall A{:}o.\forall B{:}o.\forall D{:}(\mathsf{hil}\ A \to \mathsf{hil}\ B).\exists E{:}\mathsf{hil}\,(A\ \ \mathsf{imp}\ \ B).\top$$

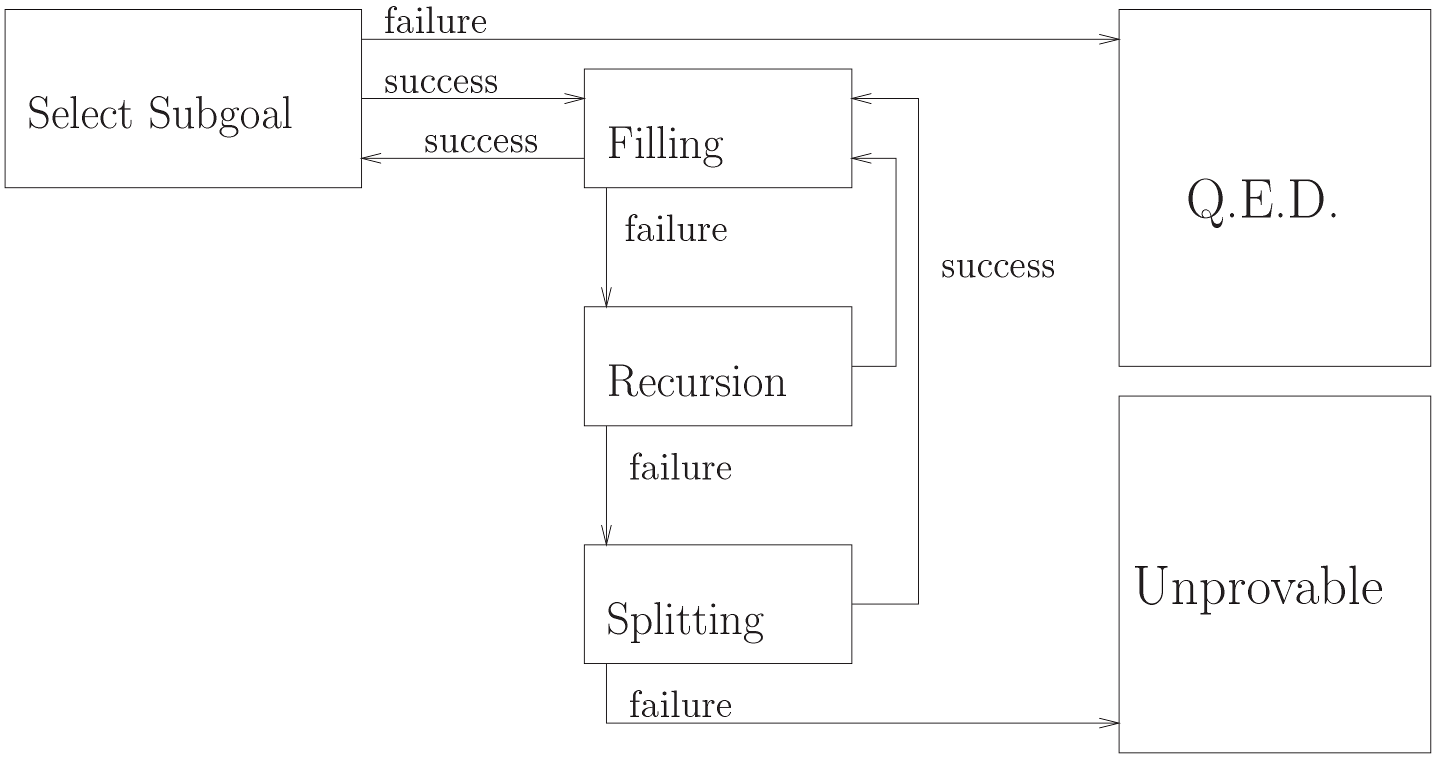
- Proof term:

$$\mathsf{abs}_{A,B} : (\mathsf{hil}\ A \to \mathsf{hil}\ B) \Rightarrow \mathsf{hil}\,(A\ \mathsf{imp}\ B)$$

$$\begin{array}{lll}
\textbf{fun}\ \mathsf{abs}\ (\lambda x.x) & = & MP\ (MP\ S\ K)\ K \\
\quad \mid \mathsf{abs}\ (\lambda x.K) & = & MP\ K\ K \\
\quad \mid \mathsf{abs}\ (\lambda x.S) & = & MP\ K\ S \\
\quad \mid \mathsf{abs}\ (\lambda x.MP\ (D_1\ x)\ (D_2\ x)) & = & \\
\qquad \textbf{let} & & \\
\qquad\qquad \textbf{val}\ E_1 = \mathsf{abs}\ (\lambda x.D_1\ x) & & \\
\qquad\qquad \textbf{val}\ E_2 = \mathsf{abs}\ (\lambda x.D_2\ x) & & \\
\qquad \textbf{in} & & \\
\qquad\qquad MP\ (MP\ S\ E_1)\ E_2 & & \\
\qquad \textbf{end} & &
\end{array}$$

Strategy

- Input: signature, theorem, bound, termination ordering



Strategy Remarks

- Bounds ensure termination, no backtracking
  - **Filling:** Size bound on terms
  - **Recursion:** Number of recursive calls
  - **Splitting:** Splitting depth
- Termination orders are lexicographic/simultaneous extensions of subterm orderings
- Dependent types restrict search space for Filling and Splitting

The Twelf Implementation

- In Standard ML
- Dependently typed explicit substitutions
- Pattern unification and constraint simplification
- Extract realizing functions as relations
- Very few optimizations (unification by effect)

Experimental Results, FRS

Experiment	**Fill**	**Split**	**Rec**	**Total**
Cartesian Closed Categories	1.000	0.004	0.036	1.099
CPM Completeness	0.916	0.010	0.117	1.134
CPM Proof equivalence: $\Rightarrow$	0.226	0.034	0.442	0.951
CPM Proof equivalence: $\Leftarrow$	0.280	0.033	0.647	1.235
Horn LP Soundness	4.336	0.004	0.049	4.501
Horn LP Canonical forms	0.028	0.009	0.107	0.303
Horn LP Completeness	0.015	0.005	0.039	0.195
Mini-ML Value soundness	0.016	0.041	0.061	0.172
Mini-ML Type preservation	0.062	0.521	0.150	0.799
Mini-ML Evaluation/Reduction	25.397	0.007	0.078	25.546
Hilbert's deduction theorem	0.197	0.004	0.010	0.322
Associativity of +	0.009	0.012	0.016	0.063
Commutativity of +	0.092	0.609	4.139	4.877

Linux 2.30, SML/NJ 110, Twelf 1.2 on Pentium II (300 Mhz)

Related Work

- $FO\lambda^{\Delta I\!N}$ over FOHH [McDowell & Miller'97]
  - definitional reflection ($\sim$ splitting)
  - only natural number induction
  - interactive
- HALF [Göteborg]
  - one-level architecture, proof editor
  - filling, splitting, recursion
- Schema checking [Rohwedder]
  - decidable meta-theory: little automation
  - larger class of proofs

Future Work

- Automatic appeal to lemmas
- Complete induction
- Complex termination orderings
- Richer meta-logic ($\mathcal{M}_\omega$, beyond)
- Reasoning about open LF objects in regular contexts

Summary

- $\mathcal{M}_2$ a simple meta-logic over LF
- Can prove a number of meta-theorems automatically
  soundness and completeness of Horn-LP, correctness of abstract machines, type preservation for Mini-ML, deduction theorem
- Simple-minded strategy
- Effectiveness due to expressive power of LF
- Ongoing work:
  - Lemmas
  - Generalization to regular hypothetical judgments
  - Efficiency and strategy improvements
  - Linear logical framework