Proofs “Я” Us

Frank Pfenning

Workshop on Logosphere and Infer!

SRI International, October 2003

Outline

- Logical Frameworks
- The LF Logical Framework
- The Twelf Implementation
- Extensions and Efficiency Improvements
- Questions for Logosphere

Logical Frameworks: The Vision

- De Bruijn's AUTOMATH [1968]: *foundationally uncommitted* framework for formally checking mathematics
- Step 1: define logic
- Step 2: present theory
- In this talk:

  theory = definitions + theorems + proofs
- Theory (incl. proofs) checked in AUTOMATH
- Reject set theory or type theory as foundation

Logical Frameworks: The Reality

- De Bruijn's dream has been realized!
- But: also many more specialized systems
- Proof checking vs. proof search
- Applications in mathematics vs. applications in computer science

Approaches to Logical Frameworks

- Rewriting logic [Maude, ELAN]
  - judgments as terms
  - deduction as rewriting
  - proofs as traces
  - no well-developed theory or practice of proof representation and checking (as yet)
  - effi cient support for equational reasoning

Approaches to Logical Frameworks

- Meta-logic [Isabelle, λProlog]
  - judgments as propositions
  - no intrinsic proofs
- Type theory [AUTOMATH, LF]
  - judgments as types
  - proofs as objects
  - proof-checking as type-checking
  - simple (fi xed) intrinsic equality

Logical Framework for Logosphere

- Desiderata
  - Express logics naturally
  - Represent theories compactly
  - Check proofs effi ciently
  - Translate between logics/theories
  - Verify properties of logics/theories
  - Tested in the battlefi eld
- LF (implemented in Twelf) is close

The LF Logical Framework

- Second generation [Harper et al.'87,'93]
- Direct descendant of AUTOMATH
- Supports
  - Variable binding and substitution
  - Parametric and hypothetical judgments
  - Higher-level judgments
- Based on dependent types
- Representing judgments as types

The Twelf Implementation

- Second generation [Schürmann & Pf'98]
- Implements LF
- In addition, offers:
  - Type reconstruction
  - Meta-programming as logic programming
  - Meta-level reasoning
  - Constraint domains
- Unification and matching are central
- Tutorials, User's Guide, etc.

Twelf Applications

- Foundational PCC [Appel et al.]
  - Represent higher-order logic (HOL) in Twelf
  - Develop theory of machine code in HOL
  - ˜100K lines of Twelf source
- Typed Assembly Language [Crary et al.]
  - Represent typed assembly language in Twelf
  - Prove soundness as meta-theorem
  - ˜60K lines of Twelf source
- Many smaller examples

Related LF Implementations

- LF_i [Necula & Lee]
  - Redundancy elimination on fragment of LF
  - Used in Touchstone certifying compiler
  - Used in Ginseng certifying Java compiler
  - Used with certifying decision procedures
  - Also: oracle strings [Necula et al.]
- Flea and Flit [Stump et al.]
  - More effi cient checking of proofs
  - Used in foundational PCC [Appel et al.]
  - Used with CVC [Barrett, Stump et al.]

The Practice of LF

- *LF is foundationally uncommitted*

  We can encode as much or as little of the semantics as we wish

- Allows inconsistent theories
- Allows typed or untyped theories
- The more is captured, the higher the benefits

Some Logics in LF

- First- and higher order logic [Harper et al.'87]
- Calculus of constructions [Pf'93]
- Martin-Löf type theory [Murthy]
- Modal and temporal logics [Bernard'02]
- Linear logic [Pf'94]

Framework Extensions

- Constraint domains
  - Now: integers, word32, rationals, strings
  - Generate and check proofs
  - Richer equational reasoning [future]
- Module system [ongoing: Watkins & Pf'01]
  - Designed but not yet implemented
  - Semantics as elaboration into LF

Efficiency Improvements

- Tabled logic programming [ongoing: Pientka'03]
- Proof irrelevance [ongoing: Pf'01, Reed'03]
  - Omit some proofs in decidable theories
- Redundancy elimination
  [ongoing: Watkins'02, Reed'03]
  - More compact representations
  - More effi cient checking

Summary

- Logical frameworks: Proofs“Я”Us
- Foundationally uncommitted
  - Step 1: encode logic
  - Step 2: encode theories
    (= defi nitions + theorems + proofs)
- LF logical framework
  - Mature implementation in Twelf
  - Signifi cant applications (FPCC, TALT, etc.)
  - Further work: modularity, effi ciency

More Information

- `http://www.twelf.org`
  - Sources (SML: SML/NJ, PolyML, MLton)
  - Binaries (Windows, Linux, MacOS X)
  - Emacs support
- Documentation and examples
  - Tutorial
  - User's Guide
  - Handbook article
  - Course notes *Computation & Deduction*

Question for Logosphere

- Representing PVS logic and proofs in LF?
- Representing other relevant logics?
- XML and/or OMDOC interfaces to Twelf?
- Total and partial translations as higher-level judgments on proofs?
- Requirements on module system?
- Requirements on constraint domains?
- Requirements on space and time efficiency?