CAS 734 Winter 2005

03 Interactive Theorem Proving Systems

Instructor: W. M. Farmer

Revised: 18 January 2005

Mechanized Mathematics Systems

- A **mechanized mathematics system** (MMS) is a software system that is intended to automate, manage, and improve parts of the mathematics process.
  - Employs deductive and computational technology.

- Three major types of MMSs:
  1. **Computer theorem proving systems**.
     - Automated theorem provers.
     - Interactive theorem proving systems (ITPSs).
     - Proof checkers.
  2. **Computer algebra systems**.
     - Macsyma, Maple, Mathematica, etc.
     - Axiom.
  3. **Interactive mathematics laboratories**.
     - Do not exist yet.

Computer Theorem Proving Systems

- Support “axiomatic mathematics”.
  - Mathematics is represented by axiomatic theories.
  - Reasoning is performed by proving conjectures.
  - Most emphasize proof checking or proof development.
- Strengths:
  - Based on rigorous logical foundations.
  - Support a wide range of mathematics.
- Weaknesses:
  - Very difficult to use.
  - Poor support for routine computation.
  - Abstract theories are emphasized over concrete structures.

Computer Algebra Systems

- Support “algorithmic mathematics”.
  - Mathematics is represented by algorithms.
  - Reasoning is performed by computation.
- Strengths:
  - Perform fast, sophisticated symbolic computations.
  - Relatively easy to use.
- Weaknesses:
  - Not based on a rigorous logical foundation.
  - Poor support for “context guided” computation.
  - Concrete structures are emphasized over abstract theories.

Interactive Mathematics Laboratories

- An **interactive mathematics laboratory (IML)** is an MMS that:
  - Is widely accessible.
  - Facilitates many kinds of mathematical activity.
  - Combines the capabilities of both computer theorem proving systems and computer algebra systems.
- An IML offers an environment that is:
  - Formal.
  - Interactive.
  - Mechanized.
- IMLs do not exist today, but much of the technology needed to build one does exist.

Components of an IML

1. Mathematics library.
   - Mathematical knowledge is stored dynamically.
   - Includes both axiomatic and algorithmic mathematics.
   - Web accessible.
2. Reasoning engine.
   - Theory development facility.
   - Deduction/computation workbench.
3. User interface.
   - Supports multiple styles of interaction.
   - Offers a range of exploratory tools.
   - Provides notational freedom.

Impact of an IML

- Transform how people learn and practice mathematics.
  - People would have greater mathematical reach.
  - Students would learn more mathematics by being able to do more mathematics.

- Students, engineers, scientists would likely benefit more than mathematicians.

- The mathematical competency of society would be raised.

Obstacles to Building an IML

1. The development cost is very high.
2. The mathematics community is apathetic.
3. Very few people have expertise or training in formalized mathematics.
4. There is very limited interaction between the computer theorem proving and computer algebra fields.
5. To be effective, a mathematics library must include many kinds of mathematics and be carefully organized.
6. The design of an IML requires sophisticated software engineering.
7. Traditional logics are not suited to be the underlying logic of an IML.

Leading ITPSs

- Coq.
- HOL.
- IMPS.
- Isabelle.
- Mizar.
- Nqthm/ACL2.
- Nuprl.
- PVS.

Logic

Kinds of logic commonly used by ITPSs:

- Equational logic.
- First-order logic.
- Type theory (higher-order logic).
  - Simple type theory (Church's type system).
  - Constructive type theory.
- Set theory.
  - Zermelo-Fraenkel set theory.
  - NBG (von-Neuman-Bernays-Gödel) set theory.
  - Stronger set theories.

Logical Foundation

- Fixed logic.
  - Single theory supplied with system.
  - User-defined theories without interoperability.
  - User-defined theories with interoperability.
- Logical framework.
  - User-defined logics without interoperability.
  - User-defined logics with interoperability.

Proof Construction

- Automated proof search.
- Proof checking.
  - The correctness of a proof tree is automatically checked.
  - A proof tree is interactively constructed, either forwards or backwards, by applying rules of inference.
  - A deduction is interactively constructed in which the inferences are automatically checked.
- Goal reduction by the application of **tactics**.
  - "Pure" tactics that only apply rules of inference.
  - "Impure" tactics that apply decision procedures, rewrite rules, algebraic simplification, etc. for which correctness is not proven in the logic.

Proof Representation

- **Descriptive**: proof object that was constructed.
  - Outside of logic.
  - Inside logic (e.g., lambda-term).
- **Prescriptive**: proof script used to construct the proof.
  - Sequence of low-level commands (not robust).
  - Sequence of high-level commands (robust).
- Both descriptive and prescriptive proof representations may be translated to natural language.

Techniques for Achieving Correctness

- Automatically check final results.

- Check final results using a small program (**de Bruijn criterion**).
  - The small program should be independently developed and could be formally verified.

- All mathematical reasoning is done within one system-supplied theory (**foundationalist criterion**).
  - A related technique is to show that all mathematical reasoning can be interpreted in the system-supplied theory.

- The correctness of computations is proved
  - Inside the logic or
  - Outside of the logic.

Some Concluding Remarks

- On one hand, the logical foundation should be as simple and familiar as possible.
  - Makes the learning process easier for users.
  - Makes it easier to design and implement the system.

- On the other hand, the logical foundation should be as expressive and powerful as possible.
  - Makes it easier to support mathematical practice.
  - Makes it easier to do everything within the logic.

- On one hand, the user needs to understand and control the reasoning process.

- On the other hand, the reasoning process needs to be highly automated.