Computers in Mathematics:

Automated and Interactive Proofs

Robert Y. Lewis

Carnegie Mellon University

6/17/2014

Why do we want computers to do math at all?

- They're accurate!
- They're fast!
- They can comprehend huge problems (in a certain sense).
- They can (sometimes) produce formal proofs.
- A standardized proof library is extremely useful.
- The math behind automated reasoning is interesting in its own right.

Automated reasoning

In the automated reasoning approach, the computer attempts to solve a problem on its own, without input from the user.

Examples of this include calculators, SAT solvers for boolean formulae, and decision procedures in general.

z3

```
>>> s = Solver()
>>> s.add(Implies(p, q))
>>> s.add(Implies(q, r))
>>> s.add(p)
>>> s.add(Not(r))
>>> s.check()
unsat
```

Interactive theorem proving

In interactive theorem proving, the user gives "hints," and the computer fills in the gaps. These proofs often involve calls to automated tools.

Environments such as Isabelle, HOL Light, and Coq have been used to formalize many proofs.

Lean

```
theorem succ_le_succ {a b: Nat} (H: a+1 < b+1) : a < b :=
  obtain (x: Nat) (Hx: a+1+x = b+1), from lt_elim H,
  have H2: a+x+1 = b+1, from (calc
    a + x + 1 = a + (x + 1) : add_assoc _ _ _
          ... = a + (1 + x) : { add_comm x 1 }
          ... = a + 1 + x : symm (add_assoc _ _ _)
          ... = b + 1 : Hx),
  have H3: a+x = b, from (succ_inj H2),
  show a < b, from (lt_intro H3)
```

Interactive theorem proving

Proof environments take "programs" and compile them into formal proofs. This certifies that the theorem proved is logically valid.

Often, interactive proofs involve calls to computational proof tactics. These are automated methods that search for a proof of a subgoal and return a proof certificate to the environment.

Real closed fields

One situation in which automated tactics come in useful is solving arithmetical inequalities on the real numbers.

$$\text{If } 0 < x < 1 \text{ and } 0 < y < 1, \text{ then } x^{500} + y^{500} > x^{500} \cdot y^{500}$$

Statements of this form make up the theory of real closed fields.

Real closed fields

The theory of real closed fields is decidable: there is an algorithm that, given any sentence of this form, will determine whether it is provable or not (Tarski, 1951). But it has many shortcomings:

- Extremely inefficient
- "Unintuitively" complex
- Can't adapt to modifications of the language
- Very hard to produce a proof certificate

Real closed fields

These shortcomings are surprising. If we consider the theory with just addition, or just multiplication, things are simple.

Can we take advantage of this?

Real closed fields

These shortcomings are surprising. If we consider the theory with just addition, or just multiplication, things are simple.

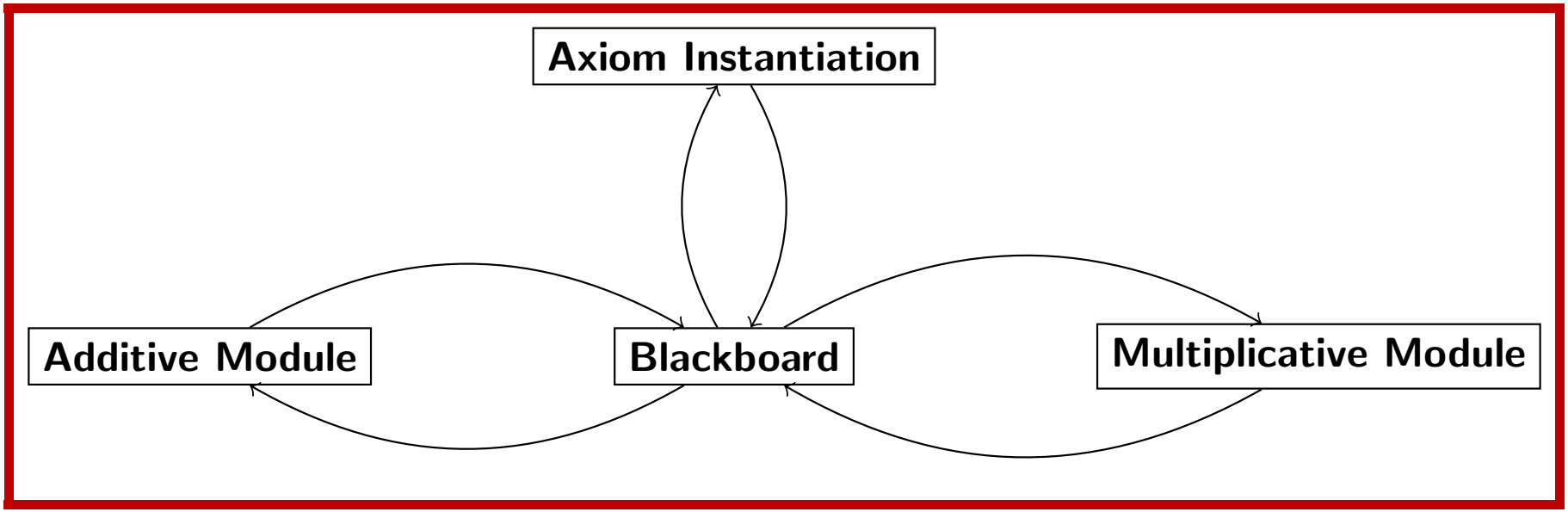
Can we take advantage of this?

Yes, if we're willing to give up decidability.

Polya: a heuristic solver

George Polya, known for his research in both problem-solving heuristics and inequalities.

Polya: an outline



Polya consists of a collection of individualized "modules" that communicate with a central database. The database stores information in a shared language.

Polya: an outline

Given a collection of inequalities, Polya searches for a contradiction according to the following procedure:

- Convert each inequality into a standard normal form.
- Find all additive and multiplicative subterms and give them names.
- Iteratively run modules for additive and multiplicative arithmetic, to learn comparisons between these subterms.
- If the central database receives contradictory information, return unsat.

This is logically equivalent to proving that a conclusion follows from certain hypotheses.

Polya: an outline

When this procedure works, it produces a simple, "unwinding" proof of contradiction. Comparisons between large terms propogate down to subterms, and vice versa.

This seems to mimic a natural or human way of reasoning.

Polya: virtues and vices

Polya can:

1. Solve many heterogeneous problems much more efficiently than the standard decision procedure.
2. Extend beyond basic arithmetic, to reason with exponentials, logarithms, partially interpreted functions, etc.
3. Efficiently produce short, simple proof certificates.
4. Create proofs via a process that is humanly surveyable.

Polya: virtues and vices

Polya cannot:

1. Solve every problem it is given. (No distribution!)
2. Solve simple problems as quickly as the standard procedure.
3. Handle second-order operators: summations, integrals, etc. (Yet!)

We see Polya as a complement, not a replacement, for the traditional decision procedure.

Why is this philosophy?

There is no general consensus about the epistemic status of formal proofs.

- What does a formal proof of a previously known theorem tell us?
- What does a formal proof of a previously unknown theorem tell us?
- Can a proof only be known with the help of computers? What does this tell us?

Thanks for listening!

Some references:

- Avigad, Lewis, and Roux. A heuristic prover for real inequalities. Forthcoming in *Interactive Theorem Proving* (Springer LNCS), 2014.
- Avigad and Friedman. Combining decision procedures for the reals. *Logical Methods in Computer Science*, 2006.
- `https://github.com/avigad/polya`