DPLL-Based Theorem Prover for Coherent Logic

Predrag Janičić
www.matf.bg.ac.rs/~janicic
Joint work with Mladen Nikolić

Work in progress

Automated Reasoning GrOup (ARGO)
Faculty of Mathematics, University of Belgrade, Serbia

Rich Model Toolkit Meeting
Lugano, October 17/18, 2010.

Motivation

- Coherent logic (CL) (also called geometric logic) is a fragment of FOL
- Good features: certain quantification allowed, direct, readable proofs, simple formal proofs...
- However, existing provers for CL are still not very efficient
- SAT and SMT solvers are at rather mature stage
- However, only universal quantification is allowed; producing readable and/or formal proofs is often challenging;
- Goal: build an efficient prover for CL based on SAT/SMT

What is Coherent Logic

- CL formulae are of the form:

$$A_1(\vec{x}) \wedge \ldots \wedge A_n(\vec{x}) \Rightarrow \exists \vec{y}_1\ B_1(\vec{x}, \vec{y}_1) \vee \ldots \vee \exists \vec{y}_m\ B_m(\vec{x}, \vec{y}_m)$$

  (A_i are atomic formulae, B_i are conjunctions of atomic formulae)
- No function symbols
- The problem of deciding $\Gamma \vdash \Phi$ is semi-decidable
- First used by Skolem, recently popularized by Bezem et al.

CL Realm

- A number of theories and theorems can be formulated directly and simply in CL
- Example (Euclidean geometry theorem): *for any two points there is a point between them*
- Most of elementary geometry belongs to CL
- Conjectures in abstract algebra, confluence theory, lattice theory, and many more (Bezem et al)

CL Proof System

- CL has a natural proof system (natural deduction style), based on forward ground reasoning with case distinction
- Existential quantifiers are eliminated by introducing witnesses
- A conjecture is kept unchanged and proved directly (refutation, Skolemization and clausal form are not used)
- CL is a suitable framework for producing readable and for producing formal proofs

CL proof procedures and provers

- Breadth-first proof procedure
- Depth-first proof procedure
- Iterative deepening proof procedure
- ... and their variants
- Several CL provers available

ArgoCLP Prover

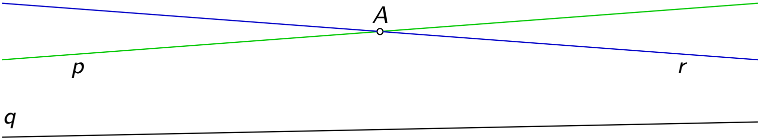
- Developed by Sana Stojanović, Vesna Pavlović, Predrag Janičić (2009)
- Iterative deepening proof procedure
- Sound and complete
- C++, $\approx$ 5000 lines of code
- Can be used for any set of CL axioms

ArgoCLP Features

- A number of techniques that increase efficiency (some of them sacrificing completeness)
- "Clean" proof trace (with all irrelevant inference steps eliminated)
- A formal (Isabelle) proof can be exported
- A proof can be exported in natural language (English, LaTeX formatting)
- Applied primarily in geometry, proved tens of theorems

Geometry Example

Assuming that $p \neq q$, and $q \neq r$, and the line p is incident to the plane α, and the line q is incident to the plane α, and the line r is incident to the plane α, and the lines p and q do not intersect, and the lines q and r do not intersect, and the point A is incident to the plane α, and the point A is incident to the line p, and the point A is incident to the line r, show that $p = r$.



Generated Proof

Let us prove that $p = r$ by reductio ad absurdum.

1. Assume that $p \neq r$.
   2. It holds that the point A is incident to the line q or the point A is not incident to the line q (by axiom of excluded middle).
      3. Assume that the point A is incident to the line q.
         4. From the facts that $p \neq q$, and the point A is incident to the line p, and the point A is incident to the line q, it holds that the lines p and q intersect (by axiom ax_D5).
         5. From the facts that the lines p and q intersect, and the lines p and q do not intersect we get a contradiction.

            Contradiction.

Generated Proof (2)

6. Assume that the point A is not incident to the line q.
   7. From the facts that the lines p and q do not intersect, it holds that the lines q and p do not intersect (by axiom ax_nint_l_l_21).
   8. From the facts that the point A is not incident to the line q, and the point A is incident to the plane α, and the line q is incident to the plane α, and the point A is incident to the line p, and the line p is incident to the plane α, and the lines q and p do not intersect, and the point A is incident to the line r, and the line r is incident to the plane α, and the lines q and r do not intersect, it holds that $p = r$ (by axiom ax_E2).
   9. From the facts that $p = r$, and $p \neq r$ we get a contradiction.

   Contradiction.

Therefore, it holds that $p = r$.

This proves the conjecture.

Nice, but...

... still not efficient

DPLL-based CL Prover — ArgoCaLyPso

- Being developed by Mladen Nikolić and Predrag Janičić
- Motivation: use SAT-like forward-chaining techniques in CL
- Uses some modules of ArgoCLP but with a new search engine
- Uses to some extent the architecture of ArgoSAT (by Filip Marić)
- C++, currently $\approx$ 10000 lines of code
- As the previous version, the prover will be forward-chaining based, but guided by DPLL-style search procedure, will use decide, backjump, learn, etc.

ArgoCaLyPso and FOL

- FOL complicates things
- The trail contains FOL literals
- The axioms make the set of clauses and the set of $\exists$ clauses
- The set of clauses can be extended by instances of existing clauses or resolvents between existing clauses and literals from the trail
- Example: if the set of clauses contains $p(x) \Rightarrow q(x)$ and the trail contains $p(a)$, then the clause $q(a)$ can be added

ArgoCaLyPso and Search

- The rule decide can be performed on ground clauses $A_1 \vee \ldots \vee A_n$ (in DPLL, decide is applied on implicit clauses $p \vee \neg p$)
- Example: *for three different collinear points A, B, and C one of them is between the other two*
- In ArgoCaLyPso, the *axiom of excluded middle* is explicit, and it is not necessarily used
- The search on one branch is finished if $\perp$ (as in SAT) or *the goal formula has been reached*
- When one branch is closed, all irrelevant preceding branching points are skipped in further search (similarly as in the backjump rule)

ArgoCaLyPso and Abstract Transition System

- Described in terms of abstract transition system
- Related to the SAT transition system by Krstić and Goel
- Correctness for SAT has been formally proved (Marić)
- Hopefully, correctness of ArgoCaLyPso could benefit from that proof

ArgoCaLyPso tasks

- Could be used as SAT solver and CL solver
- Could also used for proving by refutation
- Related to EPR solvers (checking satisfiability of $\exists\forall$ fragment)

Conclusions and Future Work

- Hopefully, efficient DPLL-based CL prover
- Applications in geometry (and education)
- Linking to SMT solvers?
- Applications in program synthesis?

Final Note: Come to FATPA 2011

- Fourth Workshop on Formal and Automated Theorem Proving
- Place and time: Belgrade, Serbia, end of January 2011
- Organization: Automated Reasoning GrOup (ARGO)
- Focus: SAT/SMT, geometry reasoning and their applications
- Related to: COST Action IC0901
- Format: informal but very inspiring and productive
- Check: `http://www.argo.matf.bg.ac.rs`/Events