Light-Weight Formal Methods with Heavy-Weight Automation
or
One Year in Sheffield is not enough

Peter Höfner

University of Sheffield

29. June 2007

Introduction

State of the Art: model checking, special purpose automated deduction or interactive theorem proving are needed for formal program development

Our Approach: off-the-shelf automated proof and counterexample search with the right kind of algebra

Results:

- off-the-shelf theorem provers are an alternative
- no special purpose prover needed
- right domain model is needed
- the verification is often done in two layers
- only a first approach
- theorem provers should be able to handle simple arithmetics
- an algebraic verification environment desireable
- a learning approach should be implemented

Prover9 / Mace4 [McCune]

Prover9

- first-order theorem prover
- successor of Otter

Mace4

- counterexample searcher
- same syntax as Prover9

Syntax

```
op(500, infix,   "+").
op(450, infix,   ";").

formulas(sos).
  x+y = y+x.                    % additive commutative monoid
  x+0 = x.
  x+(y+z) = (x+y)+z.
  x;1 = x & 1;x = x.            % multiplicative monoid
  x;(y;z) = (x;y);z.
  x+x = x.                      % additive idempotence
  0;x = 0 & x;0 = 0.            % multiplicative zeroes
  x;(y+z) = x;z+x;y.            % distributivity laws
end_of_list.

formulas(goals).
  add goal here
end_of_list.
```

Part I

Case Studies

Case Study I: Concurrency Control [HöfnerStruth07a]

The Church-Rosser Theorem (algebraic encoding) [Struth02]

$$y^* x^* \leq x^* y^* \Rightarrow (x+y)^* \leq x^* y^*$$

- repeated concurrent executions of x and y can be reduced to an x-sequence followed by a y-sequence
- sequences possible void
- it is usually proved by induction over the number of $y^* x^*$-peaks, i.e., with an external induction measure [Terese03]
- automatically proven in about 3s

Results from Case Study I

- a lot of theorems can be proved fully automatically
  e.g., in Boolean algebra

$$((v \sqcap w) \sqcup (\overline{v} \sqcap x)) \sqcap ((v \sqcap y) \sqcup \overline{\overline{v} \sqcap z}) = (v \sqcap w \sqcap \overline{y}) \sqcup (\overline{v} \sqcap x \sqcap \overline{z})$$

- around 300 theorems proved
- problems with isotonicity (in an equational setting)
- inequational reasoning desireable
- a database should be created

Case Study II: Hoare Logic [HöfnerStruth07a]

Verify the following algorithm for division of an integer n by an integer m

funct $\mathrm{Div}(n)$
  $k := 0$
  $l := n$
  while $m \leq l$ do
    $k := k + 1$
    $l := l - m$
  return k

- Precondition: $0 \leq n$
- Postconditions: $n = km + l$, $0 \leq l$, $l < m$

Translating Div

Div **in Hoare Logic**

$$\{p\} \quad x_1 \,;\, x_2 \,;\, \textsf{while}\, r\, \textsf{do}\, y_1 \,;\, y_2\, \textsf{od} \quad \{q_1 \wedge q_2 \wedge \neg r\}$$

Div **in Modal Kleene algebra** [MöllerStruth06]

$$\langle x_1 x_2 (r y_1 y_2)^* \neg r | p \leq q_1 q_2 \neg r$$

with

$$x_1 \hat{=} \{k := 0\}, \quad x_2 \hat{=} \{l := n\}, \quad y_1 \hat{=} \{k := k+1\}, \quad y_2 \hat{=} \{l := l - m\}, \quad r \hat{=} \{m \leq l\}$$
$$p \hat{=} \{0 \leq n\}, \quad q_1 \hat{=} \{n = km + l\}, \quad q_2 \hat{=} \{0 \leq l\}, \quad q_3 \hat{=} \{l < m\} = \neg r$$

A Two-Layered Proof

Step 1. (abstract simplification)

$$p \le |x_1]|x_2](q_1 q_2) \qquad \wedge \qquad q_1 q_2 r \le |y_1]|y_2](q_1 q_2)$$
$$\Rightarrow \langle x_1 x_2 (r y_1 y_2)^* \neg r | p \le q_1 q_2 \neg r$$

Step2. (concrete calculations)
assignment rule: $p[e/x] \le |\{x := e\}]\, p$

$$\begin{aligned} |x_1]|x_2](q_1 q_2) &= |\{k := 0\}]\ |\{l := n\}](q_1 q_2) \\ &\ge (\{n = km + l\}\{0 \le l\})[k/0][l/n] \\ &= \{n = 0m + n\}\{0 \le n\} \\ &= \{0 \le n\} \\ &= p \end{aligned}$$

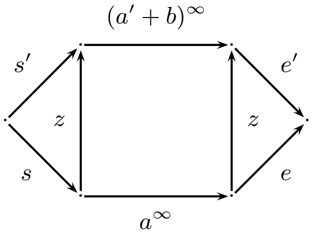
Results from Case Study II

- often two-layered proofs
- concrete calculations, e.g., simple arithmetics are needed
- arithmetics should be included in theorem provers

Case Study III: Refinement Calculus [HöfnerStruth07b]

A Classical Data Refinement Law [BackvonWright98,vonWright02]
Let $b^\infty = b^*$, $za' \le az$, $zb \le z$, $s' \le sz$ and $ze' \le e$. Then

$$s'(a'+b)^\infty e' \le sa^\infty e.$$



Results from Case Study III

- use proved lemmas
- sometimes restricted set of support
  - ping pong between Prover9 and Mace4
  - learning techniques
  - proved refinement laws instead of axioms
- more complicated theorems also possible
  e.g., Back's atomicity refinement law

$$\frac{\begin{array}{c} s \leq sq \qquad a \leq qa \qquad qb = 0 \qquad rb \leq br \\ (a+r+b)l \leq l(a+r+b) \qquad q \leq 1 \\ rq \leq qr \qquad ql \leq lq \qquad r^* = r^\infty \end{array}}{s(a+r+b+l)^\infty q \leq s(ab^\infty q + r + l)^\infty}$$

- transformation between automated proofs and diagramatic reasoning [EbertStruth05]

Part II

Towards An Algebraic Verification Environment

An Algebraic Verification Environment

Database

- create database
- check independencies
- save input/output files

GUI

- restricted set of support
- additional lemmas
- switching between different encodings (equational/inequational)

Embedding various provers

- unified syntax
- counterexample search

An Algebraic Verification Environment

Learning

- ping pong between prover and counterexample search
- restricting set of support
- random addition of verified laws

Decision procedures

- automata (GAP)
- guarded automata
- Büchi automata

different theories

- Kleene algebras [HöfnerStruth07a]
- Refinement algebras [HöfnerStruth07b]
- Relation algebras [HöfnerStruth07c]
- ...

Conclusion

- our approach is only a first step towards a light-weight formal methods with heavy-weight automation
- more than 200 theorems already proved
- complex and long-term software project
- **one year in Sheffield was not enough**

“So Long, and thanks for all the fish.”

Douglas Adams

References

- **[BackvonWright98]** R.-J. Back and J. von Wright, *Refinement Calculus: A Systematic Introduction*. Springer, 1998.
- **[EbertStruth05]** M. Ebert and G. Struth. Diagram chase in relational system development. In M. Minas, editor, *VLFM'04* , volume 127 of *ENTCS*, pages 87–105. Elsevier, 2005.
- **[HöfnerStruth07a]** P. Höfner and G. Struth. Automated Reasoning in Kleene Algebra. In F. Pfennig, editor, *CADE'07*, volume 4603 of *LNAI*, pages 279–294. Springer, 2007.
- **[HöfnerStruth07b]** P. Höfner and G. Struth. Can refinement be automated? In E. Boiten, J. Derrick, G. Smith, editors, *Refinement Workshop 2007*, *ENTCS*. Elsevier, 2007. (to appear)
- **[HöfnerStruth07c]** P. Höfner, G. Schmidt and G. Struth. Automated Reasoning in Relation Algebras and Boolean Algebras with Operators. Technical Report, University Sheffield, 2007. (to appear)
- **[McCune]** W. McCune. Prover9 and Mace4. `http://www.cs.unm.edu/~mccune/prover9`
- **[MöllerStruth06]** B. Möller and G. Struth. Algebras of modal operators and partial correctness. *Theoretical Computer Science*, 351(2):221–239, 2006.
- **[Struth02]** G. Struth. Calculating Church-Rosser proofs in Kleene algebra. In H. de Swart, editor, *RelMiCS 6*, volume 2561 of *LNCS*, pages 276–290. Springer, 2002.
- **[Terese03]** Terese, editor. *Term Rewriting Systems*. Cambridge University Press, 2003.
- **[vonWright02]** J. von Wright. From Kleene Algebra to Refinement Algebra. In E. Boiten, B. Möller, editors, *MPC'02*, volume 2386 of *LNCS*, pages 233–262. Springer, 2002.