Automated Reasoning in Kleene Algebra

Peter Höfner

August 2, 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
  variants of Kleene algebras yield good level of abstraction
- 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

- > 300 theorems proved
- applications in formal methods and computer mathematics

- most of the proofs fully automated from scratch
- some complex theorems needed lemmas (no surprise)

http://www.dcs.shef.ac.uk/∼georg/ka

Prover9 / Mace4

[McCune]

Prover9

- first-order theorem prover
- successor of Otter
- resolution and paramodulation
- software engineer's approach
  - *no* sophisticated encodings
  - *no* refined proof orderings
  - *no* hints or proof planning
  - *no* excessive running times
- stronger results achievable by specialists

Mace4

- counterexample searcher
- same syntax as Prover9

Syntax

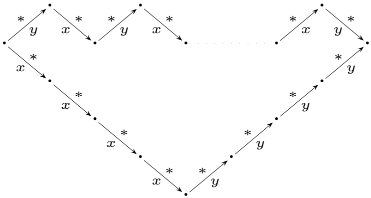
```
op(500, infix,   "+").
op(450, infix,   ";").

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.
```

I. Concurrency Control

[HöfnerStruth07a]



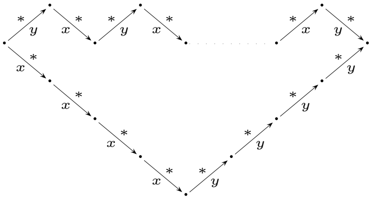
Theorem: Confluent rewrite systems have the Church-Rosser property. (repeated concurrent executions of x and y can be reduced to an x-sequence followed by a y-sequence)

Standard proof: induction over the number ofpeaks, i.e., with an external induction measure [Terese03]

Encoding in Kleene algebra: $y^*x^* \leq x^*y^* \Rightarrow (x+y)^* \leq x^*y^*$

I. Concurrency Control

[HöfnerStruth07a]



Theorem: Confluent rewrite systems have the Church-Rosser property. (repeated concurrent executions of x and y can be reduced to an x-sequence followed by a y-sequence)

Standard proof: induction over the number ofpeaks, i.e., with an external induction measure [Terese03]

Encoding in Kleene algebra: $y^*x^* \leq x^*y^* \Rightarrow (x+y)^* \leq x^*y^*$

Prover9: $< 3s$

II. Concurrency Control

[HöfnerStruth07a]

Theorem: If a rewrite system quasi-commutes over another one, then the union of the rewrite systems terminates iff the individual systems do.

Standard proof: reasoning about infinite sequences

Remark: challenge problem for computational algebras (Ernie Cohen)

II. Concurrency Control

[HöfnerStruth07a]

Theorem: If a rewrite system quasi-commutes over another one, then the union of the rewrite systems terminates iff the individual systems do.

Standard proof: reasoning about infinite sequences

Remark: challenge problem for computational algebras (Ernie Cohen)

Encoding: $yx \leq x(y+x)^* \Rightarrow ((x+y)^\omega = 0 \Leftrightarrow x^\omega + y^\omega = 0)$
$^\omega$ models infinite iteration as greatest fixedpoint

II. Concurrency Control

[HöfnerStruth07a]

Theorem: If a rewrite system quasi-commutes over another one, then the union of the rewrite systems terminates iff the individual systems do.

Standard proof: reasoning about infinite sequences

Remark: challenge problem for computational algebras (Ernie Cohen)

Encoding: $yx \leq x(y+x)^* \Rightarrow ((x+y)^\omega = 0 \Leftrightarrow x^\omega + y^\omega = 0)$
$^\omega$ models infinite iteration as greatest fixedpoint

Prover9: $\sim 235s$

Results from Case Studies I and II

- a lot of theorems can be proved fully automatically (from scratch)
- around 300 theorems proved
- problems with isotonicity (in an equational setting)
- inequational reasoning desireable
- a database should be created

III. Hoare Logic

[HöfnerStruth07a]

Exercise: Verify the following algorithm for integer division

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

- precondition: $0 \leq n$
- postconditions: $n = km + l$, $0 \leq l$, $l < m$

Encoding in Hoare Logic: $\{p\}\ x_1\,;x_2\,;\text{while}\,r\,\text{do}\,y_1\,;y_2\,\text{od}\ \{q_1 \wedge q_2 \wedge \neg r\}$

III. Hoare Logic

[HöfnerStruth07a]

Modal Kleene algebra [MöllerStruth06]

- Kleene algebra extended by *tests* and *modal operators* ($\langle x|p$, $|x\rangle p$, $[x|p$, $|x]p$)
- $\langle x|p$ is set of all states with at least one x-precessor in p

Encoding in Kleene algebra: $\langle x_1 x_2 (r y_1 y_2)^* \neg r | p \leq q_1 q_2 \neg r$

with

$$x_1 \hat{=} \{k := 0\},\quad x_2 \hat{=} \{l := n\},\quad y_1 \hat{=} \{k := k+1\},\quad y_2 \hat{=} \{l := l - m\},\quad r \hat{=} \{m \leq l\}$$

$$p \hat{=} \{0 \leq n\},\quad q_1 \hat{=} \{n = km + l\},\quad q_2 \hat{=} \{0 \leq l\},\quad q_3 \hat{=} \{l < m\} = \neg r$$

Hoare Logic

Two-layered proof:

- *Step 1 (algebraic calculation)*
  - fully automated

$$p \le |x_1]|x_2](q_1q_2) \qquad \wedge \qquad q_1q_2r \le |y_1]|y_2](q_1q_2)$$
$$\Rightarrow \langle x_1x_2(ry_1y_2)^*\neg r|p \le q_1q_2\neg r$$

- *Step 2 (domain-specific reasoning)*
  - should be automated
  - assignment rule: $p[e/x] \le |\{x := e\}]\, p$

$$\begin{aligned} |x_1]|x_2](q_1q_2) &= |\{k := 0\}]\; |\{l := n\}](q_1q_2) \\ &\ge (\{n = km + l\}\{0 \le l\})[k/0][l/n] \\ &= \{n = 0m + n\}\{0 \le n\} \\ &= \{0 \le n\} \\ &= p \end{aligned}$$

Results from Case Study III

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

IV: 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

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

$$\begin{tikzcd}
 & \cdot \arrow[r, "(a'+b)^\infty"] & \cdot \arrow[dr, "e'"] & \\
\cdot \arrow[ur, "s'"] \arrow[dr, "s"'] & & & \cdot \\
 & \cdot \arrow[uu, "z"] \arrow[r, "a^\infty"'] & \cdot \arrow[uu, "z"] \arrow[ur, "e"'] &
\end{tikzcd}$$

Results from Case Study IV

- use proved lemmas
- sometimes restricted set of support
  - ping pong between Prover9 and Mace4
  - learning techniques (SRASS)
  - 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]

Further Applications

- Linear temporal logic:
  - axioms are theorems or domain-specific
  - temporal reasoning about infinite systems
- Dynamic logic: axioms are theorems of modal Kleene algebra
- Modal correspondence theory:
  - Löb's formula related to frame property
  - calculational reasoning about infinite behaviour
  - alternative to translational approach
- Program refinement:
  - experiments in other variants of Kleene algebra
  - some complex refinement laws for action systems verified
- Relational methods [HöfnerSchmidtStruth07c]:
  - > 100 theorems in relation algebra verified
  - example: $zx \sqcap y \leq (z \sqcap yx^{\circ})(x \sqcap z^{\circ}y)$

http://www.dcs.shef.ac.uk/∼georg/ka

Research Questions

- implementation of inequational reasoning (chaining calculi)
  - we encoded inequalities as predicate
  - equational encoding fails at some points
  - problems in applying monotonicity
- integration of domain-specific solvers and decision procedures
  - e.g., Presburger arithmetics
  - promises full automatisation of partial correctness analysis

Mögliche Themen für Abschlussarbeiten

- Erstellung einer Theoremdatenbank mit passender GUI
- Vergleich und Tunign von Theorem Beweiseren im Hinblick auf Kleene Algebra
  (Prover9, Vampire, Waldmeister, SPASS, SPASS+T, E, EP, SRASS, FLOTTER...)
- Hypothesis Learning

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)
- **[HöfnerSchmidtStruth07c]** 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.