Automated Reasoning in Kleene Algebra

Peter Höfner Georg Struth

The University Of Sheffield.

July 18, 2007

Automated Deduction in Formal Methods

Observation: Formal methods are dominated by model checking and interactive theorem proving

Automated Deduction in Formal Methods

Observation: Formal methods are dominated by model checking and interactive theorem proving

Automated deduction:

- special purpose provers seem necessary
- difficult to design and implement

Automated Deduction in Formal Methods

Observation: Formal methods are dominated by model checking and interactive theorem proving

Automated deduction:

- special purpose provers seem necessary
- difficult to design and implement

Question: How can we integrate verification techniques into automated deduction?

Automated Deduction in Formal Methods

New approach: off-the-shelf theorem provers and counterexample search with computational algebras

Idea:

- algebras provide first-order equational calculus
- this can be handled by resolution and paramodulation

Results

- variants of Kleene algebras yield good level of abstraction

- > 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

The Setting

Theorem prover:

- Prover9
- 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

The Setting

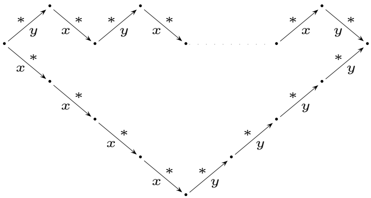
Algebra:

- Kleene algebras $(K, +, \cdot, 0, 1, ^*)$
  - elements are actions
  - $+$ models choice
  - $\cdot$ models sequential composition
  - * models finite iteration as a least fixedpoint

$$1 + xx^* = x^*, \qquad y + xz \leq z \Rightarrow x^*y \leq z$$

- rich model class: languages, relations, paths, traces, ...

Concurrency Control

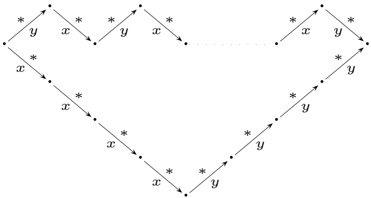


Theorem: Confluent rewrite systems have the Church-Rosser property.

Standard proof: induction over the number of peaks

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

Concurrency Control



Theorem: Confluent rewrite systems have the Church-Rosser property.

Standard proof: induction over the number of peaks

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

Prover9: $< 3s$

Remarks:

- induction handled implicitly
- refinement law for concurrent action systems

Concurrency Control

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)

Concurrency Control

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

Concurrency Control

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: $\sim 235s$

Hoare Logic

Exercise: Verify the following algorithm for integer division

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

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

Hoare Logic

Modal Kleene algebra

- 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

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

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

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

Further Applications

- Hoare logic: Hoare rules are theorems of modal Kleene algebra
- 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

some proofs require hypothesis learning

Other Applications

- Program refinement [HöfnerStruth07]:
  - experiments in other variants of Kleene algebra
  - some complex refinement laws for action systems verified

- Relational methods [HöfnerSchmidtStruth07]:
  - > 100 theorems in relation algebra verified
  - example: $zx \sqcap y \leq (z \sqcap yx^\circ)(x \sqcap z^\circ y)$
  - semantic basis for Z and B

`http://www.dcs.shef.ac.uk/∼georg/ka`

Conclusion

- automated deduction has much to offer for formal methods (Alan Bundy)
- off-the-shelf theorem provers with computational algebras works
- light-weight formal methods with heavy-weight automation
- interesting benchmarks for CADE-community
- but many questions open

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