Solution of the Robbins Problem

(and Other Adventures in Automated Deduction)

William McCune

Mathematics and Computer Science Division
Argonne National Laboratory
U.S.A.

Presented at the University of Waterloo
March 11, 1997

Outline

- The programs: Otter, MACE, EQP
- How the programs work
- Some results with Otter and MACE
- The Robbins problem
- The QED project

The Programs

- Otter and MACE apply to first-order predicate logic with equality.
  - Otter searches for proofs.
  - MACE (Models And Counter-Examples) searches for finite models.
- EQP searches for proofs in equational logic.
  - Associative-commutative unification and matching.
- Practical application
  - Otter, EQP: "small" theories.
  - MACE: simple formulas, small models.

Otter Example (EQP is similar)

A simple theorem: $xx = e$ groups are commutative.

Otter input:

```
-----------------------------------------------
op(400, xfx, *).
set(auto). clear(detailed_history).

list(usable).
e * x = x.          % A basis for group theory
i(x) * x = e.
(x * y) * z = x * (y * z).
x * x = e.          % Hypothesis
B * A != A * B.     % Denial of commutativity
end_of_list.
-----------------------------------------------
```

Otter proof:

```
---- PROOF at   0.11 sec -----

1  B*A!=A*B.          []
2  e*x=x.             []
6  (x*y)*z=x* (y*z).  []
8  x*x=e.             []
11 x* (x*y)=y.        [para_into,6,8,demod,2,flip.1]
15 x* (y* (x*y))=e.   [para_into,6,8,flip.1]
17 x*e=x.             [para_into,11,8]
25 x* (y*x)=y.        [para_from,15,11,demod,17,flip.1]
29 x*y=y*x.           [para_from,25,11]
30 $F.                [binary,29,1]
```

Paramodulation

- Paramodulation for equations:

$$\frac{\alpha_1 = \beta, \quad \gamma[\alpha_2] = \delta \quad \text{[with } \alpha_1\sigma \equiv \alpha_2\sigma\text{]}}{(\gamma[\beta] = \delta)\sigma}$$

  σ is a substitution of terms for variables that makes α_1 identical to α_2.

- Example:

$$\frac{\boxed{x' * x} = e, \quad (\boxed{y * z'}) * z = y}{e * z = z''}$$

  The terms $x * x'$ and $y * z'$ are *unified*, instantiating y to z''.

- EQP has Associative-Commutative (AC) unification.
  - If an operation is AC, build those properties into the inference rule.
  - The condition for AC paramodulation is [with $\alpha_1\sigma$ AC-identical to $\alpha_2\sigma$]

MACE Example

Find a noncommutative group.

MACE input:

```
----------------------------------------------
  op(400, xfx, *).

  list(usable).
  0 * x = x.          % A basis for group theory
  i(x) * x = 0.
  (x * y) * z = x * (y * z).

  A * B != B * A.     % Denial of commutativity
  end_of_list.
----------------------------------------------
```

MACE output (with flags -n6 -c -p):

```
  ================== Model #1 at 3.04 seconds:

    * | 0 1 2 3 4 5          i:  0 1 2 3 4 5
    --+------------          ---------------
    0 | 0 1 2 3 4 5              0 1 2 4 3 5
    1 | 1 0 3 2 5 4
    2 | 2 4 0 5 1 3
    3 | 3 5 1 4 0 2
    4 | 4 2 5 0 3 1          A:  1
    5 | 5 3 4 1 2 0          B:  2
```


How the Programs Work

Otter and EQP:

- The programs are not interactive—the user supplies a conjecture and a search strategy.
- All proofs are by contradiction.
- The search is driven by a simple loop.
  - Apply inference rules selected by the user.
  - Simplify derived clauses and test for subsumption.
  - Apply search strategies specified by the user.
  - Stop when a contradiction is found or when resources are exhausted.
- The user frequently makes a sequence of runs, examining the output and adjusting the search parameters.

MACE:

- The user doesn't need to know, because little control is possible.

Weaknesses

- Otter and EQP
  - Theories with many operations.
  - Defined operations.
  - Most proofs are longer than necessary.
  - Non-Horn theories, set theory (Otter).
  - You have to learn a lot about the programs before you can use them for difficult problems.
- MACE
  - Complex statements.
  - Large models.
  - If finds too many isomorphic models.

Some Results From the Programs

- Otter
  - Single axioms for group-like and lattice-like algebras
    Groups: $x \cdot (y \cdot (((z \cdot z^{-1}) \cdot (u \cdot y)^{-1}) \cdot x))^{-1} = u$
    Lattices: 355 symbols $\longrightarrow$ 79 symbols
  - First first-order proofs
  - Algebraic geometry
  - Semigroup test cases
- MACE
  - Quasigroup problems from Frank Bennett
  - Lattice-like problems from Padmanabhan
- *Automated Deduction in Equational Logic and Cubic Curves*, by McCune, Padmanabhan, Otter, and MACE, Springer LNCS #1095 (1996).
- EQP
  - Lemmas related to the Robbins problem
  - The Robbins problem

The Robbins Problem

- A well-known basis for Boolean algebra [Huntington, 1933]:

$x + y = y + x$	(C) Commutativity
$(x + y) + z = x + (y + z)$	(A) Associativity
$\overline{(\overline{x} + y)} + \overline{(\overline{x} + \overline{y})} = x$	(H) Huntington

- The Robbins Problem:

$$\overline{(\overline{(x + \overline{y})} + \overline{(x + y)})} = x \quad \text{(R) Robbins}$$

  Is $\{C, A, R\}$ basis for Boolean algebra?

- A proof of the following would solve the problem.

  Robbins $\Longrightarrow_{\mathrm{AC}}$ Huntington.

Easy Lemmas About the Robbins Equation

An easy one:

$$\left\{ \begin{array}{l} \text{Robbins} \\ \forall x\ (\overline{\overline{x}} = x) \end{array} \right\} \Longrightarrow_{\mathrm{AC}} \text{Huntington.}$$

Another easy one (but not first-order):

$$\left\{ \begin{array}{l} \text{Robbins} \\ \text{over a finite set} \end{array} \right\} \Longrightarrow_{\mathrm{AC}} \text{Huntington.}$$

Existence of a zero:

$$\left\{ \begin{array}{l} \text{Robbins} \\ \exists x \forall y\ (x + y = y) \end{array} \right\} \Longrightarrow_{\mathrm{AC}} \text{Huntington.}$$

Idempotence:

$$\left\{ \begin{array}{l} \text{Robbins} \\ \forall x\ (x + x = x) \end{array} \right\} \Longrightarrow_{\mathrm{AC}} \text{Huntington.}$$

Existence of an idempotent element:

$$\left\{ \begin{array}{l} \text{Robbins} \\ \exists x\ (x + x = x) \end{array} \right\} \Longrightarrow_{\mathrm{AC}} \text{Huntington.}$$

(Try these with *Son of BirdBrain*.)

Hard Lemmas About the Robbins Equation

These were formulated and proved by Steve Winker c. 1980 [J. Algebra 1992].

Lemma 1 (An absorption):

$$\left\{ \begin{array}{l} \text{Robbins} \\ \exists x \exists y\ (x + y = x) \end{array} \right\} \Longrightarrow_{\mathrm{AC}} \text{Huntington.}$$

Lemma 2 (An absorption within complement):

$$\left\{ \begin{array}{l} \text{Robbins} \\ \exists x \exists y\ (\overline{x + y} = \overline{x}) \end{array} \right\} \Longrightarrow_{\mathrm{AC}} \text{Huntington.}$$

Winker's proofs are not entirely first-order (they use some induction).

EQP proved these lemmas automatically in 1992 and 1995. (Proofs are on the Web.)

Search Strategy

Working on hard problems is *not* simple matter of pushing the $\boxed{\text{start}}$ button.

- The basic search method for equational problems is similar to Knuth-Bendix completion.
- Search parameters
  - unification restrictions *
  - paramodulation restrictions *
  - rewrite strategy
  - size of inference tasks
  - order of inference tasks
  - conserving resources
- Achieving well-behaved searches
  - Experiment with many search strategies.
  - Kill searches that don't look good, adjust some search parameters, and try again.

Unification Restrictions

Associative-commutative unification of a pair of terms:

- An *AC unifier* is a substitution of terms for variables that makes two terms AC-identical.
- Find a set S of AC-unifiers such that every AC-unifier is an instance of a member of S. (A finite S always exists.)
- In theory, the smallest S can be doubly exponential.
- In practice, S is frequently very big. The pair of terms $x + x + x$ and $y + z + u + v$ has 1,044,569 most general AC unifiers.

Super-n: A new strategy for AC unification

- Use only the simplest AC unifiers.
- This is an incomplete (and severe) restriction.
- Applied to the prolific pair above with $n = 0$, we get 139 AC unifiers.
- In practice, it doesn't seem to block many proofs.

Paramodulation Restrictions

- We use the Knuth-Bendix (superposition) restrictions.
- "Basic" paramodulation
  - Reduce redundancy in the search.
  - Controls the *order* of equality replacement steps.
  - Prevents replacement of terms that arise by instantiation (deeply nested terms).
  - Complete for several paramodulation proof procedures.
  - See the EQP proof below.

The Final Assault

- Five weeks of real time, five weeks of CPU time.
- About 50 searches with various strategies, including 14 multiday searches.
  - Attempts to achieve well-behaved search
- Success: Robbins $\Longrightarrow_{\text{AC}} \exists x \exists y \; (\overline{x + y} = \overline{x})$
  - 8 day search, 30 megabytes of memory.
  - "Basic" paramodulation, super-0 AC restriction.
  - Small inference tasks.
  - Equal parts breadth-first, shortest-first search.
  - Discard equations with more than 70 symbols.
- Better proofs and shorter proofs were found later.

Solutions of the Robbins Problem

First proof (EQP, 8 days of search):

$$\text{Robbins} \Longrightarrow_{\text{AC}} \exists x \exists y\ (\overline{x+y} = \overline{x})$$

Another proof (EQP, a bit more search):

$$\text{Robbins} \Longrightarrow_{\text{AC}} \exists x \exists y\ (x+y = x)$$

Yet another proof (EQP, 10%–90% more search):

$$\text{Robbins} \Longrightarrow_{\text{AC}} \text{Huntington}$$

A Proof that Solves the Robbins Problem

Theorem. *Robbins* $\Longrightarrow_{\text{AC}}$ $\exists c \exists d\ (c+d=c)$.

Outline of proof.

7	$\overline{\overline{\overline{x}+y}+\overline{x+y}} = y$	[Robbins Equation]
10	$\overline{\overline{\overline{x+y}+\overline{x}+y}+y} = \overline{x+y}$	[7 → 7]
11	$\overline{\overline{\overline{\overline{x}+y}+x+y}+y} = \overline{\overline{x}+y}$	[7 → 7]
29	$\overline{\overline{\overline{\overline{x}+y}+x+2y}+\overline{\overline{x}+y}} = y$	[11 → 7]
54	$\overline{\overline{\overline{\overline{\overline{x}+y}+x+2y}+\overline{\overline{x}+y}+z}+\overline{y+z}} = z$	[29 → 7]
217	$\overline{\overline{\overline{\overline{\overline{x}+y}+x+2y}+\overline{\overline{x}+y}+\overline{y+z}+z}+z} = \overline{y+z}$	[54 → 7]
674	$\overline{\overline{\overline{\overline{\overline{\overline{x}+y}+x+2y}+\overline{\overline{x}+y}+\overline{y+z}+z}+z+u}+\overline{\overline{y+z}+u}} = u$	[217 → 7]
6736	$\overline{\overline{\overline{\overline{3x}+x}+\overline{3x}}+\overline{\overline{\overline{3x}+x}+5x}} = \overline{\overline{3x}+x}$	[10 → 674]
8855	$\overline{\overline{\overline{3x}+x}+5x} = \overline{3x}$	[6736 → 7 :54]
8865	$\overline{\overline{\overline{\overline{3x}+x}+\overline{3x}+2x}+\overline{3x}} = \overline{\overline{3x}+x}+2x$	[8855 → 7]
8866	$\overline{\overline{\overline{3x}+x}+\overline{3x}} = x$	[8855 → 7: 11]
8870	$\overline{\overline{\overline{\overline{3x}+x}+\overline{3x}+y}+\overline{x+y}} = y$	[8866 → 7]
8871	$\overline{\overline{3x}+x}+2x = 2x$	[8865 :8870]

Equation 8871 asserts the existence of an object c, namely $2x$, and an object d, namely $\overline{\overline{3x}+x}$, such that $c+d=c$. Q.E.D.

Step 6736 in Detail

Premises:

$$\overline{\overline{\overline{x+y}+\overline{x}+y}+y} = \overline{x+y} \qquad (10)$$

$$\overline{\overline{\overline{\overline{\overline{\overline{x}+y}+x+2y}+\overline{\overline{x}+y}+\overline{y+z}+z}+z+u}+\overline{\overline{y+z}+u}} = u \qquad (674)$$

With (10), let x be $3v$, and y be $\overline{\overline{3v}+v}+2v$:

$$\underbrace{\overline{\overline{\overline{3v+\overline{\overline{3v}+v}+2v}+\overline{3v}+\overline{\overline{3v}+v}+2v}+\overline{\overline{3v}+v}+2v}} = \overline{3v+\overline{\overline{3v}+v}+2v} \qquad (10')$$

With (674), let x be $3v$, y be v, z be $2v$, and u be $\overline{\overline{3v}+v}$:

$$\overline{\underbrace{\overline{\overline{\overline{\overline{\overline{3v}+v}+5v}+\overline{\overline{3v}+v}+\overline{3v}+2v}+2v+\overline{\overline{3v}+v}}}+\overline{\overline{3v}+\overline{\overline{3v}+v}}} = \overline{\overline{3v}+v} \qquad (674')$$

Replace the braced term of (674′), which is AC-identical to the left-hand side of (10′), with the right-hand side of (10′):

$$\overline{\overline{3v+\overline{\overline{3v}+v}+2v}+\overline{\overline{3v}+\overline{\overline{3v}+v}}} = \overline{\overline{3v}+v}.$$

Rename the variable and rearrange:

$$\overline{\overline{\overline{\overline{3x}+x}+\overline{3x}}+\overline{\overline{\overline{3x}+x}+5x}} = \overline{\overline{3x}+x}. \qquad [10{\rightarrow}674]\ (6736)$$

EQP's Robbins Proof

- Why did it work?
  - a few small equations
  - lots of experience with easier problems
  - new search pruning methods
  - efficient code
  - luck
- Will it work on other hard problems?

Is The Proof Correct???

- Computer proofs are always questionable.
- We have an explicit proof
  - checked by several people
  - checked by many other programs
  - checked by a special proof-checking program
- Other computer proofs
  - The four-color theorem
  - Finite projective plane of order 10

The Proposed QED Project

- Formalize and computer-check big chunks of (all of) mainstream mathematics.
- To the foundations.
- Applications
  - Pure mathematics
  - System verification
  - Modeling and simulation
- So that nonexperts can use the mathematics correctly, with confidence.
- The MIZAR project – a good start (pure mathematics).

Network Addresses

- The Robbins Problem:

  `http://www.mcs.anl.gov/home/mccune/ar/robbins/`

- Automated Deduction at Argonne:

  `http://www.mcs.anl.gov/home/mccune/ar/`

  - including Otter, MACE, EQP, papers, QED, and *Son of BirdBrain*

- Me:

  `mccune@mcs.anl.gov`
  `http://www.mcs.anl.gov/home/mccune/`

- These slides:

`ftp://info.mcs.anl.gov/pub/Otter/Slides/waterloo-97.ps.Z`