MSc COMP6012

Automated Reasoning

Who, What, When, Where, Why?

Renate Schmidt

(email: `schmidt@cs.man.ac.uk`)

Alan Williams

(email: `alanw@cs.man.ac.uk`)

September 2007

Why?

System Design:

- The Pentium Bug
- The Pentium II Bug
- Arriane 5 Failure, 4 June 1996
- software + hardware specification and design errors . . .
- increasing design complexity . . .

Or. . .

- Mathematical Logical Foundations

What?

- (System property or component description via formal logic)
- Logical reasoning
- Automation: decision procedures
- Advanced techniques for efficiency
- Associated theoretical concepts, e.g. soundness and completeness

Course Outline

When?	**Where?**
Period 1, Semester 1	Lectures: 2.15
Mondays	Labs: 2.25a

A Course of Two Halves:

1. Formal Logic and Automated Reasoning (AJW)
2. Advanced Automated Reasoning (RenS)

Pre-requisites: Familiarity with Propositional Logic

Part I: Formal Logic and Automated Reasoning

- Classical Propositional Logic
- First-order Predicate Logic
- Automated Reasoning: Methods and Tools, including
  - resolution
  - logic programming

Reasoning Example

Assumptions:

IF I live in MANCHESTER THEN it is SUNNY

IF it is SUNNY THEN I need a PARASOL

Conclusion:

IF I live in MANCHESTER THEN I need an PARASOL

The Resolution Principle

Assumptions: $(A \vee B) \quad (C \vee \neg B)$

Conclusion: $(A \vee C)$

The basis of

- Automated Theorem-proving: e.g. Vampire (Andrei Voronkov)
- Logic Programming: e.g. Prolog

Logic Programming and Prolog

Prolog Program — rules and facts:

```
ancestor(X,Y) :- parent(X,Y).

ancestor(X,Y) :- parent(X,Z),
                 ancestor(Z,Y).

parent(sue,toby).
parent(roy,sue).
```

Run program:

```
?- ancestor(roy,X).

X = sue;
X = toby;
```

Part II: Advanced Techniques

Why:

- The *basic resolution calculus* is very simple
  - Just two rules
  - Extremely prolific at generating new conclusions
  - Inefficient, impracticable
- Advanced techniques are available
- Part II is devoted to *Advanced Automated Reasoning*

Emphasis in Part II

- *Introduction* to advanced theorem proving
- *Fundamentals of first-order resolution*
  - Selection of important topics
  - Many examples and exercises
- Important basic properties of inference systems
  - *Soundness* $\leadsto$ no false conclusions are drawn
  - *Completeness* $\leadsto$ all true conclusions are drawn
  - *Efficiency* $\leadsto$ avoid unnecessary inferences

Modern Resolution Framework

- Best provers use resolution
- Modern resolution framework = an extension of *basic resolution calculus* with:
  - Powerful *search control* mechanisms
    $\leadsto$ ordering and selection refinements
  - General notion of *redundancy*
    $\leadsto$ simplification and optimisation techniques
  - *optimised transformations into clausal form*
- Has many uses and applications
  - This course: *verfication of Neuman-Stubblebine key exchange protocol*
- Fast implementations: *Vampire*, *(M)SPASS*

Topics of Current Research

- Developing practical *decision procedures*
- Handling *specific theories* (equality, transitive relations, . . .) or *logics* (description logics, modal logics, . . .)
- Implementing *fast* automated theorem provers
- *Relationship* between different proof methods (resolution, tableau, ND, . . .)
- *Combining* different proof methods and different provers
- Specific applications:
  - *Software engineering*
  - *Ontologies and the semantic web*
  - *Multi-agent systems*

Lectures:

- include Examples Classes
- paper-based Exercises (some assessed)

Labs:

- Approximately 35% of Teaching Time is lab
- Prolog
  - build a resolution theorem-prover
  - extend with advanced techniques
- try out MSPASS, Vampire

Reading List

- ‘Course Text’:

  Kelly, J. (1997), *The Essence of Logic.* Prentice Hall.
- Recommended:

  Schöning, U. (1989), *Logic for Computer Scientists.* Birkhäuser.

  Fitting, M. (1990), *First-Order Logic and Automated Theorem Proving*. Springer.

Assessment

- Examination (40%)
  - open book
- Exercises and labs (60%)