COMP 426:Automated Reasoning
Advanced undergraduate course

Brigitte Pientka

School of Computer Science

McGill University

Outline

- Motivation
- Administrative issues

Motivation

- Hardware and software is pervasively used in many (safety-critical) applications.

Motivation

- Hardware and software is pervasively used in many (safety-critical) applications.
- We need to understand how to
  - formally characterize and prove important invariants
  - increase the confidence in software

Motivation

- Hardware and software is pervasively used in many (safety-critical) applications.
- We need to understand how to
  - formally characterize and prove important invariants
  - increase the confidence in software
- We need tools to
  - specify and verify formal statements
  - catch flaws early in the development

Human costs of bugs

- Software failure can cause loss of lifes.

Human costs of bugs

- Software failure can cause loss of lifes.
- Bugs in medical software
  - 1985: Software-design flaws lead to radiation overdosis in US. and Canadian patients (Therac-25)
  - 1997: Software-logic error causes infusion pump to deliver lethal doses of morphine sulfate
  - 2001: Panamian cancer patients die following overdosis of radiation due to faulty software

Economic costs of bugs

- Software bugs cost north-american economy over 10 billion US Dollars per year.

Economic costs of bugs

- Software bugs cost north-american economy over 10 billion US Dollars per year.
- Bugs in hardware and software systems
  - 1994: FDIV bug (floating point arithmetic) in Intel Pentium processor

    Cost: US 500 million

Economic costs of bugs

- Software bugs cost north-american economy over 10 billion US Dollars per year.
- Bugs in hardware and software systems
  - 1994: FDIV bug (floating point arithmetic) in Intel Pentium processor
    Cost: US 500 million
  - 1996: Ariane 5 space craft crashes 40 sec after take-off due to floating point conversion error
    Cost: US 500 million

Economic costs of bugs

- Software bugs cost north-american economy over 10 billion US Dollars per year.
- Bugs in hardware and software systems
  - 1994: FDIV bug (floating point arithmetic) in Intel Pentium processor
    Cost: US 500 million
  - 1996: Ariane 5 space craft crashes 40 sec after take-off due to floating point conversion error
    Cost: US 500 million
  - 2004: Mars Rover is paralyzed for 5 days. After communication breakdown, it starts continuously rebooting.

Limits of testing

- Testing is good, but not good enough.

Limits of testing

- Testing is good, but not good enough.
- Slow
- Too many possibilities
  2^{160} possible input pairs for floating point adder
  even higher number of states for more complex architectures
- Sometimes even a huge weight of empirical evidence is misleading.

Industrial efforts in automated reasoning

- Hardware verification: IBM, Intel, Motorolla, Bell Laboratories, Saab Ericsson, etc
- Software verification examples:
  - Active Missile Decoy 'Nulka' system
  - Hong Kong Mass Transit Railway Corporation
  - Safety analysis for Queensland Rail.
  - Smart Card Verification

Future of software

- Correctness = major concern for all software producers.
- Especially for providers of platform software.
- Today: Microsoft employs a variety of reasoning tools
  - theorem provers, model checkers, type inference engines, static analysis tools
  - Since 2001: annotate legacy code and verify important properties
  - Today: Every code developed must pass certain checks before it can be committed to repository
  - Every computer has these tools installed!

Academic efforts: Verified Software

- Neglected aspect: language we write programs in
- We need tools to
  - Model and specify programming languages
  - Experiment easily with language extensions
  - Mechanically check their meta-theoretic properties
- POPLmark Challenge [Pierce et al 05]
  "Mechanically check every POPL paper by 2010!"

Logical framework allows us to represent, execute, and reason about formal systems.

Academic efforts: Flyspeck

- Verify Kepler Conjecture
  - Oldest problem in discrete geometry
  - Part of Hilbert's 18th problem
- Proven in 1989 by Thomas Hales

  It is one of the most complicated mathematical proofs!
- After 4 years of refereeing, 12 referees stated they are 99% certain of the correctnes of the proof

Verified Mathematics ?

"The news from the referees is bad, from my perspective. They have not been able to certify the correctness of the proof, and will not be able to certify it in the future, because they have run out of energy to devote to the problem. This is not what I had hoped for."

Robert MacPherson, editor of the Annals of Mathematics

- These situations will occur more often.
- What level of correctness do we want?
- No computer proof will be accepted.

This course

- A thorough introduction to modern constructive logic and its properties.

  Logic is the foundation for specifying formal systems, and has numerous numerous applications in automated reasoning, formal software and hardware verification, programming languages, mathematics etc.

- Centerpiece : Design and implementation of a succession of different theorem provers

- Fundamental principles which lead to efficient implementations

Warning:

- Speed is not everything!
- I am not interested in performance tuning.
- I am interested in
  - transferrable techniques.
  - correctness.
  - completeness.
  - deep fundamental principles.
  - elegant proofs.

Learning outcome

- Good working knowledge of logic
  - Connection between logic and computation
  - Prove statements!
  - Recognize a proof and know how to do it!
  - Fundamental principles underlying logic
- Understand fundamental techniques for designing and implementing automated theorem provers
- Be able to transfer techniques to different logics

Topics

- Natural deduction
- Sequent calculus
- Curry-Howard Isomorphism
- Propositional theorem proving
- First-order theorem proving
- Logic and functional programming
- . . .

Coordinates

Prof	Brigitte Pientka
e-mail	bp@cs.mcgill.ca
office	107N McConnell
office hours	Wed, 2:00pm – 3:00pm
TA	David Xi Li
e-mail	xli53@cs.mcgill.ca
office hours	tba.

Course web-page

http://www.cs.mcgill.ca/~bpientka/courses/atp

Acknowledgements

Part of this course material is based on material which was mainly developed by

Frank Pfenning, Carnegie Mellon University