Verification of Systems With Non-Boolean Variables

How to find errors in computer programs

Jesper Møller [jm@it.edu]

Department of Innovation
IT University of Copenhagen
Denmark

Introduction

- More and more products contain **embedded computers:**

Introduction

- More and more products contain **embedded computers:**
  - mobile phones
  - microwave ovens
  - medical equipment
  - cars
  - satellites
  - airplanes

Introduction

- More and more products contain **embedded computers:**
  - mobile phones
  - microwave ovens
  - medical equipment
  - cars
  - satellites
  - airplanes
- Designed by humans $\Longrightarrow$ May contain errors

Introduction

- More and more products contain **embedded computers:**
  - mobile phones
  - microwave ovens
  - medical equipment
  - cars
  - satellites
  - airplanes
- Designed by humans $\Longrightarrow$ May contain errors
- Types of errors:
  - Irritating/harmless
  - Catastrophic

Catastrophic Errors

Therac-25 radiation therapy machine [1985–1987]

- Patients receive massive overdoses of X-rays
- Two deaths, others injured and handicapped

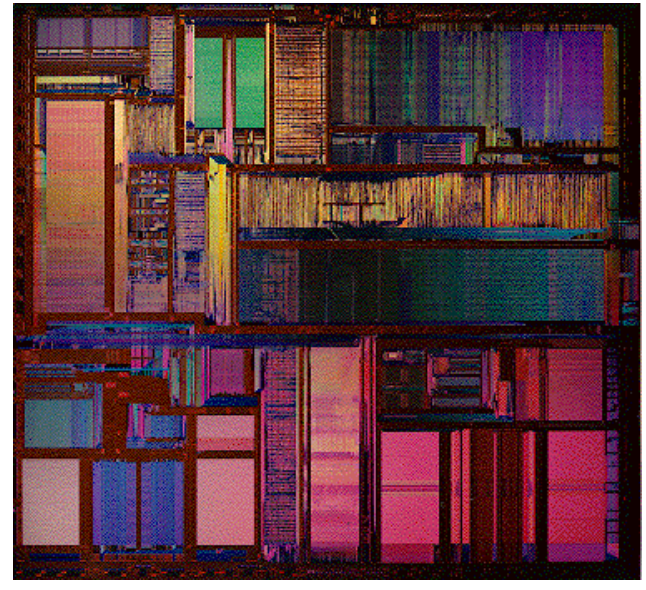
Catastrophic Errors

Therac-25 radiation therapy machine [1985–1987]

- Patients receive massive overdoses of X-rays
- Two deaths, others injured and handicapped

Intel's Pentium division bug [Dec. 1994]

- 4.999999/14.999999 gives 0.333329, but should be 0.3333**33**29
- Intel have to replace millions of defect processors
- Costs several 100 million $



Catastrophic Errors

Ariane 5 rocket [4 June 1996]

- Floating-point conversion fails, and the computer crashes
- Rocket self-destructs 40 seconds after lift-off
- Costs ESA 600 million $

Catastrophic Errors

Ariane 5 rocket [4 June 1996]

- Floating-point conversion fails, and the computer crashes
- Rocket self-destructs 40 seconds after lift-off
- Costs ESA 600 million $

NASA's Mars Climate Orbiter [23 Sep. 1999]

- One team uses English units, another metric units
- Landing procedure doesn't start timely
- Spacecraft explodes in Mars's atmosphere

Testing

- The Problem: How do we ensure that computer programs are error–free?

Testing

- The Problem: How do we ensure that computer programs are error–free?
- 10 years ago: Testing and simulation

Testing

- The Problem: How do we ensure that computer programs are error–free?
- 10 years ago: Testing and simulation
- But testing only shows the presence of errors, not the absence of errors

Formal Verification

- Last 5–10 years: **Formal verification**
  - Construct a mathematical *model* of the system
  - Specify the *requirements*
  - *Prove* that the model satisfies the requirements

Formal Verification

- Last 5–10 years: **Formal verification**
  - Construct a mathematical *model* of the system
  - Specify the *requirements*
  - *Prove* that the model satisfies the requirements
- Gives a 100% guarantee of correctness

Formal Verification

- Last 5–10 years: **Formal verification**
  - Construct a mathematical *model* of the system
  - Specify the *requirements*
  - *Prove* that the model satisfies the requirements
- Gives a 100% guarantee of correctness
- Proofs can be automated for programs with true/false variables

Formal Verification

- Last 5–10 years: **Formal verification**
  - Construct a mathematical *model* of the system
  - Specify the *requirements*
  - *Prove* that the model satisfies the requirements
- Gives a 100% guarantee of correctness
- Proofs can be automated for programs with true/false variables
- Easy to check that “the processor divides correctly”

Formal Verification

- My research: Verification of program with true/false variables + **continuous variables** (e.g., time or temperature)

Formal Verification

- My research: Verification of program with true/false variables + **continuous variables** (e.g., time or temperature)
- Example: check that “the processor divides correctly within 10 microseconds”

Formal Verification

- My research: Verification of program with true/false variables + **continuous variables** (e.g., time or temperature)
- Example: check that “the processor divides correctly within 10 microseconds”
- High demand for tools from the hardware industry

Formal Verification

- My research: Verification of program with true/false variables + **continuous variables** (e.g., time or temperature)
- Example: check that “the processor divides correctly within 10 microseconds”
- High demand for tools from the hardware industry
- Challenging because it’s harder to automate proofs

Formal Verification

- My research: Verification of program with true/false variables + **continuous variables** (e.g., time or temperature)
- Example: check that “the processor divides correctly within 10 microseconds”
- High demand for tools from the hardware industry
- Challenging because it’s harder to automate proofs
- Perspective: Verification can be used to check Java programs

Summary

It’s fun!

- Use math to find computer bugs
- 10 months at University of British Columbia

Summary

It's fun!

- Use math to find computer bugs
- 10 months at University of British Columbia

It's innovative

- Co-inventor of two patents: "A data structure and its use" [1999], and "Virtual tabulation" [2000]

Summary

It's fun!

- Use math to find computer bugs
- 10 months at University of British Columbia

It's innovative

- Co-inventor of two patents: "A data structure and its use" [1999], and "Virtual tabulation" [2000]

It's commercial

- Collaborate with researchers from SRI International
- Co-founder of the spin-off company [www.configit-software.com] [1999]