

CS6133

Software Specification and Verification

Lecture 1

Introduction and Administration

Course Information

- ◆ Course web page
<http://www.cs.utsa.edu/~niu/teaching/cs6133>
- ◆ Office hours
by appointment
- ◆ Text books
 - Michael Huth and Mark Ryan, "Logic in Computer Science", Cambridge University Press, 2004

Grading Policy

- ◆ Individual assignments and quizzes: 20%
- ◆ Course participation: 10%
- ◆ Research presentation: 20%
- ◆ Group project: 50%

Course Objectives

- ◆ Introduce you to fundamental concepts of formal methods
 - How to model software systems using formal specification notations
 - How to analyze software systems using a variety of verification techniques
- ◆ Provide a map to guide you towards future research in this area
- ◆ Give you hands-on experience with formal method tools

CS6133

5

Course Topics

- ◆ Formal notations and their semantics
 - Finite-state-machine-based notations
 - Process algebras
 - Template Semantics
- ◆ Formal verification techniques
 - Temporal logic and algorithmic verification
 - Binary decision diagram and symbolic model checking
 - Theorem proving
- ◆ Automated analysis tools
 - LTSA
 - NuSMV
 -

CS6133

6

What Are Formal Methods

- ◆ Formal methods are a collection of notations and techniques for precisely describing and analyzing computer systems
 - They are based on mathematical theories, such as logic, set, and automaton
 - Formal specification is precise and useful in eliminating misunderstandings
 - Verification techniques and tools verify that a system meets the specification and satisfies desired properties
 - They are aimed at enhancing the quality of systems

CS6133

7

When Apply Formal Methods

- ◆ Formal methods can be used at various stages during software development process
- ◆ It is more effective to apply formal methods during early development stages
 - Requirements and design artifacts are relatively small
 - Errors that are found in early stages are usually cheaper to fix
- ◆ Formal verification techniques with tool support are employed widely to examine computer systems

CS6133

8

Why Study Formal Methods

- ◆ Computer-based systems control many aspects of our lives
- ◆ Such systems are often complex
 - Run behind schedule,
 - Are over budget
 - Have errors, which can cause loss of lives and money

CS6133

9

Loss of Life: Therac-25

- ◆ A computer-controlled radiation therapy machine called the Therac-25 overdosed six people between June, 1985 and January 1987 - with resultant deaths and serious injuries
- ◆ The accidents occurred when the high-power electron beam was activated instead of the intended low power beam, and without the beam spreader plate rotated into place
- ◆ Neither was the machine's software well-documented nor thoroughly validated

CS6133

10

Loss of Life: Therac-25

- ◆ The Therac-25 could deliver radiation as either a beam of electrons or a beam of X-rays. If the operator entered "x" for x rays, the setting of the magnets took 8 seconds. If the operator discovered she had made a mistake and fixed the entry to be "e" within that 8 seconds, even though the screen reflected the change, the change did not affect a part of the program.
- ◆ Leveson and Turner, "An investigation of the Therac-25 accidents". [LT93]

CS6133

11

Loss of Money: Pentium FDIV bug

- ◆ The floating point processor of Intel Pentium chips manufactured before a certain date had a bug in the FDIV operation. Approximately 2 million chips had the flaw.
- ◆ The FDIV operation didn't always give precise results. According to sources on the web, the bug could be seen by entering the following formula in the Windows calculator:
$$(4195835/3145727) * 3145727 - 4195835$$

It produced 512 instead of 0.

CS6133

12

Loss of Money: Pentium FDIV bug

- ◆ Intel identified the problem in testing, summer, 1994. Prof. Thomas Nicely, Lynchburg College, first identified in email to colleagues Oct, 1994.
- ◆ EE Times had an article on it in Nov, 1994.

"This was a very rare condition that happened once every 9 to 10 billion operand pairs," said Steve Smith, a Pentium engineering manager at Intel. Intel's Smith emphasized that the anomaly would not affect the average user. Speaking of Nicely, Smith said: "He's the most extreme user. He spends round-the-clock time calculating reciprocals. What he observed after running this for months is an instance where we have eight decimal points correct, and the ninth not showing up correctly. So you get an error in the ninth decimal digit to the right of the mantissa. I think even if you're an engineer, you're not going to see this."

CS6133

13

Loss of Money: Pentium FDIV bug

- ◆ Intel offers to replace it for those customers based on "need".
- ◆ Stock prices fell, great concern.
- ◆ Dec, 1994, Intel offers to exchange the processor for anyone who asks in Dec, 1994.
- ◆ Cost to Intel estimated at \$475 million.
- ◆ Intel hired many people who did formal verification after this incident and they continue to carry out research in hardware verification and are applying these techniques in practice.

CS6133

14

Loss of Money: Banking

- ◆ February 1994, automated teller machines at Chemical Bank in New York City mistakenly deducted a total of approximately \$15 million dollars from about a hundred thousand customer accounts. Until the problem was discovered, any customers making withdrawal were charged double the withdrawal's actual amount on their accounts, although the printed transaction slip showed the correct amount. The culprit proved to be a flawed instruction - a single line in updated computer program the company had installed day before the problem surfaced. [Pet96]

CS6133

15

Why Study Formal Methods

- ◆ Testing is not enough: testing only show the presence of bugs not their absence
- ◆ Formal method is an area of research whose goal is to create techniques for analyzing systems to find subtle, critical logic, and safety errors
- ◆ Greater confidence can be achieved by using formal methods

CS6133

16

Propositional Logic: Overview

- ◆ The aim of logic in computer science is to develop languages to model situations, in such a way that we can reason about them formally
- ◆ Syntax: define well-formed formula
- ◆ Semantics: define the meaning of the formula using truth tables
- ◆ Proof theory

Review Propositional Logic: Syntax

- ◆ Two constant symbols: true and false
- ◆ Proposition letters: represent declarative sentences
- ◆ Logical connectives and their precedence
- ◆ Brackets

Review Propositional Logic: Semantics

- ◆ Related two worlds: mapping expressions in one world in terms of values in another world using semantic function
- ◆ Classical logic is two-valued: T and F
- ◆ The domain of the semantic function is the syntax
- ◆ The range of the semantic function is the set of truth values
- ◆ Truth table

Review Propositional Logic: Proof Procedures

- ◆ We can always determine if a formula is a tautology by using truth tables: can be tedious
- ◆ Proof procedures are alternative means to determine tautologies
- ◆ Forward and backward proof
- ◆ Example procedures include: resolution, natural deduction, and etc

Review Predicate Logic

- ◆ Predicate logic is for expressing properties about fixed-valued variables
- ◆ A predicate logic formula is evaluated with respect to a particular assignment of values to variables

Review Predicate Logic

- ◆ Set of typed variables
- ◆ Functions on typed variables
- ◆ Predicates
- ◆ Equivalence
- ◆ Propositional logic connectives
- ◆ Quantifiers

References

- ◆ [LT93] Nancy G. Leveson and Clark S. Turner. An investigation of the Therac-25 accidents. *Computer*, 26(7):18-41, July 1993.
- ◆ [Pet96] Ivars Peterson. *Fatal Defect: Chasing Killer Computer Bugs*. Vintage Books, New York, 1996.