EECE 513: Error-Resilient Computing Systems

Lecture 0: Introduction to the course

(Karthik Pattabiraman)

Who am I ?

- Associate Professor of Electrical and Computer Engineering (ECE)– joined UBC in 2010
  - PhD from UIUC, Post-doc with MS Research
  - Research in fault-tolerant and secure computing
  - Use static analysis techniques in our research
- Fifth time I'm teaching this course (EECE513) – second time as the MEL version
  - Taught grad course in program analysis (EECE 571P)
  - Teach undergrad courses on software design/web/OS

Who are you ?

- Your name, department, advisor (if applicable)
- What interests you ? What do you want to do after you finish your current degree ?
- What made you choose this class (or did you) ? What do you want out of the course ?

Administrivia

- **Class hours:** Mondays 2 to 5 PM
- **Office hours**: By Appointment (send me a private note on Piazza)
- **Textbook:** No text book (I will distribute notes/slides online)
- **Reference Books:**
  - D. P. Siewiorek and R. S. Swarz, *Reliable Computer Systems - Design and Evaluation*, 3rd edition, 1998, A.K. Peters, Limited.
  - K. Trivedi, *Probability and Statistics with Reliability, Queuing and Computer Science Applications*, 2nd edition, 2001, John Wiley & Sons.
  - Dhiraj K. Pradhan, Fault tolerant computer system design, Prentice Hall, 1996.
  - Laura Pullum, Software Fault-tolerance techniques and implementation, Artech House.

Prior Background

- **Mathematical background**: I assume you've had some basic exposure to probability theory
  - Both discrete and continuous probability distributions
  - Can pick up easily from the Trivedi book (Chap 1-4)
  - Talk to me if you don't have this background (today)

- **Programming/software background**
  - At least one course in software design (e.g., software engg., operating systems) – proficient in programming
  - Knowledge of C/C++ programming language (assignments)

Evaluation - 1

- **Assignments (55 %)**
  - One assignment involving pencil and paper (15%)
  - Other two assignments involving C/C++ programming and using LLFI (40%)
    - Assignment 1: Inject faults and study their effects (15%)
    - Assignment 2: Build an error-resilient program to tolerate different kinds of faults and evaluate it (25%)
      - Need to show a demo to the TA (Date: TBD)
      - Will require much higher time and effort than the first one

Evaluation - 2

- **Class participation in class/Piazza** (5%)
  - Participation does not mean just attending class
  - Asking and answering questions in class/piazza

- **Exams** (40%)
  - Midterm exam - in class (15%)
  - Final exam - take home (20%)
  - Exams will draw upon material covered in lectures

Why Study Error Resilient Computing ?

- **Traditional needs**
  - Long-life applications (e.g. space missions)
  - Life-critical, short-term applications (e.g., aircraft engine control, fly-by-wire)
  - Defense applications and Nuclear industry

- **Newer critical-computation applications**
  - Health industry
  - Automotive industry
  - Industrial control systems, production lines

Why Study Error Resilient Computing ?

- **Networks and Internet**
  - Wired and wireless networked applications
  - Large data-centers such as Google, Amazon
  - E-commerce, Web 2.0 applications

- **Scientific computing**
  - Reliability is an important issue with the advent of large-scale machines (e.g., Blue Waters will have on the order of hundred thousand processors)
  - Check-pointing and recovery don't scale well

Why Study Error Resilient Computing ?

- **Desktop computing**
  - Reliability problems can impact system security
  - Power consumption is becoming important, so we cannot overprovision as we used to in the past
  - Users don't want to deal with system management

- **Ubiquitous computing**
  - We are seeing an increase in the number of devices around us, and are increasingly relying on their correct operation (e.g., smart phones, sensors …)

What will you learn in this course ?

- **Dependable systems design**
  - Hardware dependability
  - Duplication and TMR
  - Software approaches
  - Parallel systems
  - Distributed systems
  - Case studies of real-world systems
- **Dependability evaluation techniques**
  - Combinatorial methods
  - Fault-injection
  - State-based Methods
  - Statistical methods

Why take this course ?

- Exposure to state of the art and traditional techniques in fault-tolerant systems design
- Deep understanding of design choices and trade-offs in real-world fault-tolerant systems
- Rigorous methods and tools for dependability evaluation (useful for industry and research)
  - Opportunity to learn to use tools of the trade

Policies etc.

- Please do NOT send me email unless it's an emergency – use Piazza for this purpose
- Late submissions will be penalized by 10% for everyday they are late up to 3 days maximum
- Please review UBC's policy on plagiarism and academic conduct. Ignorance of the same will not be an acceptable excuse for lapses.
- You are expected to attend (and participate in) all lectures and discussions in class
  - Not everything covered in class will be posted online

Final thoughts

- The goal is to have fun while we learn !
  - I certainly hope you have fun in this course …

- I am always open to suggestions and critical comments on the course. **Such critical comments will not impact your grade in any way.** On the other hand, good suggestions may even earn you some extra credit/cookies