Static Analysis for Software Engineering
ECE750-T5, Spring 2012

Patrick Lam

Code review is a key technique for ensuring software quality. However, human reviewers have limitations: limited time, limited attention spans, and a limited understanding of the implications of a software modification (due to interactions between parts of a potentially vast codebase).

Computers can successfully carry out many tasks at which humans fail. A goal of my research is to find classes of properties which are amenable to automatic verification, using static analysis techniques (which trace their roots to optimizing compilers). Many challenges exist. The most notorious problems include the undecidability of the halting problem (which we overcome using approximations) and the need to deal with unknown input values. More recently, large codebases and plugin-based software architectures pose additional challenges to static analysis.

Nevertheless, a number of static analysis techniques for software engineering have recently been proposed and even deployed in commercial development environments. Microsoft's Static Driver Verifier is the most prominent example of a static analysis technique that has escaped the laboratory.

In this seminar course, we will first briefly explore the strengths and limitations of program analysis techniques. These techniques traditionally come from the compiler research community. The bulk of this course, however, will consist of a discussion of current research papers in the field of software verification using program analysis techniques.

Objectives. After this course, you will:

- understand the strengths and limitations of program analysis techniques, particularly static analysis techniques;
- be familiar with the research literature on static analysis for software verification.
- have carried out a small research project implementing a program analysis and evaluating its efficacy at solving a software engineering problem.

General Information

Course Web Page: `http://patricklam.ca/sase`

Lectures: Wednesday, 11:00-13:50, EIT 3141

Instructor:

Prof. Patrick Lam
Office: DC2534
Office Hours: By appointment
Email: `p.lam@ece.uwaterloo.ca`
Phone: Use email instead!

Course Description

Overview of techniques used in static and dynamic analysis, including dataflow analysis, type systems, and pointer analysis. Typestate properties. Applications to software engineering (notably concurrency and security).

Here is a week-by-week schedule:

1–2	Introductory material
3	Applications of Type Systems
4	Typestate (dynamic)
5	Alias/pointer analysis
6	Security
7	Interprocedural taint-style analyses
8	Concurrency
9	Testing (and Dynamic Analysis)
10	Typestate (static)
11	Using static analysis results

Reference Material

The reference material for this course consists of my notes for the first two lectures and the research papers that we'll discuss every week. Most of the papers have been posted on the authors' webpages, and I've included those links when possible. You may need to be on campus (or proxying) to download the ACM links to a couple of papers.

Evaluation

You will be expected to present two papers to the class and to complete a course project, which includes a short presentation on the last day of class. Due to departmental regulations, there will also be a final examination.

Course project	50%	due last day of class
Project presentation	5%	during last class
Paper presentations	10% each	throughout term
Final exam	25%	final exam period

Presentation. You will present two of the papers that we are discussing in this class. Please let me know which papers you have chosen by the end of the second week of classes. The presentations will be an hour long. Be prepared to answer questions on the papers.

I will provide submarks for both delivery and content of your paper and project presentations, and I'll send you timely feedback and suggestions on how to improve your presentations.

Course project. The course project gives you an opportunity to work on a particular application of program analysis techniques to software engineering issues. I expect that most projects will include an implementation component. To help you stay on track, I will expect a project proposal by the third week of class and a mid-term project update (1 page). (Start early!) I will post a list of suggested projects on the webpage. Please come see me to talk about possible projects!

Final examination. The open-notes final examination will ask you to synthesize the information you've seen throughout the semester. Potential questions include summarizing key points from selected papers, or sketching out how the techniques from one paper might apply to the problems that another paper addresses.

Lateness. This is a graduate seminar, so I will offer some flexibility on due dates. However, this flexibility is often not in your best interest. You must consult with me before you hand in something late; otherwise, you will lose 5% on the project mark for a late project proposal or mid-term project report. The default mark for a late project submission is 0.

Required inclusions

Academic Integrity: In order to maintain a culture of academic integrity, members of the University of Waterloo community are expected to promote honesty, trust, fairness, respect and responsibility. [Check `www.uwaterloo.ca/academicintegrity/` for more information.]

Grievance: A student who believes that a decision affecting some aspect of his/her university life has been unfair or unreasonable may have grounds for initiating a grievance. Read Policy 70, Student Petitions and Grievances, Section 4, `www.adm.uwaterloo.ca/infosec/Policies/policy70.htm`. When in doubt please be certain to contact the departments administrative assistant who will provide further assistance.

Discipline: A student is expected to know what constitutes academic integrity [check `www.uwaterloo.ca/academicintegrity/`] to avoid committing an academic offence, and to take responsibility for his/her actions. A student who is unsure whether an action constitutes an offence, or who needs help in learning how to avoid offences (e.g., plagiarism, cheating) or about rules for group work/collaboration should seek guidance from the course instructor, academic advisor, or the undergraduate Associate Dean. For information on categories of offences and types of penalties, students should refer to Policy 71, Student Discipline, `www.adm.uwaterloo.ca/infosec/Policies/policy71.htm`. For typical penalties check Guidelines for the Assessment of Penalties, www.adm.uwaterloo.ca/infosec/guidelines/penaltyguidelines.htm.

Appeals: A decision made or penalty imposed under Policy 70 (Student Petitions and Grievances) (other than a petition) or Policy 71 (Student Discipline) may be appealed if there is a ground. A student who believes he/she has a ground for an appeal should refer to Policy 72 (Student Appeals) `www.adm.uwaterloo.ca/infosec/Policies/policy72.htm`. Note for Students with Disabilities: The Office for Persons with Disabilities (OPD), located in Needles Hall, Room 1132, collaborates with all academic departments to arrange appropriate accommodations for students with disabilities without compromising the academic integrity of the curriculum. If you require academic accommodations to lessen the impact of your disability, please register with the OPD at the beginning of each academic term.