CS 521/621
Paper Selection and Idea Proposal

Due: **Monday, Oct 6, 2014, 9:00 AM EDT**. Submit paper selection via http://goo.gl/Q3hRLb and idea proposal via Moodle. Late assignments will not be accepted without **prior** permission.

This assignment has two parts:

1. Selecting research papers to present, and
2. Generating great research ideas and starting to discuss them with others.

Completing the first part is simple. You will be given with a list of papers, and you will pick **five** (or more, if you want) papers from that list. Submit via http://goo.gl/Q3hRLb

Completing the second part (submit via Moodle) involves:

1. coming up with a creative new research idea,
2. writing a 1-page description of this idea, and
3. giving a 5-minute in-class presentation on Wednesday, October 8.

To complete this assignment, **every student** must (1) submit a list of **five** (or more, if you want) research papers, of which the instructor will selected papers for the student to present. Additionally, every **CS 621 student** must do **one** research idea write-up and presentation. Meanwhile, the **CS 521** students may **optionally** do **one** research idea write-up and presentation.

To understand the reason for this complexity, we must first consider the goals for this semester:

For **CS 621** students, the future is clear. This semester, each **CS 621** student will present **one** research paper, **and** do **one** research project.

For **CS 521** students, decisions await. This semester, each **CS 521** student will either:

1. Present **two** research papers, *or*
2. Present **one** research paper, **and** do a research project.

The **CS 521** students will have until October 7 to decide which option they prefer. They can listen to all the presentations, talk to potential partners, form teams of 1–4 students, and then decide. If you are sure you want to do a project, it is recommended (but not required) you give a presentation.

If you are confused by the requirements, contact the instructor as soon as possible!

Research paper list

Submission site: http://goo.gl/Q3hRLb

Select papers from this list. You must select **at least five** papers. If your interests are broad, you may include more papers on your list, though I will try to assign you papers from your top five choices. (Obviously, if every single student picks the exact same five papers, I won't be able to make everyone happy, and having additional choices may help your cause.)

All paper presentations will be 20 minutes. Most (all, if possible) papers will have **two** or **three** students assigned, so you will work with partners for the presentation. All presentations will be during regular class times. The dates on which each paper is presented will be assigned by the instructor (as soon as he knows how many CS 521 students are doing two presentations). *You will be given more guidance later on how to present a paper.*

Choosing the papers can take a bit of time. To make the most well-educated choices, you should take at least a brief look at each paper and read its abstract. Each paper below includes a URL. Some papers are available for free, and others are available for free only to you as UMass students. On campus, the links will just work. If you are off campus, you may access the papers for free by adding `.silk.library.umass.edu` to the end of the domian. For example, http://dl.acm.org/citation.cfm?id=2491459 becomes http://dl.acm.org.silk.library.umass.edu/citation.cfm?id=2491459. (Note that the addition does not go to the end of the URL, but rather the end of the domain.) You will be asked to log in (and one time only fill out a survey). Again, submit your selections via http://goo.gl/Q3hRLb

List of potential papers:

1. Automatic Patch Generation Learned from Human-Written Patches. By Dongsun Kim, Jaechang Nam, Jaewoo Song, and Sunghun Kim. ICSE 2013. http://www.cse.ust.hk/~hunkim/papers/kim-icse2013.pdf
2. Automatic Recovery from Runtime Failures. By Antonio Carzaniga, Alessandra Gorla, Andrea Mattavelli, Nicolo Perino, and Mauro Pezze. ICSE 2013. http://dl.acm.org/citation.cfm?id=2486891
3. SemFix: Program Repair via Semantic Analysis. By Hoang Duong Thien Nguyen, Dawei Qi, Abhik Roychoudhury, and Satish Chandra. ICSE 2013. http://www.comp.nus.edu.sg/~abhik/pdf/ICSE13-SEMFIX.pdf
4. Modular and Verified Automatic Program Repair. By Francesco Logozzo and Thomas Ball. OOPSLA 2012. http://research.microsoft.com/pubs/170385/res0099-logozzo.pdf
5. CodeHint: Dynamic and Interactive Synthesis of Code Snippets. By Joel Galenson, Philip Reames, Rastislav Bodik, Björn Hartmann, and Koushik Sen. ICSE 2014. https://jgalenson.github.io/papers/icse2014.pdf
6. Checking App Behavior Against App Descriptions. By Alessandra Gorla, Ilaria Tavecchia, Florian Gross, and Andreas Zeller. ICSE 2014. http://www.st.cs.uni-saarland.de/chabada/CHABADA.pdf
7. Finding your way in the testing jungle: A learning approach to web security testing. By Omer Tripp, Omri Weisman, and Lotem Guy. ISSTA 2013. http://dl.acm.org/citation.cfm?id=2483776
8. Automated testing with targeted event sequence generation. By Casper S. Jensen, Mukul R. Prasad, and Anders Moller. ISSTA 2013. http://dl.acm.org/citation.cfm?id=2483777
9. Guided Test Generation for Web Applications. By Suresh Thummalapenta, K. Vasanta Lakshmi, Saurabh Sinha, Nishant Sinha, and Satish Chandra. ICSE 2013. http://dl.acm.org/citation.cfm?id=2486810
10. Are Mutants a Valid Substitute for Real Faults in Software Testing? By Rene Just, Darioush Jalali, Laura Inozemtseva, Michael D. Ernst, Reid Holmes, Gordon Fraser. FSE 2014. http://homes.cs.washington.edu/~rjust/publ/mutants_real_faults_fse_2014.pdf
11. Coverage Is Not Strongly Correlated with Test Suite Effectiveness. By Laura Inozemtseva and Reid Holmes. ICSE 2014. http://www.linozemtseva.com/research/2014/icse/coverage/coverage_paper.pdf

12. Aluminum: Principled Scenario Exploration through Minimality. By Tim Nelson, Salman Saghafi, Daniel J. Dougherty, Kathi Fisler, and Shriram Krishnamurthi. ICSE 2013. http://cs.brown.edu/~sk/Publications/Papers/Published/nsfdk-aluminum-scen-expl-min/paper.pdf
13. Interaction-Based Test-Suite Minimization. By Dale Blue, Itai Segall, Rachel Tzoref-Brill, and Aviad Zlotnick. ICSE 2013. http://researcher.watson.ibm.com/researcher/files/il-RACHELT/icse13main-p118-p-15780-preprint.pdf
14. Termination Proofs from Tests. By Aditya V. Nori and Rahul Sharma. FSE 2013. http://research.microsoft.com/pubs/193427/final-version.pdf
15. Enhancing Symbolic Execution with Veritesting. By Thanassis Avgerinos, Alexandre Rebert, Sang Kil Cha, and David Brumley. ICSE 2014. http://users.ece.cmu.edu/~aavgerin/papers/veritesting-icse-2014.pdf
16. RefaFlex: Safer Refactorings for Reflective Java Programs. By Andreas Thies and Eric Bodden. ISSTA 2012. http://www.bodden.de/pubs/tb12refaflex.pdf
17. Refactoring with Synthesis. By Veselin Raychev, Max Schafer, Manu Sridharan, and Martin Vechev. OOPSLA 2013. http://researcher.watson.ibm.com/researcher/files/us-msridhar/OOPSLA13Refactoring.pdf
18. Drag-and-Drop Refactoring: Intuitive and Efficient Program Transformation. By Yun Young Lee, Nicholas Chen, and Ralph E. Johnson. ICSE 2013. http://www.ideals.illinois.edu/bitstream/handle/2142/33793/DNDRefactoring.pdf
19. Making Offline Analyses Continuous. By Kvanç Muşlu, Yuriy Brun, Michael D. Ernst, and David Notkin. FSE 2013. http://cs.umass.edu/~brun/pubs/pubs/Muslu13fse.pdf
20. SPLat: Lightweight dynamic analysis for reducing combinatorics in testing configurable systems. By Chang Hwan Peter Kim, Darko Marinov, Sarfraz Khurshid, Don Batory, Sabrina Souto, Paulo Barros, and Marcelo D'Amorim. FSE 2013. http://dl.acm.org/citation.cfm?id=2491459
21. Automated Diagnosis of Software Configuration Errors. By Sai Zhang and Michael D. Ernst. ICSE 2013. http://homes.cs.washington.edu/~szhang/pdf/confdiagnoser-icse13-zhang.pdf
22. Using likely invariants for automated software fault localization. By Swarup Kumar Sahoo, John Criswell, Chase Geigle, and Vikram Adve. ASPLOS 2013. http://dl.acm.org/citation.cfm?id=2451131
23. Injecting Mechanical Faults to Localize Developer Faults for Evolving Software. By Lingming Zhang, Lu Zhang, and Sarfraz Khurshid. OOPSLA 2013. http://sei.pku.edu.cn/~zhanglu/Download/OOPSLA13.pdf
24. Automated Memory Leak Detection for Production Use. By Changhee Jung, Sangho Lee, Easwaran Raman, and Santosh Pande. ICSE 2014. http://dl.acm.org/citation.cfm?id=2568311
25. Automatic Detection of Floating-Point Exceptions. By Earl T. Barr, Thanh Vo Vu Le, and Zhendong Su. POPL 2013. http://dl.acm.org/citation.cfm?id=2429133
26. Be Conservative: Enhancing Failure Diagnosis with Proactive Logging. By Ding Yuan, Soyeon Park, Peng Huang, Yang Liu, Michael M. Lee, Xiaoming Tang, Yuanyuan Zhou, and Stefan Savage. OSDI 2012. http://opera.ucsd.edu/paper/osdi12-errlog.pdf
27. Cassandra: Proactive Conflict Minimization through Optimized Task Scheduling. By Bakhtiar Khan Kasi and Anita Sarma. ICSE 2013. http://dl.acm.org/citation.cfm?id=2486884
28. Data Clone Detection and Visualization in Spreadsheets. By Felienne Hermans, Ben Sedee, Martin Pinzger, and Arie van Deursen. ICSE 2013. http://dl.acm.org/citation.cfm?id=2486827
29. Sample Size vs. Bias in Defect Prediction. By Foyzur Rahman, Daryl Posnett, Israel Herraiz, and Premkumar Devanbu. FSE 2013. http://dl.acm.org/citation.cfm?id=2491411.2491418
30. Will You Still Compile Me Tomorrow? Static Cross-Version Compiler Validation. By Chris Hawblitzel, Shuvendu Lahiri, Kshama Pawar, Hammad Hashmi, Sedar Gokbulut, Lakshan Fernando, Dave Detlefs, and Scott Wadsworth. FSE 2013. http://dl.acm.org/citation.cfm?id=2491442
31. Averroes: Whole-Program Analysis without the Whole Program. By Karim Ali, and Ondrej Lhotak. ECOOP 2013. http://plg.uwaterloo.ca/~olhotak/pubs/ecoop13.pdf

Research idea write-up and presentation

This part of the assignment consists of:

1. An up to 1-page write-up describing your research idea.
2. A 5-minute presentation, given in class on Wednesday, October 8.

Overview

Your primary job in this part of the assignment is twofold:

1. To describe your proposed research goal so that people understand what it is and why it is valuable. This must include a *research question* you will try to answer.
2. To describe how you will accomplish your research goal and how you will evaluate it so that it is clear how a team of up to four students can answer the research question in approximately 10 weeks.

You will present your idea to the class. Everyone will then have the opportunity to review the presentations and form groups of up to four students to actually explore the research idea! These groups may be made up for CS 521 or CS 621 students. Any mix is OK.

One of the purposes of identifying the research idea is to find an area of software engineering research that is interesting to you. The idea will evolve over time, especially as you read the related work. While this initial idea may differ significantly from the final research question you tackle, the initial idea will serve an important role in focusing you on a particular area of software engineering.

Deliverables

Everyone must submit:

- An ordered list of at least **five** papers from the above list. Submit here: `http://goo.gl/Q3hRLb`

Every **CS 621**, and, optionally, any **CS 521** student must submit:

- An up to 1-page description of the research idea, in a .pdf, uploaded to Moodle. Don't forget to put your name on it.
- A digital presentation (keynote, powerpoint, or pdf) also uploaded to Moodle. Don't forget to put your name on it.
- An in-class presentation. The delivery should take a maximum of 5 minutes. You will be cut off after 5 minutes! Prepare and rehearse your presentation.

For the **research idea** part of the assignment, the written description **and** the presentation must **each** contain:

- Research question. This must be in the form of a question and describe what you will know after this project is finished that the world does not know today. Examples of reasonable research questions include: "RQ1: Can mutations used in mutation testing generate the kinds of bugs observed in real, open-source development?" and "RQ2: Can the k-tail algorithm for model inference be augmented with information about pre- and post-conditions (inferred by dynamic analysis) to improve the inferred models' precision and recall?"
- The key idea behind the new technique you will develop. For example, for RQ2, the idea may be "Preventing k-tail from merging states if the pre- and post-conditions do not match."
- A concrete evaluation plan that you will use to determine when answering your research question is a success. For example, for RQ1, the plan may be "We will survey the mutation testing literature to enumerate at least 10 top mutation operators. We will then find, on `github.com`, at least 6 open-source programs. Each program will have at least 10K lines of code, 100 actively maintained tests, and at least 1000 commits in its history. We will analyze the commit history to find regression bugs (times when a test that previously was passing but then started failing), isolate the changes that resulted in each of these bugs, and determine whether these changes could be been generated automatically using the mutation operators we identified."

You may use any resource you wish in this assignment but you must list your collaborators and cite all your sources. Failure to do so will result in a grade of 0.