CS 520

Theory and Practice of Software Engineering
Fall 2017

Introduction to empirical research

October 05, 2017

Today

- Overview of class projects.
- A short quiz.
- Paper discussion:
  A Practical Guide for Using Statistical Tests to Assess Randomized Algorithms in Software Engineering

Class projects: overview

Logistics

- 4-5 students per project group.
- Group selection until 10/13 (discussion on Piazza).
- 2 informal presentations:
  - 10min + Q&A (early feedback).
  - Poster session and demo (final presentation).

High-level topics

1. Build system inference
2. API for code coverage tools
3. Elevation-based Navigation
4. Automated Software Fairness Testing
5. Model inference for inferring processes
6. Commit minimization

Project: Build system inference

Goal:

Given a project's build file (e.g., Apache Ant's build.xml), automatically determine (infer) relevant properties.

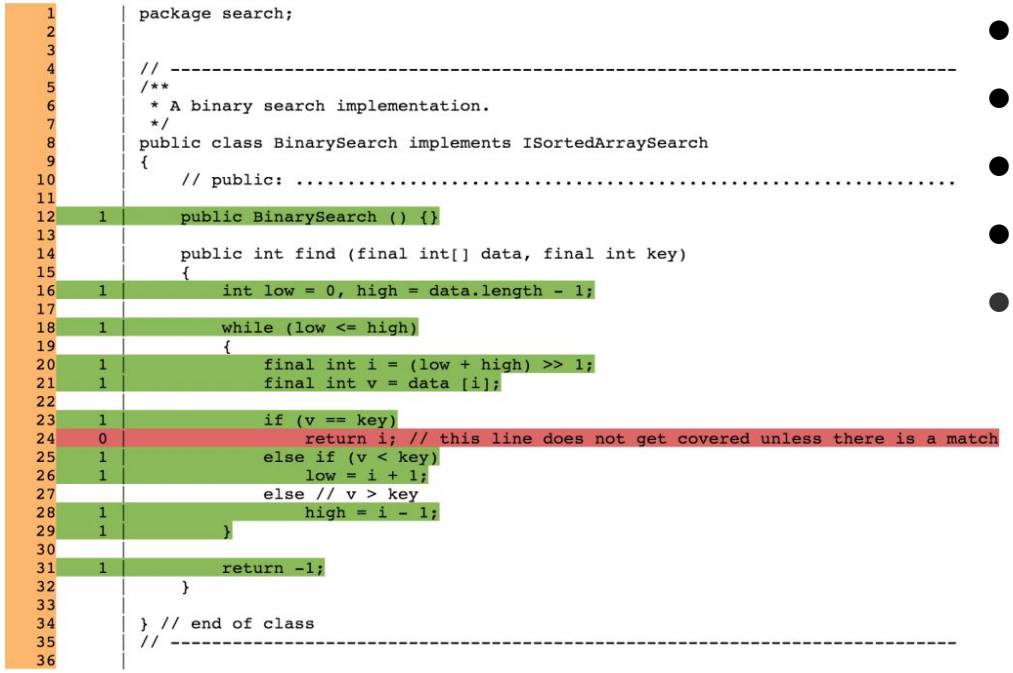
```
<project name="Example" default="compile" basedir=".">

<!-- Compile the project -->
<target name="compile" depends="init" description="Compile">
    <javac includeantruntime="true"
                        srcdir="src"
                       destdir="bin"
                         debug="yes">
        <classpath location="lib/junit.jar"/>
    </javac>
</target>
```

- Where are the sources?
- Where are the tests?
- What's the classpath?
- ...

Project: API for code coverage tools

Goal:

Design a Java API that defines a common abstraction for code coverage tools, and implement a concrete instance for Cobertura.



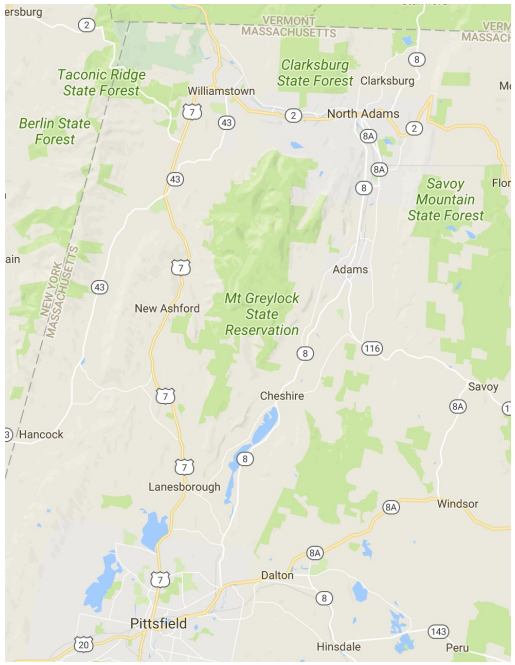
```
 1       |  package search;
 2       |
 3       |
 4       |  // ----------------------------------------------------------------------
 5       |  /**
 6       |   * A binary search implementation.
 7       |   */
 8       |  public class BinarySearch implements ISortedArraySearch
 9       |  {
10       |      // public: ..........................................................
11       |
12     1 |      public BinarySearch () {}
13       |
14       |      public int find (final int[] data, final int key)
15       |      {
16     1 |          int low = 0, high = data.length - 1;
17       |
18     1 |          while (low <= high)
19       |          {
20     1 |              final int i = (low + high) >> 1;
21     1 |              final int v = data [i];
22       |
23     1 |              if (v == key)
24     0 |                  return i; // this line does not get covered unless there is a match
25     1 |              else if (v < key)
26     1 |                  low = i + 1;
27       |              else // v > key
28     1 |                  high = i - 1;
29     1 |          }
30       |
31     1 |          return -1;
32       |      }
33       |
34       |  } // end of class
35       |  // ----------------------------------------------------------------------
36       |
```

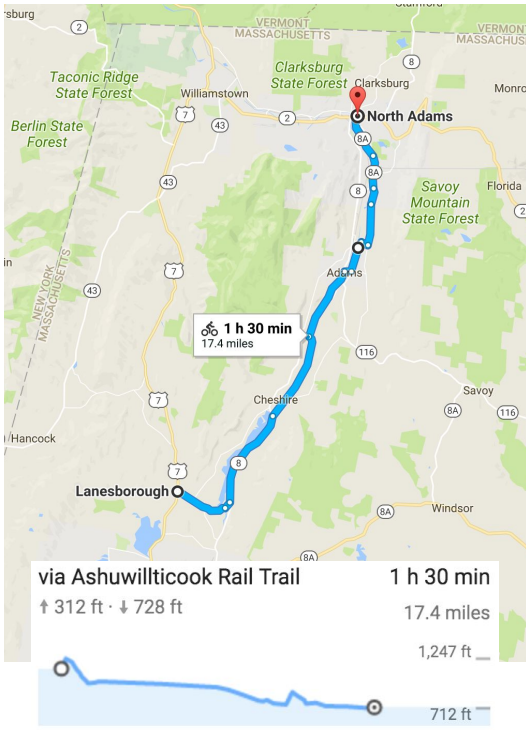
- Is line x covered?
- How often is it covered?
- How many lines are covered overall?
- How many lines exist in method y?
- ...

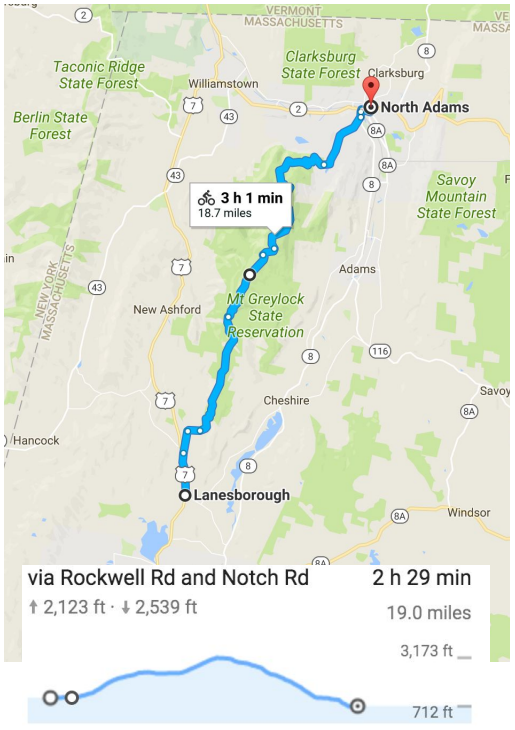
Project: EleNa (Elevation-based Navigation)

Goal:

Given a start and end location, determine the route that maximizes or minimizes elevation gain, while limiting the total distance between the two locations to x% of the shortest path.







Project: Automated Software Fairness Testing

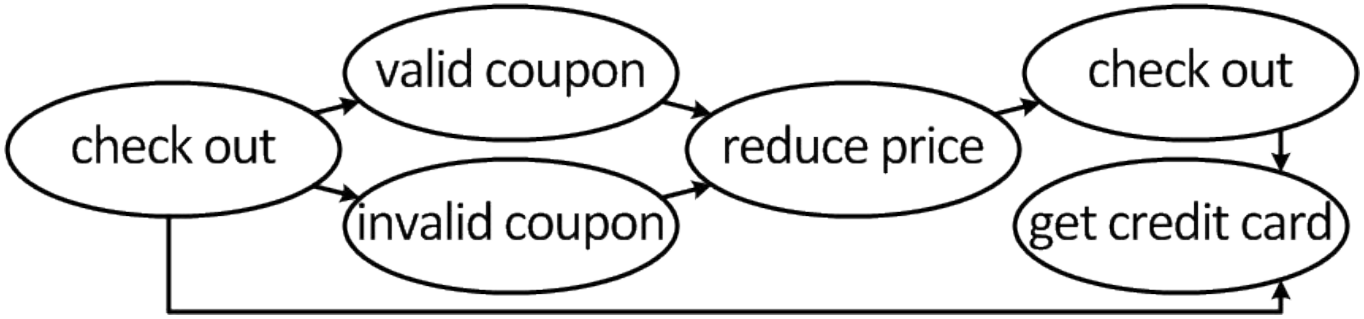
Goal:

Find open-source software systems on GitHub with categorical inputs and binary outputs (many ML-based classification systems fit these requirements) and generate test suites for these projects with either EvoSuite or Randoop. Evaluate whether these test suites are adequate to measure causal discrimination, as defined in: *Fairness Testing: Testing Software for Discrimination (http://people.cs.umass.edu/~brun/pubs/pubs/Galhotra17fse.pdf)*.

Project: Model Inference for Inferring Processes

Goal:

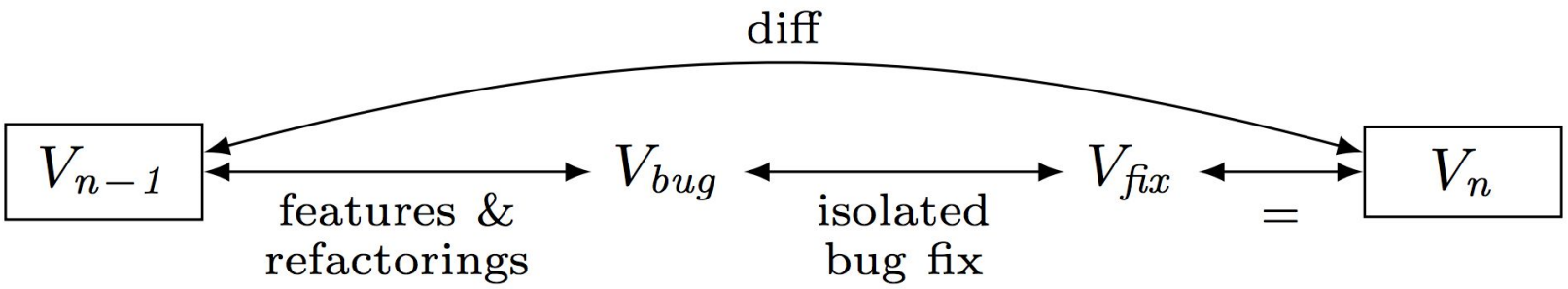
Collect multiple traces of human-driven process descriptions (e.g., imagine writing down every step you take of the process of cooking a pie, in great detail) for the same process, and use model inference techniques (e.g., *Synoptic*) to infer a model that captures all the variations in the process. Evaluate the generative properties of the model. Perform sensitivity analysis on the inclusion of more or fewer traces.



Project: Commit minimization

Goal:

Given a bug-fixing commit and a test suite that failed before and passes after that commit, automatically minimize the changes in that commit such that only changes relevant to the bug fix remain.



Introduction to empirical studies and research

A little quiz

1. What is the difference between statistical and practical significance?
2. What is a Type I error, what is a Type II error?
3. What is the interpretation of the p value and what is effect size?
4. What is the multiple-comparison problem?
5. What is the difference between parametric and non-parametric statistics?
6. Name one parametric and one non-parametric statistical test.
7. Name one parametric and one non-parametric effect size.

Paper discussion

A Practical Guide for Using Statistical Tests to Assess Randomized Algorithms in Software Engineering

High-level topics

- Significance and effect size.
- Parametric vs. non-parametric statistics.
- Proper baselines.

Open discussion

- Why do we need statistical tests?
- Why is this important for randomized algorithms?
- Is this only important for randomized algorithms?
- What's wrong with the current state of the art?