Teamline: Visualizing small team code contributions

Nick Bradley and Felix Grund
CPSC 547 Course Project
April 25, 2017
Background

• CPSC 310
  • Term-long coding project
  • Teams of 2-3 students
  • 3 automatically graded deliverables
Dataset

• >44,000 commit records from 139 teams and 285 students

• Grading system records metrics on every commit:
  • team number, deliverable, student (GitHub ID), commit ID, timestamp
  • pass rate: number of instructor-written tests that were passed
  • coverage: the proportion of code executed by student-written tests
  • grade: 80% pass rate + 20% coverage
Motivation

• TAs scale back grades of students who contribute less
  • Retrospectives after each deliverable
  • Assess contribution by talking to team members individually

• This can be hard and time consuming
  • Students may cover for partner
  • May need to examine code diffs over many commits

• Create derived data
  • Contribution for both pass rate and coverage
  • Contribution uniformity: how evenly did the members contribute
Derived Attributes

• Pass rate contribution: increases to pass rate
  # tests passed first time/total number passed tests

• Coverage contribution:
  increase in max coverage/coverage grade

• Overall contribution: 80% pass rate contrib. + 20% coverage contrib.

• Contribution uniformity: sum of pairwise differences of overall contributions.

\[
CU = 1 - \sum_{i=1}^{m-1} |u_i - u_{i+1}|
\]
Demo
Analysis

• What: data
  • Table of graded commits
• What: derived
  • Measure of contribution
• Why: tasks
  • Present uniformity of contributions
  • Summarize commit history

• How: facet
  • Overview+detail
  • Partition into side-by-side views
• How: encode
  • Heatmap and line charts
  • Marks positioned on common time scale; color indicates attribute
• How: embed
  • Superimpose grade in heatmap cells
• How: reduce
  • Filter by team and deliverable
Discussion

• Derived attributes are all-or-nothing
  • Ignores code contributions that don't increase either pass rate or coverage
  • Partly mitigated by having fine-grained tests

• Other metrics could help fill the gap
  • Code churn: number of lines of code added/removed
  • Trace changes using diffs and coverage report
    • Give credit also for code and not tests passed
  • Integration with task tracking: limit evaluation of contribution to assigned tasks
Future Work

• Make contribution measure more robust
• Provide view of entire project
• Support live data
  • Direct connection to auto grading database
  • Handle partially complete deliverables
• Evaluate effectiveness of Teamline
  • Are grades more fairly scaled back by TAs?
More Use-Cases

• Teams should be able to see their own Teamline
  • Help with team dynamic

• Instructors
  • Overview: dashboard view of student grades
  • Detail view: team conflicts

• TAs can identify students that
  • may be struggling
  • require more motivation

• Idea of visualizing code contributions could be extended to industry
  • Team awareness
  • Task allocation
  • Augment pull requests
Questions?
Current Approach – roll into demo...Team78

• Talk to students
• Look at individual commits
• Use GitHub graphs
  • Contributor graph
  • Network graph