PRINCETON UNIVERSITY

A Modular Framework for Developing, Deploying, and Evaluating Game-Theoretic Strategy Design Exercises

COS

Andrew Wonnacott, advised by Prof. Dave Walker and Prof. Matthew Weinberg

Problem Background: COS 445 Problem Sets

Students submit an implementation of a simple interface corresponding to a simple game representing real-world applications of course content

Student strategies are graded based on their performance against other student strategies

Course staff need tools to implement different games as strategy design exercises each semester

Students need better tools to test their strategies

Course staff need better tools to automate grading

Sample Programming Assignments

Prisoner's Dilemma	Cooperate	Defect
Cooperate	(3, 3)	(0, 5)
Defect	(5, 0)	(1, 1)

```
public interface Prisoner {
  // true to cooperate, false to defect
  public boolean getAction();

  // callback to receive action
  public void addResult(boolean action);
}
```

Google AdWords

```
public interface Bidder {
  // Return your bid for the current day
  public double getBid(double dailyValue);

  // if you won, how much the winners paid
  public void addResults(List<Double> bids,
        int myBid, double myPayment
  );
}
```

CollegeBoard

```
public interface Student {
  public int[] getApplications(int N,
        double max_SAT, double max_rank,
        double max_syn, double aptitude,
        double[] schools,
        double[] synergies
  );
}
```

Related Work

Past COS 445 course staff have developed tools as needed, but these tools were not reusable across years and could not easily be modified to develop a new assignment

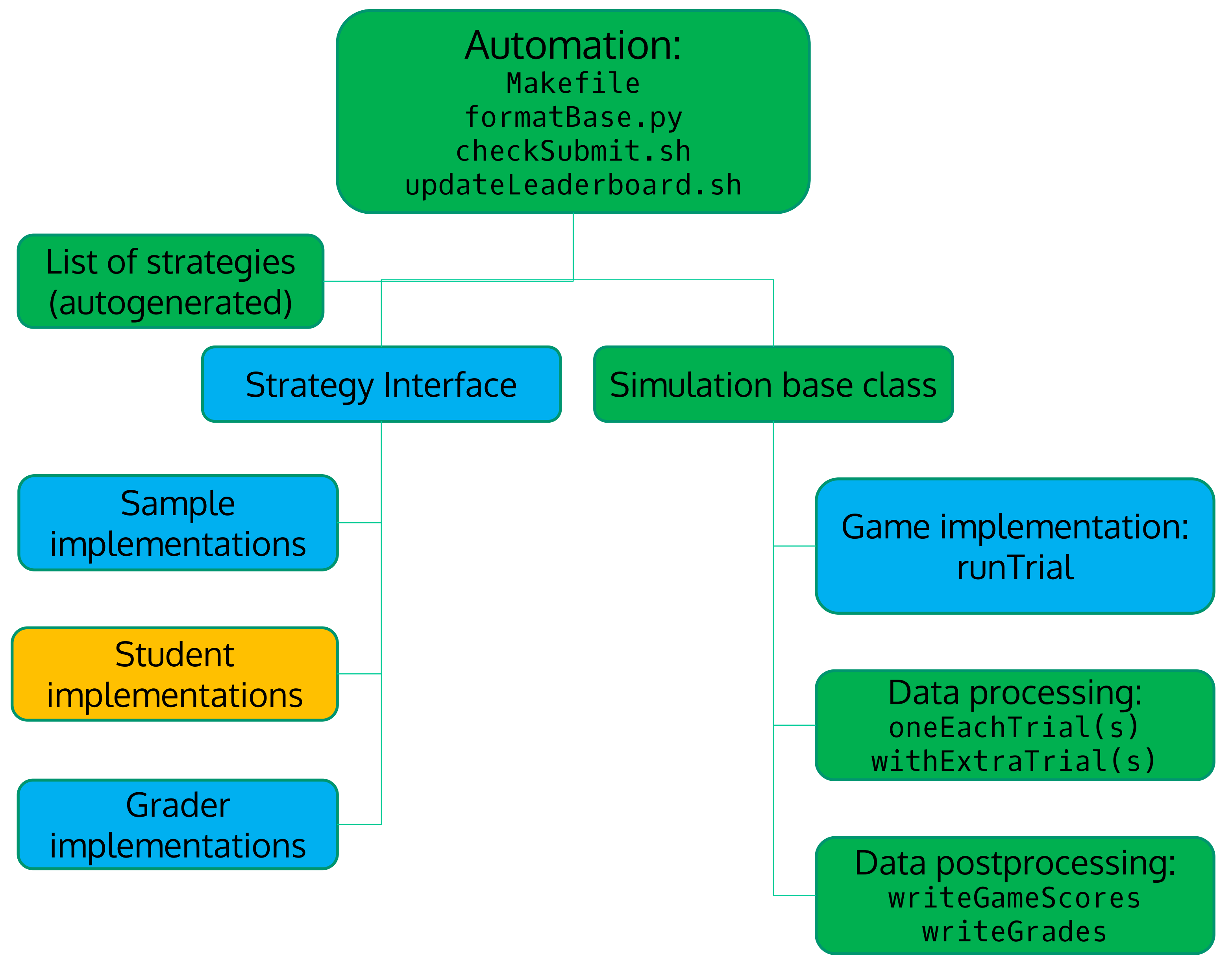
Harvard CS 136 has a similar assignment, but no publicly available tooling which could be adapted for this use-case

Approach

Developed simulation tools which allow students or instructors to evaluate and compare performance of a collection of strategies

All code in Java except for interfaces with existing systems: allows students to understand and modify simulation tools

Assignment Handout and Auto-grader Design



Assignment-Specific Components

Strategy interface documents the game to be played and specifies how students should communicate their strategy

Game implementation describes how to simulate one full tournament between a collection of provided strategies

Sample implementations are simple examples for students

Grader implementations are compared to student implementations to assign student grades

Reusable Components

Data processing methods instantiate student and instructor strategies based on rewriting from a configuration file and amortize results across randomized tournaments

Data postprocessing methods return simulation results of configured strategies

Student handout code displays average game scores

Auto-grader also compares each student to each instructor and outputs grades based on configurable rubric

All components are configured executed from environment-specific automation code

Student Submission

Each student submits their strategy as an implementation of the given interface

Results

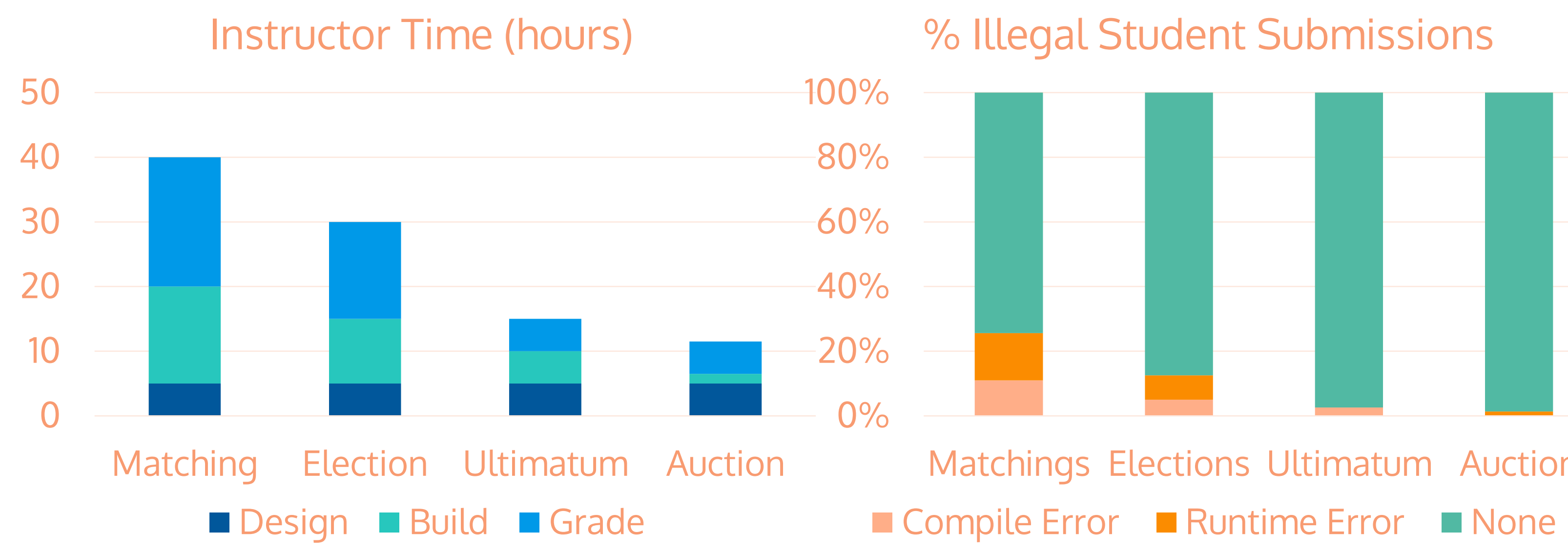
Developed four programming assignments using this framework for the Spring 2018 offering of COS 445

Developed extensions beyond student handouts and instructor auto-graders:

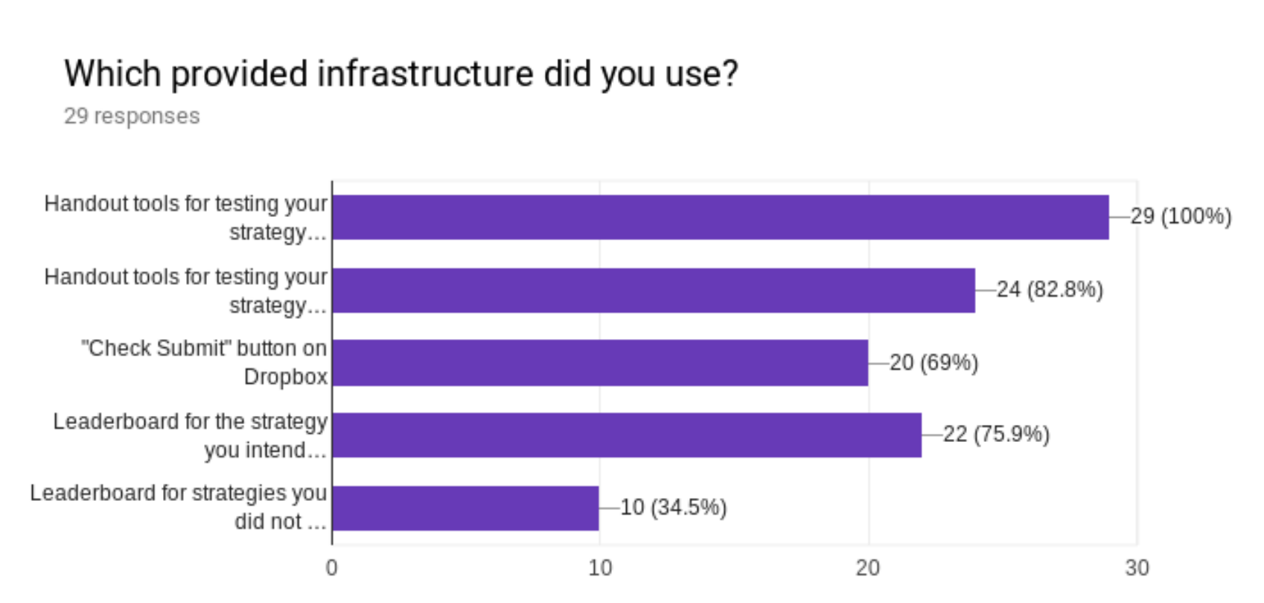
CS Dropbox Check Submit functionality

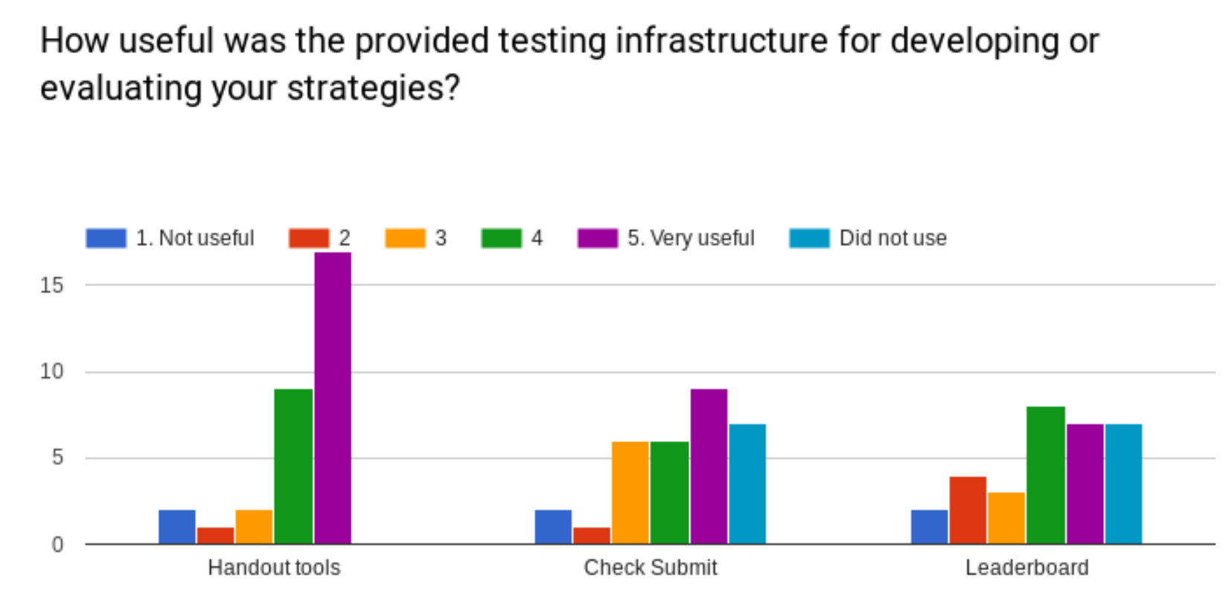
Opt-in auto-updating leaderboard displaying performance of student submissions

Decreased instructor time spent developing and grading



We eliminated student submissions of broken strategies





Students used these tools and provided positive feedback

Conclusions

COS 445's strategy exercises are highly modular and have been heavily automated

Successfully built tools which have improved student and instructor experiences in a popular theoretical CS course

This framework will be used in future offerings of the course

Future Work

I will reprise my role as an undergraduate course assistant for the course to continue to improve these resources

I have documented my framework so that future course assistants will be able to continue my work

Prof. Weinberg will ensure these tools are provided to future course assistants