Streamlining Grading toward Better Feedback

Streamlining Grading toward Better Feedback

Tommy MacWilliam
School of Engineering and Applied Sciences
Harvard University
Cambridge, Massachusetts, USA
tmacwilliam@cs.harvard.edu

David J. Malan
School of Engineering and Applied Sciences
Harvard University
Cambridge, Massachusetts, USA
malan@harvard.edu

ABSTRACT

CS50 is Harvard University's introductory course aimed at majors and non-majors alike. Each week, students complete programming assignments and have traditionally received feedback from staff in the form of comments on PDFs of their code. Staff have historically reported spending significant amounts of time grading because of bottlenecks that included generating PDF documents and manually emailing feedback to students. Because we preferred that staff spend less of their time on grading logistics and more time providing feedback and helping students online or in person, we set out to improve the efficiency of the grading process.

In Fall 2012, we developed and deployed CS50 Submit, a web-based utility through which staff can leave feedback for students via inline "sticky notes." Following the introduction of CS50 Submit, staff reported grading for 10% fewer hours (i.e., 42 minutes) per week and 13% fewer minutes (i.e., 4 minutes) per student, even while providing as much or more feedback. Meanwhile, we observed significantly higher levels of engagement with the course's online discussion board among staff, suggesting a more favorable distribution of staff workload. With CS50 Submit, we have also been able to audit exactly how much time staff spent grading each week in order to identify additional bottlenecks.

Using CS50 Submit, we also observed that, on average, 9% of students each week never read their graders' comments, with a peak one week of 14%. The number of students who did not read feedback increased with time, which has led us to question whether asynchronous, textual comments are the most effective feedback mechanisms for students. In future terms, we plan to experiment with in-person, interactive means of delivering feedback to students.

In this paper, we present CS50 Submit and the insights it has yielded into the behavior of students and staff alike.

ITiCSE'13, July 1–3, 2013, Canterbury, England, UK.

Categories and Subject Descriptors

K.3.1 [**Computer Uses in Education**]: Computer-assisted Instruction, Distance Learning; K.3.2 [**Computer and Information Science Education**]: Computer science education, Curriculum

General Terms

Design, Experimentation, Human Factors

Keywords

assignment submission, online feedback

1. INTRODUCTION

CS50 is Harvard University's introduction to computer science that combines concepts typically taught in CS1 and CS2 into a one-semester course aimed at majors and non-majors alike. Over the course of a 12-week semester, students complete seven programming assignments and receive feedback from teaching staff in the form of inline comments on source code. So that students can take graders' feedback into account on future assignments, staff generally return feedback to students as quickly as possible, typically within seven days of submission.

Historically, the workflow for providing qualitative feedback has been inefficient for staff. In years past, staff have rendered and annotated PDF documents of students' source code. However, downloading students' submissions, navigating long PDFs, and individually emailing students their feedback have proven to be bottlenecks in our grading process. As a result, staff have reported spending disproportionate amounts of their time per week providing feedback as opposed to teaching or interacting directly with students: in Fall 2011, staff graded for an average of 7.2 hours out of the 17.2 hours (42%) they worked each week. We preferred, though, that staff spend more of their time providing feedback to students, whether online or in person, and less of their time on logistics, so we set out to improve the efficiency of the grading process.

Additionally, we worried that many students never took the time to read their graders' comments. Because PDFs were simply emailed to students, we did not know if students were even reviewing their comments. We thus sought to confirm or deny these suspicions in order to decide if we should explore new methods of feedback so that we might better connect with students.

And so, in order to streamline the grading and submission process in Fall 2012, we deployed CS50 Submit, a web-based

utility with which staff could review problem sets' submissions and attach comments to source code via inline "sticky notes," to approximately 70 graders and 700 students. With the introduction of CS50 Submit, staff reported grading for fewer minutes per student (a reduction of 13%) and fewer total hours per week (a reduction of 10%), which takes time spent on overarching setup and distributing feedback into account. At the same time, the average amount of time spent by staff answering students' questions online via the course's discussion forum more than doubled. (To be fair, we also transitioned to new forum software in Fall 2012, which might also explain the increase in engagement.) Furthermore, we observed that an average of 9% of students did not review their graders' feedback each week, and the number of students who failed to do so increased over time, peaking at 14%.

We present in this paper the pedagogy behind the design of CS50 Submit and its impact on staff and students alike. We first describe the grading workflow that CS50 Submit aimed to improve in Section 2. In Section 3, we explore related work. In Section 4, we describe the motivations for CS50 Submit's functionality, and in Section 5, we present our findings. We describe our future work in Section 6, and in Section 7, we conclude.

2. BACKGROUND

CS50 students complete seven problem sets of the course of the semester. The first six of these problem sets use C, while the final problem set challenges students to implement a database-backed website using PHP and SQL. Problem set submissions typically consist of three or more source code files, each of which are reviewed by teaching staff. At the beginning of the semester, typical submissions comprise around 50 lines of code, while by term's end, problem sets' implementations consist of several hundred lines of code (including starter code distributed with the problem set's specification). Problem sets are assigned to students weekly, and staff are given several days to grade a section of about 10 students.

Problem sets are graded along four axes: scope (how much of the problem set was attempted), correctness (the degree to which programs' output conforms to problems' specifications), design (a measure of the code's organization, clarity, and elegance), and style (how well-formatted and -commented the code is). Each axis is typically graded on a five-point scale, with a 5 representing "best," a 3 representing "good," and a 1 representing "poor." We provide teaching staff with rubrics with which they can objectively evaluate the axes of scope and correctness, but we leave the axes of design and style to a more open-ended, subjective analysis. We normalize grades at end of term to account for differences among graders.

Staff also provide qualitative feedback on students' code in addition to quantitative scores. Years ago, staff provided feedback by handwriting comments on physical, printed copies of students' source code, which proved to be an inefficient and unnecessarily costly workflow. More recently, staff downloaded their students' submissions from a central repository and created PDFs of students' source code using utilities including CS50 Render [5] and provided feedback through PDF annotation software such as Adobe Acrobat [1] and Bluebeam Revu [2]. Typically, qualitative feedback on PDFs took the form of short, inline annotations on source code as well as longer, overarching comments on the first page of the PDF. While we provided graders with Windows tablets in 2008 to augment our PDF-based workflow [7], staff have long reported that the process of creating, annotating, and distributing source code PDFs was still time-consuming. Prior to Fall 2012, graders reported that writing feedback consumed 7 to 8 hours per week (i.e., over 40% of their weekly time commitment) and over 30 minutes per student. These bottlenecks in workflow also detracted from time spent with students, as the time consumed by setting up a grading environment could be better spent leaving higher-quality feedback or answering students' questions.

Moreover, we have historically lacked insight into the diligence with which staff reviewed submissions, which we suspect is correlated with the amount of time spent grading and the length of feedback. Without collecting annotated PDFs from graders and manually auditing comments, which would prove to be a logistical challenge with 70 staff, we have traditionally been unable to evaluate the quality of staff's feedback. Similarly, we have not been able to identify how many students actually take the time to review their graders' feedback on a weekly basis.

3. RELATED WORK

Others have found a link between the introduction of web-based feedback mechanisms and improvements in staff's grading workflow. In a comparison of email-based feedback and online feedback, Jones and Jamieson [9] found that web-based systems allow staff to grade significantly faster, and automation in the grading workflow decreased grading time. Heaney and Daly [8] similarly report that course staff preferred leave feedback online, and in particular, staff liked the ability to attach comments inline with source code. Finally, Bridge and Appleyard [4] observe that the turnaround time for online feedback was shorter than that for traditional mechanisms.

Not only can web-based grading workflows benefit staff, but students also seem to prefer viewing feedback electronically. Dalgarno et al. [6] found that students supported the widespread use of online feedback, with a majority of students preferring to both submit assignments and view feedback online. Similarly, Bridge and Appleyard [4] observed that students found submitting assignments online to be faster and easier, and more than half of students preferred online feedback. Heaney and Daly [8] also report that students found electronic systems to be easy to use, and the introduction of an online feedback mechanism correlated with higher exam scores.

However, others note potential pitfalls with electronic submission tools. For example, Bridge and Appleyard [3] note that students unfamiliar with online submission utilities may prefer more traditional means, and instructors must be prepared to deal with the possibility of systems' failure.

Many Learning Management Systems, such as Moodle [10], include modules for assignment submission. However, CS50 Submit also allows instructors to attach comments to files inline while giving students the ability to submit code via the command-line and browse their submission histories. Other platforms designed specifically for source code submissions, such as Web-CAT [11], integrate unit tests into the submission process. While CS50 Submit does not automatically test source code, it is designed to accept a wider variety of file types, including Word documents, PDFs, and images.

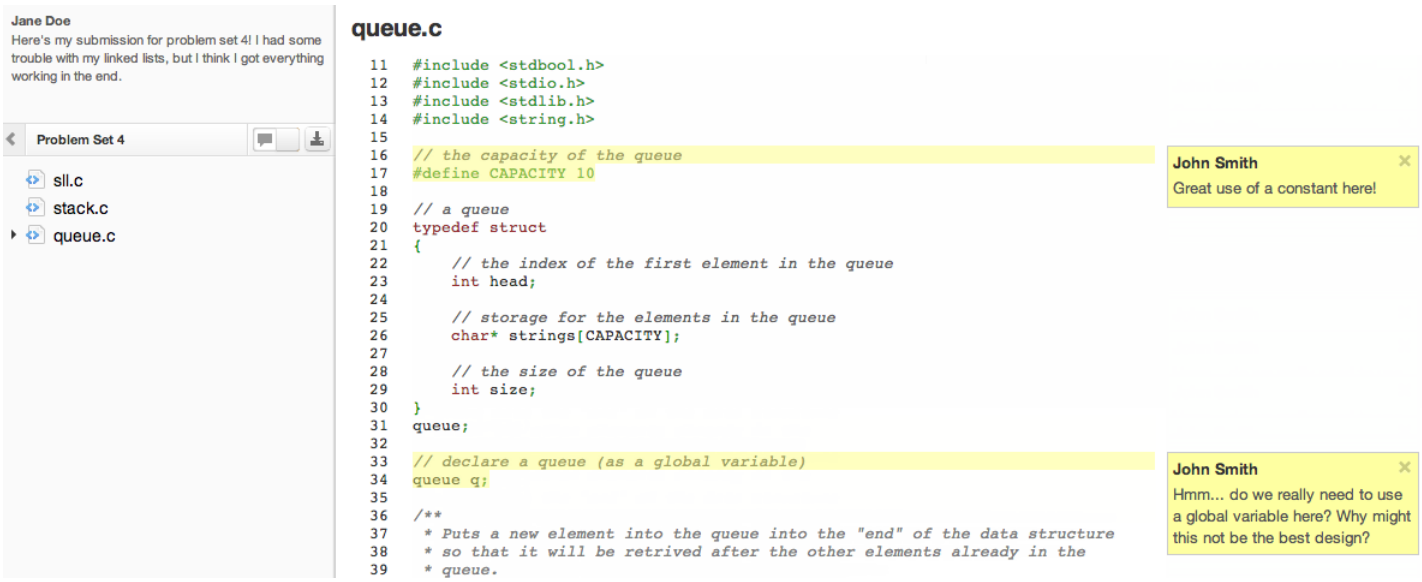

Figure 1: Using CS50 Submit, graders can attach inline comments to source code in the form of "sticky notes" that can be viewed by students. Depicted here are John Smith's comments on Jane Doe's work.

4. CS50 Submit

CS50 Submit is a web-based system that facilitates the collection of problem set submissions and annotation of source code. Using CS50 Submit, students can submit problem sets using either a command-line utility or a web browser. CS50 Submit tracks students' submission history for all problem sets, so students can receive preliminary feedback from graders as well as reference earlier submissions during development. Per Figure 1, CS50 Submit renders source code files in a web browser and allows graders to highlight lines of code and attach inline comments in the form of "sticky notes" that can be viewed by students. Using inline comments, graders can more easily contextualize their feedback, as students can clearly see the lines of code to which each comment pertains. Nevertheless, because graders may wish to leave overarching comments that pertain to the submission as a whole, CS50 Submit also allows staff to create a longer "summary comment" that can be viewed alongside inline comments. Students can also download PDFs of their submission (with comments attached) for their records. By centralizing the submission and grading process with a single web application, CS50 Submit seeks to remove logistical bottlenecks in the grading process (e.g., rendering PDFs of source code and emailing students individually), which allows staff to spend a larger proportion of their time leaving feedback.

So that we can better understand the grading process from the perspectives of students and staff alike, CS50 Submit also logs certain actions. For example, CS50 Submit tracks not only the number and length of comments left by staff, but also the amount of time graders spend interacting with the web interface. Furthermore, CS50 Submit automatically notifies students once their graders have finished commenting on their submission in order to encourage students to review their feedback. CS50 Submit then records how many students actually reviewed this feedback online, allowing us to determine whether or not students actually review their graders' feedback.

5. RESULTS

CS50 Submit has yielded insights into the behavior of staff and students alike, including the amount of time staff spent grading, the number and length of comments left by staff, and the number of students who reviewed their feedback.

5.1 Impact on Staff

End-of-term surveys of staff indicate that CS50 Submit has made the grading workflow more efficient for staff. In Fall 2011, when staff graded problem sets by attaching comments to rendered PDFs of source code, graders reported that, on average, the grading process consumed 7.2 hours per week and 30.75 minutes per student. When asked which parts of the grading workflow proved to be the largest bottlenecks, representative staff responses included "getting all the files organized so you can grade them" and "there was just way too much downloading/setting up." Similarly, many graders commented that the grading workflow could be improved by "automat[ing] the collection and unpacking of student code and the generation of PDFs" and "a more convenient way to send students their PDFs." Past years' surveys echoed these sentiments as well: Fall 2010's staff reported grading for 7.95 hours per week and 32.69 minutes per student, and Fall 2009's staff reported grading for 7.25 hours per week and 37.39 minutes per student. With the introduction of CS50 Submit in Fall 2012, staff reported that grading consumed only 5.3 hours per week (–10% versus Fall 2011, normalizing for section size) and 26.86 minutes per student (–13% versus Fall 2011). Prior teaching experience did not significantly affect these results, as first-time and returning graders alike reported a decrease in grading time on average. In addition, far fewer responses to the same questions regarding grading bottlenecks and potential improvements cited setup and feedback distribution as issues with the grading workflow, which corroborates our perception of graders' improvements in efficiency.

Not only did the amount of time staff spent grading de-

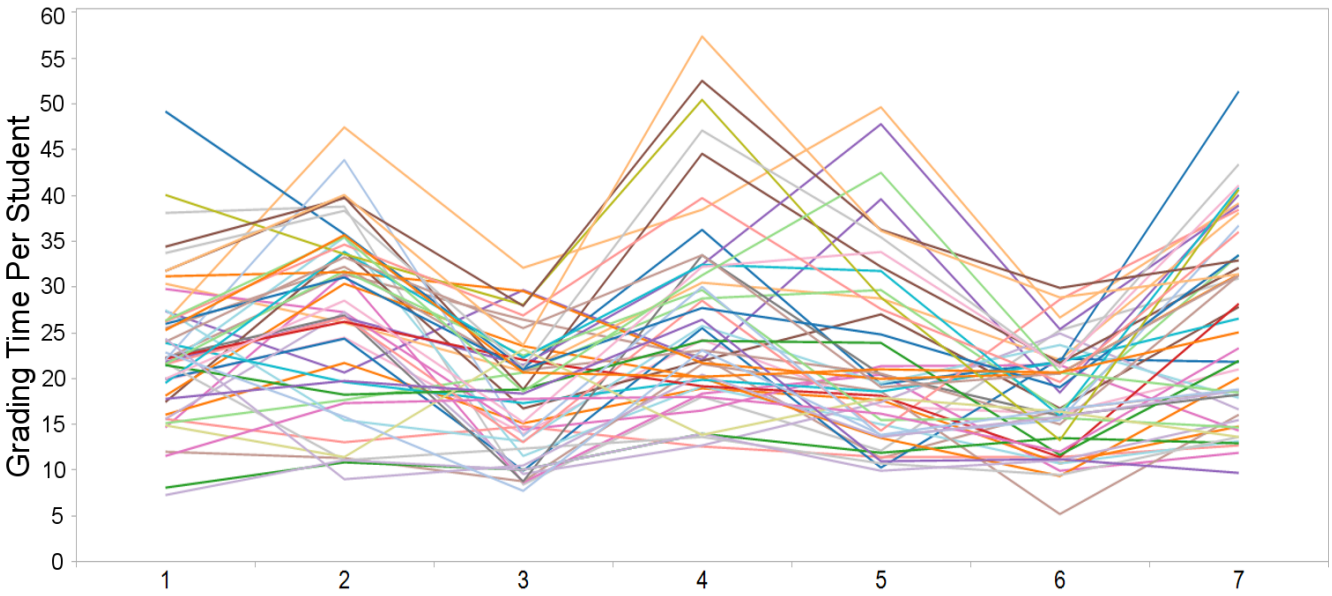

Figure 2: With CS50 Submit, we can track how many minutes our staff members spent grading over the course of the semester. Here, peaks at Problem Sets 2 and 4 (based on averages) indicate potential grading bottlenecks for those particular assignments. Each line represents one of the course's 70 graders. Omitted as outliers are staff who did not grade every week and graders whose peak times exceeded 60 minutes.

crease, but the amount of time staff spent helping students through other means also increased. For example, the amount of time spent by staff answering questions on the course's online discussion forum increased significantly. In Fall 2011, individual staff members reported spending an average of 0.47 hours per week answering forum questions, while in Fall 2012, staff reported spending an average of 1.11 hours per week (+136%). Furthermore, while in Fall 2011 62.2% of staff members reported they did not field questions on the discussion board on a weekly basis, only 12.3% of staff members reported they did not regularly answer questions online in Fall 2012. (Though the introduction of new discussion software in Fall 2012 may also have contributed to increases in participation). Similarly, staff also reported spending more time per week answering students' questions over email, with Fall 2011 staff reporting 1.65 hours per week and Fall 2012 staff reporting 1.82 hours per week (+10%).

Data obtained from CS50 Submit corroborates the results of end-of-term staff surveys. Figure 2 shows the number of minutes per student each staff member spent grading, according to CS50 Submit's logs, which is consistent with our self-reported data. This same graph reveals opportunities for more fine-grained improvement. For example, Problem Set 4 (which called for staff to inspect manually student-generated images for correctness) and Problem Set 2 had the highest grading times. In future semesters, we will explore the development of additional grading scripts specific to these problem sets to improve graders' efficiency. In the same vein, Problem Sets 3 and 6 required less time to grade (perhaps because they were easier), which suggests that we can channel staff's time into other activities, such as meeting with students or peer evaluations, those weeks. In future terms, we plan to use this same data to determine which graders might be spending too much time or too little time grading. We expect that analyzing grading times can help course heads identify and assist staff who are not grading efficiently, determine which sections of students are struggling, and pinpoint staff with particularly excellent feedback.

In addition to total time spent grading, CS50 Submit records the number and length of comments left by graders. Per Figure 3, peaks in grading time corresponded both to a larger number of comments and longer comments, as we expected. However, while the length of comments left by staff declined after the semester's midway point, with median word counts dropping from 105 to 75, the median number of comments on each submission did not change. Because these trends indicate that individual comments decreased in length over time, we plan to investigate whether the quality of feedback declined over the course of the semester or whether students' submissions improved and required less feedback.

5.2 Impact on Students

Using CS50 Submit, we also determined that, on average, 9% of students did not review their graders' feedback each week. Per Figure 4, the number of students who did not read feedback increased over the course of the semester. While fewer than 2% of students did not read their feedback for Problem Set 1, 12% of students were not reviewing feedback by term's end, with a peak of 14% at Problem Set 5. However, the amount of time graders spent writing comments and the length of graders' comments were not correlated with the number of students who did not review feedback.

Because students' interest in reading feedback appears to be waning over time, we intend to explore the effectiveness of other feedback mechanisms. Asynchronous and longhand comments may not be the most effective means of delivering feedback to students, particularly given that the number

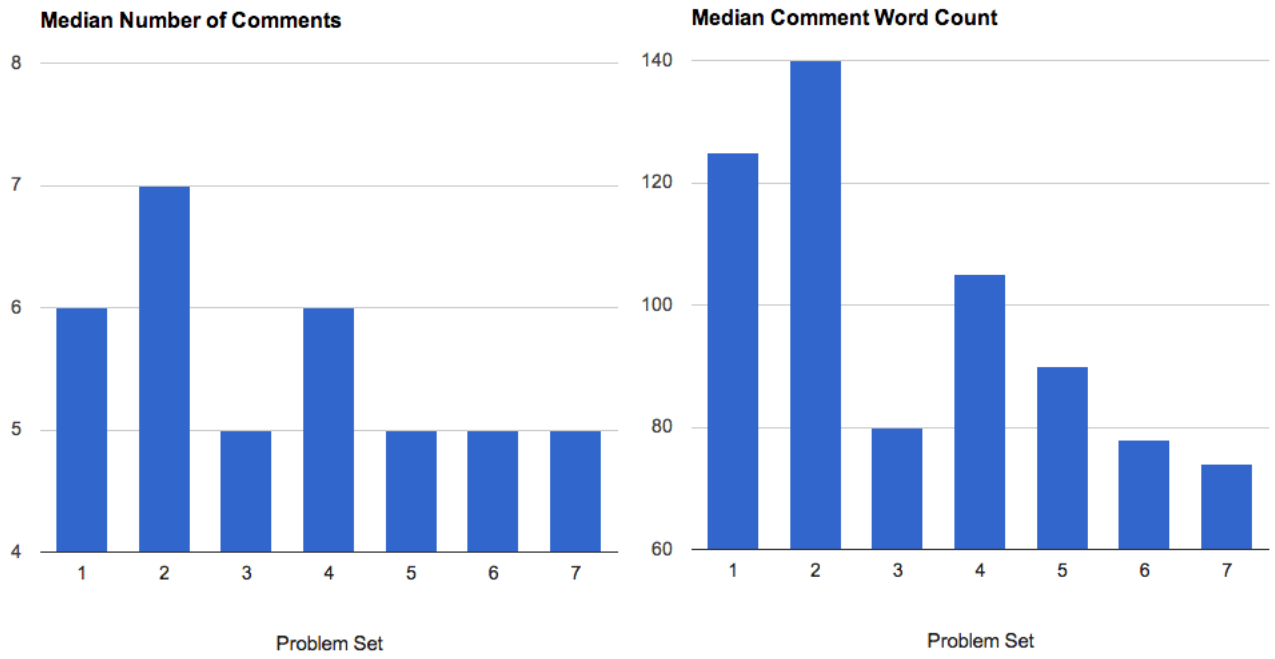

Figure 3: While the number of comments graders left on each submission remained the same for the second half of the semester (left), the length of comments decreased (right). We plan to investigate whether the quality of staff's comments declined or whether students' code improved and required less feedback.

of students reading textual comments decreased over the course of the semester. We now wonder, based on these results, whether more interactive, in-person code reviews might instead prove to be more effective delivery mechanisms for feedback. In future terms, we also plan to survey students to understand their motivations for not reviewing feedback.

Though staff spent less time grading in Fall 2012, students' perceptions of feedback were not significantly different compared to past terms, as the percentage of students satisfied with the amount of feedback they received has remained largely unchanged. In a course-wide, end-of-term survey, 86% of students in Fall 2012 reported that they received "enough feedback," compared to 89% in Fall 2011 and 85% in Fall 2010. Even so, we believe that experimenting with more interactive feedback mechanisms such as code reviews may increase students' satisfaction with their feedback on problem sets.

While CS50 Submit is not intended to replace source control management software, many students took advantage of CS50 Submit's ability to track submission history. On average, 23% of students submitted each problem set two or more times, and the number of students who submitted only once declined over the course of the semester. We thus hypothesize that a non-trivial number of students used CS50 Submit to back up their code, receive preliminary feedback from graders, and submit early to ensure a version of their code was received before the deadline, though we plan to investigate these habits further.

6. FUTURE WORK

In future terms, we plan to investigate the trend of students' declining interest in reviewing their graders' comments. First, we intend to survey students in order to better understand their motivations for not reviewing. Our questions include:

- Do students find written feedback to be less helpful on later problem sets, when they have more comfort with course material?
- Do students have less time to read long comments later in the semester?
- Do students find feedback to be less relevant to future problem sets as the term progresses?
- Does the quality of feedback from staff decline over time?

Once we have a better sense of students' reasons for neglecting feedback, we will explore alternative means of assessment. In-person code reviews are of particular interest, as we hypothesize they will prove more engaging than asynchronous, textual comments. We intend to experiment with the implementation of interactive feedback sessions, which could take the form of one-on-one meetings between students and staff or larger sessions led by one staff member and several students. Furthermore, we hope to motivate students to review their feedback via email reminders to those students who have become unengaged. CS50 Submit could also be used to facilitate collaboration among students, as students can comment on each other's code via the web interface.

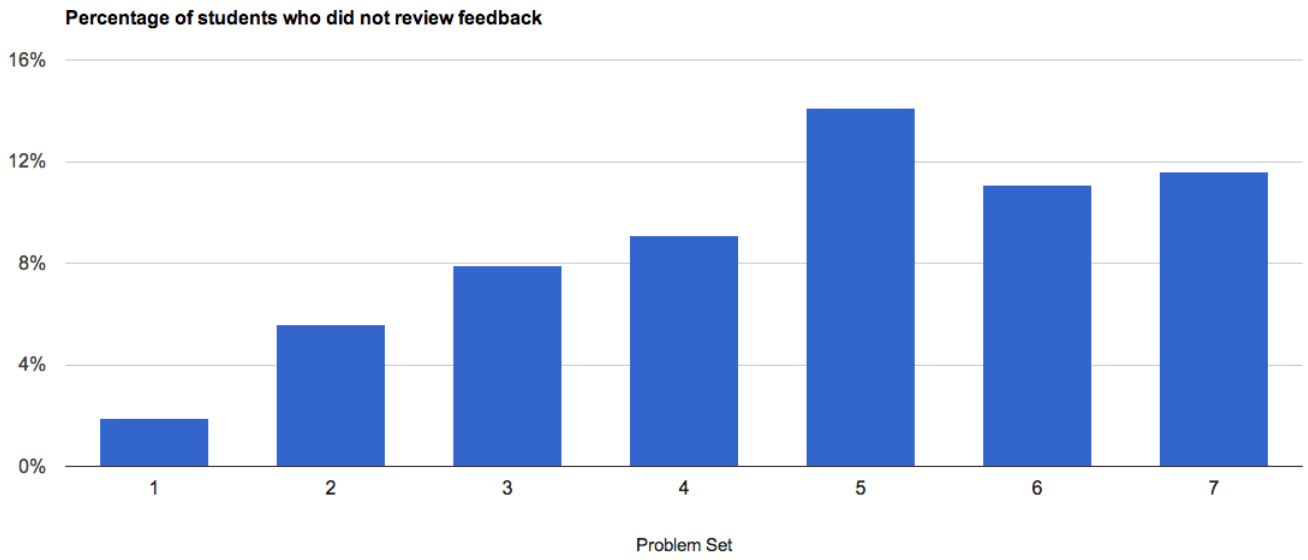

Figure 4: The percentage of students who did not review their graders' comments increased over the course of the semester, reaching 12% by the end of the term and peaking at 14% for Problem Set 5.

We also intend to use the data collected by CS50 Submit to make adjustments to our grading process mid-semester. For example, if in future terms we again observe a decline in average comment length, we can audit the quality of graders' feedback before additional problem sets are returned to students, lest the trend indicate declining quality of feedback on our part. Moreover, grading-time data will help us identify staff who are struggling to keep pace with the grading process and correct new bottlenecks in the grading process throughout the term. Finally, we intend to reach out to students who become disengaged with the feedback process so that we might find more beneficial ways to connect with them.

7. CONCLUSIONS

Teaching staff in CS50 have historically spent a large proportion of their time writing qualitative comments on students' weekly problem set submissions, in part because of bottlenecks in the grading process. We have also lacked insights into how many students actually took the time to review their graders' comments. We thus developed and deployed CS50 Submit, a browser-based submission and commenting utility, in Fall 2012. Ultimately, our goal was to increase graders' efficiency by removing grading bottlenecks (e.g., generating PDFs of source code and emailing students individually) so that graders could spend less time with logistics and more time leaving feedback.

Upon CS50 Submit's introduction, staff reported not only a 13% decrease in grading time per student, but also an increase in the amount of time spent answering students' questions online. We also observed that an average of 9% of students did not review their graders' comments, with the number of students neglecting their feedback increasing over the course of the semester (peaking at 14%). As a result, we plan to experiment with code reviews in future terms, with the hope that staff's feedback can more effectively help students learn.

8. ACKNOWLEDGMENTS

Many thanks to R.J. Aquino, Rob Bowden, and Joseph Ong for their contributions to this work.

9. REFERENCES

[1] Adobe Systems, Inc. Adobe Acrobat. http://www.adobe.com/products/acrobat.html.

[2] Bluebeam Software, Inc. Bluebeam Revu. http://www.bluebeam.com/us/products/revu/.

[3] P. Bridge and R. Appleyard. System failure: A comparison of electronic and paper-based assignment submission, marking, and feedback. *British Journal of Educational Technology*, 36(4):669, 2005.

[4] P. Bridge and R. Appleyard. A comparison of electronic and paper-based assignment submission and feedback. *British Journal of Educational Technology*, 39(4):644–650, 2008.

[5] CS50 Render. http://github.com/cs50.

[6] B. Dalgarno, A. Chan, P. Adams, P. Roy, and D. Miller. On campus and distance student attitudes towards paperless assessment and feedback. *Proceedings of the ICT: Providing choices for learners and learning. Proceedings ascilite Singapore 2007*, pages 168–178, 2007.

[7] David J. Malan. Grading qualitatively with tablet PCs. *Workshop on the Impact of Pen-Based Technology on Education*, 2009.

[8] D. Heaney and C. Daly. Mass production of individual feedback. In *ACM SIGCSE Bulletin*, volume 36, pages 117–121. ACM, 2004.

[9] D. Jones and B. Jamieson. Three generations of online assignment management. *IN Kevill, R., Oliver, R. & Phillips, R.(Eds.) ASCILITE*, 97, 1997.

[10] Moodle. http://moodle.org.

[11] Web-CAT. http://web-cat.org/group/web-cat.