Graphing Performance on Programming Assignments to Improve Student Understanding

Stephen H. Edwards, Manuel A. Pérez-Quiñones, Matthew Phillips, and Johnny RajKumar
Department of Computer Science, Virginia Tech, 660 McBryde Hall (0106),
Blacksburg, VA 24061 {edwards, perez}@cs.vt.edu, {mphillip, johnny}@vt.edu

Abstract **- Within computer science education, automated grading systems are used by many institutions. This paper summarizes an investigation into how the data collected by an electronic submission system can be used to aid students and instructors. Rather than simply providing feedback on a single submission, a grading system can give a student summary information about individual improvement over time, as well as where the student stands with respect to his or her peers. We explore graphical presentations—in the form of bar charts, histograms, and line charts—of a student's personal progress over time, as well as the student's current performance in relationship to the remainder of the class body. Particular attention is paid to how graphs can help the student understand likely future outcomes on assignments based on current effort expended, and on "calibrating" one's own understanding of how the rest of the class is performing.**

Index Terms - Computer science education, automated grading, information visualization, learning management systems, Web-CAT.

INTRODUCTION

Within computer science education, automated grading systems have been in use for many years at a variety of institutions. Such systems allow students to submit programming assignments, which are then compiled, executed, and automatically assessed. Automated tools can provide directed, concrete feedback to students with rapid turn-around times, increasing the number of feedback cycles students participate in.

This paper reports on a preliminary investigation into how the data collected by an electronic submission system can be used to aid students and instructors. Rather than simply providing feedback on a single submission, a grading system can give a student summary information about individual improvement over time, as well as where the student stands with respect to his or her peers. To this end, we explore graphical presentations—in the form of bar charts, histograms, and line charts—of a student's personal progress over time, as well as the student's current performance in relationship to the remainder of the class body. The paper examines in detail four visualizations that were created for student use, and four visualizations created for instructor tracking purposes. Particular attention is paid to how graphs can help the student understand likely future outcomes on assignments based on current effort expended, and on "calibrating" one's own understanding of how the rest of the class is performing.

The graphical visualizations discussed were implemented in an existing automated grading system that is used by the CS departments at several universities. Experiences with these visualizations in the classroom are discussed, together with the results of a survey of undergraduate student reactions to the graphs. While the work is discussed in the context of computer science courses, similar techniques can also be applied to help students visualize performance on other kinds of assignments in other disciplines, perhaps as a course management system feature.

BACKGROUND AND RELATED WORK

Web-CAT is an automatic grading system that evaluates student programming projects [2] [3]. One of Web-CAT's most prominent features is that it allows instructors to set up assignments that require students to submit test cases along with their programs so that both can be graded together. Once the student submits their program and its test cases to Web-CAT, Web-CAT grades their submission and immediately returns scoring information to the student. If the student isn't happy with her score, she can alter her program and/or her test cases and resubmit to Web-CAT. Students are allowed multiple submissions for each programming project.

Prior to the work reported in this paper, when Web-CAT would return a submission score to a student, he/she would not receive any information about how others had performed, such as the average submission score for the class, how one's submission scores have improved over time, or where the best opportunities for improvement were. The lack of such information limits the student's insights into their progress. In this paper we present visualizations that have been added to Web-CAT to aid students.

Course management systems (CMS) help facilitate online student learning and management of large courses [1]. Systems like Web-CAT [2] [3] extend the traditional CMS by providing automated grading for student programming projects. This automated grading generates data on how the student performed in relation to himself/herself and in relation to his/her peers. It can be useful to visualize this grading data in order to view trends and gain an understanding of how the student is progressing in the course [4] [5] [6].

Visualizing information such as data logged from Web-CAT can lead to the realization of trends in data. Schneiderman [4] discusses that visualizations can provide a much higher bandwidth of information when compared to textual data. This higher bandwidth can allow users to quickly scan data and look for changes in size, color, and shape.

While little or no specific work has been published on creating visualizations for student progress in a course, Mazza and Dimitrova [6] have examined visualizations in course management systems (CMS). Their work focused on visualizing log data generated by student activity (accessing of web pages, participation in course discussion boards, etc.), but they did touch on visualizing a student's performance on quizzes. They found that matrix plots were useful visualizations. To build the matrix visualization, they entered values associated with a student's performance on a quiz and mapped each value to a color in a range of colors from white (being low performance) to black (being high performance). Once these colors were shown graphically, it was easy to see trends showing what concepts on the quizzes students didn't understand. We can use this visualization technique and this research in general as a starting point for visualizing student performance in Web-CAT.

INFORMING STUDENTS THROUGH VISUALIZATIONS

By creating visualizations for the student in Web-CAT, we can increase the student's understanding of how they are performing on their current and past submissions, their current and past assignments, and overall in the course.

In order to create the proper visualizations from the proper data in Web-CAT we went through an iterative process of discussing student needs with instructors that use Web-CAT in their courses. The product of this iterative process was a set of four refined graphs that visually provide valuable information to the student. We then evaluated these visualizations and proceeded to implement some of them and add others.

I. Assignment Score Distribution for the Class

The first visualization that we made available to students is a histogram titled *Score Distribution for the Class*. An example of this graph is shown in Figure 1. The goal of this visualization is to provide the student with an insight into how he/she is performing on the current assignment when compared to the rest of the class. We chose a standard histogram format since most students are familiar with such graphs and can interpret them readily. The student's individual performance is marked by a top-to-bottom red line laid over the histogram's broader bars.

Prior to this histogram, Web-CAT did not have any method for showing a student how her/his score on an assignment compared to the scores of his/her peers. The students often would post in the forum questions like "I am getting a 92, does anybody else have a score higher than that?" Showing performance within the class allows the student to assess their progress compared to their peers. We feel that this histogram is an ideal way of showing the student this information.

II. Opportunities for Improvement

The second visualization that we made available to students is an area chart that shows opportunities for improvement,

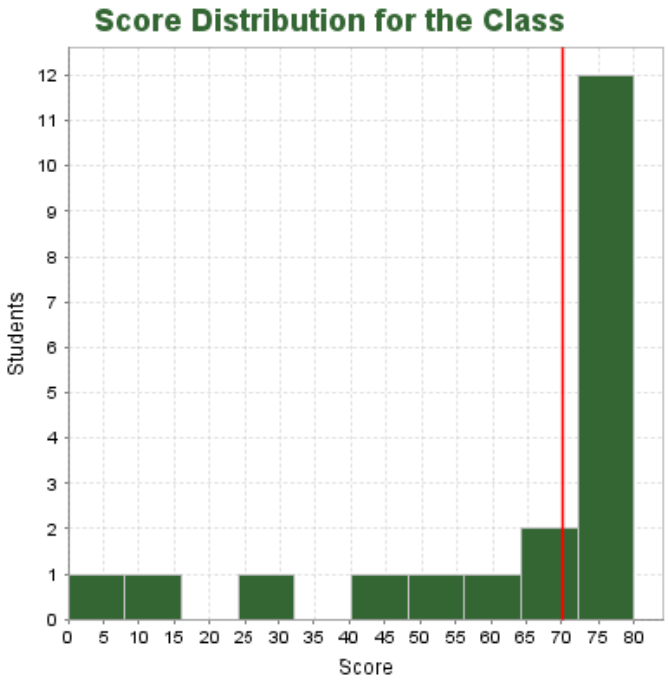

FIGURE 1
THE STUDENT'S SCORE IS SHOWN IN THE CONTEXT OF THE TOTAL DISTRIBUTION ACROSS THE CLASS.

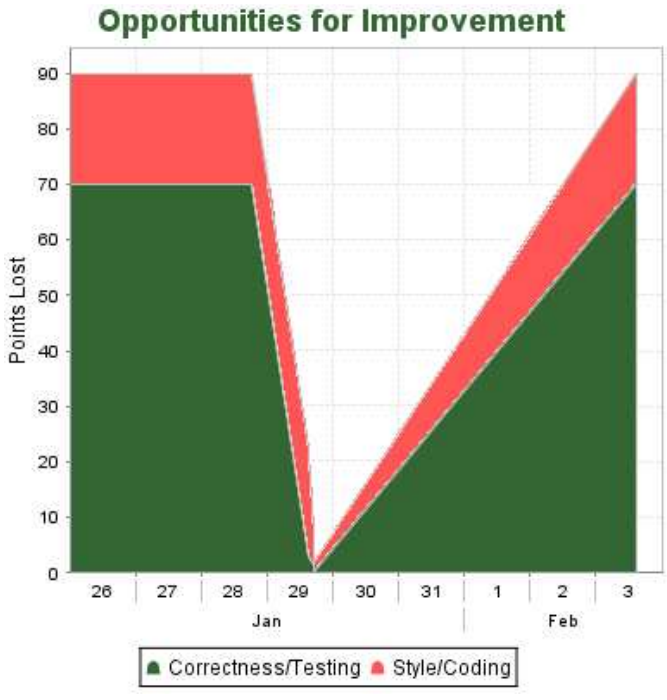

FIGURE 2
A STACKED AREA CHART SHOWS BREAKDOWN OF POINTS LOST ON PAST SUBMISSIONS, INDICATING OPPORTUNITIES FOR IMPROVEMENT.

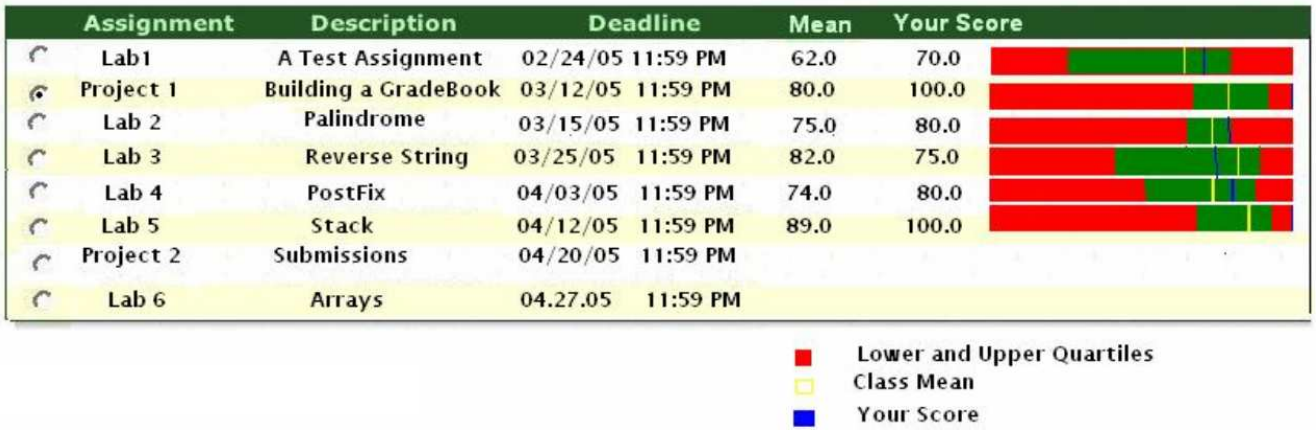

FIGURE 3
A HORIZONTAL VARIANT OF A CANDLESTICK CHART DISPLAYS QUARTILES, THE MEAN, AND THE INDIVIDUAL STUDENT SCORE FOR EACH ASSIGNMENT.

illustrated in Figure 2. The goal of this visualization is to provide the student with an insight into where he/she is losing points over multiple submissions. During the submission process, Web-CAT grades a submission for two items: Coding/Style and Correctness/Testing. The Coding/Style category is used to award points for things like documentation, indentation, and general style. The Correctness/Testing category is used to award points to students for things like extensiveness of testing, error checking, and correctness of behavior. Figure 2 shows a snapshot in time for a student who has made a series of submissions over several days. The area chart shows the amount of credit that has been lost for each of the two assessment categories.

We have found students who leave the Coding/Style portion of their work for late in the submission period may scramble to do a good job. They often feel like the documentation can be done at the end. This graph clearly shows the relationship of the two parts of their grade. Students should be able to see what area(s) they need to improve in their submissions in order to get their desired grade. In particular situations, this graph should help student see trends in their programming habit. For example, a student might start loosing Coding/Style points as he/she adds new functionality without documenting it properly. At the same time, the new functionality might afford the student higher points in Correctness/Testing. Such a situation might produce the same score as before, but this graph should communicate to the student clearly how the composition of the score changed over time.

III. Quartiles with Mean Score over Several Assignments

The third visualization that we made available to students, shown in Figure 3, is a horizontal bar chart that is similar to the traditional notion of a candlestick chart. The goal of this visualization is to provide the student with a general idea of how he/she is doing compared to his/her peers within each of the assignments in the class. The horizontal bars appear in Web-CAT's interface when students are choosing among a list of assignments, for example, when a student is selecting among all past assignments to view feedback on a previous submission.

Beside each listed assignment is a horizontal bar graph. The outermost left and right red regions depict the lower and upper quartiles. The green central region depicts the two middle quartiles. The mean assignment score for the class is drawn with a vertical yellow line that splits the green central region into the second and third quartiles. Finally, the student's individual score appears as a vertical blue line overlaying the graph.

Using these horizontal bar charts, the student can quickly get a general idea of how he/she is performing compared to his/her peers on various assignments.

IV. Projected Score for Next Submission

The fourth visualization that we designed is a line chart that projects the score for a student's next submission, as exemplified in Figure 4. The goal of this visualization is provide the student with an estimate of how he/she is likely to perform on his/her next submission.

The projected scores are calculated by taking a sliding window of previous scores and performing a simple linear regression on those scores. From the linear regression we are able to calculate and plot the next extrapolated point. Obviously, the further the data is extrapolated, the higher the error margin becomes. We explain to the student that the projected points are prone to error.

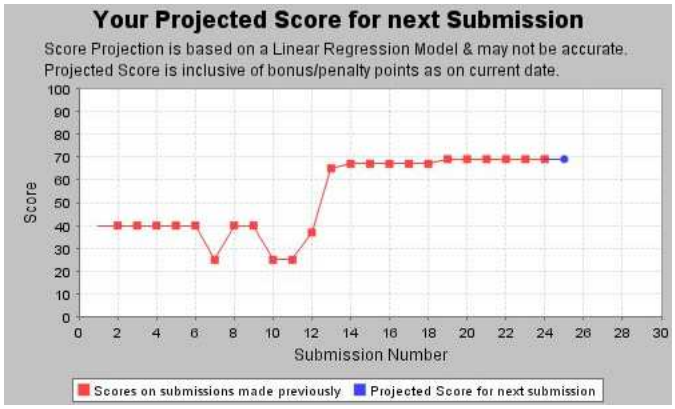

FIGURE 4
A STATISTICAL PREDICTION OF PERFORMANCE ON THE NEXT SUBMISSION.

Having information on projected scores provides the student with insight on how he/she should budget their time for future submissions. That is, if the student is looking to get a certain score on the assignment due date, he/she can get a feeling for how many submissions he/she will need to make to get that score if he/she continues at his/her current rate.

On a higher level, this visualization can provide insight to the student about how he/she is refining his/her project for each submission. For example, it can indicate to the student that he/she should revise his/her programs more heavily between submissions if he/she wants to see a greater increase rate in his/her submission score.

INFORMATION FOR THE INSTRUCTOR

In addition to supporting students, we are also interested in helping instructors manage their electronically graded assignments. By creating visualizations for the instructor in Web-CAT we can assist the instructor in managing student progress on assignments and increase the instructors knowledge of the problems that students face when trying to complete an assignment.

In order to create the proper visualizations from the proper data in Web-CAT we went through an iterative process of discussing our and other instructor's needs with the instructors that use Web-CAT to manage programming assignments in their courses. The product of this iterative process is four refined visualizations that provide valuable information to the instructor.

I. Class Performance on the Current Assignment

The first visualization that we made available to the instructor is a histogram showing the score distribution for one assignment across all students. This graph is nearly identical to the student-oriented graph shown in Figure 1, except that there is no marker for one's individual score. This visualization also provides class median and class average scores below the title.

Using this histogram, the instructor can gain insight into how a class is performing as a whole on the current assignment. This visualization can help the instructor spot problems with the assignment if the score distribution looks odd. For instance, if it appears that no student is scoring higher than 40%, this might indicate to the instructor that he/she has an error in the program specification or that the class does not understand a key concept.

II. Number of Submissions per Student

Figure 5 shows the second visualization for instructors, a histogram showing the number of submission attempts completed by each student, arranged in increasing order. Web-CAT allows instructors to set limits on the number of submissions a student can make for a given assignment, or leave submissions unlimited. In Figure 4, each vertical bar represents the submissions made by one student, and the horizontal axis represents that students "rank" in terms of number of submissions made. This graph gives an instructor a better feel for both how many students have made submissions, and the distribution of how many submissions each student is making.

III. Number of Student Submissions across Time

Figure 6 shows the third visualization designed for instructor use, a histogram that shows the number of student submission across time. The goal of this visualization is to aid the instructor in understanding student submission patterns, particularly as an assignment deadline approaches. For example, an instructor might want to know a week before the deadline how many submissions have been made.

Using this histogram, an instructor is able to gain insight into the submission patterns of students and adjust due dates, early bonuses, and late penalties if needed. The instructor can

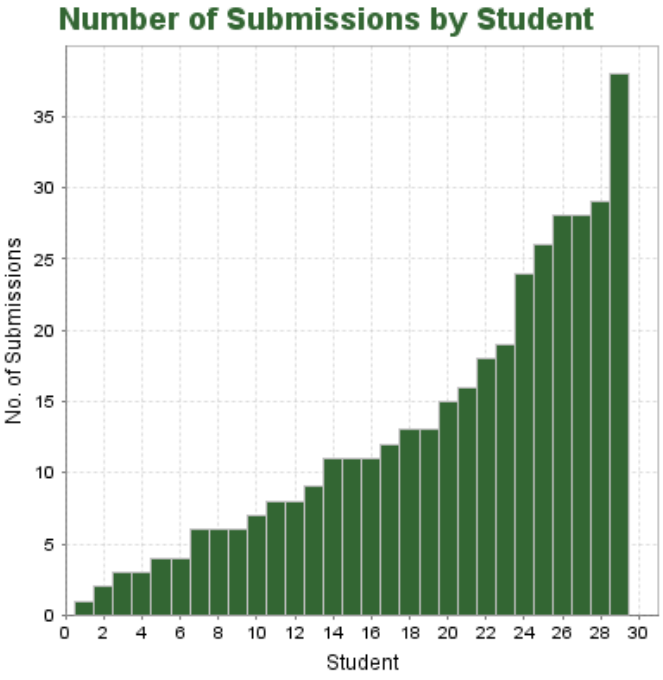

FIGURE 5
DISTRIBUTION OF THE NUMBER OF SUBMISSIONS MADE BY EACH STUDENT, PLOTTED IN INCREASING ORDER.

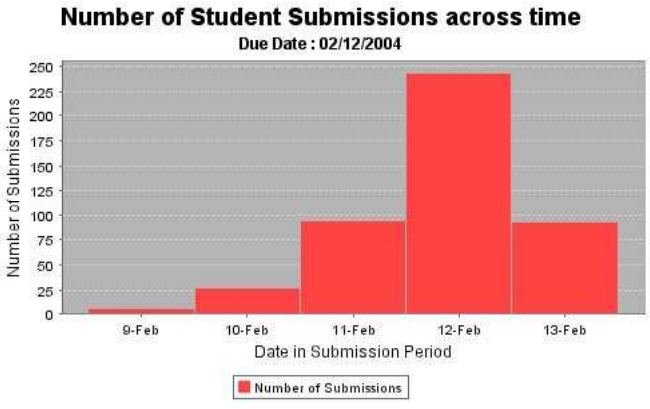

FIGURE 6
INSTRUCTORS CAN VIEW CHANGES IN THE FREQUENCY OF STUDENT SUBMISSIONS OVER TIME.

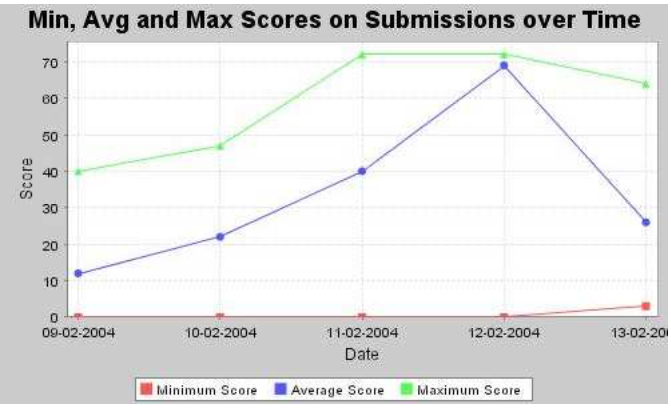

FIGURE 7
MINIMUM, AVERAGE AND MAXIMUM SCORES ACROSS ALL STUDENTS CAN BE TRACKED OVER TIME TO GAUGE CLASS PROGRESS.

also identify situations when an unusual number of students might be starting a project too late. This might lead to a discussion in class to try to identify the problem.

IV. Score Progression over Time

Figure 7 presents the fourth instructor-oriented visualization, which is a line chart that shows the minimum, mean and maximum score achieved by anyone in the course over time. The goal of this visualization is to help the instructor gain insight into the range of student scores for the current assignment, and how this distribution changes as an assignment deadline approaches.

The maximum points are calculated by taking the highest submission score by any student for that day. The minimum points are calculated similarly. The average points are plotted by taking the average of all the submission scores for that day.

Under normal circumstances, this graph should be monotonically increasing. Using this line chart, the instructor can gain insight into the spread of the class scores. This insight may help an instructor guide a class along if he sees that there are odd shapes to the graph. For example, if the instructor sees that all three lines are very close together, he might infer that there is a problem with the Web-CAT grading or that the students are struggling with a certain key algorithm.

EVALUATION

We performed a preliminary evaluation of the visualizations designed for student use presented above. For the evaluation we used mock-up versions of the visualizations. We evaluated the four student visualizations using two sections of a sophomore Introduction to Object-Oriented Development class at Virginia Tech. These sections regularly used Web-CAT for all of their closed lab assignments and class projects. The visualizations were evaluated on three different aspects: clarity and intuitiveness, syntax, and usefulness. A total of 24 students completed the survey.

To test the clarity and intuitiveness of the visualization, we asked the students to indicate how strongly they agreed with the following statement: "This graph is clear and concise. It is intuitive and stands on its own." We used a Likert scale with the following values: Strongly Disagree, Disagree, Neither Agree nor Disagree, Agree, Strongly Agree.

We assessed the student's understanding of the syntax of each visualization by asking a question that required the student to interpret the graph. For example, the question for the *Score Distribution for the Class* graph (Figure 1) was "How many students have a score between 60 and 80?" The full set of task-oriented questions appear in Table 1.

TABLE I
VISUALIZATION AND UNDERSTANDING EVALUATION

Visualization	Question
Score Distribution for the Class (Figure 1)	How many students have a score between 60 and 80?
Opportunities for Improvement (Figure 2)	How many correctness/score points were lost on the fourth submission?
Quartiles with Mean Score over Assignments (Figure 3)	For Lab 1, is your score above or below the mean?
Projected Score for Next Submission (Figure 4)	What is the approximate projected score for your the next submission

To test the usefulness of the graphs shown in Figures 1 and 3, we asked the student to indicate a level of agreement with the following statement: "This graph would be useful in Web-CAT when trying to determine how your score on the current assignment compares to the scores of your classmates." We asked a very similar question for Figures 2 and 4, with the wording slightly changed to assess if the visualization was useful when evaluating their own submissions.

Students liked the class distribution histogram shown in Figure 1. 19 out of 24 students agreed or strongly agreed that it was both clear and intuitive and 21 out of 24 agreed that it was useful. 19 out of 24 students answered the corresponding question from Table 1 correctly, showing that they understood the syntax.

Students were also for the most part in favor of Figure 2. 14 out of 24 students agreed or strongly agreed that it was both clear and intuitive and 17 out of 24 agreed that it was useful. 19 out of 24 students answered the corresponding question from Table 1 correctly, showing that they understood the syntax.

Students seemed to also favor Figure 3. 15 out of 24 students agreed or strongly agreed that it was both clear and intuitive and 18 out of 24 students agreed or strongly agreed that it was useful. 23 out of 24 students answered the corresponding question from Table 1 correctly, showing that they understood the syntax of Figure 3.

Students seemed to have mixed feelings about the projection graph in Figure 4. 22 out of 24 students agreed or strongly agreed that it was both clear and intuitive, but only 8 out 24 agreed or strongly agreed that it was useful. However, the students seemed to understand the syntax of Figure 4 very well, with 24 out of 24 answered the question from Table 1 correctly.

We believe that students were mixed on Figure 4 for two reasons. First, the wording of the question was not as clear and succinct as it should have been. Second, the visualization is built to predict the score for the next submission. That is,

submission numbers are placed along the X axis. It might be more useful if the visualization predicted scores according to date, with dates placed along the X axis.

IMPLEMENTATION

As a result of the promising feedback from the student survey, we have implemented several of the visualizations and are now actively using them in class. Web-CAT is implemented in Java, so we have used the JFreeChart library to generate graphs dynamically from data stored in the system. Figures 1, 2, and 5 are actually live images produced by Web-CAT from real submission data.

Because of the results of our preliminary evaluation, we also made several changes to our visualizations. First, note that the evaluation results indicated that students had mixed feelings about the clarity and intuitive nature of the horizontal bars in Figure 3, which were intended to summarize class performance on a series of assignments. We surmised that this may be due to the use of a candlestick-like depiction, since such graphs are known to be less intuitive. As a result, we modified the visualization when it was implemented, so that Web-CAT would present the class distribution in the form of a more traditional distribution histogram. Figure 8 shows the result.

Second, the preliminary evaluation indicated that students also had mixed feelings about the clarity and intuitiveness of Figure 2, which shows points lost, or opportunities for improvement. We surmised that this might be due to the fact that the graph as shown is really a "negative" plot, since it shows points lost rather than points earned. When we implemented this visualization, we decided to generate two versions of the graph: the first shows points earned broken down by category, in the same format as Figure 2. The second is the original visualization from Figure 2. Providing both views allows students a more conventional view of the data while also emphasizing the information content that originally inspired Figure 2.

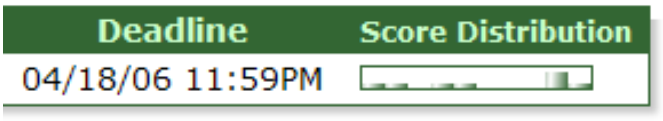

FIGURE 8
A CROPPED PORTION OF A LIST OF ASSIGNMENTS SHOWS A MINIATURE SCORE DISTRIBUTION HISTOGRAM, REPLACING THE PROTOTYPE SHOWN IN FIGURE 3.

CONCLUSIONS AND FUTURE WORK

Augmenting Web-CAT with visualizations can help the instructor and the student gain insights into their work in a course. This was shown through the survey results and through the informal discussions with instructors. Even though we have improved Web-CAT by adding the visualizations, there are several areas where additional work is possible.

The line chart projection shown in Figure 4 could be improved so that it shows predicted submission scores for dates. This will help the student know what their projected score is on the assignment due date. This visualization has not yet been implemented, but its value to students makes it a prime future target.

The line chart summarizing class performance in Figure 7 could be improved so that it splits the range of student scores up into quartiles and plots the quartiles in bands as a stacked area chart. This visualization, along with the one in Figure 6, remain to be implemented for instructors. Another possible instructor-oriented visualization might show if there is any correlation between a student's score on an assignment and the number of submissions the student made for that assignment. This might be shown through a histogram or a scatter plot. Alternatively, a variation on Figure 6 might show how many unique users have made submissions over a period of time.

Overall, we have found that graphically depicting scoring trends for individual students or an entire class can be a powerful way to give both students and instructors a feel for how they are making progress on a programming assignment. This can significantly reduce student frustration by giving them an accurate idea of how they are performing with respect to the remainder of the class.

REFERENCES

[1] Reek, K.A. A software infrastructure to support introductory computer science courses. In *Proc. 27th SIGCSE Tech. Symp. Computer Science Education*, ACM, 1996, pp. 125 – 129.

[2] Edwards, S.H. Improving student performance by evaluating how well students test their own programs. *J. Educational Resources in Computing*, 3(3):1-24, Sept. 2003.

[3] Edwards, S.H. Using software testing to move students from trial-and-error to reflection-in-action. In *Proc. 35th SIGCSE Tech. Symp. Computer Science Education*, ACM, 2004, pp. 26-30.

[4] Shneiderman, B. The eyes have it: a task by data type taxonomy for information visualizations. In *Proc. IEEE Symp. Visual Languages*, IEEE, 1996, pp. 336-343.

[5] Naps, T., *et al.* Evaluating the educational impact of visualization. In *Proc. 4th Annual SIGCSE/SIGCUE ITiCSE Conf. Innovation and Technology in Computer Science Education*, ACM, 1999, pp. 143-146.

[6] Mazza, R. and Dimitrova, V. Visualising student tracking data to support instructors in web-based distance education. In *Proc. 13th Int'l World Wide Web Conf. Alternate Track Papers & Posters* (New York, NY, USA, May 19 - 21, 2004), ACM, 2004, pp. 154-161.