Learning by Reviewing Paper-Based Programming Assessments

Yancy Vance Paredes[1(✉)], David Azcona[2], I-Han Hsiao[1], and Alan Smeaton[2]

[1] Arizona State University, Tempe, AZ 85281, USA
{yvmparedes,Sharon.Hsiao}@asu.edu
[2] Dublin City University, Glasnevin, Dublin 9, Ireland
{David.Azcona,Alan.Smeaton}@insight-centre.org

Abstract. This paper presents a retrospective analysis of students' use of self-regulated learning strategies while using an educational technology that connects physical and digital learning spaces. A classroom study was carried out in a Data Structures & Algorithms course offered by the School of Computer Science. Students' reviewing behaviors were logged and the associated learning impacts were analyzed by monitoring their progress throughout the course. The study confirmed that students who had an improvement in their performance spent more time and effort reviewing formal assessments, particularly their mistakes. These students also demonstrated consistency in their reviewing behavior throughout the semester. In contrast, students who fell behind in class ineffectively reviewed their graded assessments by focusing mostly on what they already knew instead of their knowledge misconceptions.

Keywords: Programming learning · Reviewing behavior
Educational technology · Educational data mining
Behavioral analytics

1 Introduction

Successful learners monitor their own memory, comprehension, and performance to evaluate their progress. They use this information to adapt their current strategies and behavior [1]. Aside from motivation, metacognition, and resource management strategy, being able to monitor one's progress and understanding is critical to succeed in problem solving in programming learning [2–5]. Unfortunately, novices and experts employ different such self-regulated learning (SRL) strategies [6].

This raises several research questions that are worth investigating as research on SRL in programming learning is still limited. However, due to the complex nature of programming problem solving, research in this discipline involves using qualitative methods, such as questionnaires, think-aloud protocols, and interviews. These are used to code student's behaviors according to corresponding

V. Pammer-Schindler et al. (Eds.): EC-TEL 2018, LNCS 11082, pp. 510–523, 2018.
https://doi.org/10.1007/978-3-319-98572-5_39

SRL motivation and strategies. We have begun to see more empirical and qualitative mixed method studies reporting SRL during programming problem solving. For instance, students solve code rearrangement Parson problems using sub-goal labelling [7]; the iterative programming process framework supports SRL activities [3]; adequate prior knowledge affects the searching and evaluating processes in programming problem solving [8].

To address such limitations, researchers have started developing technologies that focus on integrating and modelling physical learning activities while making use of advanced learning analytics. Clickers [9] and multi-touch tabletops [10] are some of the examples. In our case, we developed a system that captures and connects multimodal learning analytics from both the physical and the digital worlds in the programming learning domain. It has the capability of digitizing paper-based artifacts, such as paper assessments, and providing an interface for grading and delivery of feedback to classes with large number of students. It logs how students interact with it (timing, frequency, sequence, attention, and changes of patterns when performing the reflecting actions towards their learning). Most importantly, the system supports students in managing their learning by integrating *assessment content*, *feedback*, and *learning outcome* in classes with blended instruction. Our goal is to systematically track students' learning activities across the physical and the digital spaces. In this paper, we focused on the monitoring and reflecting SRL behaviors.
The rest of the paper aims to answer the following research questions:

RQ1: What are the behavioral differences between high-achieving and low-achieving students in terms of monitoring and reviewing? Do high-achieving students review more thoroughly or frequently?
RQ2: What is the magnitude of difference in reviewing behavior of students when grouped according to their performance trajectories? Do *improving students* review differently from others?
RQ3: Which reviewing behaviors are more effective towards learning?

The paper is organized as follow. First, we discuss the role of feedback and behavioral analytics in programming learning. Next, we provide an overview of our research platform and the data gathering approach. Finally, we present the evaluation results along with its educational implications.

2 Literature Review

2.1 Feedback in Programming Learning

Feedback has been considered one of the most influential factors that affect educational achievement [11]. In Science, Technology, Engineering and Mathematics (STEM) subjects, such as programming, physics, or math, automated grading of assessment is one of the most popular methods in providing feedback. Such method is particularly pertinent for large classes as it guarantees a short turnaround time. Systems like WEB-CAT [12] or ASSYST [13] apply pattern-matching techniques that verify students' answers by running a set of unit test

cases and comparing them with the correct answers. Unfortunately, in the programming learning domain, these platforms typically check the concrete aspects of the solutions. The logic and reasoning of students are often neglected. As a result, instructors had to manually examine the program quality. Several alternative approaches have been proposed to address the issue of providing semantic and constructive feedback as well as the issue of scaling of the generation of feedback. For example: crowdsourcing code solutions which will then be suggested to students [14]; using parameterized exercises to create a sizable collection of questions to facilitate automatic programming evaluation [15]; PeerGrader [8] and PeerWise [16] utilizing student cohorts to provide peer feedback.

Regardless of the feedback generation methods, all of the above mentioned systems and approaches focused on evaluating digital artifacts, less is discussed in assessing paper-based programming problems. There has been a few relevant early innovations addressing the problem by digitizing exams (e.g. GradeScope [17]). Digitization essentially provides several advantages (e.g. some default feedback can be kept on the digital pages with the predefined rubrics; students' identity can be kept anonymous which eliminates any of the grader's biases, etc.) As our system has the ability to capture how students attend to their graded assessments, we explored these reviewing behaviors to understand their impacts on learning.

2.2 Behavioral Analytics in Programming Learning

Modelling student's programming learning is not a new topic. Student models reside in intelligent tutors or any adaptive educational systems. Student's learning is typically estimated based on their behavior logs, such as the interactions with tutors resulting in the updates on the knowledge components. In modelling programming language learning, several parameters are used to estimate students' coding knowledge. For instance, learning can be gauged based on the sequence of programming problem solving success [18], programming assignments progression [19], dialogic strategies [20], programming information seeking strategies [21], assignment submission compilation behavior [22], troubleshooting & testing behaviors [23], code snapshot process state [24], and generic Error Quotient measures [24]. Additionally, Educational Data Mining (EDM) techniques have helped educational researchers to analyze snapshots of learning processes, such as a combination of automated and semi-automated real-time coding to identify meaningful meta-cognitive planning processes in an online virtual lab environment [25]; supervised and unsupervised classification on log files and eye-tracking data to find meaningful events in an exploratory learning environment [26]; the sequences of reviewing and reflecting behaviors Hidden Markov Models (HMM) to predict students' learning performances [27]. In learning analytics literature, Blikstein [28] proposed an automatic analytic tool to access student's learning in an open-ended environment. This considers a range of behavioral analytics to predict learning, such as the amount of code changing, compilation behavior, and code editing.

3 Research Methodology

3.1 Research Platform and Data Collection

WebPGA[1] was developed to serve as a platform that connects the physical and the digital learning spaces in programming learning. This system enables the digitization, grading, and distribution of paper-based assessments. Further details regarding the rationale and the design of the platform can be found in [27]. All events (which mostly are students' clickstream) are logged along with their timestamp. Examples of which include: logging in and out, clicking on a question to review, bookmarking a question, navigating through an exam, and taking of notes.

The data were collected from a classroom study conducted in a Data Structure and Algorithms course offered during the Fall 2016 semester. This class had a total of 3 exams and 13 quizzes. Among the 13 quizzes, only 6 were graded while the remaining 7 were recorded only for attendance (full credit was given regardless of the answers). There were 283 students enrolled in the class but only 246 (86.93%) were included in the study as those who dropped the course in the middle of the semester, did not take the three exams, or did not use the reviewing platform at all had to be removed. In this study, we analyzed *review actions* performed by students. A review action is an event where a student examines his or her graded answer. It includes reading the question, the answer, the assigned score and the feedback provided by the grader (see Fig. 1).

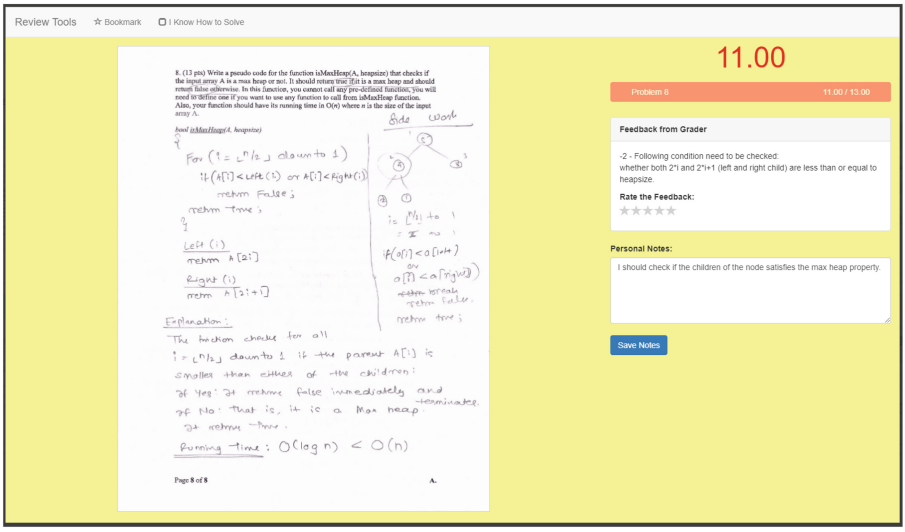


Fig. 1. Screenshot of what the student sees when reviewing his or her graded answer

[1] https://cidsewpga.fulton.asu.edu/.

3.2 Data Processing

In order to understand how students' monitoring and reviewing behavior affect their learning, students were labeled and grouped in two different ways. First, they were labeled according to their overall academic performance. Second, they were labeled according to their performance trajectory in a given period.

Overall Academic Performance. The average of the three exams was used to determine the overall academic performance of a student. Students were divided into two groups: *high-achieving* and *low-achieving*. Figure 2 shows the grades' distribution. Jenks natural breaks classification method [29] was used to identify the optimal break-point (77.60%) to divide the two groups.

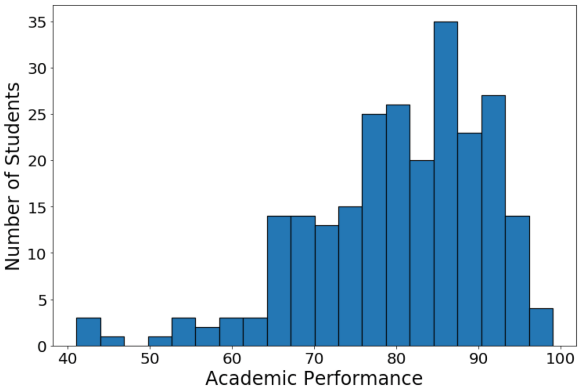


Fig. 2. Distribution of academic performance of students

Performance Trajectory. The exams served as milestones to identify the change in the performance of the students in a given period. There were two time periods in this analysis, namely: *Exam1-Exam2*, between the first and the second exam, and *Exam2-Exam3*, between the second and the third. The difference in the scores between the second and the first exam in a given period is computed. Students are labeled *improving* if the difference is positive; *dropping* if negative; *retaining* if zero.

Reviewing Behavior. A total of N = 17,518 review actions were extracted from the logs for this analysis. The score of the student in a particular question determines the label of a review action. The review action is labeled as *r_correct* if the student got the question right. Otherwise, it is labeled as *r_incorrect*.

3.3 Descriptive Data

An exam is considered reviewed if at least one of its questions is reviewed. Table 1 shows an overview of how students reviewed their exams. This includes the average performance of the class, the number of students who reviewed them, and the average time it took students before their first review attempt (hereinafter referred to as "reviewing delay"). A downward trend can be seen for both the number of students reviewing and their reviewing delay.

Table 1. Overview of students' reviewing behavior

Exam	Avg. score	No. of students who reviewed the exam	Avg. time before first review attempt	Standard deviation
Exam1	81.2%	230 (93.50%)	4.5 days	14 days
Exam2	78.7%	224 (91.06%)	2 days	6 days
Exam3	80.6%	196 (79.67%)	0.8 days	2.4 days

In terms of exam reviewing behaviors, most students reviewed past exams before taking the next one. During the *Exam1-Exam2* time period, there were 217 students (88.21%) who reviewed Exam 1 prior to taking Exam 2. During the *Exam2-Exam3*, it was also 217 students (88.21%) who reviewed Exam 1 or Exam 2 (or both) prior to taking Exam 3. However, these may not necessarily be the same set of students.

4 Evaluation Results

4.1 Association Between Reviewing Behavior and Learning Performance

To identify the impact of reviewing exams on the learning performance of a student, we compared the effort exerted by high-achieving students and low-achieving students. To answer this question, the contents of the first two exams were manually inspected. Based on the number of questions in the two exams, a student only needs to perform at least 16 review actions to cover all the items. Following the Pigeonhole principle, performing more than 16 review actions could indicate that a question is reviewed more than once. Furthermore, performing less than 16 review actions could indicate that not all questions were reviewed. Review actions for the third exam were omitted since the impacts of these actions cannot be captured and measured anymore (the class has ended).

Table 2 summarizes the average number of review actions performed by the two groups. Using t-test, we found that high-achieving students significantly ($t = -2.16, p = 0.03$) did more reviews than low-achieving students. It is interesting to note that, on average, high-achieving students performed 20.3 review actions. It could indicate that they reviewed their exams after it was made available and possibly prior to taking the next exam. This reflects their effort

in studying the material. On the other hand, low-achieving students, on average, performed 15.4 review actions. This shows how they barely reviewed their exams. This is clearly a bad habit since students are not able to take advantage of learning from the feedback they were provided, which could help them correct any of their misconceptions.

Table 2. Average review actions prior to Exam 3

Group	No. of students	Avg. review actions on exams	Standard deviation
High-achieving	158	20.3	19.1
Low-achieving	88	15.4	12.2

Doing more review does not necessarily translate to an effective one. Students may be doing a lot of review but not on the items where they really need to focus—their mistakes. Unfortunately, with the current grouping of students, it would not be surprising to find that majority of the review actions done by high-achieving students would be *r_correct* (answers they got correctly). This is because it is dependent on their academic performance. Therefore, a different grouping was used to answer this question.

4.2 Effectiveness of Reviewing Behavior

To address the issue mentioned above, students were grouped according to their performance trajectory in a given period. Table 3 summarizes the average number of review actions done by the improving and dropping students. The retaining group was omitted since it only has few students. During the Exam1-Exam2 period, there was no significant difference on the number of review actions performed by the two groups. Interestingly, during the Exam2-Exam3 period, dropping students performed significantly more review actions. This possibly happened because during the Exam1-Exam2 period, students only had one exam to review, while during the Exam2-Exam3 they had two. This led us to investigate why there was a drop in the grades of those who reviewed more.

Table 3. Review count of improving and dropping students

Group	Exam1-Exam2			Exam2-Exam3		
	n	Mean	SD	n	Mean	SD
Improving	104	9.57	8.88	115	10.23	11.52
Dropping	109	8.93	8.45	100	11.89	12.54

Table 4. Reviewing behavior of improving and dropping students

Review action	Group	Exam1-Exam2		Exam2-Exam3	
		Mean	SD	Mean	SD
R_Correct	Improving	0.16	0.18	0.22	0.25
	Dropping	0.22	0.24	0.36	0.28
R_Incorrect	Improving	0.84	0.18	0.78	0.25
	Dropping	0.78	0.24	0.64	0.28

Improving Group Reviewed Strategically and Effectively. An improving student may not necessarily be a high-achieving student. It is interesting to investigate what led to the improvement of their exam scores. Table 4 summarizes the reviewing behavior of both the improving and dropping students. Although not statistically significant, improving students during the Exam1-Exam2 period reviewed their mistakes more than the dropping students ($t = -1.82, p = 0.07$). During the Exam2-Exam3 period, a similar trend can be seen, but now statistically significant ($t = -3.69, p < 0.05$). This shows that this strategy, where you focus on your mistakes to get them right, helps in improving your grade.

Dropping Group Reviewed Ineffectively. During the Exam1-Exam2 period, dropping students reviewed their correct answers more than the improving students, though not statistically significant ($t = -1.82, p = 0.07$). However, during the Exam2-Exam3 period, the same trend was seen and is statistically significant ($t = -3.69, p < 0.05$). Although dropping students devoted more time in reviewing their mistakes, the effort they spent was not enough. There was no improvement in their grades. It is also possible that they may have overlooked their mistakes. Since this effect was found in both time periods, this demonstrates the persistent ineffectiveness in reviewing of the dropping students. This is concerning especially for students who are struggling or experiencing difficulties in class. Intervention strategies should be developed and applied.

4.3 Reviewing Behavior Efficiency

The reviewing delay of students was modeled as a function of their review efficiency and their effort in learning the material. The average reviewing delay for each student (the average of all the delays for each assessment the student reviewed) was computed. Afterwards, it was correlated to their academic performance. It was found that there is a significant negative linear correlation that exists (Pearson's), $r = -0.24, p < 0.05$. This means that better performing students tend to attend and review their graded answers sooner.

The trend on how students attended to their assessments throughout the semester was visualized (see Fig. 3). Students were grouped according to their academic performance. The groups' average reviewing delay was computed. The first three quizzes were omitted since the logging feature was only introduced after the third quiz. It can be seen that high-achieving students generally spent less time before they begin to review their assessments (notice that the green

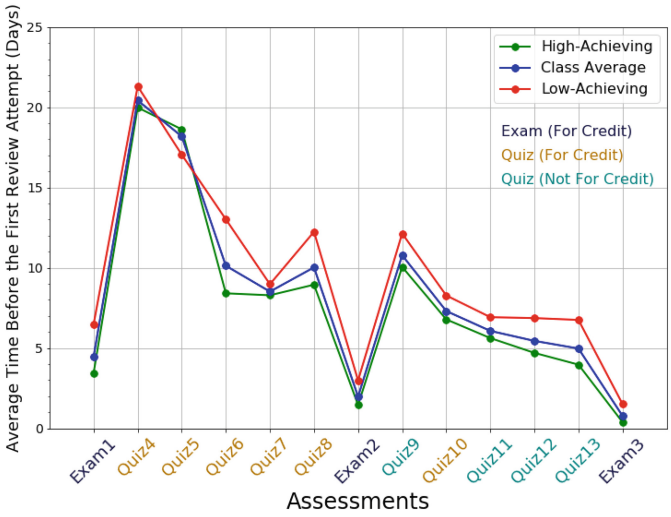
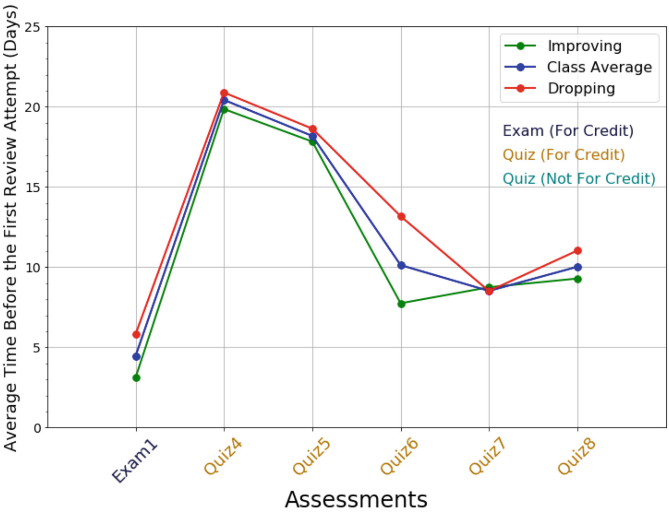


Fig. 3. The reviewing delay curve of students when grouped according to their overall academic performance (Color figure online)

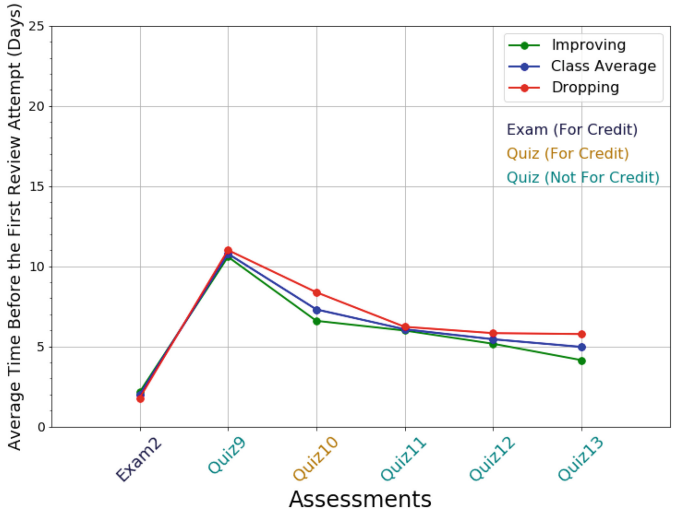
line is generally the lowest line throughout the semester). All students were more attentive in reviewing the three exams (shown by the dips) than the quizzes. This is not surprising. This suggests that the higher the credits that are at stake, the more attentive students become. One possible reason why students took longer time before they reviewed between the fourth to the sixth quizzes is that they did not review it right after it was made available. Students may have only reviewed them prior to taking Exam2. The chart also shows that students learned to use the platform over time as indicated by the downward trend. Interestingly, students started to review assessments sooner, even when the quiz was not for credit. This is an encouraging note and an evidence how students self-regulated their own learning in reviewing assessments.

The same steps were undertaken to investigate if similar findings could be obtained if students are grouped according to their performance trajectory. Unfortunately, there was no significant correlation between their magnitude of change (difference in their exam scores) and their average reviewing delay for both time periods.

Lastly, the trend for the two periods: Exam1-Exam2 (Fig. 4(a)) and Exam2-Exam3 (Fig. 4(b)) were visualized. It was not surprising to see that the improving group attended to their assessments sooner (notice that the green line is generally lower than the red line). This is consistent with the earlier finding which indicates that better performing students review sooner. This showed that being more vigilant in reviewing could potentially be associated to an improvement in grades. Another interpretation is that students who get better grades started seriously preparing for the exam earlier. Students likely reviewed their past exams at the start of preparing for the exam, so the fact that high-performing students did that earlier is not surprising.



(a) During the Exam1-Exam2 Period



(b) During the Exam2-Exam3 Period

Fig. 4. The reviewing delay curve of students when grouped according to their performance trajectory (Color figure online)

4.4 Subjective Evaluation

An online survey was administered at the end of the semester to know the experience of students when using the system. Also, to identify possible features that could help them review effectively and efficiently. Only 74 students (30.10%)

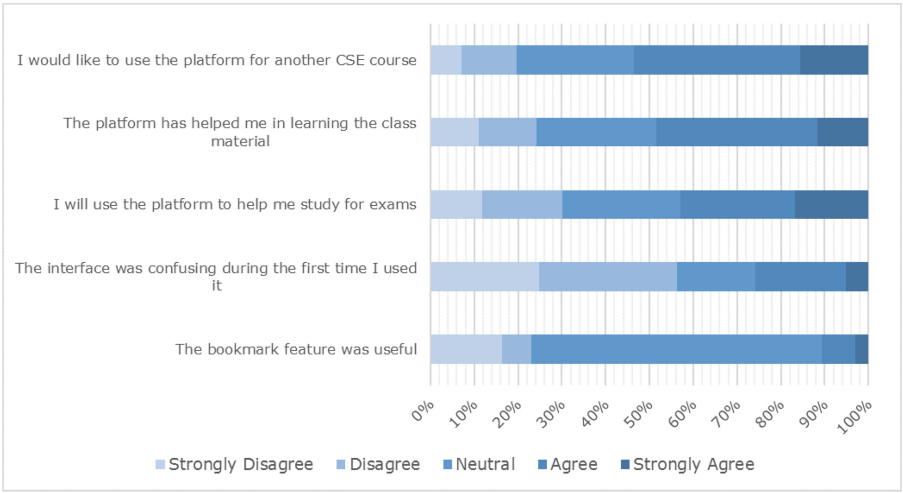


Fig. 5. Selected questions from the online survey

responded to the survey. Figure 5 shows some of the questions and the students' responses.

Learning and the Reviewing Platform. More than half of the respondents (54.1%) believed that the system helped them learn the class material. When asked how they prepare for programming exams, they would review lecture notes (78.2%) or past assignments and assessments (68%). Some even create study guides (45.7%). We also found that 60% would even use our system. All these strategies involve a range of reviewing activities.

Ease of Using the Platform. The system enables the students to access and review their graded assessments anytime and anywhere. Majority (60.3%) of the respondents found the system easy to navigate and use as it only took them around 1–2 quizzes to be comfortable using it.

Awareness of Features. A color coding scheme was used to display the graded answers of the students, which majority of the respondents were aware of. However, some features, such as bookmarking and filtering were not used as they were not aware of the existence of such features. Finally, we asked for suggestions on how to improve user experience. One popular suggestion was for the system to be able to inform students what content or question to focus on when reviewing (51.2%).

5 Conclusion

This study focused on analyzing and understanding student reviewing and learning behaviors captured by an educational tool that enables students to review their paper-based assessments. A classroom study was conducted where data from a Data Structure & Algorithms class were collected. Students were grouped based on their overall performance: high-achieving and low-achieving; and based on their performance in a given time period: improving, retaining, and dropping. By comparing their reviewing behaviors, high-achievers were found to review more and quicker than low-achievers. Both improving and dropping students reviewed their mistakes. However, improving students reviewed and focused on their mistakes more than dropping students. This clearly indicates the effectiveness and the willingness of improving students to learn more from their mistakes. It also indicates a failure on the part of the dropping students to pay enough attention to address their misconceptions.

In addition, this study provides empirical data on how students review their paper-based assessments. This contribution could be used to improve the design of existing educational technologies. Letting the students focus on their mistakes (guided navigation) and advising them to attend to their graded assessments sooner (through prompts) would have a positive impact on their learning.

Finally, reviewing patterns can be extracted from the student behavioral actions and leveraged to train predictive models. These models will enable the further personalization of the feedback and potential interventions that will be provided to future students such as suggested reviewing assessments and material.

6 Future Work and Limitations

There are a number of limitations in the current study. The analysis only focused on students' voluntarily reviewing behavior to signify one of the self-regulated learning processes: in the abstract form of monitoring and reviewing their own learning. In the future, a more comprehensive scenario such as planning, comprehension monitoring, and self-explaining will be considered. In addition, the platform was informally introduced to students and no tutorial on how to use it was provided. They had to familiarize it on their own. The usability of the platform is currently being studied. Finally, this study will be further extended to other courses and cohorts to investigate the generalizability of these findings in Computer Science Education.

References

1. Butler, D.L., Winne, P.H.: Feedback and self-regulated learning: a theoretical synthesis. Rev. Educ. Res. **65**(3), 245–281 (1995)
2. Bergin, S., Reilly, R.: The influence of motivation and comfort-level on learning to program. In: Proceedings of the 17th Workshop of the Psychology of Programming Interest Group, Sussex, UK, Psychology of Programming Interest Group, pp. 293–304 (2005)
3. Loksa, D., Ko, A.J.: The role of self-regulation in programming problem solving process and success. In: ICER, pp. 83–91. ACM, New York (2016)
4. Eteläpelto, A.: Metacognition and the expertise of computer program comprehension. Scand. J. Educ. Res. **37**(3), 243–254 (1993)
5. Hsiao, I.H., Bakalov, F., Brusilovsky, P., König-Ries, B.: Progressor: social navigation support through open social student modeling. New Rev. Hypermedia Multimed. **19**(2), 112–131 (2013)
6. Falkner, K., Vivian, R., Falkner, N.J.: Identifying computer science self-regulated learning strategies. In: Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education, pp. 291–296. ACM, New York (2014)
7. Morrison, B.B., Decker, A., Margulieux, L.E.: Learning loops: a replication study illuminates impact of hs courses. In: Proceedings of the 2016 ACM Conference on International Computing Education Research, pp. 221–230. ACM, New York (2016)
8. Gehringer, E.F.: Electronic peer review and peer grading in computer-science courses. ACM SIGCSE Bull. **33**(1), 139–143 (2001)
9. Trees, A.R., Jackson, M.H.: The learning environment in clicker classrooms: student processes of learning and involvement in large university-level courses using student response systems. Learn. Media Technol. **32**(1), 21–40 (2007)
10. Martinez-Maldonado, R., Dimitriadis, Y., Martinez-Monés, A., Kay, J., Yacef, K.: Capturing and analyzing verbal and physical collaborative learning interactions at an enriched interactive tabletop. Int. J. Comput. Support. Collab. Learn. **8**(4), 455–485 (2013)
11. Hattie, J., Timperley, H.: The power of feedback. Rev. Educ. Res. **77**(1), 81–112 (2007)
12. Edwards, S.H., Perez-Quinones, M.A.: Web-cat: automatically grading programming assignments. In: ACM SIGCSE Bulletin, vol. 40, pp. 328–328. ACM, New York (2008)
13. Jackson, D., Usher, M.: Grading student programs using assyst. In: ACM SIGCSE Bulletin, vol. 29, pp. 335–339. ACM, New York (1997)
14. Hartmann, B., MacDougall, D., Brandt, J., Klemmer, S.R.: What would other programmers do: suggesting solutions to error messages. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 1019–1028. ACM, New York (2010)
15. Hsiao, I.H., Sosnovsky, S., Brusilovsky, P.: Guiding students to the right questions: adaptive navigation support in an E-learning system for Java programming. J. Comput. Assist. Learn. **26**(4), 270–283 (2010)
16. Denny, P., Luxton-Reilly, A., Hamer, J.: Student use of the peerwise system. In: ACM SIGCSE Bulletin, vol. 40, pp. 73–77. ACM, New York (2008)
17. Singh, A., Karayev, S., Gutowski, K., Abbeel, P.: Gradescope: A fast, flexible, and fair system for scalable assessment of handwritten work. In: Proceedings of the Fourth ACM Conference on Learning@ Scale, pp. 81–88. ACM, New York (2017)

18. Guerra, J., Sahebi, S., Lin, Y.R., Brusilovsky, P.: The problem solving genome: analyzing sequential patterns of student work with parameterized exercises. In: Educational Data Mining, EDM, North Carolina (2014)
19. Piech, C., Sahami, M., Koller, D., Cooper, S., Blikstein, P.: Modeling how students learn to program. In: Proceedings of the 43rd ACM Technical Symposium on Computer Science Education, pp. 153–160. ACM, New York (2012)
20. Boyer, K.E., et al.: Investigating the relationship between dialogue structure and tutoring effectiveness: a hidden Markov modeling approach. Int. J. Artif. Intell. Educ. **21**(1–2), 65–81 (2011)
21. Lu, Y., Sharon, I., Hsiao, H.: Seeking programming-related information from large scaled discussion forums, help or harm? In: Proceedings of the 9th International Conference on Educational Data Mining, EDM, North Carolina, pp. 442–447 (2016)
22. Altadmri, A., Brown, N.C.: 37 million compilations: investigating novice programming mistakes in large-scale student data. In: Proceedings of the 46th ACM Technical Symposium on Computer Science Education, pp. 522–527. ACM, NY (2015)
23. Buffardi, K., Edwards, S.H.: Effective and ineffective software testing behaviors by novice programmers. In: Proceedings of the Ninth Annual International ACM Conference on International Computing Education Research, pp. 83–90. ACM, New York (2013)
24. Carter, A.S., Hundhausen, C.D., Adesope, O.: The normalized programming state model: predicting student performance in computing courses based on programming behavior. In: Proceedings of the Eleventh Annual International Conference on International Computing Education Research, pp. 141–150. ACM, New York (2015)
25. Montalvo, O., Baker, R.S., Sao Pedro, M.A., Nakama, A., Gobert, J.D.: Identifying students' inquiry planning using machine learning. In: Educational Data Mining, EDM, North Carolina (2010)
26. Bernardini, A., Conati, C.: Discovering and recognizing student interaction patterns in exploratory learning environments. In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010. LNCS, vol. 6094, pp. 125–134. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13388-6_17
27. Hsiao, I.H., Huang, P.K., Murphy, H.: Uncovering reviewing and reflecting behaviors from paper-based formal assessment. In: Proceedings of the Seventh International Learning Analytics & Knowledge Conference, pp. 319–328. ACM, New York (2017)
28. Blikstein, P.: Using learning analytics to assess students' behavior in open-ended programming tasks. In: Proceedings of the 1st International Conference on Learning Analytics and Knowledge, pp. 110–116. ACM, New York (2011)
29. Jenks, G.F.: The data model concept in statistical mapping. Int. Yearb. Cartogr. **7**, 186–190 (1967)