Results of an Online Algorithms Course with Mastery Grading and Optional Oral Examination

Matthew Ferland
Computer Science
University of Southern California
Los Angeles, CA 90007
mferland@usc.edu

Abstract

In the summer of 2023, a mastery grading scheme with weekly exams was implemented in an online upper-divsion introductory algorithms course with 123 enrolled students (117 after the drop deadline), with optional oral exams for additional attempt opportunities. Student reception to this system was overwhelmingly positive, with high student final grades, while still maintaining a high level of academic rigor.

1 Introduction

The traditional grading system has well documented flaws [5, 6], and many alternative systems have been developed to attempt to address them [2, 3, 8]. This experience report uses a system which addresses the issue of single-chance exams. That is to say, the system where, for each topic in the course, it is assessed on exactly one exam, and if a student performs poorly on that exam, there are no opportunities to make up the grade by demonstrating later mastery of the subject.

The most common reason for not implementing multiple-chance systems is due to grading overhead. In courses such as an introductory algorithms class, automatic grading is often infeasible, because designing algorithms and writing mathematical proofs are common forms of assessment [7]. As such, manually grading exams across all topics in the course every week could be impracticable, particularly for those with strained course staff to student ratios.

Furthermore, creating new questions each week is also a challenge. While topics such as Dynamic Programming and NP-completeness are rich domains

to create problems from, topics such as Greedy and Divide and Conquer algorithms have notably smaller design spaces.

Despite these constraints, multiple-chance grading schemes have been explored in algorithms courses in the past. The earliest example we are aware of is [9], where the authors explored having the exam grade split by topic, and made the final exam a "second chance" for each one. [1] allowed reassessment on topics via a limited number of "tokens" to limit grading load, while [10] didn't have proctored exams, and instead had assessments entirely through take-home assignments, allowing students to resubmit any programming assignment freely, and each week resubmit 1 or 2 written assignments from a previous week. However, all of these differ from our work, since none of these systems had weekly proctored exams on every topic.

2 The Course in Context

The course explored in this experience report was taught at UCLA, a large and very selective R1, over the summer of 2023, where initially 123 students enrolled (6 of whom eventually dropped), plus an additional 5 students audited, and one student made up an incomplete (which involved taking the exams, so they are part of the data). There was one student who never submitted any assignment/assessment, and never attended lecture. This student will be discarded from the data from this point on. The course staff consisted of the instructor of record and 2 PhD student teaching assistants. The instructor of record was teaching the course for the first time at this institution. The PhD students were required to hold exactly 2 hours of office hours each week (and no more by union contract). Both graded 1/3 of the exams (with the instructor grading the final 3rd), and assisted the proctoring of the exams. Over the summer, UCLA allows visiting students from other universities to take classes for course credit at their home institution. While numbers were not provided, a sizable percentage of the students enrolled were visiting students, perhaps as high as one third of the student population.

The course was 9 weeks long and taught entirely online, with lectures taking place over a Zoom classroom. Lecture attendance was typically between 40 and 60 students. Around 50 students or so noted inability to attend lecture live due to difficulties with time zones, internships, or other factors. Exams were conducted through the LMS, with a lockdown browser requirement and a recorded live Zoom proctoring to attempt to crackdown on cheating. There were no reported incidents of cheating, though there was one suspected case (the student failed regardless).

The grade in the course had two components. The first component was comprised of summative assessments. This was referred to as the "baseline

grade." The second was comprised of formative assessments, and was referred to as "grade modifications."

The baseline grade was based on the pass/fail status of the 4 core topics of the course: dynamic programming, greedy algorithms, divide and conquer algorithms, and NP-completeness. To pass, students needed to solve a single exam problem (giving and algorithm and/or proof). If at the end of the course, for each of those 4 topics, they received a pass on the corresponding topic on at least one exam, then they would receive a "baseline A." If they passed three out of the four, then they would receive a "baseline C." Otherwise, they would receive a "baseline F" and have no way to pass the course.

The grade modifications were just another way to present the formative assessments. Every one of the four topics had a reading assignment (from a Zybooks textbook [4]) and a homework assignment (composed of 2 questions for the topic). Dynamic Programming also had a programming assignment. Each of these was graded on a pass/fail basis, and for each "fail," the letter grade was lowered by one step (e.g.: an A would become an A-, an A- would become a B+, and so on). The homework assignments had two chances to pass. Either the submission would be completely correct, or, so long as the submission was "in good faith," a "reflection" could be submitted to pass, where students would note what mistakes they made, why they made them, and what they would do differently in the future to not make the same mistake again.

Finally, there was one "extra credit" opportunity based on a bonus lecture about network flow. The week 9 exams and final exam each had a bonus question involving network flow. If students passed one of these questions, then their grade would be raised one step. Notably, this was the only way to have an A+ in the course, which UCLA allows (but does not effect GPA).

2.1 Exam details

There were two exam slots each week, held 8 hours apart. As previously mentioned, these were done via the LMS on a lockdown browser, while being proctored on Zoom. The reason for having two timeslots was to accommodate students abroad and with internships. Also, students were allowed to (and several did) attend both sessions if they were able to and wished to have additional attempt opportunities. Furthermore, all topics additionally appeared on a final exam in the final week, and the first 3 topics had 2 bonus opportunities attached to review sessions, while the final topic (NP-completeness) had 6 bonus opportunities attached to review sessions.

Students were also allowed attempt oral exams. The student could schedule a time with a member of the course staff, who would give a problem to the student via the Zoom whiteboard. If the student was able to solve it, then they would pass the topic. If not, then they would get feedback to improve.

In total, excluding oral exams and counting both morning and afternoon exams, there were 16 dynamic programming exam chances, 13 greedy chances, 8 divide and conquer chances, and 11 NP-completeness chances. All questions were pulled from a database of questions made by the instructor, a TA, or taken and modified from a question from another instructor (with permission).

3 Data

Grading Type	Pre-N	Pre-μ	Pre-σ	Post-N	Post-μ	Post-σ	Δ (Conf. Int.)	P-value	T-score	df	σ_M
Curved	62	4.60	3.23	34	4.88	2.88	0.29 ($\pm$1.32)	0.6678	0.4305	94	0.663
Weighted	**62**	**7.21**	**2.08**	**34**	**6.26**	**1.91**	**-0.94 ($\pm$0.86)**	**0.0311**	**2.1883**	**94**	**0.432**
Standards-Based	62	7.66	1.90	33	7.67	2.17	0.01 ($\pm$0.85)	0.9901	0.0125	93	0.431
Pass/Fail	62	6.23	2.36	32	6.53	2.60	0.31 ($\pm$1.05)	0.5670	0.5745	92	0.532
Partial Credit	**62**	**9.24**	**1.29**	**31**	**7.81**	**1.66**	**-1.44 ($\pm$0.62)**	**0.0001**	**4.5867**	**91**	**0.313**
One Chance	62	3.79	2.28	31	2.90	1.83	-0.89 ($\pm$0.93)	0.0634	1.8794	91	0.472
Multiple Chance	62	8.84	1.27	31	8.87	1.98	0.03 ($\pm$0.67)	0.9244	0.0952	91	0.339

Table 1: Data from student rating of grading systems, testing whether the change of rating is significant using an unpaired two-tailed t-test. "Pre" refers to the Pre-survey and "Post" refers to the post survey. N refers to the sample size, μ refers to the mean, and σ refers to the standard deviation. Δ (Conf. Int.) contains the change in mean and confidence interval with $\alpha = 0.05$. The bolded entries are the statistically significant ones. σ_M is the standard error.

The course had both a pre-course survey, given before the first day of class, and a post-course survey, given after the final exam. In both, students were given a description of each system, then asked to rate them from from 1-10. The data from this can be seen in Table 1.

In both the pre-survey and post-survey, students were asked whether they would prefer a multiple-chance system graded as pass/fail or a single-chance system graded with partial credit. In the pre-survey, 80.6% of respondents said they would prefer the multiple-chance system as opposed to 19.4% the single-chance system. The post-survey widened the gap to 87.9% vs 12.1%.

Question	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
This grading system made me enjoy the course more than I would under a traditional system	1	1	0	12	20
This grading system made me less stressed about exams	2	2	3	9	18
The grading system helped me learn iteratively over time	0	1	3	5	25
I would take a course with a similar system to this again	2	0	0	5	27

Table 2: Course grading system questions

There were also class-specific questions. Opinions about the course grading system are shown in table 2. When asked about the helpfulness of the bonus sessions, 14 gave a 5/5, 4 a 4/5, 2 a 3/5, and 1 a 2/5. When asked about the most helpful part about them, 10 students indicated the exam chance, 4 the review session, and 8 students that both were equally helpful. The 8 respondents whom had taken oral exams were asked about the helpfulness of them. 3 students said 5/5, while 5 indicated 3/5. Finally, Table 3 shows the number of students that passed each topic each week. Only 6 of the passes were from oral exams (3 DP, 2 Greedy, and 1 Divide and Conquer).

Exam Time	Topic			
	DP	Greedy	D&C	NP
Week 3	56	0	0	0
Week 4	29	0	0	0
Week 5	4	46	0	0
Week 6	6	33	66	0
Week 7	14	18	31	0
Week 8	5	8	12	61
Bonus Exams	3	4	3	36
Week 9	4	4	5	8
Final	0	1	0	7
Never passed	0	3	0	5

Table 3: Number of students that passed each topic in each week. Some students that passed DP later dropped.

4 Discussion and Conclusion

4.1 Threats to Validity

The post-course survey had less respondents (34) than the pre-course survey (62). This could bias the results, since the population that completed the post-course survey may be different than those that completed the pre-course survey.

4.2 Student Suggestions

Student feedback about the system, received via survey, course evaluations, email, or private discussions, can broadly be broken up into the following themes:

1. Have it be easier to check what one's current course grade is, and provide better feedback on when one is "in danger"
2. Make the drop from a baseline A to a baseline C less harsh somehow
3. Introduce some sort of partial credit into the system, especially for the programming assignment

The first is simple enough and will be done in future instances of this system being used. The other two are are more difficult to address. One possible idea is using a "conditional pass" on sufficiently close answers, where the student must complete an alternative assignment to demonstrate full mastery.

4.3 Instructor Reflection

Even with a large course, the grading burden was bearable, The largest number of unpassed topics in any week of the course was during week 6, where there were around 220 unpassed topics. Furthermore, the actual number of attempts was lower. So, even in the worst week, making worst-case assumptions when splitting work equally among the course staff, at a grading time of 10 minutes a question, grading would take no more than 10 hours, with most weeks being far less. Thus, with a sufficiently large question bank, the work for the course is manageable even when teaching 2, or possibly even 3, courses during the academic term.

The system being pass/fail with multiple chances allowed the course to have a policy about "good faith" submissions. In essence, students were expected to be truthful and not submit an answer they knew was wrong. If the student didn't know, they were expected to report what they attempted, and what difficulties they encountered. This was beneficial when grading, as regularly students would report their answer to be wrong, and write what was confusing them or causing them difficulty, removing the time spent grading trying to figure out if an unorthodox solution is correct, and blithely reporting to a student why it doesn't work. Instead, the time could be spent simply looking at what students report difficulty with, and giving advice directly for those points.

Another thing to note is that the entire first class was spent discussing the philosophy of grading and why the grading system is what it is. Anecdotally, students responded very well to this, and it made the class excited to try the system and see how it worked for them. Discussing the motivations for the "unnattractive" lacking of partial credit seemingly lead to less students being upset about it, and indeed, the only statistically significant change in their view of grading systems was a less positive view of partial credit systems.

References

[1] Lijun Chen et al. "Experience report: Standards-based grading at scale in algorithms". In: *Proceedings of the 27th ACM Conference on on Innovation and Technology in Computer Science Education Vol. 1*. 2022, pp. 221–227.

[2] Andrew A Cooper. "Techniques grading: Mastery grading for proofs courses". In: *PRIMUS* 30.8-10 (2020), pp. 1071–1086.

[3] Joe Feldman. *Grading for equity: What it is, why it matters, and how it can transform schools and classrooms*. Corwin Press, 2023.

[4] Chelsea Gordon, Roman Lysecky, and Frank Vahid. "The shift from static college textbooks to customizable content: A case study at zyBooks". In: *2021 IEEE Frontiers in Education Conference (FIE)*. IEEE. 2021, pp. 1–7.

[5] Danielle L Iamarino. "The benefits of standards-based grading: A critical evaluation of modern grading practices". In: *Current Issues in Education* 17.2 (2014).

[6] Laura J Link and Thomas R Guskey. "How traditional grading contribute to student inequities and how to fix it". In: *Curriculum in Context* 45.1 (2019).

[7] Michael Luu et al. "What is an Algorithms Course? Survey Results of Introductory Undergraduate Algorithms Courses in the US". In: *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*. 2023, pp. 284–290.

[8] Linda B Nilson and Claudia J Stanny. *Specifications grading: Restoring rigor, motivating students, and saving faculty time*. Routledge, 2015.

[9] Michael Shindler et al. "Experience report: preemptive final exams for computer science theory classes". In: *Journal of Computing Sciences in Colleges* 35.10 (2020), pp. 9–14.

[10] Robbie Weber. "Using Alternative Grading in a Non-Major Algorithms Course". In: *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*. 2023, pp. 638–644.