Measuring the Score Advantage on Asynchronous Exams in an Undergraduate CS Course

Mariana Silva
mfsilva@illinois.edu
University of Illinois at Urbana-Champaign

Matthew West
mwest@illinois.edu
University of Illinois at Urbana-Champaign

Craig Zilles
zilles@illinois.edu
University of Illinois at Urbana-Champaign

ABSTRACT

This paper presents the results of a controlled crossover experiment designed to measure the score advantage that students have when taking exams asynchronously (i.e., the students can select a time to take the exam in a multi-day window) compared to synchronous exams (i.e., all students take the exam at the same time). The study was performed in an upper-division undergraduate computer science course with 321 students. Stratified sampling was used to randomly assign the students to two groups that alternated between the two treatments (synchronous versus asynchronous exams) across a series of four exams during the semester. These non-programming exams consisted of a mix of multiple choice, checkbox, and numeric input questions. For some questions, the parameters were randomized so that students received different versions of the question and some questions were identical for all students. In our results, students taking the exams asynchronously had scores that were on average only 3% higher (0.2 of a standard deviation). Furthermore, we found that the score advantage was decreased by the use of randomized questions, and it did not significantly differ based on the type of question. Thus, our results suggest that asynchronous exams can be a compelling alternative to synchronous exams.

KEYWORDS

asynchronous exams, computer-based testing, cheating, randomized questions

ACM Reference Format:
Mariana Silva, Matthew West, and Craig Zilles. 2020. Measuring the Score Advantage on Asynchronous Exams in an Undergraduate CS Course. In *The 51st ACM Technical Symposium on Computer Science Education (SIGCSE '20), March 11–14, 2020, Portland, OR, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3328778.3366859

1 INTRODUCTION

In the US and internationally, computer science departments are wrestling with course enrollment surges resulting from both increased numbers of majors and more courses taken by non-majors [5, 10, 13, 22]. Unsurprisingly, many CS faculty are employing automation (e.g., online auto-graded assignments) as one approach to managing this growth [3, 12, 14, 16, 24, 27]. Relevant to this work, computer-based exams have been proposed both as providing a more authentic testing environment for CS exams (by allowing compilers and debuggers to be used) and as a means to reduce grading effort (e.g., graders no longer need to compile hand-written code in their heads) [1, 2, 4, 6, 9, 15, 19, 21]. The total overhead of running exams for large classes can be further reduced by centralizing the running of exams to a computer-based testing facility [11, 17, 20, 23, 28], so that courses no longer need to proctor their exams or deal with student time conflicts and accommodations.

A key feature of these computer-based testing facilities [29], which enables them to efficiently deal with student time conflicts, is that exams are offered asynchronously, meaning that students are not all taking the exam at the same time. Instead, students are offered a multi-day window in which to take the exam, and they make a reservation for a time that fits in their schedule. Asynchronous exams not only completely eliminate the overhead of scheduling and separately proctoring conflict exams, but the flexibility is greatly appreciated by students who can schedule their exams around other academic, work, and family (e.g., non-traditional aged students with children) obligations.

The primary concern around asynchronous exams is what has previously been called *collaborative cheating* [7], where one student takes the exam early in the exam period and then tells another student (taking the exam later) what was on their exam. Previous work attempted to characterize the impact of collaborative cheating by *a posteriori* analysis of exams consisting of a mix of questions drawn from homework and hidden problems [8]. Students engaging in collaborative cheating were identified by looking for students that disproportionately re-studied exactly the subset of homework problems that were present on the exam. Such "cheaters" appeared to gain a 12 percentage point score advantage on problems drawn from the homework if every student received the same question, but their advantage dropped to 2 to 3 percentage points when the question was randomly drawn from a pool of questions. As discussed in Section 4.5, we observe a similar benefit from randomization.

In this work, we designed a controlled experiment (Section 3) to directly measure the score advantage students gain from an asynchronous exam. In particular, this paper finds the following:

(1) Across the four exams, students asynchronously taking the exam had a small score advantage, more specifically an average 3% score advantage, which was statistically significantly above zero.
(2) When we look at each exam individually, only one of the exams had a statistically significant asynchronous score advantage, and the score advantage appears to be correlated to exam construction.

SIGCSE '20, March 11–14, 2020, Portland, OR, USA

ACM ISBN 978-1-4503-6793-6/20/03…$15.00
https://doi.org/10.1145/3328778.3366859

(3) While there was no statistically significant score advantage difference based on question type (e.g., multiple choice, checkbox, numeric), questions that were not randomized contributed more to the score advantage to a statistically significant degree.

2 BACKGROUND

The focus of this paper is a series of exams that were conducted in a computer-based testing facility using the PrairieLearn LMS. As computer-based exams are not commonplace, we provide some background on both of the capabilities of the exam delivery tool and how the exams are managed.

2.1 The PrairieLearn LMS

PrairieLearn is an online problem posing system that permits the authoring of *item generators*, each of which is capable of generating a range of parameterized question (*item*) instances [25]. PrairieLearn permits a broad range of question types, including but not limited to numeric, graphical, symbolic, programming, and drawing problems. PrairieLearn can be used for both homework and exams.

Exams in PrairieLearn are constructed by specifying a series of *slots* on the exam. Each slot is associated with either a single problem or a pool of problems from which every student gets a random draw. Typically, faculty construct pools to have similar topic coverage and difficulty. When a student begins an exam, an exam is constructed for them by randomly drawing questions from pools and then randomly parameterizing those problems. Students grade their exams interactively during the exam and exam authors can permit students to have multiple attempts on problems. Each slot is assigned a point value and (optionally) a partial-credit schedule for multiple attempts.

2.2 Computer-based Testing Facility (CBTF)

Our computer-based testing facility [30] is a pair of computer labs that together have roughly 120 seats for students and another 5 seats in a reduced-distraction environment for students registered with the disability resource center. The networking and file system of the computers are strictly controlled and computers have privacy screens to prevents reading from neighboring computers. The facility is open and proctored 12 hours a day, 7 days a week to accommodate roughly four thousand exams per week. Proctors verify student identity, and students are randomly (by the scheduling software) assigned to a computer to deter coordinated cheating. Students are not permitted to take cell phones/smart watches, written notes, or other records into or out of the exam area.

Exams are generally run asynchronously, meaning that classes assign a three or four-day period for the students to take a midterm exam. Students use an online tool to make a reservation at any available time during the exam period. Students with accommodations permitting them additional time on exams are handled automatically by the scheduling software. Exam periods from different classes frequently overlap, and there are almost always several distinct exams running concurrently. The scheduling software attempts to seat students so that adjacent students are taking different exams.

3 METHODS

This study took place in a large public research university during the Spring 2019 semester. The data was drawn from a required upper-division undergraduate computer science (CS) course, where students were expected to have an introductory programming course as prerequisite. The course offered all of the homework assignments and exams via the PrairieLearn LMS, with a total of 9 exams held in the CBTF. With IRB approval, we obtained all the exam records for the 359 students registered in the course. The data used in this study consists of exam records from 321 students, since we removed the students with test accommodations or students that did not complete all the CBTF exams during their designated time. Of the students that participated in the study, 5% were sophomores, 42% were juniors and 53% were seniors.

Table 1: Exam schedule for groups X and Y. The groups alternated taking exams with two different treatments: synchronous (syn) and asynchronous (asyn)

Group	Exam 1	Exam 2	Exam 3	Exam 4
X	asyn	syn	asyn	syn
Y	syn	asyn	syn	asyn

Stratified sampling was used to randomly split the students into two groups (X and Y). All students took the same exams (subject to randomization as explained above), but the two groups had different exam schedules, alternating between the two treatments, i.e., the synchronous (syn) and asynchronous (asyn) exams, as illustrated in Table 1. During the semester, students had to take eight 50-minute exams and a 3-hour final exam. However, only four exams were used for this crossover study, namely the third, fifth, sixth and seventh exams, since they only included a mixture of multiple choice, checkbox and numeric input questions and had a relatively uniform score distribution in previous semesters. Other exams not included in the study also included a mixture of programming questions. The exams included in the study were administered in weeks 7, 11, 13 and 15 of the semester, which consisted of a total of 16 weeks.

Table 2: Major distribution in groups X and Y.

Group	CS	CS+X	Engineering	Others
X	69	35	32	24
Y	72	33	31	25

To create groups X and Y, we stratified on gender and majors. As a result, group X consisted of 73% male (160 students total) and group Y consisted of 74% male (161 students total). For the major stratification, we considered four different sub-groups: CS, CS + X, Engineering and Others. The final student major distribution in each group is shown in Table 2.

The course instructor announced during the first lecture that they would be performing a study to investigate the difference between synchronous and asynchronous tests, and explained how the

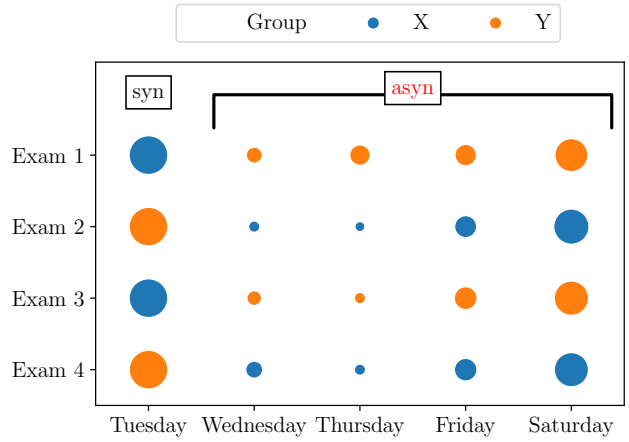

Figure 1: Exam schedule for groups X and Y. Students taking the exam synchronously were scheduled to take the exam on Tuesday during lecture time. Students taking the exam asynchronously had the option to take the exam from Wednesday to Saturday. The area of the circle indicates the number of students taking the exam on that given day.

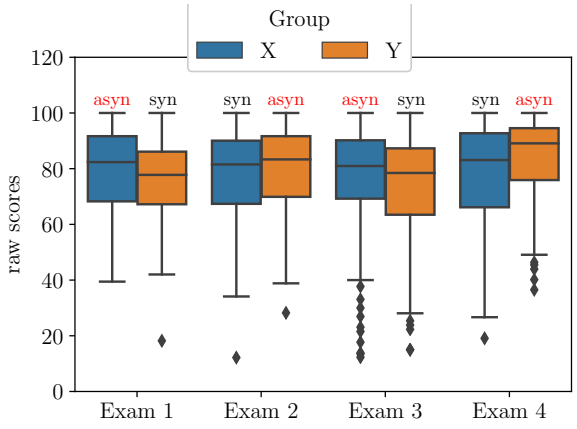

Figure 2: Box plot showing the distribution of the exam raw scores for groups X and Y. We use the labels 'syn' and 'asyn' to indicate the synchronous and asynchronous exam treatments.

exams would be offered following an alternate schedule. However, students were not aware of the parameters used to form the groups.

Students scheduled to take an exam asynchronously could sign up for an exam time of their choosing, with sign-ups starting two weeks prior to the start of the exam period, which ran from Wednesday to Saturday. On the other hand, students scheduled to take an exam synchronously were pre-scheduled by the CBTF administration to take the exam during lecture time on Tuesday and were not able to modify their registration. This ensured that all students taking the exam synchronously took the exam before any of the students taking it asynchronously.

Due to the course's large size and lecture hall availability, two lecture sections of the course were offered, scheduled in consecutive hours. Consequently, students in the group taking the exams synchronously were also scheduled at two different consecutive time slots based on their lecture time. Students taking the exam during the first synchronous time slot were not able to leave the CBTF room until the end of the exam time and exited through a different door than the students entering the room for the second synchronous time slot, eliminating the possibility of communication among students taking the exam synchronously. For consistency, students taking the exam during the second synchronous time slot were also held in the room until the end of the exam time.

Figure 1 shows an illustration of the exam schedule. The area of each circle indicates the number of students taking an exam on a given day. The number of students taking the exam on Tuesday is fixed and determined by the size of each group (around 160 students). For the group taking the exam during the asynchronous period, we note that a large percent of the students choose to take the exam on the last day of the exam period (Saturday), as has been previously observed [7, 26].

4 RESULTS

4.1 Overall exam scores

We first consider the overall score distribution for groups X and Y for all four exams. Figure 2 shows a box plot of the raw data for both groups and all four exams, indicating the exam treatment for each group. Table 3 summarizes the mean scores and standard deviations.

Table 3: Summary information for exams used in the study.

Exam	Group	Treatment	Mean	Std dev
Exam 1	X	asyn	78.31	15.23
Exam 1	Y	syn	76.05	14.38
Exam 2	X	syn	78.62	15.63
Exam 2	Y	asyn	80.18	14.72
Exam 3	X	asyn	75.79	20.44
Exam 3	Y	syn	73.52	18.85
Exam 4	X	syn	83.70	14.84
Exam 4	Y	asyn	77.59	18.73

Further analysis of the data show that the score distributions are not normal, but that they deviate from normal in a structured way. All exams have mean scores well above 50% and, as depicted in Fig. 3, they are also negative skewed with mostly positive excess kurtosis (that is, above the normal distribution value of 3), but with less-skewed exams having less (or even negative) excess kurtosis (we used the Pearson's moment coefficient of skewness, and the same Pearson's definition for the kurtosis). These features have been observed in exams both historically [18] and recently [7], and are consistent with exam scores being limited at 100%, which tends to produce a "piling up" of high scores.

To maintain simplicity of the analysis, we will utilize statistical tests and linear regression models that assume normality of the

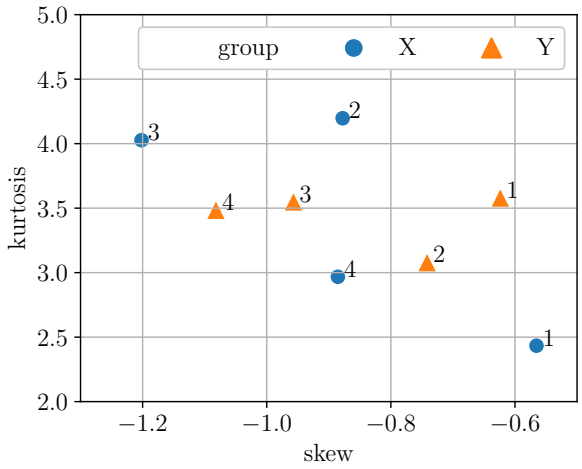

Figure 3: Kurtosis and skewness for the exam raw score distributions for groups X and Y. The exam scores distributions are not normal, in a way that is consistent with censoring (limiting of scores) at 0% and 100%.

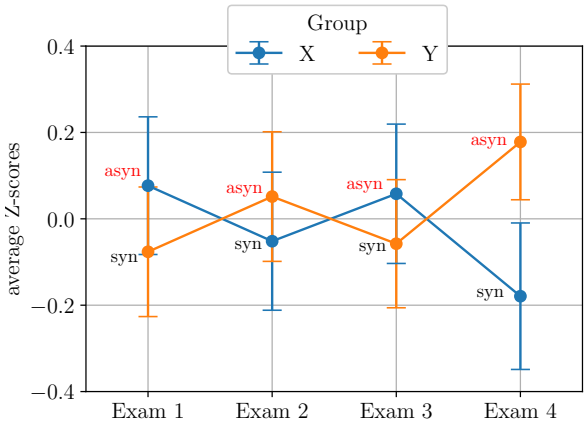

Figure 4: Mean z-scores for groups X and Y for each exam. The errors bars are the 95% confidence intervals (assuming normality). The exam treatment given to each group is indicated by the labels 'syn' and 'asyn'.

data. However, we will later use a bootstrap to verify our results and confirm that non-normality has not changed any of our conclusions.

Recall from the section Methods that each group (X and Y) were formed by two sub-groups of students since the course was split into two different lecture sections. While taking the synchronous exams, one of the sub-groups took the exam during the first lecture period, and the second sub-group took the exam immediately after at the second lecture period. We performed one-way ANOVA tests with the null hypothesis that the two sub-groups that formed group X and the two sub-groups that formed group Y have the same mean score. Table 4 summarizes the results and shows that none of the exams have a p-value that is statistically significant, supporting our decision to treat the two sub-groups as one group in the analyses that follow.

Table 4: The p-values from one-way ANOVA to test the null hypothesis that the two lecture sub-groups that form groups X and Y have the same mean scores.

Group	Exam 1	Exam 2	Exam 3	Exam 4
X	0.082	0.232	0.166	0.256
Y	0.311	0.407	0.495	0.375

4.2 Comparing scores from synchronous and asynchronous treatments

We standardized the scores from each exam, using all scores from both groups X and Y together. Figure 4 shows the mean of the z-scores for groups X and Y for each exam. We observe that the alternating values of the mean score for groups X and Y follow the same alternating pattern of the exam treatment: the group that takes the exam synchronously have negative mean scores, while the group that takes the exam asynchronously have positive mean scores.

Results from one-way ANOVA tests using the exam score distributions from groups X and Y indicate that Exam 4 is the only one in which the means are statistically different ($p = 0.00131$), confirming the results illustrated in Fig. 4.

4.3 Score advantage for students taking asynchronous exams

For each exam, we fitted an ordinary least squares (OLS) model of the form

$$z_{ij} = \sigma_j + \alpha_i + \beta A_{ij}, \tag{1}$$

where z_{ij} is the standardized z-score that student i received in exam j, A_{ij} is 1 if student i took the exam j in the asynchronous schedule, otherwise A_{ij} is 0, and β, α_i and σ_j are the parameters we want to estimate, which can be interpreted as:

- σ_j: the mean score of exam j,
- α_i: the ability of student i,
- β: the score advantage for students taking an exam asynchronously rather than synchronously.

Here we are mostly interested in the coefficient β which represents the effect size of the score advantage (in units of standard deviations) when students take an exam asynchronously. We find $\beta = 0.182$ (95% CI [0.107,0.257], $p < 0.0001$), meaning that a student exam score increases roughly by 0.18 of a standard deviation if the exam is taken asynchronously rather than synchronously.

We also performed linear regression to fit Eq.1 using the raw scores as the left-hand values. Table 5 summarizes the coefficients β and σ_i in this case. We note that the values of σ are close to the actual values of the exam averages, as expected (see Table 3). The β value indicates that students taking the exam asynchronously had on average a 3.05 percentage point score advantage, which is small

Table 5: Linear regression coefficients from Eq. 1 using raw exam scores, with corresponding 95% confidence intervals and p-values.

Coefficient	Value	p-value	95% CI
β	3.05	<0.0001	[1.792,4.316]
σ_1	79.3	—	[68, 90.7]
σ_2	81.6	—	[70.2, 92.9]
σ_3	76.8	—	[65.4, 88.2]
σ_4	82.8	—	[71.4, 94.2]

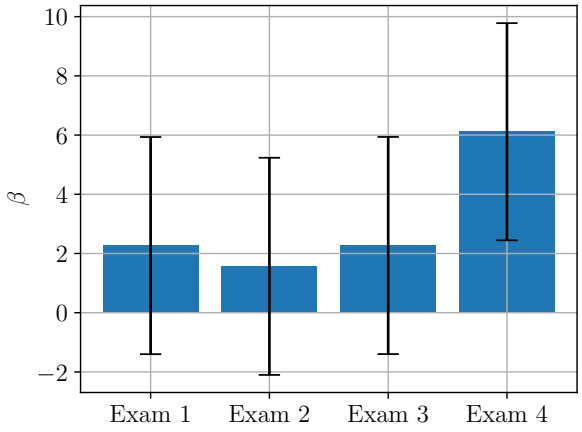

Figure 5: Linear regression coefficients from Eq. 2 using raw exam scores, with corresponding 95% confidence intervals.

relative to the range of exam score but is statistically significantly above zero.

The results above were obtained from linear regression models with standardized and non-standardized exam scores without any extra treatment for non-normality. To validate our results, we computed bootstrap estimates of β and its confidence interval. From 10,000 bootstrap samples, we obtained $\beta = 0.182$ (95% CI [0.103,0.261], $p < 0.0001$) for z-scores and $\beta = 3.05$ (95% CI [1.694, 4.405], $p < 0.0001$) for raw scores, which are almost identical to the OLS estimates above.

4.4 Score advantage in each exam

In this second analysis, we fit an OLS model of the form

$$s_{ij} = \sigma_j + \beta_j A_{ij}, \tag{2}$$

where we will now obtain one β_j value for each exam, indicating the score advantage that students get in exam j if that exam was taken asynchronously. For this analysis, we define s_{ij} as the raw scores, and σ_j and β_j are the parameters we want to estimate, which can be interpreted as:

- σ_j: the mean score of exam j,
- β_j: the score advantage students get when taking exam j asynchronously.

The resulting coefficients for σ_j are representative of the mean scores, as expected, with values of 76, 78.6, 73.5 and 77.6 respectively.

The results for the β coefficients are shown in Fig.5. We observe that the coefficients for exams 1, 2 and 3 are not significantly different from zero and are quite similar, while the coefficient for exam 4 is an outlier. In other words, only one of the exams had a statistically significant asynchronous score advantage, which is consistent with the results presented in Fig.4.

To better understand what made exam 4 an outlier, we analyzed the construction of the exams, by comparing the type of questions included in each one of them. Did exam 4 have characteristics that facilitated the spread of information (i.e., it was easier to cheat) and therefore gave an increased advantage to students that took the exam in the asynchronous schedule? We explore this question in the next section.

4.5 Score advantage per question type

The exams used in this study consisted of a mix of question formats: multiple choice (M), checkbox (C), and numeric input (N). In many of the questions the parameters were randomized so that students received different versions of the question, while some questions were not randomized and were thus identical for all students. Table 6 describes the type of questions included in each exam.

Table 6: Question types included in each exam. M: multiple choice, C: checkbox, N: numeric input, R: randomized parameters.

Question	Exam 1	Exam 2	Exam 3	Exam 4
1	N/R	N/R	N/R	C
2	N/R	M/R	N/R	N/R
3	M	N/R	C/R	M
4	N/R	M	N/R	N
5	N/R	M/R	N/R	N/R
6	C/R	N/R	N/R	M
7	M/R	M/R	M/R	N/R
8	N/R	N/R	N/R	N/R
9	M/R	C/R	N/R	N/R
10	N/R	N/R	C/R	N/R
11	N/R	N	N/R	C/R
12	N/R	N/R	N/R	-
13	-	-	N/R	-

In this last analysis, we considered the scores for each individual question included in the exams. We fit an OLS model of the form

$$q_{ik} = \sigma_k + \alpha_i + \Big(\delta_M M_k + \delta_C C_k + \delta_N N_k + \delta_R R_k\Big) A_{ik}, \tag{3}$$

where q_{ik} is the raw score that student i received on question k (all questions are worth 10 points.), A_{ik} is 1 if student i received question k during an asynchronous exam, otherwise A_{ik} is 0, and M_k, C_k, N_k, and R_k are all 0 or 1 depending on whether question k is multiple choice, checkbox, numeric, or randomized, respectively.

We want to estimate the coefficients δ_M, δ_C, δ_N, δ_R, α_i, and σ_k, which can be interpreted as:

- σ_k: the mean score of question k,
- α_i: ability of student i,

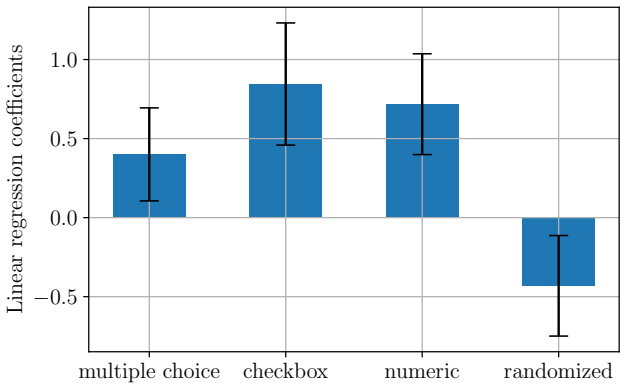

Figure 6: Linear regression parameters from Eq. 3, with corresponding 95% confidence intervals. Multiple choice: δ_M, checkbox: δ_C, numeric input: δ_N, randomized: δ_R.

- δ_M: the score advantage for multiple choice questions when the exam is taken asynchronously,
- δ_C: the score advantage for checkbox questions when the exam is taken asynchronously,
- δ_N: the score advantage for numeric questions when the exam is taken asynchronously,
- δ_R: the score advantage for randomized questions when the exam is taken asynchronously.

We will focus on δ_M, δ_C, δ_N, and δ_R, since we want to know how the question type and randomization affects the student score advantage when taking exams asynchronously. We plotted these coefficients with their corresponding 95% confidence intervals in Fig. 6. The units of these values is "question points".

Questions of different formats (multiple choice, checkbox, or numeric) are not statistically-significantly different in how much advantage is gained from asynchronous exams. However, randomized questions give statistically-significantly less advantage ($\delta_R = -0.431$, 95% CI [-0.749,-0.113], $p = 0.008$) than non-randomized questions on asynchronous exams. Speaking somewhat loosely, this shows that questions of different formats (multiple choice, checkbox, numeric) are all roughly equally easy to cheat on, while randomized questions are harder to cheat on. This is consistent with prior work [8] that found randomization of questions to significantly decrease the score advantage from collaborative cheating.

In Section 4.4, we observed that only exam 4 had a statistically significant asynchronous score advantage (β_j). Table 7 shows the percent of randomized questions in each exam, where we see that exam 4 has fewer randomized questions than the other exams, which might explain why it had a larger β_j.

Table 7: Fraction of questions on each exam that are of each type. The fractions do not sum to 100% because questions can have multiple types, as shown in Table 6.

Type	Exam 1	Exam 2	Exam 3	Exam 4
Multiple choice (M)	25%	34%	8%	18%
Checkbox (C)	8%	8%	15%	18%
Numeric (N)	67%	58%	77%	64%
Randomized (R)	92%	83%	100%	64%

5 DISCUSSION AND CONCLUSIONS

In this paper, we perform, to our knowledge, the first controlled experiment to measure the score advantage resulting from running exams asynchronously, where students are allowed to select their exam time. In aggregate, across the four exams, we find that the effect size is modest (0.182 of a standard deviation, equivalent to 3.05 percentage points). Furthermore, we find that the strongest correlation to an exam's score advantage is the number of non-randomizing questions that the exam includes. In this work, we did not investigate the underlying causes for this score advantage. Two plausible hypotheses are collaborative cheating and improved confidence/preparation resulting from the flexibility of deciding when to take an exam, but further study is needed.

The contribution of collaborative cheating to the score advantage is not surprising. We know anecdotally that students communicate about the exam, and trying to prevent that communication is probably a losing battle. Nevertheless, we find it reassuring that the impact of that communication appears to be modest, at least in this course, suggesting that asynchronous exams can be a reasonably-secure alternative to traditional synchronous exams.

The key mechanism to mitigate that score advantage seems to be randomization. From our data, we can extrapolate that had all of the problems been parameterized so that each student would get a different version of the question, the aggregate score advantage would drop from 3.05 percentage points to 2.39 percentage points. Furthermore, the exams studied had only minimal use of problem pools; in over 90% of the slots all students received the same question generator. Studies of the score advantage reduction resulting from using problem pools is important future research.

There are several limitations of our results that restrict the extent to which they can be generalized to other contexts. For example, we used data from one large, highly-selective, research university, with a student population that was majority male and US domestic, and the exam questions were short-answer and multiple-choice. It would be interesting to see similar studies at other universities to understand asynchronous exams in different environments.

We feel that our results have potential implications for other contexts in which exams are run asynchronously. Recently, an assortment of commercial online proctoring services have become available that similarly offer students the ability to take exams at a time of their choosing. While we hesitate to predict the magnitude of the score advantage resulting from the asynchronicity of online proctored exams, we are comfortable predicting that increased randomization of exams would reduce the score advantage in that context as well.

ACKNOWLEDGMENTS

We acknowledge David Mussulman and Carleen Sacris for their invaluable assistance in making the synchronous exam schedule possible at the CBTF.

REFERENCES

[1] Joao Paulo Barros, LuÃŋs Estevens, Rui Dias, Rui Pais, and Elisabete Soeiro. 2003. Using lab exams to ensure programming practice in an introductory programming course. In *Proceedings of the 8th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education, (ITiCSE)*. 16–20.

[2] Jens Bennedsen and Michael E. Caspersen. 2007. Assessing Process and Product: A Practical Lab Exam for an Introductory Programming Course. *Innovation in Teaching and Learning in Information and Computer Sciences* 6, 4 (2007), 183–202. https://doi.org/10.11120/ital.2007.06040183 arXiv:http://www.tandfonline.com/doi/pdf/10.11120/ital.2007.06040183

[3] Luciana Benotti, Federico Aloi, Franco Bulgarelli, and Marcos J. Gomez. 2018. The Effect of a Web-based Coding Tool with Automatic Feedback on Students' Performance and Perceptions. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education (SIGCSE '18)*. ACM, New York, NY, USA, 2–7. https://doi.org/10.1145/3159450.3159579

[4] Mary Elaine Califf and Mary Goodwin. 2002. Testing Skills and Knowledge: Introducing a Laboratory Exam in CS1. In *Proceedings of the 33rd SIGCSE Technical Symposium on Computer Science Education (SIGCSE '02)*. ACM, New York, NY, USA, 217–221. https://doi.org/10.1145/563340.563425

[5] Tracy Camp, W. Richards Adrion, Betsy Bizot, Susan Davidson, Mary Hall, Susanne Hambrusch, Ellen Walker, and Stuart Zweben. 2017. Generation CS: The Mixed News on Diversity and the Enrollment Surge. *ACM Inroads* 8, 3 (July 2017), 36–42. https://doi.org/10.1145/3103175

[6] Jacabo Carrasquel, Dennis R. Goldenson, and Philip L. Miller. 1985. Competency testing in introductory computer science: the mastery examination at Carnegie-Mellon University. In *SIGCSE '85*.

[7] Binglin Chen, Matthew West, and Craig Zilles. 2017. Do Performance Trends Suggest Wide-spread Collaborative Cheating on Asynchronous Exams?. In *Learning at Scale*.

[8] Binglin Chen, Matthew West, and Craig Zilles. 2018. How much randomization is needed to deter collaborative cheating on asynchronous exams?. In *Learning at Scale*.

[9] Yuting W. Chen. 2019. Work in Progress: A Balancing Act - Evolution of Assessments in An Introductory Programming Course in ECE After Curriculum Redesign. In *2019 ASEE Annual Conference & Exposition*. ASEE Conferences, Tampa, Florida. https://peer.asee.org/32380.

[10] Computing Research Association. 2017. Generation CS: Computer Science Undergraduate Enrollments Surge Since 2006. https://cra.org/data/Generation-CS.

[11] Ronald F. DeMara, Navid Khoshavi, Steven D. Pyle, John Edison, Richard Hartshorne, Baiyun Chen, and Michael Georgiopoulos. 2016. Redesigning Computer Engineering Gateway Courses Using a Novel Remediation Hierarchy. In *2016 ASEE Annual Conference & Exposition*. ASEE Conferences, New Orleans, Louisiana. https://peer.asee.org/26063.

[12] Margaret Ellis, Clifford A. Shaffer, and Stephen H. Edwards. 2019. Approaches for Coordinating eTextbooks, Online Programming Practice, Automated Grading, and More into One Course. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE '19)*. ACM, New York, NY, USA, 126–132. https://doi.org/10.1145/3287324.3287487

[13] Daniel T. Fokum, Daniel N. Coore, Eyton Ferguson, Gunjan Mansingh, and Carl Beckford. 2019. Student Performance in Computing Courses in the Face of Growing Enrollments. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE '19)*. ACM, New York, NY, USA, 43–48. https://doi.org/10.1145/3287324.3287354

[14] Georgiana Haldeman, Andrew Tjang, Monica Babeş-Vroman, Stephen Bartos, Jay Shah, Danielle Yucht, and Thu D. Nguyen. 2018. Providing Meaningful Feedback for Autograding of Programming Assignments. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education (SIGCSE '18)*. ACM, New York, NY, USA, 278–283. https://doi.org/10.1145/3159450.3159502

[15] Norman Jacobson. 2000. Using On-computer Exams to Ensure Beginning Students' Programming Competency. *SIGCSE Bull.* 32, 4 (Dec. 2000), 53–56. https://doi.org/10.1145/369295.369324

[16] David Joyner, Ryan Arrison, Mehnaz Ruksana, Evi Salguero, Zida Wang, Ben Wellington, and Kevin Yin. 2019. From Clusters to Content: Using Code Clustering for Course Improvement. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE '19)*. ACM, New York, NY, USA, 780–786. https://doi.org/10.1145/3287324.3287459

[17] Alan C. Bugbee Jr. and Frank M. Bernt. 1990. Testing by Computer: Findings in Six Years of Use 1982-1988. *Journal of Research on Computing in Education* 23, 1 (1990), 87–100. https://doi.org/10.1080/08886504.1990.10781945 arXiv:https://doi.org/10.1080/08886504.1990.10781945

[18] Frederic M. Lord. 1955. A Survey of Observed Test-Score Distributions With Respect to Skewness and Kurtosis. *Educational and Psychological Measurement* 15, 4 (1955), 383–389. https://doi.org/10.1177/001316445501500406

[19] Teemu Rajala, Erkki Kaila, Rolf Lindén, Einari Kurvinen, Erno Lokkila, Mikko-Jussi Laakso, and Tapio Salakoski. 2016. Automatically Assessed Electronic Exams in Programming Courses. In *Proceedings of the Australasian Computer Science Week Multiconference (ACSW '16)*. ACM, New York, NY, USA, Article 11, 8 pages. https://doi.org/10.1145/2843043.2843062

[20] Anni RytkÃűnen and Liisa Myyry. 2014. Student experiences on taking electronic exams at the University of Helsinki. In *World Conference on Educational Multimedia, Hypermedia and Telecommunications*. 2114–2121.

[21] Mika Saari and Timo MÃďkinen. 2016. Utilizing Electronic Exams in Programming Courses: A Case Study. In *EDULEARN16 Proceedings : 8th International Conference on Education and New Learning Technologies (EDULEARN proceedings)*. 7155–7160. https://doi.org/10.21125/edulearn.2016.0560

[22] Mehran Sahami and Chris Piech. 2016. As CS Enrollments Grow, Are We Attracting Weaker Students?. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education (SIGCSE '16)*. ACM, New York, NY, USA, 54–59. https://doi.org/10.1145/2839509.2844621

[23] Mordechai Shacham. 1998. Computer-based exams in undergraduate engineering courses. *Computer Applications in Engineering Education* 6, 3 (1998), 201–209.

[24] Leo C. Ureel II and Charles Wallace. 2019. Automated Critique of Early Programming Antipatterns. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE '19)*. ACM, New York, NY, USA, 738–744. https://doi.org/10.1145/3287324.3287463

[25] Matthew West, Geoffrey L. Herman, and Craig Zilles. 2015. PrairieLearn: Mastery-based Online Problem Solving with Adaptive Scoring and Recommendations Driven by Machine Learning. In *2015 ASEE Annual Conference & Exposition*. ASEE Conferences, Seattle, Washington.

[26] M. West and C. Zilles. 2016. Modeling student scheduling preferences in a computer-based testing facility. In *Third Annual ACM Conference on Learning at Scale*. 309–312. https://doi.org/10.1145/2876034.2893441

[27] Laura Zavala and Benito Mendoza. 2018. On the Use of Semantic-Based AIG to Automatically Generate Programming Exercises. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education (SIGCSE '18)*. ACM, New York, NY, USA, 14–19. https://doi.org/10.1145/3159450.3159608

[28] C. Zilles, R. T. Deloatch, J. Bailey, B. B. Khattar, W. Fagen, C. Heeren, D Mussulman, and M. West. 2015. Computerized Testing: A Vision and Initial Experiences. In *American Society for Engineering Education (ASEE) Annual Conference*.

[29] Craig Zilles, Matthew West, Geoffrey Herman, and Timothy Bretl. 2019. Every university should have a computer-based testing facility. In *Proceedings of the 11th International Conference on Computer Supported Education (CSEDU)*.

[30] Craig Zilles, Matthew West, David Mussulman, and Timothy Bretl. 2018. Making testing less trying: Lessons learned from operating a Computer-Based Testing Facility. In *2018 IEEE Frontiers in Education (FIE) Conference*. San Jose, California.