Engineering Educators Bringing the World Together
2025 ASEE Annual Conference & Exposition
Palais des congrès de Montréal, Montréal, QC • June 22–25, 2025
Paper ID #48619

Experiences with the GraySim CPU Scheduling Simulator

Lt. Sierra Zoe Bennett-Manke, United States Military Academy

Sierra Zoe Bennett-Manke is a recent graduate of the United States Military Academy at West Point, where she studied Computer Science and German. She will be serving as a Second Lieutenant in the U.S. Army at Fort Novosel in the Aviation branch where she hopes to fly Chinooks.

Dr. Maria R. Ebling, United States Military Academy

Maria R. Ebling is an assistant professor at the United States Military Academy at West Point in the Department of Electrical Engineering and Computer Science. She earned both a Ph.D. and M.S. degree in Computer Science from Carnegie Mellon University and a B.S. in Mathematics from Harvey Mudd College. Her research interest includes pervasive computing, the Internet of Things, and Computer Science education. Prior to joining the faculty of West Point, Dr. Ebling spent three years as the Chief Technology Officer at Medaptive Health and nearly 20 years at the IBM T. J. Watson Research Center. Dr. Ebling is an ACM Distinguished Scientist.

©American Society for Engineering Education, 2025

Experiences with the
GraySim CPU Scheduling Simulator

Abstract

An undergraduate course in Operating Systems contains many difficult topics, including CPU scheduling. One approach to helping students test their understanding is using a paper-based worksheet that allows them to record which process runs at each point in time. Another approach, which has been used in the past with apparent success, is the use of simulations. Many of these simulations have been lost to the ravages of time and are not readily accessible today. Others are designed to show students the solutions, but not to help them assess their own understanding. None of the papers that we found that describe these simulations performed a study on the learning outcomes of the students using the tools. This paper describes our experience in deploying the GraySim tool in an OS course. GraySim gives students the opportunity to test their understanding of five common CPU scheduling policies and provides them with feedback about any errors they make. GraySim allows students to specify the policy they wish to practice. The students indicate which process is running at each point in time. The student can then view feedback on their solution. If their solution is correct, the simulator tells them so; otherwise, the feedback identifies the types of errors the student has made, such as scheduling a process for too little or too much time, scheduling the process before its arrival time, or scheduling it at the same time as another process. The simulation also offers policy-specific feedback, such as that the student has preempted a process when using a non-preemptive policy or that they appear to have used the incorrect policy. We report the results of a study on the effectiveness of GraySim in the classroom, comparing the learning outcomes between two cohorts of students: one that had access to GraySim and the other that did not. We also present the results of a qualitative study that asked students about their experience using GraySim. We close with a discussion of the limitations of the study and threats to its validity, and conclude with a discussion of future research.

Introduction

Students often look upon an undergraduate course on Operating Systems (OS) with trepidation because this course covers many complex and abstract topics and requires low-level C programming. One approach many instructors have used in undergraduate OS courses is using simulations. Although these simulations do not address the challenges surrounding low-level C programming, they can help make complex, abstract topics more concrete and easier to understand, thereby giving students a more stable foundation from which to start programming.

To aid students in learning CPU scheduling policies, we built GraySim, an interactive simulation that allows students to practice their understanding of CPU scheduling policies and receive relevant feedback. We then conducted a qualitative study on GraySim to obtain feedback from students who had recently completed an undergraduate OS course regarding whether they believed the simulation would have been effective in the classroom [1]. The results were encouraging, and we deployed GraySim in our institution's OS course the following semester. This paper reports on our experiences with this deployment and our answers to two research questions: (1) whether it would improve student learning and (2) whether students find value in using it. We present the results of both a quantitative study and a qualitative study. Our quantitative study compares the performance of students during the current course offering with that of students enrolled during the previous course offering before the introduction of GraySim. Our qualitative study examines students' perceptions about its utility and value.

The remainder of this paper is organized as follows. We begin with a discussion of related work. We then describe the GraySim simulation with a focus on the differences between this work and the prior publication. Following that, we summarize the qualitative and quantitative methods. Next, we present the results of the study as well as a discussion of their implications. We then discuss the limitations of the study and threats to its validity. We conclude the paper and describe opportunities for further work.

Related Work

Many educators have experimented with the use of simulations in their undergraduate OS courses. Robbins experimented with numerous simulations in his OS course and published a number of papers about those simulations [2] - [7]. His final paper on this topic [8] explained that although he did not perform a formal evaluation of his simulators, "students seem to enjoy using [them]". None of these simulators are readily available today.

Maia and his colleagues used the SOsim graphical simulator in their OS course [9]. SOsim emulates the main subsystems of a multi-programming operating system, including the scheduler. Their paper presented the results of a user study with both quantitative and qualitative feedback from a cohort of 30 students. Maia's results indicated support for more simulations in the classroom; however, their study did not include learning outcomes or quantitative improvement in student understanding of the subject. In addition, access to the SOsim scheduler is limited.

Paschoal and his colleagues described their educational tool, called SSP-Edu, and how it fills the gaps in the current simulation literature [10]. They addressed the question of whether SSP-Edu helped the student get better results in learning process scheduling. However, because the quantitative data was taken from a small sample size of only 10 students, no statistically relevant conclusions could be drawn from the results.

Jain and Suresh present an application that visualizes various scheduling policies and the effectiveness of each [11]. Accessing this tool requires contacting the authors personally. In addition, the authors did not study whether the simulation improved student learning.

Students who took our OS course in Fall 2023 reported using the CPU Scheduling Visualizer [12]. This simulation allows students to configure processes and scheduling policies in a manner similar to GraySim. This visualizer allows students to see differences in efficiency of these scheduling policies. The simulation does not test student understanding, nor does it provide educational feedback. Despite its use by students, no research has been conducted on its effectiveness in the classroom or its impact on student learning.

GraySim CPU Simulation

As described previously [1], the purpose of GraySim is to provide students with an interactive, visual representation of CPU scheduling policies. GraySim supports five uniprocessor scheduling policies: First-In-First-Out (FIFO), Shortest Job First (SJF), Shortest Time-to-Completion First (STCF), Round Robin (RR), and Multi-level Feedback Queue (MLFQ). GraySim allows students to practice their understanding of the policies and provides educational feedback to improve their understanding. This feedback is either specific to the policy, such as when a student preempts a process when using a non-preemptive scheduling policy, or generalized, such as when a student has started a process before its start time. The GraySim repository is publicly available via GitHub [13].

Since that original paper was published, GraySim has experienced some design changes. GraySim now supports some randomized process configurations. Every time GraySim is run from the command line, a random selection of processes is selected. GraySim also presents a new tab-based interface, which allows students to practice the same process configuration on all supported scheduling policies. The initial window (not shown) contains six tabs and shows a summary of the process-related metadata. Each tab, located at the top of the window, displays the abbreviations for one scheduling policy. When a tab is selected, for example MLFQ, the window changes to Figure 1. At this point, the interface looks much like that described in the original publication. The left-most area of the window holds the same process-related metadata shown by the simulator on the initial window. The right-most area shows three control buttons, which perform the following actions:

- **Show Policy –** Displays a description of the policy
- **Show Feedback –** Displays both general and policy-specific feedback
- **Show Solution –** Displays a solution

As before, the center area of the window allows students to select which process is running at any given time based on their understanding of the selected policy. As they click on the boxes, the background color of the box darkens. Figure 2 shows the state of GraySim after a student clicked "Show Policy" and (incorrectly) scheduled the processes according to the MLFQ scheduling policy (where each queue, i, uses a quanta equal to 2^i).

When the "Show Feedback" button is selected, the feedback contains both specific feedback for the current policy and general feedback common to multiple policies. Policy-specific feedback addresses common mistakes that students make with a particular policy. For example, specific feedback is generated to remind a student that a policy is non-preemptive, as seen in Figure 3 for non-preemptive processes such as FIFO and SJF. General feedback, in contrast, applies to every policy. An example of general feedback is that a student has allocated too much (or too little) time to a process. Figure 4 illustrates many examples of generic feedback. These general and

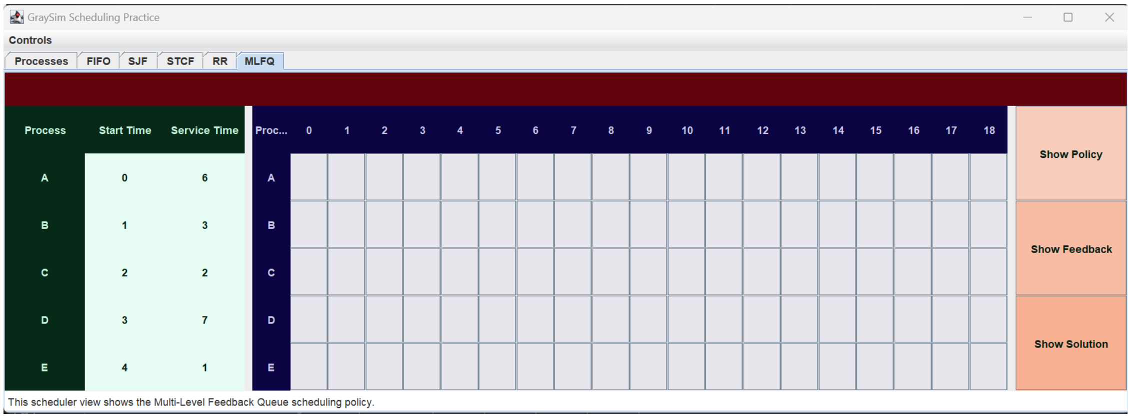


Figure 1: GraySim Displaying Multi-Level Feedback Queue

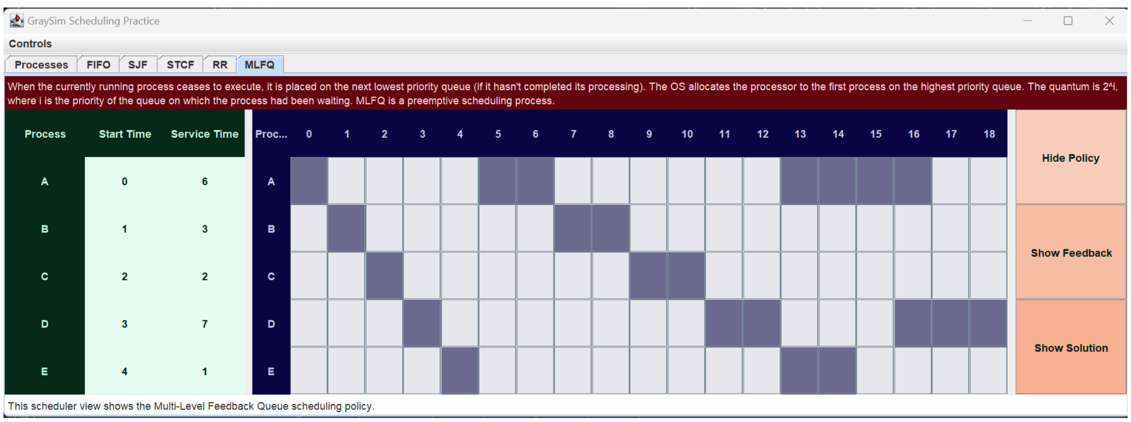


Figure 2: Student Selection of Answer

policy-specific feedback messages allow students to reflect on their mistakes without actually viewing the solution. In addition, they reduce the amount of direct feedback the instructor must provide.

When the "Show Solution" button is selected, the solution is shown to the student by placing an 'X' in each square as shown in Figure 5. If students have made selections, their selections will remain highlighted, allowing students to compare their answer with the correct solution (not shown here).

Method

We investigated the use of GraySim in an undergraduate OS course to answer two research questions: (1) Does the use of GraySim improve student learning? and (2) Do students perceive value to GraySim? Our study, which was reviewed by our local Institutional Review Board (#CA-2024-165), included both quantitative and qualitative aspects.

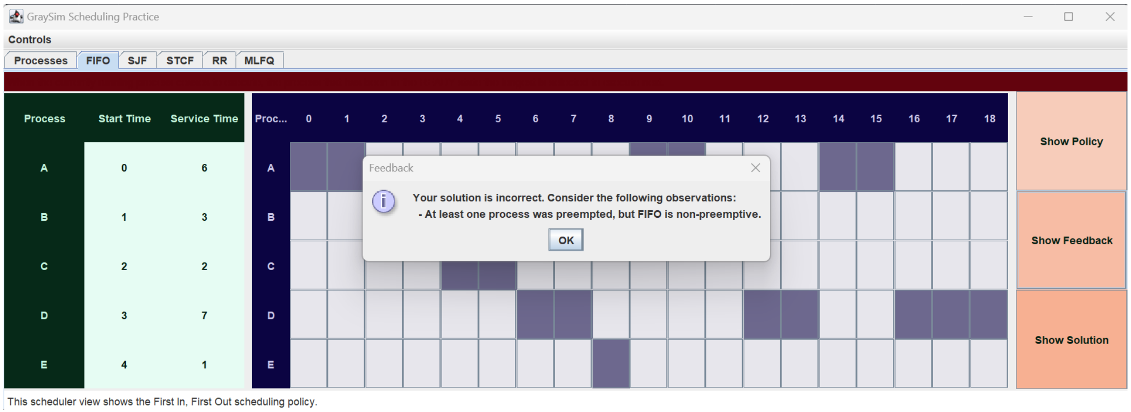


Figure 3: Specific Feedback on FIFO

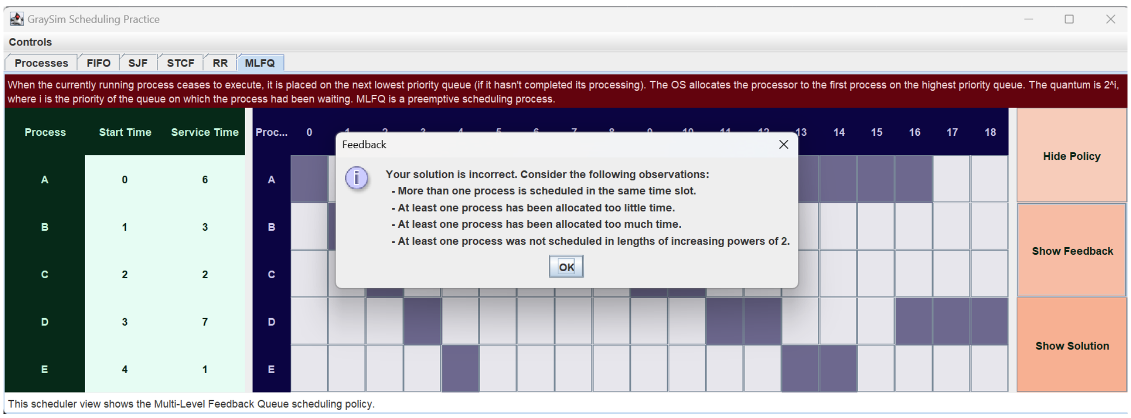


Figure 4: Generic Feedback on MLFQ

Quantitative Study

Our primary purpose in conducting the quantitative study was to answer the question: Does the use of the GraySim CPU simulator improve student learning? We use student performance on an exam as our measure of student learning. We collected exam responses from two sample groups: a group of students who did not have access to GraySim while taking OS and a group who did. We conducted a two-proportion z-test [14] for each tested scheduling policy: FIFO, RR with quantum $q = 2$, STCF, and MLFQ. (Note that SJF was not tested on the exam.) We used two hypotheses for each scheduling policy, where X is FIFO, RR, STCF, and MLFQ:

- H_0: The students' performance with the X scheduling policy with access to GraySim is the same as the students' performance before using the simulator.
- H_1: The students' performance with the X scheduling policy with access to GraySim is different from the students' performance before using the simulator.

The exam had a question that evaluated students' understanding of CPU scheduling policies. It presented five processes together with their arrival and service times. The question had four parts, one for each tested scheduling policy. Each part presented a grid with the time units marked on

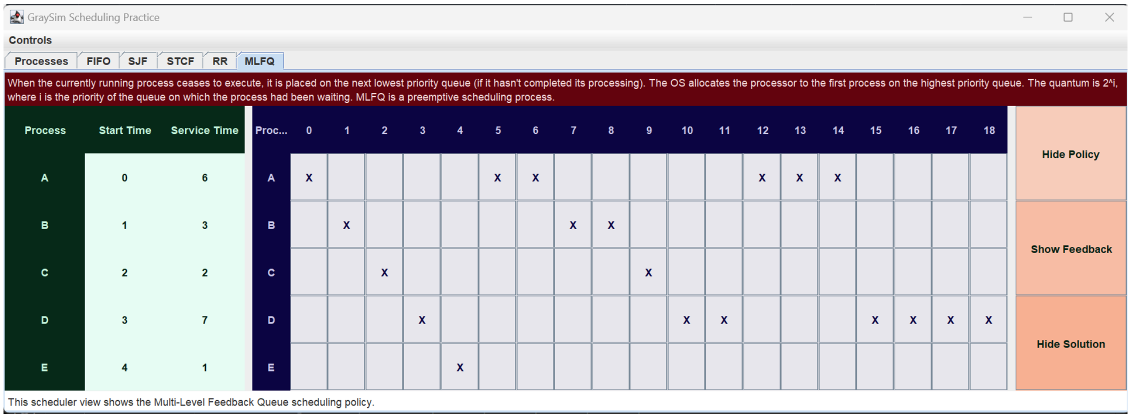
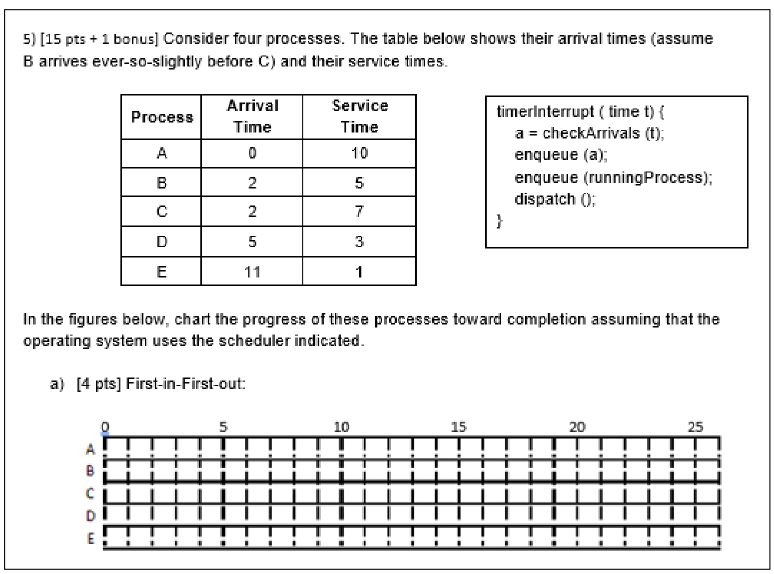


Figure 5: Solution to MLFQ

the top and the processes to be scheduled on the far left as shown in Figure 6. (Note that when learning CPU scheduling algorithms, both cohorts of students received a paper-based, in-class exercise very similar to this exam question with space for *all* five policies, including two different quanta for round robin.) Students filled in the grid to indicate which process ran at each time unit.



5) [15 pts + 1 bonus] Consider four processes. The table below shows their arrival times (assume B arrives ever-so-slightly before C) and their service times.

Process	Arrival Time	Service Time
A	0	10
B	2	5
C	2	7
D	5	3
E	11	1

```
timerInterrupt ( time t) {
    a = checkArrivals (t);
    enqueue (a);
    enqueue (runningProcess);
    dispatch ();
}
```

In the figures below, chart the progress of these processes toward completion assuming that the operating system uses the scheduler indicated.

a) [4 pts] First-in-First-out:

Figure 6: Format of CPU Scheduling Policy Exam Question

Our observational units for our two-proportion z-test were the students' responses to each scheduling policy. The identified categorical variable for these observational units is the correctness of the student's response. If a student answered the subpart correctly they received a score of 1. If they answered incorrectly they received a score of 0. For the Round Robin scheduling policy, several students in the Fall 2024 semester misread the question and scheduled their answer with a quantum of $q = 1$ instead of $q = 2$. We scored their response as a one because they demonstrated full understanding of the scheduling policy, just incomplete comprehension of the question.

We drew the two samples from the cohorts of students enrolled in our OS course over two academic years. One cohort had access to GraySim; the other did not. The exact same question was posed to each cohort on an exam; the exam was not released to either cohort.

We collected these data on student success in a spreadsheet. We then calculated the proportions of student success from these values. At this point, we conducted a two-proportion z-test, comparing the sample proportions. We used p_1 to represent the observed sample proportion in Fall 2023 and p_2 to represent the observed sample proportion in Fall 2024. Similarly, we used n_1 to represent the sample size in the Fall 2023 cohort and n_2 to represent the sample size in the Fall 2024 cohort. We then used the following equations:

$$p = \frac{4_i n_1 + p_2 n_2}{n_1 + n_2} \qquad z = \frac{p_1 - p_2}{\sqrt{(1-p)(\frac{1}{n_1} + \frac{1}{n_2})}}$$

If the p-value that corresponds to the test statistic z for a policy was less than the chosen significance level of $\alpha = 0.025$, we rejected the null hypothesis. Note that we reduced the level of α due to the multiple comparisons included in this study for each scheduling policy.

Qualitative Study

Our primary purpose in conducting the qualitative study was to gain an understanding of how students who used GraySim in a classroom environment perceived its value. Students were provided access to GraySim when learning about CPU scheduling policies in the classroom, and the survey was distributed by a third party after the relevant midterm exam. All students enrolled in the course were invited to participate. If more than 80% of the students completed the survey, everyone in the course received five bonus points. Five bonus points represented a minimal incentive because the total number of available points was 1000.

The survey questions covered a number of resources that may have impacted their learning experience, including GraySim. The questions related to GraySim are as follows:

1. How helpful did you find [the CPU Scheduling Simulation] in studying for the exam? (Five-point Likert scale from “Very Helpful” to “Very Unhelpful”)
2. Rank (from most important to least important) the study aids you used for succeeding on the exams. (7 study aids: Text, Lesson, ICE, Pintos Boot Process Video, Pintos System Call Handling Video, CPU Scheduling Simulation, Other). The results show only the ranking of the CPU Scheduling Simulation.
3. How many times did you use [the CPU Scheduling Simulation] (Number entry)
4. What do you like about the CPU Scheduling Simulation? (Text entry)
5. What could be improved in the CPU Scheduling Simulation? (Text entry)
6. How impactful (if at all) was having this simulation available to you? (Text entry)

Results

For both the quantitative and qualitative study, the total sample size was 91 students, with 45 taking the course in Fall 2023 and 46 taking the course in Fall 2024. Those who took the course in 2023 did not have access to GraySim, whereas those who took the course in 2024 had access to GraySim. Both cohorts were in their final year of a Computer Science or Cyber Science major before graduating from our institution.

Quantitative Study

All but one student answered all of the scheduling policy questions on the exam. This student left the MLFQ sub-question entirely blank. We scored their result as a 0, or failure, because this question was not the last question on the exam. If it had been, the lack of a response to this question might have indicated a lack of time to complete the exam as opposed to a lack of understanding. However, because the student responded to questions following the scheduling policies, we chose to assume that they did not know the answer to the question.

When calculating the proportions, the two authors compared individual scores of the student responses. We noted four discrepancies between our answers and reviewed our analysis of these answers before continuing. We note that the number of “successes” and “failures” in each population was greater than 10, and the samples were independent.

Table 1 displays the overall results of our analysis. The far left column refers to the different scheduling policy questions: FIFO, RR, STCF, and MLFQ. The top row of the table has the following labels: Fall 2023, Fall 2024, z, p-value, and standard error. The columns labeled Fall 2023 and Fall 2024 represent the proportion of successes for each scheduling policy question in that cohort of students. The two-sided p-value is calculated from the z test statistic to compare it with the significance level. The standard error of a statistic is the standard deviation of its sampling distribution.

Table 1: Overall Quantitative Results

	Fall 2023	**Fall 2024**	z	p **-value**	**Standard Error**
FIFO	0.9778	0.9782	0.0137	0.9890	0.0307
RR	0.5555	0.4565	-0.9449	0.3447	0.1043
STCF	0.7333	0.6304	-1.0536	0.2921	0.0970
MLFQ	0.5111	0.4565	-0.5212	0.6022	0.1046

We failed to reject the null hypothesis for each policy. None of their p-values are less than our significance level α. Thus, we do not have sufficient evidence to say that the proportion of successful answers for the FIFO, RR, STCF, and MLFQ CPU scheduling policies are different between the two cohorts.

Qualitative Study

A total of 39 students participated in the qualtitative study, which is 84.8% of the 46 enrolled students. (All students received the five bonus points.) The response to the question asking how helpful students found GraySim was overwhelmingly positive (Figure 7). Twenty-three students responded that it was “Very Helpful” and only five students responded “Neutral” or below. In addition, three students responded that this question did not apply to them.

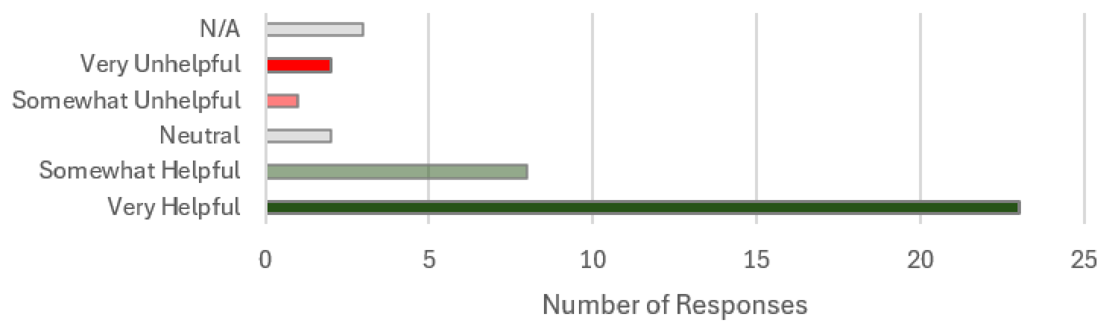


Figure 7: Helpfulness of GraySim

The next question asked students to rank each study aid from most important to least. Figure 8 shows the results. GraySim falls slightly above the average with 24 students rating it in the top half of the study aids and 10 students rating it in the bottom half.

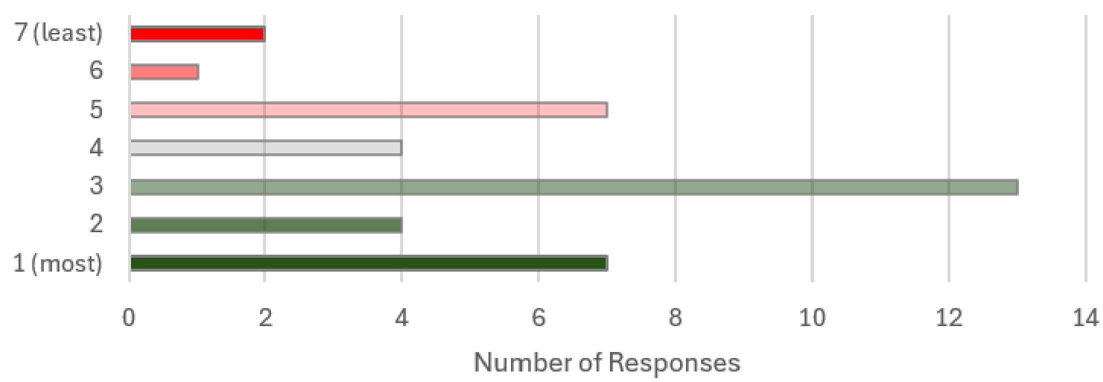


Figure 8: Importance of GraySim

Students reported using the CPU Scheduling Simulation an average of 6.68 times, with a maximum of 25 and minimum of 1. Two students did not answer this question.

The next three questions asked what students liked about the simulation, the improvements they would like to see, and the impact of having it available. All three were open-ended. Any student who failed to answer the question was removed from the pool. These responses were coded independently by both authors. We then compared our coding. We found ten differences in our coding. We agreed to combine two codes (“good for testing” and “good for practice”) and reached consensus on all responses. The final results are shown in Figure 9. Overall, three features stood out that the students liked: the feedback provided; the intuitive interface; and the opportunity to test, learn, and practice.

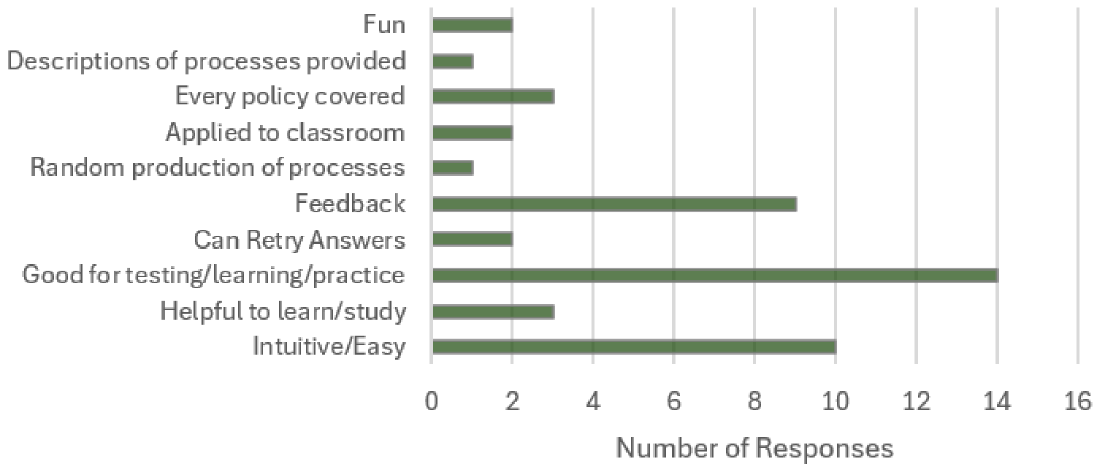


Figure 9: Features Receiving Positive Feedback

The students also suggested improvements to the simulation, seen in Figure 10. The most common recommendation that students made was to allow the learner to change the process configuration inside the simulation rather than requiring them to restart the simulator. The second-most requested improvement was more feedback for incorrect answers. Three students suggested a reset button to clear previous selections from the simulation window. In addition, three students described the simulation as unintuitive, an apparent contradiction from the ten who thought it was intuitive, which suggests further user interface improvements are needed.

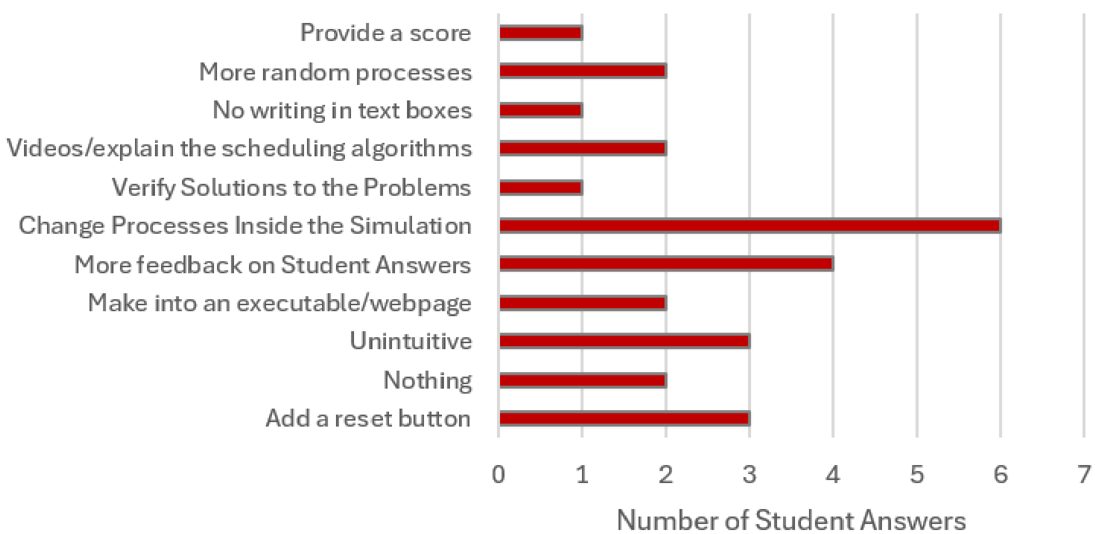


Figure 10: Suggested Improvements

The final written question asked about the impact the simulation had on student learning. Although this was an open-ended question, the answers predominantly fit those of a Likert scale. We present the results as such. Of the 27 students who provided an answer to this question, 13 of them found GraySim to be “Moderately Impactful,” while 12 found the simulation to be “Very Impactful,” as seen in Figure 11. Several students included aspects of the simulation that they liked in their answer to this question. The researchers compared the response IDs of those who provided these types of comments; if the respondent had not already provided that feedback to the prior question, the researchers moved the aspects they liked to the relevant question. (The data shown in Figure 9 contain these responses.)

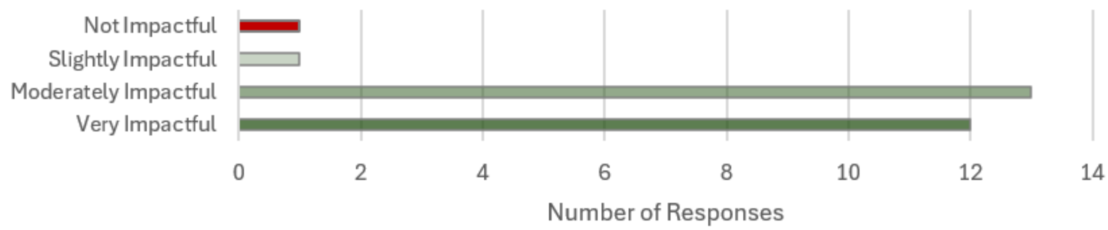


Figure 11: Impact of GraySim

Limitations & Threats to Validity

When considering our research, there are several limitations and threats to the validity of our study. The first threat to our quantitative results relates to the way the relevant lessons were taught. When teaching the concepts of CPU Scheduling in Fall 2024, the instructor used slides posted on the local course website for students to access. In contrast, during Fall 2023, the instructor wrote substantively similar material on the white board. This difference in how the lesson was taught had to do with differences in the way the respective classrooms were laid out. This difference in presentation could have impacted the way the students learned between the two year groups and may have influenced our results.

The second threat to the study’s validity is our use of binarization. As in all statistical evaluations, information is lost when binarizing information that does not have to be binary. Every student who got the question 100% correct was included in the proportion of students who successfully understood the question. However, if they made even a simple error, such as switching two boxes, they were marked as not understanding the concept. This may have skewed the results by being overly conservative about whether students understood the simulation or not. To evaluate this threat, we objectively graded student responses by the number of boxes they marked correct and performed a preliminary two-sample t-test on the data. The p-values for each question were well above our significance level $\alpha = 0.025$. We conclude that binarization did not influence our results.

A third threat is our lack of control for either student ability or differing external demands. Both of these influences could be a potential source of confounding variables. For example, the association between student ability and performance on different problems was not accounted for in any of our data. Likewise, the demands on student time can vary from year to year. These demands could have affected the results of our quantitative study. However, we do not believe that either of these variables affected our results.

The final threat we consider is the lack of recording of simulation usage and the consequential need to use self-reporting of this information. Both the number of times GraySim CPU Scheduling Simulator was used by each student and their results for each scheduling policy were self-reported, introducing bias into the results. This threat also relates to students' demonstrated understanding during use of the simulation. For example, if a student practiced all scheduling policies twice but got all of them correct, we would expect them to do well on the exam; whereas, if a student practiced twice on one scheduling policy but was wrong both times, we would expect their results on the exam to reflect that lack of understanding. Because this stand-alone simulation does not record student usage in any way, we are unable to track that activity, removing a possible indicator of students' overall understanding of the scheduling policies. In addition, we were unable to match the self-reported usage with the learning outcome due to the anonymous nature of the survey.

Conclusion

This paper presents the results of both a quantitative study that compares learning objectives between two course offerings with and without access to GraySim and a qualitative study that asks students about their actual experience using GraySim. Our results show no statistical significance to support our hypothesis that the proportion of successful answers for any of the scheduling policies is different between the two cohorts, meaning GraySim did not have a statistical impact on student understanding of CPU scheduling policies. The reason the statistical difference is not significant could be due to the smaller size of the student cohorts (about 50 in each) or the small size of the learning difference (e.g., there is very little room for improvement in the FIFO scheduling policy).

Our students provided overwhelmingly positive feedback in the qualitative study that evaluated the utility of the simulation and provided insightful comments about it. Our students reported the simulation to be very helpful and moderately important to student learning in an undergraduate OS course. These results reinforce our prior study that looked at students' expectations for the utility of the simulation [1] by reporting on the actual experiences of students enrolled in an undergraduate OS course. We note that Robbins reported similar experiences with his simulations [8]. Our students also provided additional insights to aspects of the simulation they liked, such as its use for practice and the provided feedback, and suggested numerous improvements, such as being able to change the process configuration while inside the simulation.

The students also recommended improvements such as more specific feedback on their incorrect solutions. One way this could be realized is by evaluating student answers and determining the most common mistakes that students made. These common mistakes could be compiled and added into the feedback evaluation that takes place inside GraySim. The other common recommendations made by students were to make the simulation more intuitive and to provide a reset, or clear, button.

Finally, without a record of student use, we cannot evaluate the direct impact the simulation had on student learning. If GraySim were to be integrated into a web server, this system could record the number of times students used the simulation with each policy and how often they answered correctly. We could then incorporate this information into a paired t-test. Such a change could help to show statistical significance in learning outcomes.

We are currently implementing the suggested improvements and will contribute those improvements to the GraySim repository for others to use. We are also working to incorporate GraySim into a web-based system. Once available, we will perform a follow-up study to examine learning outcomes in light of usage, possibly also incorporating students' incoming major GPA to account for the effect of student ability.

Acknowledgments

The authors wish to thank the students who participated in the study for their valuable feedback and guidance as well as COL Christa Chewar for the inspiration and encouragement. The authors also wish to thank LTC Mike Powell who helped them with using RStudio, the method of evaluation for the quantitative results, and the evaluation of the quantitative results. The views expressed here are those of the authors and do not reflect the official policy or position of the Department of the Army, Department of Defense, or the U.S. Government.

References

[1] S. Z. Bennett-Manke and M. R. Ebling. "GraySim: An OS Scheduling Simulator". In: *J. Comput. Sci. Coll.* 39.8 (May 2024), pp. 55–69. ISSN: 1937-4771.

[2] S. Robbins. "Introducing Empirical Investigation in Undergraduate Operating Systems". In: *SIGOPS Oper. Syst. Rev.* 32.4 (Oct. 1998), pp. 77–80. ISSN: 0163-5980. DOI: 10.1145/302350.302366. URL: https://doi.org/10.1145/302350.302366.

[3] S. Robbins. "Experimentation with Bounded Buffer Synchronization". In: *Proceedings of the Thirty-First SIGCSE Technical Symposium on Computer Science Education*. SIGCSE '00. Austin, Texas, USA: Association for Computing Machinery, 2000, pp. 330–334. ISBN: 1581132131. DOI: 10.1145/330908.331880. URL: https://doi.org/10.1145/330908.331880.

[4] S. Robbins. "Starving Philosophers: Experimentation with Monitor Synchronization". In: *Proceedings of the Thirty-Second SIGCSE Technical Symposium on Computer Science Education*. SIGCSE '01. Charlotte, North Carolina, USA: Association for Computing Machinery, 2001, pp. 317–321. ISBN: 1581133294. DOI: 10.1145/364447.364612. URL: https://doi.org/10.1145/364447.364612.

[5] S. Robbins. "A Disk Head Scheduling Simulator". In: *Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education*. SIGCSE '04. Norfolk, Virginia, USA: Association for Computing Machinery, 2004, pp. 325–329. ISBN: 1581137982. DOI: 10.1145/971300.971413. URL: https://doi.org/10.1145/971300.971413.

[6] S. Robbins. "An Address Translation Simulator". In: *Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education*. SIGCSE '05. St. Louis, Missouri, USA: Association for Computing Machinery, 2005, pp. 515–519. ISBN: 1581139977. DOI: 10.1145/1047344.1047507. URL: https://doi.org/10.1145/1047344.1047507.

[7] S. Robbins. "A UNIX Concurrent I/O Simulator". In: *Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education*. SIGCSE '06. Houston, Texas, USA: Association for Computing Machinery, 2006, pp. 303–307. ISBN: 1595932593. DOI: 10.1145/1121341.1121435. URL: https://doi.org/10.1145/1121341.1121435.

[8] S. Robbins. "A Three Pronged Approach to Teaching Undergraduate Operating Systems". In: *SIGOPS Oper. Syst. Rev.* 42.6 (Oct. 2008), pp. 93–100. ISSN: 0163-5980. DOI: 10.1145/1453775.1453792. URL: https://doi.org/10.1145/1453775.1453792.

[9] L. P. Maia, F. B. Machado, and A. C. Pacheco. "A Constructivist Framework for Operating Systems Education: A Pedagogic Proposal Using the SOsim". In: *Proceedings of the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*. ITiCSE '05. Caparica, Portugal: Association for Computing Machinery, 2005, pp. 218–222. ISBN: 1595930248. DOI: 10.1145/1067445.1067505. URL: https://doi.org/10.1145/1067445.1067505.

[10] L. N. Paschoal et al. "Towards an Educational Simulator to Support the CPU Scheduling Algorithm Education". In: *2019 International Symposium on Computers in Education (SIIE)*. Tomar, Portugal: IEEE, 2019, pp. 1–6. DOI: https://doi.org/10.1109/SIIE48397.2019.8970116.

[11] N. Jain and P. V. Suresh. "A java based visual tool to learn CPU scheduling algorithms". In: *2014 International Conference on Computing for Sustainable Global Development (INDIACom)*. New Delhi, India: IEEE, 2014, pp. 519–523. DOI: https://doi.org/10.1109/IndiaCom.2014.6828013.

[12] M. Agrawal. *CPU Scheduler Visualiser*. https://github.com/mukul2310/cpu-scheduler-visualiser. Accessed: 2024-07-09. Feb. 2021.

[13] S. Z. Bennett-Manke and M. R. Ebling. *GraySim*. https://github.com/usma-eecs/GraySim. Accessed: 2024-12-30. Feb. 2024.

[14] Z. Bobbitt. *Two Proportion Z-Test: Definition, Formula, and Example*. https://www.statology.org/two-proportion-z-test/. Accessed: 2024-11-1. Apr. 2020.