Journal of Information Technology Education: Innovations in Practice

An Official Publication of the Informing Science Institute
InformingScience.org

JITEiip.org

Volume 23, 2024

VIRTUAL SIMULATIONS TOOL FOR OPERATING SYSTEMS: ADVANCING E-LEARNING IN COMPUTING EDUCATION

Jyoti Wadmare*	K. J. Somaiya Institute of Technology, Sion, Mumbai, India.	jyoti@somaiya.edu
Dakshita Kolte	K. J. Somaiya Institute of Technology, Sion, Mumbai, India.	d.kolte@somaiya.edu
Kapil Bhatia	K. J. Somaiya Institute of Technology, Sion, Mumbai, India.	kapil.bhatia@somaiya.edu
Palak Desai	K. J. Somaiya Institute of Technology, Sion, Mumbai, India.	palak.pd@somaiya.edu
Ganesh Wadmare	K. J. Somaiya Institute of Technology, Sion, Mumbai, India.	gwadmare@somaiya.edu

* Corresponding author

ABSTRACT

Aim/Purpose	This paper highlights an innovative and impactful online operating system algorithms e-learning tool in engineering education.
Background	Common teaching methodologies make it difficult to teach complex algorithms of operating systems. This paper presents a solution to this problem by providing simulations of different complex algorithms to enable students to visualize and perform hands-on experiments. Developing these simulations offered different hurdles, which included step-by-step precise computations, managing edge circumstances, creation of dynamic representations like Gantt charts and disk scheduling graphs, strong input validation, user-friendly customization, and real-time performance. The developed simulations also observed some limitations, like the Process Scheduling simulator, which can be improved from the aspect of context switching overheads. Disk Scheduling simulators can include different evaluation parameters, such as fairness and starvation avoidance. Banker's Algorithm can address circumstances such as invalid resource requests, resource deadlock, and resource exhaustion to model real-world system behavior.

Accepting Editor Stamatis Papadakis | Received: July 2, 2024 | Revised: September 2, September 24, November 8, November 12, 2024 | Accepted: November 17, 2024.
Cite as: Wadmare, J., Kolte, D., Bhatia, K., Desai, P., & Wadmare, G. (2024). Virtual simulations tool for operating systems: Advancing e-learning in computing education. *Journal of Information Technology Education: Innovations in Practice, 23,* Article 18. https://doi.org/10.28945/5404

Methodology	The study focuses on the development of an e-learning tool that consists of the simulation of 13 different operating system algorithms, such as Process Scheduling, Disk Scheduling, and Banker's Algorithm.
Contribution	This study contributes to the body of knowledge by presenting an e-learning educational tool that bridges the gap between theory and practice in operating system algorithms, thus boosting student engagement and understanding.
Findings	The findings of the work comprise the analysis of 276 student feedbacks demonstrating a significant favorable influence on students' learning and engagement with operating system algorithms through the use of the built-in e-learning tool.
Recommendations for Practitioners	It is advised that educators integrate this e-learning tool into their curriculum to boost student understanding and engagement in operating system courses.
Recommendations for Researchers	Future studies should aim to broaden the range of algorithms contained in the tool and analyze its potential for applicability in other areas of computer science education.
Impact on Society	Operating system algorithms have a profound societal impact by enabling the development of efficient, reliable, and secure software systems that power everything from personal devices to critical infrastructure. Through engineering education, students learn these foundational principles, allowing them to innovate and create software solutions that enhance productivity, security, and connectivity in daily life. This knowledge contributes to the advancement of technology, fostering societal progress in areas like healthcare, communication, and automation while promoting digital security and accessibility.
Future Research	Future research should focus on the development of different e-learning tools for different disciplines of engineering education and evaluate their efficiency in different ways of learning.
Keywords	simulation, process scheduling, disk scheduling, banker's algorithm, resource allocation

INTRODUCTION

Operating systems are an integral part of modern computing, allowing resource management and scheduling of activities with stability in the system. However, increased complexity is a major challenge for trainers. The main problem that students face is understanding advanced algorithms or concepts. On the other hand, the other problems that come up for teachers are how to explain things clearly, provide good exercises to learn, and address resource constraints. According to Genkov and Slavov (2021), there is a growing demand for new teaching methodologies for operating system education challenges. Altalbe (2019) suggests using simulators as supplementary learning tools in interaction with both practical and teaching components that improve the learning process among students and instructors. This method aims to have a clearer understanding of the theory of operating systems and ease the learning process by developing specialized software simulators that focus on algorithms as well as managing resources.

There are many simulators of operating system algorithms, but a one-stop platform integrating several simulations into one comprehensive solution is yet to be found. Some of the already existing simulations include Process Scheduling Solver (Boonsuen, n.d.) and CPU Scheduling Sim (CPU

Scheduling Simulator, n.d.). These simulations tackle the computation of waiting time and turnaround time, making only Gantt charts. However, these simulations do not offer full step-by-step calculations, detailed CPU utilization assessments, or pictorial illustrations of the ready queue at each processing level. Further, disk scheduling simulators, such as Disk Scheduler Visualisation (Seek-Time, n.d.) and Disk Scheduling Solver (AssistedCoding, n.d.-a), measure seek time and generate graphical outputs but do not provide detailed comparisons or tailored computations designed to the specific input sequences implemented. Safe resource allocations can be performed along with the entry in the matrices for different Banker's Algorithm tools like Banker's Algorithm Calculator (Pisqre, n.d.-a) and Banker's Algorithm Simulator (Kanwar Adnan, n.d.). It does not display the available matrix. These constraints underline the need for more powerful simulation tools that give in-depth comprehension and step-by-step representations, increasing both student learning and teaching approaches.

The proposed simulation software attempts to provide an interactive, hands-on method of teaching operating system topics. It allows students to understand complex concepts through effective visualization and complete implementations of algorithms. This method adds innovative dimensions to the teaching of operating systems and further improves the efficiency of learning among students.

This paper discusses the impact and usefulness of the e-learning tool on the performance of the students and canvases directions for new innovations in this tool that could eventually revolutionize algorithms that govern the teaching and learning operating system. The next section entails an introduction to operating systems and their underlying algorithms. This is followed by an explanation of the hypotheses that are part of this research and an analysis of the tools currently available in simulating operating systems and their limitations. The following section provides the learning resources, such as theoretical concepts, flowcharts, and algorithms, for the 13 key processes associated with an operating system. Then, the results of the different simulations that were implemented are presented, along with discussions. Finally, the conclusion gives an overview of the findings and recommends future exploration options.

RESEARCH HYPOTHESIS

The implementation of e-learning tools into computer science education, particularly in the field of Operating Systems, incorporates strong simulation tools with environments that are automated to provide hands-on learning experiences. This is grounded in constructivist theories, which encourage active participation and the production of knowledge and experience. Different simulation techniques are used, for example, simulating some of the complex algorithms, such as algorithms for CPU scheduling and strategies for avoiding deadlocks, in an e-learning tool, thereby making this place controlled yet interactive where learning takes place effectively.

The hypothesis is that such e-learning tools are able to overcome the boundaries of time and space, hence maintaining educational continuity and producing better results in learning. The use of advanced simulation tools and multimedia resources in these labs will enable students to understand abstract and complex systems in a more profound way and, hence, may lead to enhanced knowledge of basic and advanced operating systems concepts.

LITERATURE SURVEY

E-learning tools such as virtual labs have recently become an effective teaching aid in many subjects, including but not limited to Engineering, Computer engineering, and vocational training. They are a partial substitute for the relatively unchanged old-fashioned physical labs with their high cost, limited space, and real-practical barriers to accessibility. Constructivist learning theory emphasizes the active

role of involvement in the learning process, which is the reason why virtual labs represent an excellent tool for this topic. They contain practical experience to make students more engrossed and motivated for learning and promote understanding and production of knowledge.

Studies of authors such as Holovnia and Oleksiuk (2022) and Kleine and Pessot (2023) indicate the use of private cloud-based virtual labs as an opportunity to improve learning results in the education of operating systems and engineering concepts, respectively. Such findings speak to the need to develop user-friendly, interactive, and scalable environments for education that could be required by modern learners, particularly in virtual settings. Reginald (2023) also highlighted that virtual labs play a greater role in self-regulation during the COVID-19 period among learners in online learning environments, and they should be free to learn without time bounds and space bounds.

Research by Frady (2022) focuses on the fact that virtual labs have been integrated into educational technology and vocational training through advanced technologies such as artificial intelligence, virtual reality, and cloud systems for an interactive learning experience. Genkov and Slavov's (2021) design also included virtual labs with virtual desktop infrastructure for overcoming the barriers of distance learning in the disciplines of environmental and computer science to create a chance for experimentation.

One example is the employment of virtual labs even in subjects like physics, a trend observed by May et al. (2022), in which teachers' experiences of the transition to online-only lab modules during lockdown were assessed. The response was wonderful, and it made sure that not only are virtual labs viable, but they are also essential for keeping the learning process running. In this regard, Garcia et al. (2021) have offered synchronous programming labs that have been successful during the COVID-19 lockdown and maintained adult learners' focus on learning even as external events were going on around them.

Besides, the automation of virtual labs has been one of the critical components that should be considered in enhancing the efficiency of education. For example, a system developed by L. Wang et al. (2020), which generates and tests lab environments automatically, highly decreases the setup time. Do Hoang et al. (2022) also presented approaches to IT training laboratories through concurrent schemes and local repositories such that large-scale virtual classroom setups work well.

Virtual laboratories open new possibilities for experience and learning in specialized fields, such as robotics and optical physics. Salas and Ho (2021) proposed a cloud-based virtual laboratory in robotics using Kubernetes orchestration, thereby permitting students to control robots remotely. Gamo (2019) designed OPTILAB as a simulation tool for optical diffraction, which enriches the students' learning experience with fewer errors and previews of the experimental outcome.

Other research agendas targeted virtual labs in complex physical systems and network planning. For instance, Martin-Villalba and Urquia (2022) researched the application of Modelica to represent physical systems in engineering education, providing students with authentic experience concerning challenging concepts of engineering. Zapata-Rivera and Aranzazu-Suescun (2020) presented the potential use of virtual simulation models and educational video games in order to support learners in treating various contemporary challenges, such as antenna dispersion in wireless networks.

Vayadande et al. (2023) stated that other key features of virtual labs include high-performance computing operations, such as CPU scheduling. In this simulation, several scheduling algorithms and parameters, such as completion time and CPU utilization, were tested for various students to further enhance their understanding of operating system processes. Huang and Song (2023) especially emphasized that virtual labs are pertinent to computing learning, where high-performance simulations are linked to the operating systems, allowing teachers to get real-time statistics of how students are doing in their classes.

Moreover, virtual labs have been used in the context of deadlock prevention strategies in computing, such as the work by Y. Wang et al. (2022) and XiaoLing (2019), where advanced algorithms and

models, such as the banker's algorithm and Petri net models, are used to simulate deadlock scenarios and develop efficient solutions. These studies highlight the practicality of virtual labs to study of abstract computing concepts.

Virtual labs have further seen improvements with AI-based methods, as brought forward in Abd El-Haleem et al. (2022). A study was done that developed an AI-driven performance assessment that supports grading and assistance given to students in online labs. In teacher education, McGarr (2020) discussed the role of virtual simulations in developing pre-service teachers' classroom management and reflective practices. His study highlighted how virtual simulations allow teachers to practice in a safe and controlled environment, preparing them for real-world classroom scenarios.

Furthermore, Panasiuk et al. (2021) have discussed the advantages and disadvantages of virtual labs in engineering education. The authors consider that virtual labs can provide access at any time and at any place so the future of engineering education might be blended learning of virtual and real labs.

Altalbe (2019) highlighted the application of realistic and convincing simulation technologies in laboratory learning and developed a theoretical model that is based on usability and learning objectives. The information about students' perceptions and educational benefits of using virtual laboratories was obtained through the research. Finally, Sáenz et al. (2021) focused on the aspects of modularity and access of virtual labs by outlining solutions that guarantee usability as well as reliability across varying educational contexts. Such evolutions show how virtual labs are continuously evolving in satisfying the needs of modern education.

In conclusion, virtual labs are a versatile platform to be adapted to the different needs of educational institutions. They allow students to perform experiential learning activities, solve complex problems, and gain hands-on experience in a cost-effective, scalable manner, as several studies have shown in many different domains. More than that, Luse et al. (2021) have tried to solve the problem of the education in wireless technology by introducing a virtual 802.11 lab with an integration of USB hubs and wireless adapters, minimizing physical setups, but not diminishing the quality of learning.

In such highly specialized areas as optical communication, Dahan et al. (2022) proposed a universal virtual lab for performance assessment in wideband DWDM systems. This lab has simulated real-time scenarios of transmitter and receiver imperfection with a great speed-up factor and therefore is highly efficient in its learning utility for the students. Along the same line, server-deployed virtual labs by Rassudov and Korunets (2022) simulates real-time operations of the hardware which also contributed to distance learning but supported in-person training of the engineering students. Such research describe the versatility of virtual labs and their effects on current education.

The proposed system is compared with existing simulations, highlighting its features and identifying gaps in the existing systems, as outlined in Table 1.

Table 1. Comparison of proposed simulation with existing simulations available

Operating system algorithm simulations	Existing simulations features	Gaps in existing simulations
Process Scheduling (FCFS, SJF, Priority Preemptive, Priority Non-Preemptive, SRTF) For more information, see: (Boonsuen, n.d.), (CPU Scheduling Sim, n.d.), (Pisqre, n.d.-b), (AssistedCoding, n.d.-b), (Choudhary, n.d.).	Calculates Waiting Time. Calculates Turn Around Time. Plots Gantt Chart.	Waiting and Turnaround Time calculation at each step is not defined. CPU utilization at each step is not calculated. Ready Queue is not defined for each step.
Disk Scheduling (FCFS, SSTF, SCAN, CSCAN, LOOK, CLOOK) For more information, see: (SeekTime, n.d.), (AssistedCoding, n.d.-a), (Makaroff, n.d.), (Khushali, n.d.).	Directly print the value of seek time calculated. Plots graph of different disk scheduling algorithms.	Step by step calculations for seek time is not defined. No comparison with different disk scheduling algorithms for a given input sequence.
Banker's Algorithm For more information, see: (Pisqre, n.d.-a), (Jangid, n.d.), (Kanwar Adnan, n.d.), (Anish-U, n.d.).	Inputs Allocation Matrix. Inputs Maximum Matrix. Displays safe or unsafe sequence.	Available Matrix is not plotted.

The proposed system of simulations addresses the gaps observed in existing systems by offering comprehensive solutions. The novel contributions in the proposed system for process scheduling algorithms include the calculation of step-by-step waiting and turnaround time and defining ready queue and CPU utilization for each step. Similarly, in disk scheduling, seek time is computed for every step, followed by a comparison chart of different algorithms for a given input sequence of cylinders. Furthermore, the Banker's Algorithm simulation displays the available matrix. These enhancements signify the superiority of the proposed simulation over existing ones, providing a more thorough and informative learning experience.

METHODOLOGY

This research paper presents the development of an e-learning tool for Operating System algorithms. The platform covers all the basic topics of operating systems and is quite comprehensive and interactive. A systematic and organized approach is implemented in the system for every method, which includes theoretical explanation, simulation, practical examples, numerical exercises, and quizzes. This design is centered on enhancing the learning experience. The e-learning tool also features a discussion forum that fosters user interaction and collaborative learning, making it an unequaled educational resource.

The simulations are based on the three major algorithms of the Operating System:

(i) Process Scheduling Algorithms
(ii) Disk Scheduling Algorithms
(iii) Banker's Algorithm

Simulations for Process Scheduling Algorithms and Banker's Algorithm are developed using Java and Java Swing. Here, the algorithms based on Java serve as the base code; interfaces use a variety of Java Swing components such as JLabels, JButtons, JComboBox, Textfields, EventListeners, JMenus, JPanels, JScrollPanes, and Jtables. Simulations of the Disk Scheduling Algorithm are built with the Plotly API in HTML, allowing for an interactive environment in which the user can explore and understand how different algorithms work step by step. The work provides users with detailed information on Operating System algorithms and offers a feedback mechanism for continuous improvement. Figure 1 highlights the e-learning tool that covers different algorithms in the subject of Operating Systems.

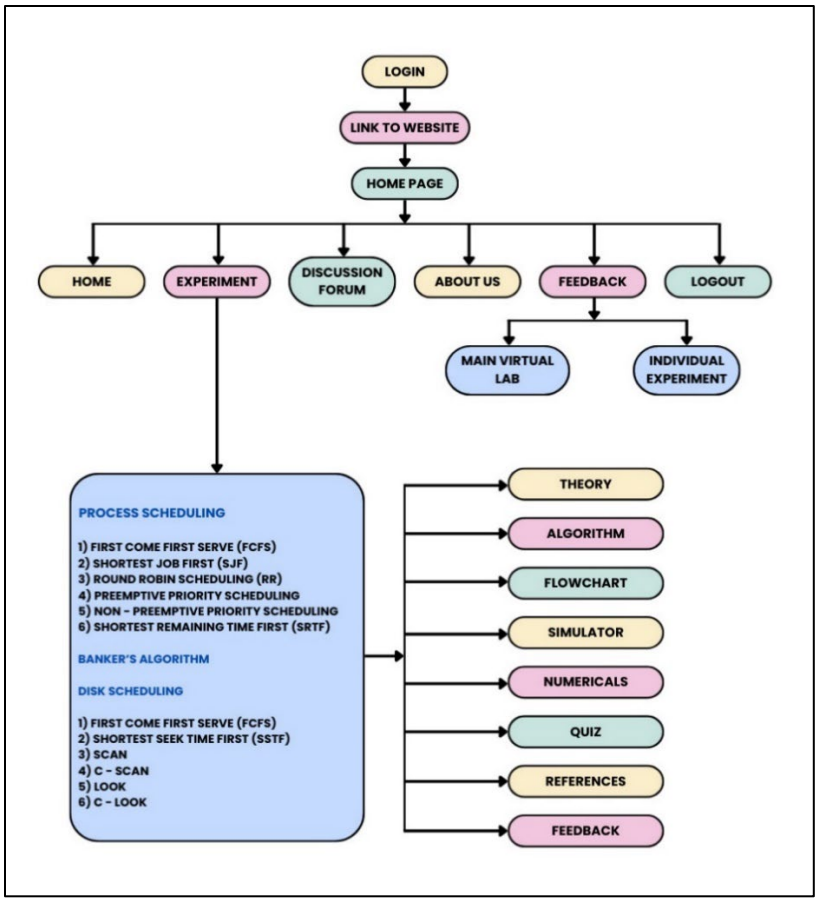


Figure 1. Flow diagram of system implementation

The study encompasses a comprehensive analysis of various process scheduling and disk scheduling algorithms. In the realm of process scheduling, the algorithms covered include First-Come-First-Serve (FCFS), Shortest Job First (SJF), Round Robin (RR), Preemptive Priority, Non-Preemptive Priority, and Shortest Remaining Time First (SRTF). These algorithms play a pivotal role in optimizing CPU utilization and improving system performance by determining the order in which processes are executed. Furthermore, the simulation of the Banker's algorithm is included to address deadlock avoidance in resource allocation.

The research study delves into the algorithms necessary to manage I/O requests efficiently on the front of disk scheduling. They are, therefore, classified into the following: the FCFS, SSTF, SCAN, C-SCAN, LOOK, and C-LOOK algorithms. All the algorithms contribute towards the optimization of the movements of the disk arm and the minimization of seek time. An in-depth understanding of the operational characteristics and performance implications in different system environments is what the research imparts. Components of e-learning tool are:

(a) *Theory:* A comprehensive explanation of the algorithm or topic is provided, covering its principles, concepts, and relevant background information.

(b) *Algorithm:* The step-by-step algorithm or procedure of the specific topic is presented, detailing how it operates and its key components.

(c) *Flowchart:* A visual representation of the algorithm is presented in the form of a flowchart, providing a clear and intuitive understanding of the process flow.

(d) *Simulator:* A simulator is included to allow users to interactively visualize how the algorithm works. It may provide a graphical representation or simulation of the algorithm's behavior.

(e) *Numericals:* Numerical examples are given to demonstrate the application of the algorithm in practical scenarios, helping users grasp its real-world usage.

(f) *Quiz:* A quiz section is included to test users' understanding of the algorithms. It may consist of multiple-choice questions or other types of assessments.

(g) *References:* References and additional resources are listed at the end of each experiment, offering further reading material for those interested in diving deeper into the topic.

(h) *Feedback:* Users are encouraged to provide feedback on the experiment they completed to aid in improvement and enhancement.

This paper provides an intuitive, interactive user platform that focuses on the comprehensive exploration of operating system algorithms so that they are better comprehended. Placed in a permanent position of updating, based on feedback from users, it was sure to give users optimum learning experiences.

RESULTS

The operating system e-learning tool is an innovative, interactive instructional tool developed to assist in a complete understanding of operating system algorithms. Registered users can access different modules of this tool, which include process scheduling, disk scheduling, and Banker's Algorithm simulators so that one can practice the techniques in a hands-on way.

SIMULATION OF PROCESS SCHEDULING ALGORITHMS:

Accessing and using the simulator

The interface of the process scheduling simulator includes crucial components such as a job pool, ready queue, CPU utilization metrics, and Gantt charts, as illustrated in Figure 2. These features are to visualize the outcome of scheduling algorithms. Critical metrics, such as waiting time and turnaround time, must be evaluated to assess the efficiency of the implemented scheduling strategies in evaluating system performance.

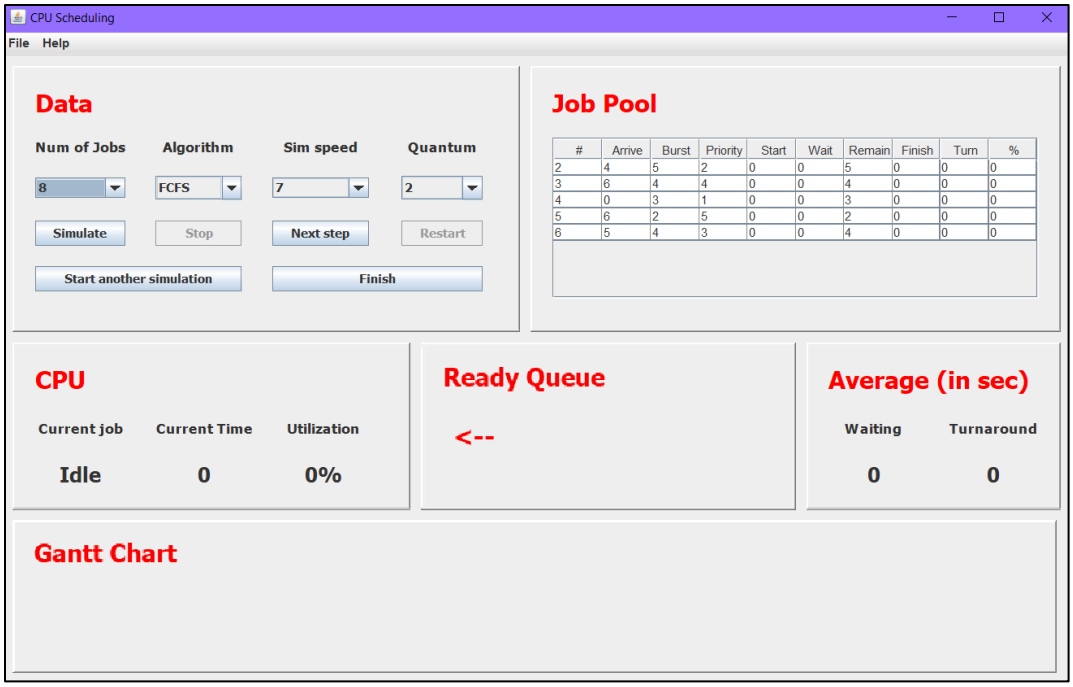


Figure 2. Interface of process scheduling simulator

Process data, including arrival, burst, and priority time, needs to be given input to the simulator to run it, as shown in Figure 3. It can be entered manually, as shown in Figure 4, or default data can be loaded in the simulator, as shown in Figure 5. Moreover, different parameters, such as the number of jobs, algorithm needed to perform the simulation, simulation speed, and quantum time, need to be selected to run the simulation successfully.

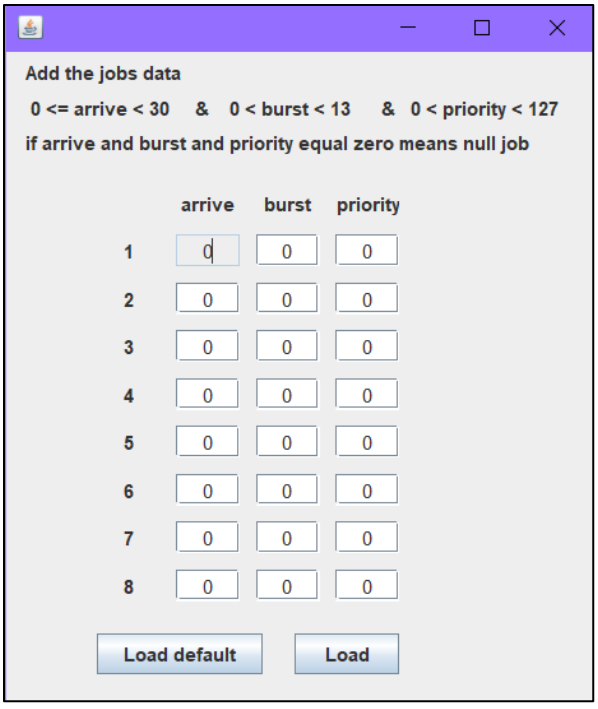


Figure 3. Data to be added to perform simulation

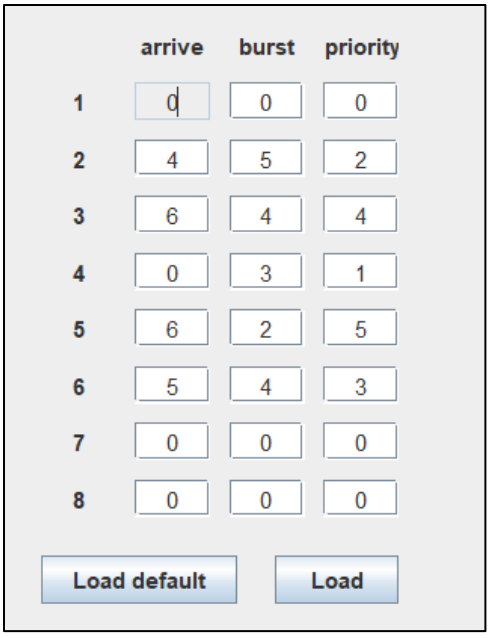


Figure 4. Sample format to add data added manually

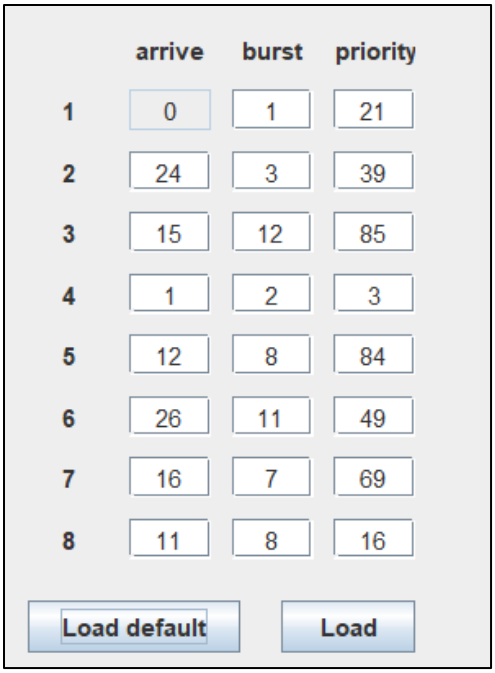


Figure 5. Sample format of data loaded by default

Different algorithms present in the simulator of process scheduling algorithms are as follows:

1. *First-Come-First-Serve (FCFS):* This is one sort of scheduling approach where processes are processed based on their arrival sequence. The simulator provides users with the visual rationale for the execution of processes based on their arrival times, displaying the sequence as a Gantt Chart. Calculations are done for various performance measures, such as waiting time, turnaround time, and CPU utilization. Users can gain a better view of how scheduling influences their overall system performance.

2. *Shortest Job First:* SJF schedules operations with the lowest burst time, hence minimum wait times for the shortest jobs. This technique should be illustrated by the simulator because it reduces backlog times with variant process lengths. Using such a simulator, users can input custom job data or employ a few preloaded examples where they can see that the average turnaround time and average waiting time both vary.
3. *Round Robin:* This algorithm splits CPU time in a stepwise manner according to the prespecified time quantum. The simulator allows users to observe how different quantum sizes impact context switching and waiting times. The high context switch rate of Round Robin makes it more suitable for time-sharing systems. The simulator then provides rapid visual feedback regarding the performance of the system in various configurations.
4. *Priority Scheduling:* In Priority Scheduling, processes are executed based on their priority levels. It might be either preemptive or non-preemptive. This displays how the change in the priority affects the execution order as well as system performance parameters such as waiting time and turnaround time.

Users can view the preemptive and non-preemptive versions and discuss the difference between these two types of priority scheduling in realistic scenarios with varying priority work. Various evaluative metrics are used in a process scheduling simulator.

1. **Gantt Chart**
   In process scheduling simulation, the visual illustration of the timeline of the execution of processes is shown in a Gantt chart as shown in Figure 6. Each process start and end is explained to make it easy to analyze CPU allocation while evaluating the waiting time, turnaround time, and effectiveness of scheduling algorithms.

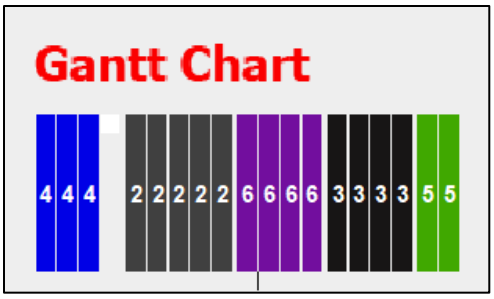


Figure 6. Gantt Chart

2. **Ready Queue**
   For process scheduling methodologies, the ready queue shown in Figure 7 exemplifies a pictorial representation of the pool of waiting processes for CPU allocation. The Gantt chart depicts how activities are conducted in temporal sequence, while the ready queue represents how operations are queued and ready for scheduling. This lets users determine which processes are entering and leaving the queue and provides key information on how efficiently the scheduling algorithm is performing the transitions of processes and resource allocation.

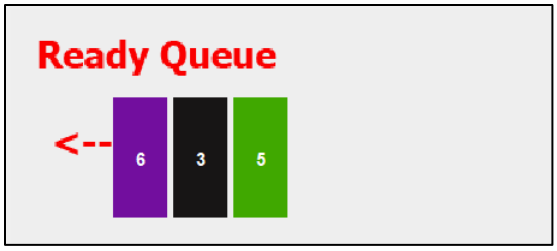


Figure 7. Ready queue shows the next procedure in line

3. **Average Waiting and Turnaround Time**
The mean waiting time is the average duration that a process spends in the ready queue before it gets allocated the CPU resources, thus showing the responsiveness of the system. Average turn-around time spans the whole duration from the submission of a process to its completion and offers critical information about the overall efficiency of the system.

The formula for the computation of Turnaround and Waiting Time is described in equations (1) and (2) as follows:

Turnaround Time = Completion Time – Arrival Time (1)

Waiting Time = Turnaround Time – Burst Time (2)

The formulas in (3) and (4) below describe how to compute Average Turnaround and Waiting Time, respectively.

$$\text{Average Turnaround Time} = \frac{1}{n} \sum_{i=1}^{n} \text{TAT} \quad (3)$$

$$\text{Average Waiting Time} = \frac{1}{n} \sum_{i=1}^{n} \text{WT} \quad (4)$$

These metrics will be used to evaluate and compare the effectiveness of different scheduling algorithms, which will allow us to optimize process management and resource allocation. A lower mean turnaround time implies faster process completion and better system performance. Likewise, a lower average waiting time reflects better algorithm responsiveness, where the higher the system performance, the lesser the number of process delays. Figure 8 shows the representation of these metrics in the simulation.

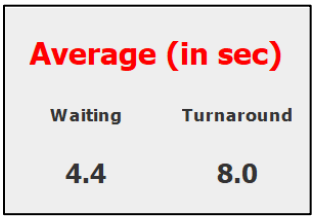


Figure 8. Mean waiting time and mean turnaround time (in seconds)

4. **CPU utilization**
The CPU utilization is a measure of CPU usage efficiency through the execution of a task by measuring, as shown in Figure 9, the ratio of time the CPU spends processing tasks relative to the time it is idle. High CPU usage means the scheduling system efficiently allocates resources so that the CPU is not idle most of the time, thus improving system performance generally.

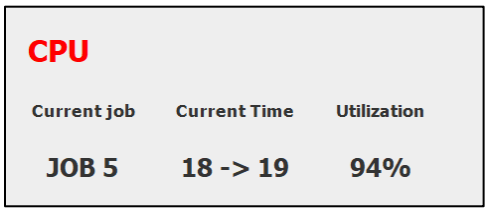


Figure 9. CPU utilization window of simulator

Simulation of Banker's Algorithm

The simulation of the Banker's Algorithm serves to adequately present the necessary requirements for distributing resources and preventing deadlocks. Participants then introduce an input of allocation, maximum, and available resource matrices and then compute the requirement matrix to assess the capacity at which the system can safely allocate resources. The simulator tests the status of the system by producing a safe sequence if the conditions are secure; otherwise, it shall alert the users that the system is deemed unsafe.

The Banker's Algorithm is very relevant to the understanding of deadlock prevention in real multiple-process scenarios. By simulating even complex requests for resources and allocation, users can see how the algorithm ensures that the system will function safely and avoid deadlocks. In addition, simulation also includes invalid resource requests and how resources are depleted, thereby helping users understand other practical challenges encountered in resource allocation.

Within the Banker's Algorithm, the allocation, maximum, and available resource matrices are essential metrics in determining and comparing resource distribution so that processes can be allowed to continue without deadlocks. The following discussion is facilitated by understanding how the algorithm works from the use of a simulator in Figure 10.

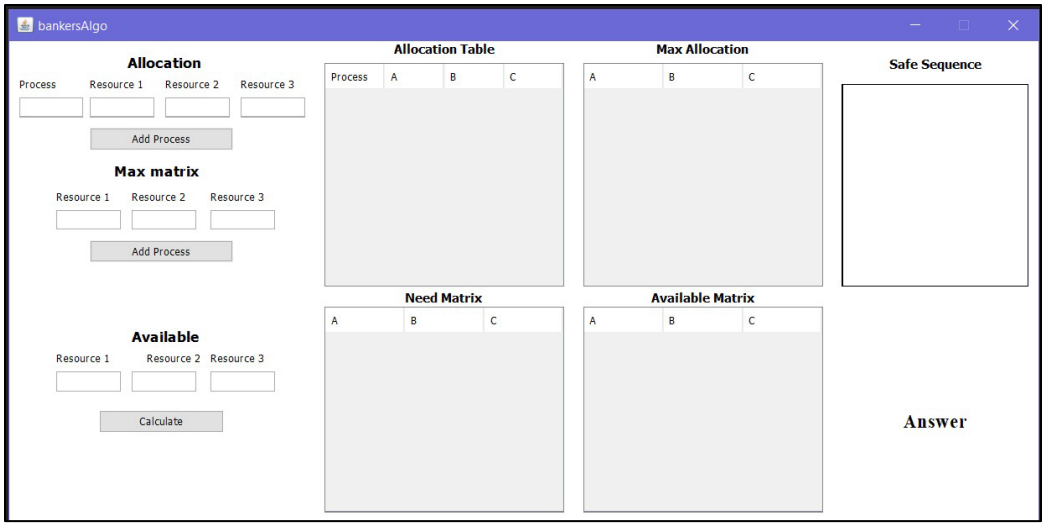


Figure 10. Interface of Banker's Algorithm simulator

Short descriptions of different matrices used in Banker's Algorithms are as follows:

a) *Allocation Matrix:* The allocation matrix, finally, indicates how resources are currently allocated to many activities. Each entry in the matrix indicates the number of units of each of those resources currently allocated to some activity. Essentially, it reflects the resources that are in use by each process at a given moment, as shown in Figure 11.

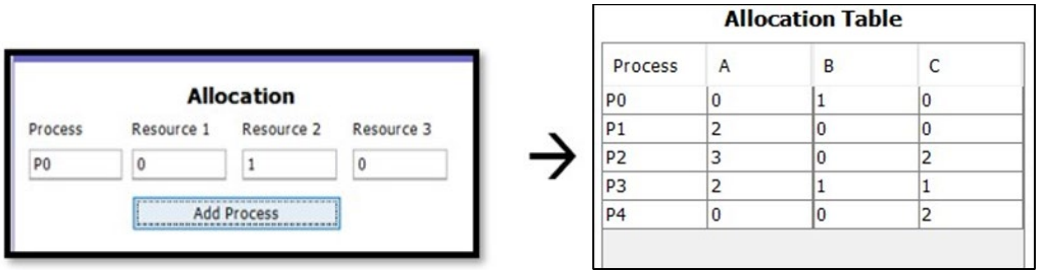


Figure 11. Input for the allocation matrix

To add to the next process, repeat the same procedure. All the processes need to be added one by one, and once added, the Allocation Table looks like it is shown in Figure 12. The matrix form for the allocation matrix is represented as:

$$\text{Allocation} = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1m} \\ A_{21} & A_{11} & \cdots & A_{11} \\ \cdots & \cdots & \ddots & \cdots \\ A_{n1} & A_{n2} & \cdots & A_{nm} \end{bmatrix} \quad (5)$$

where

- A_{ij}: The number of instances of resource type j allocated to process i.
- n: Number of processes.
- m: Number of resource types.

b) *Maximum Matrix:* The maximum matrix defines the maximum demand of each process for each type of resource. This matrix outlines the upper limit on the number of resource units that a process may need to complete its execution, as shown in Figure 12.

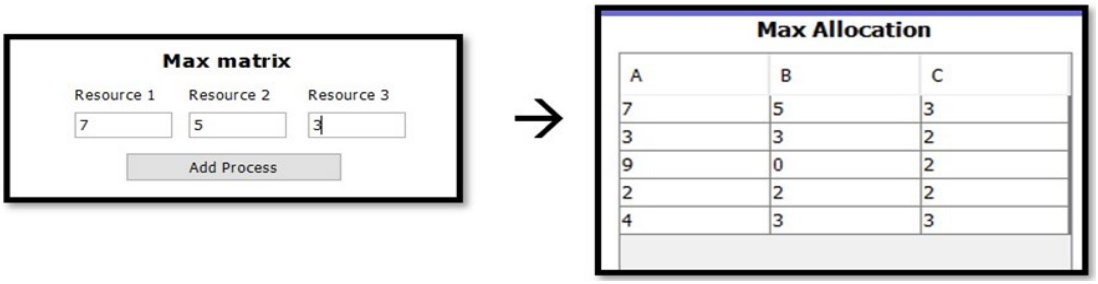


Figure 12. Input for the maximum matrix

The Matrix Form for Maximum Matrix is represented as:

$$Maximum = \begin{bmatrix} M_{11} & M_{12} & \cdots & M_{1m} \\ M_{21} & M_{11} & \cdots & M_{11} \\ \cdots & \cdots & \ddots & \cdots \\ M_{n1} & M_{n2} & \cdots & M_{nm} \end{bmatrix} \quad (6)$$

where

- M_{ij}: The maximum number of instances of resource type j needed by process i.

c) *Available Matrix (or Vector):* The available matrix, typically displayed as a vector, indicates the total number of each resource type currently free for allocation to processes. This is illustrated in Figure 13 and represents the resources not yet allocated.

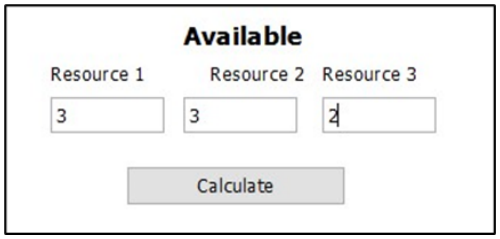


Figure 13. Input to available matrix

Matrix Form for Available Matrix is represented as:

$$\text{Available} = [A1 \quad A2 \quad \cdots \quad Am] \tag{7}$$

where

- A_i: The number of available instances of resource type i.

d) *Need Matrix:* The Need Matrix calculates the remaining resources required by each process to complete its execution. It is derived from the Max and Allocation matrices.

The Need Matrix can be determined using the following formula, as represented in Equation 8.

$$Need_{ij} = M_{ij} - A_{ij} \tag{8}$$

where

- Mij is the Maximum Matrix given input by user
- Aij is the Allocation Matrix given input by user

Based on the evaluation of different metrics, the simulator provides two possible outcomes: "Safely Allocated" and "Unsafe," as illustrated in Figure 14 and Figure 15.

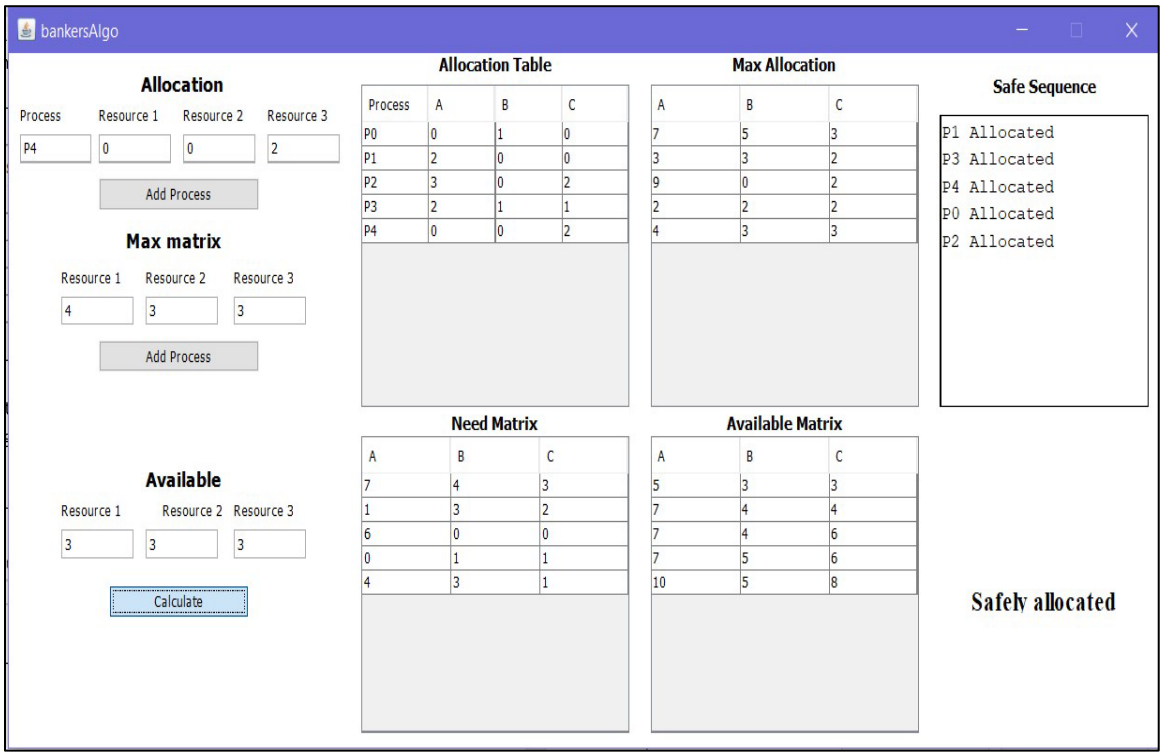


Figure 14. All resources allocated safely

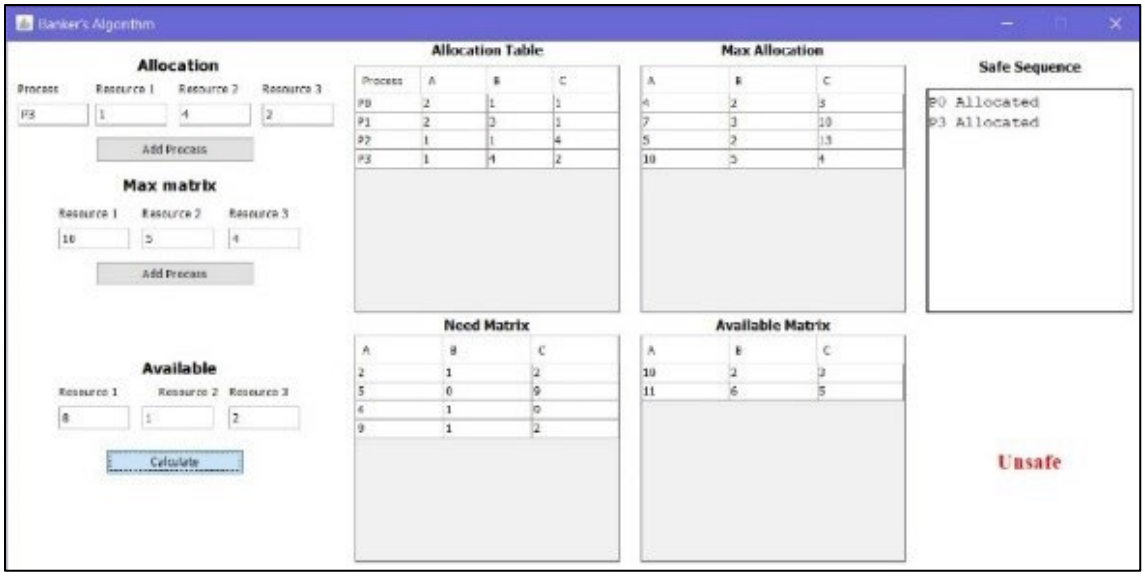


Figure 15. Unsafe sequence allocated

Simulation of disk scheduling algorithms

A disk scheduling algorithm simulation explains the concept through which techniques like FCFS, SSTF, SCAN, C-SCAN, LOOK, and C-LOOK have reduced disk access time. Users can analyze various algorithms for best performance based on the relative effectiveness of scheduling approaches by submitting a set of disk cylinder requests and finding the seek time achieved. This simulation allows users to appreciate the impact of each algorithm on disk head movement and its consequent effect on overall system performance. To perform this simulation successfully, different input parameters required are algorithm to be used, the direction of disk head movement, the sequence of cylinder requests, and the initial and final cylinder positions, as depicted in Figure 16.

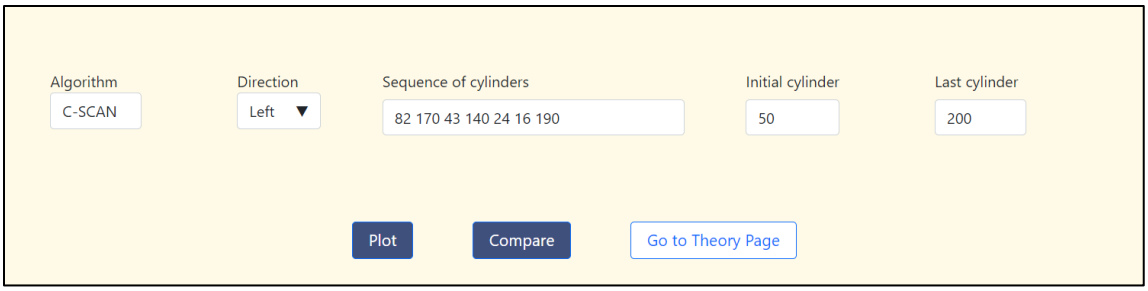


Figure 16. Disk scheduling simulator interface

This disk scheduling part contains six algorithms intended to help the users familiarize themselves:

1. First Come First Served (FCFS): It is the fundamental algorithm that processes the disk requests in the order of their arrival. Though simple, it may at times cause suboptimal head movement and increase seek time significantly due to imbalanced requests.
2. Shortest Seek Time First (SSTF): The SSTF will consider the disk request nearest to the current position of the head; this will reduce seek time and improve efficiency, especially when demands are closely grouped.
3. SCAN: The SCAN algorithm works on the principle of scanning the disk in one direction only by moving the disk arm. It processes requests until a terminal point, after which it reverses its direction of movement. This design offers a more balanced mechanism for handling distributed requests to disks.

4. C-SCAN: The disk arm moves in one direction with a head reset when it reaches the end of the disk in this C-SCAN algorithm. This technique has less variability in the timing of disk requests as the head does not need to be reversed.
5. LOOK directs the arm of the disk toward the nearest pending request and does not overshoot toward the ends of the disk, hence minimizing needless traversal when requests are spread apart.
6. CLOOK extends the principles of LOOK, handling requests within the currently active range of cylinders and restarting immediately at the head of the request queue, therefore further improving head movement as well as general system performance.

These algorithms provide alternative techniques to optimize disk access, and those techniques reduce seek time and can thereby improve system performance based on the nature of the request sequences.

The result for the C-SCAN algorithm, exhibiting disk head movements, is provided in Figure 17. Additionally, a comparison chart for the specified series of cylinder requests is provided, indicating the seek times generated by various disk scheduling algorithms, as illustrated in Figure 18. This comparative analysis reveals the performance disparities among the algorithms based on the calculated seek times.

Figure 17. C-SCAN algorithm seek time calculation

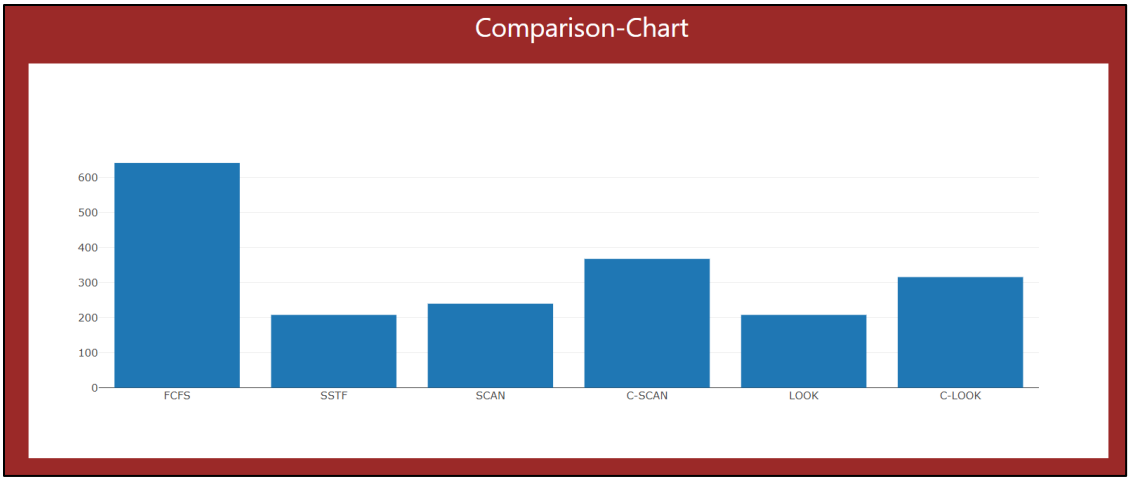


Figure 18. Comparison of various disk scheduling algorithms

Performance Analysis of Process and Disk Scheduling Algorithms

The simulation tool facilitates an in-depth understanding of operating system algorithms by delivering an interactive and hands-on learning experience. By simulating and comparing various process and disk scheduling techniques, users may analyze the real-time impact on important performance indicators such as waiting time and seek time. This hands-on approach makes abstract ideas into real-life consequences, better explains trade-offs and efficiency of various algorithms, and improves learning in the classroom with empirical experimentation and monitoring of performance.

Comparison of different process scheduling algorithms

The sample input to the process scheduling algorithm is given in Table 2.

Table 2. Process scheduling algorithms sample input

Job	Arrival time	Burst time	Priority
1	0	10	3
2	2	4	1
3	4	2	4
4	6	6	2
5	8	8	5

Figure 19 presents the comparison of the average waiting time and turnaround time of various process scheduling algorithms. The graph indicates that SJF and SRTF algorithms significantly reduce waiting time because they favor short jobs. On the other hand, Round Robin tends to increase the waiting time because its context switches are mostly in great numbers, especially if implemented with low quantum time. The two fundamental preemptive and non-preemptive priority algorithms create natural trade-offs. The highest-priority tasks get the advantage of minimum turnaround time, while the lowest-priority tasks endure the maximum waiting time. This comparison analysis enables the learners to comprehend the real-time consequences of various scheduling choices and select the optimal strategy for certain workload scenarios.

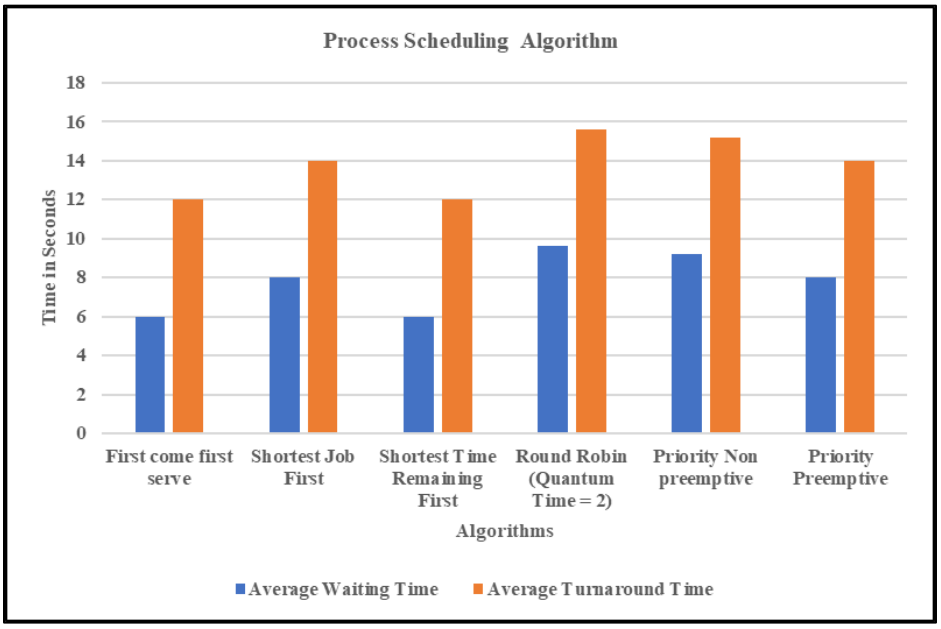


Figure 19. Comparison of average waiting and turnaround time of process scheduling algorithms

Comparison of different disk scheduling algorithms

Disk Scheduling Algorithms Sample Input

Disk Requests: 45, 20, 80, 150, 10, 75, 130.

Initial Head Position: 50.

Total Disk Size: 200 cylinders (assuming disk cylinders range from 0 to 199).

Figure 20 illustrates a comparison of seek times for different disk scheduling strategies. This graph provides students with practical insights into how each algorithm affects disk head movement. The research highlights trade-offs, such as the instant seek time decrease achieved by Shortest Seek Time First (SSTF), whereas SCAN and C-SCAN offer more balanced seek durations across requests. On the other hand, the FCFS scheduling algorithm can cause poor disk head movement and delayed seek times. This experimental study further develops students' knowledge on theoretical grounds by focusing on the practical strengths and weaknesses of each disk scheduling method for easy application of academic know-how in the real world.

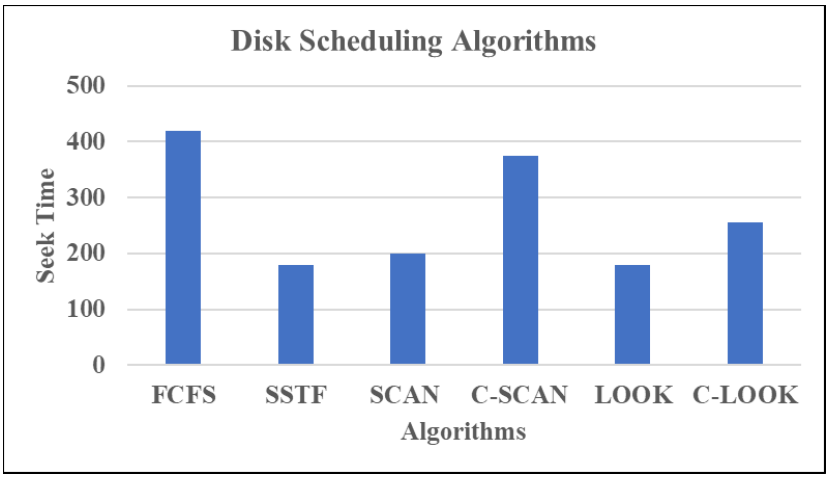


Figure 20. Seek time comparison of various disk scheduling algorithms

IMPACT ANALYSIS

Research studies have extensively been conducted on the effectiveness of constant interaction with e-learning tool settings on the results of student learning. In this tool, numerical exercises and quizzes are incorporated, which equip students with efficient resources to check their understanding of basic concepts. Unlike previous pedagogical techniques for teaching operating system algorithms, the proposed simulation platform enables students to view the execution process and forecast the following steps inside each simulation. Additionally, the incorporation of audio-visual explanations of numerical problems aids problem-solving based on multiple methods, hence boosting conceptual understanding.

The virtual laboratory has proven to be a highly effective tool for lecturers in teaching operating systems courses. It provides a visual and audio-visual application of abstract concepts, which promotes students' understanding of the same. Additionally, the addition of quizzes and numerical exercises allows lecturers to identify whether and to what extent understanding of the material covered is registered in students' heads.

Student learning outcomes were assessed based on feedback from a group of 276 students, and the results are briefly represented in Figure 21. This commentary provides insights into how effective the learning tool is in helping students learn operating system algorithms and areas for further development.

In feedback analysis of simulation parameters, the process scheduling algorithm received the highest rating of 65%, followed by disk scheduling with 58%, and Banker's Algorithm with 71%. For the quiz parameter, the highest rating for the process scheduling algorithm is 70%, followed by disk scheduling at 54% and Banker's Algorithm at 78%. The ratings of the remaining parameters are shown in Figure 21.

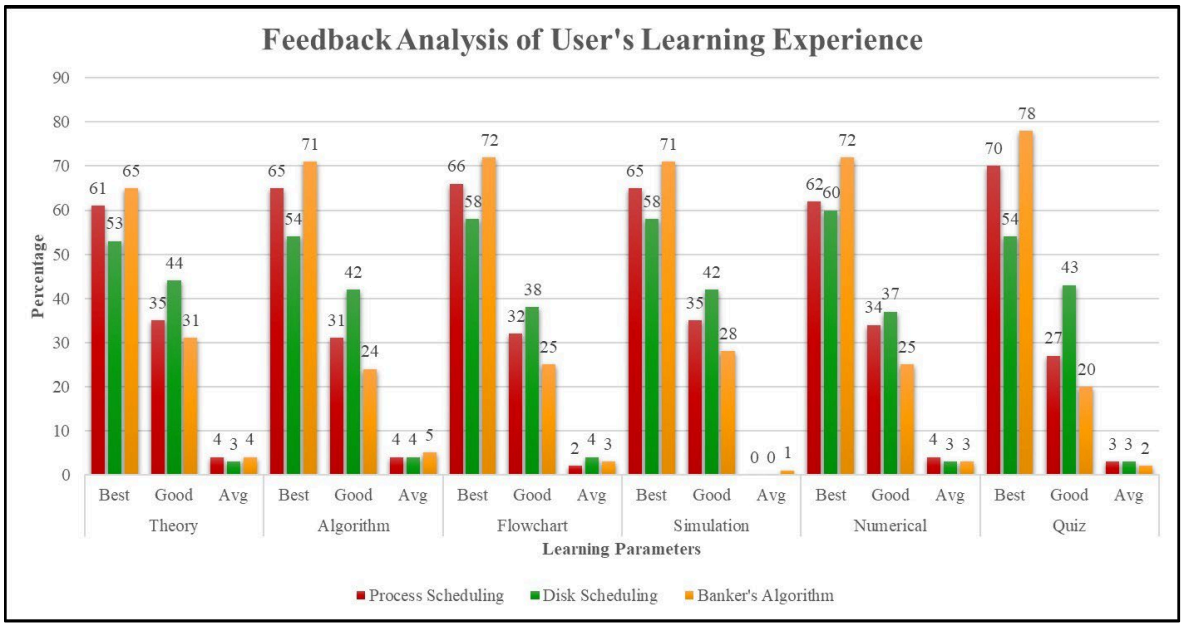


Figure 21. Feedback analysis of user's learning experience

DISCUSSION

The Operating System e-learning tool is a giant leap in educational tools designed to teach complex operating system algorithms. The interactive platform has provided complex simulations for most of the algorithms – process scheduling, disk scheduling, and Banker's Algorithm – that deal with such topics. It will greatly help students learn these central ideas by providing hands-on experience on an interactive platform.

PROCESS SCHEDULING ALGORITHM SIMULATION

The e-learning tool contains a Process Scheduling Simulator that allows users to test with many CPU scheduling algorithms, such as First-Come-First-Serve (FCFS), Shortest Job Next (SJN), Round Robin, and Priority Scheduling. These simulations vividly describe how varying scheduling strategies affect the process scheduling and system performance. Users can upload their own data to really understand the theoretical properties of these methods.

A major aspect of this simulation is its capability to calculate and display waiting and turnaround times for each scheduling method, providing crucial insights into their performance, as illustrated in Figure 22. For instance, Figure 22(a) illustrates that for Job 5, at time 18-19 seconds, the average waiting time is 2.375 seconds, and the turnaround time is 4.75 seconds. Figure 22(b) depicts the ensuing time period, 19-20 seconds, where the average waiting time increases to 2.75 seconds, and the turnaround time rises to 5.25 seconds.

The simulation also estimates CPU consumption, as seen in Figure 23. Figure 23(a) demonstrates that at time 2-3 seconds, CPU utilization is 50%, but Figure 23(b) shows a decline to 33% at time 3-4 seconds. These measures align with previous studies that illustrate the significance of CPU utilization in assessing scheduling methodologies.

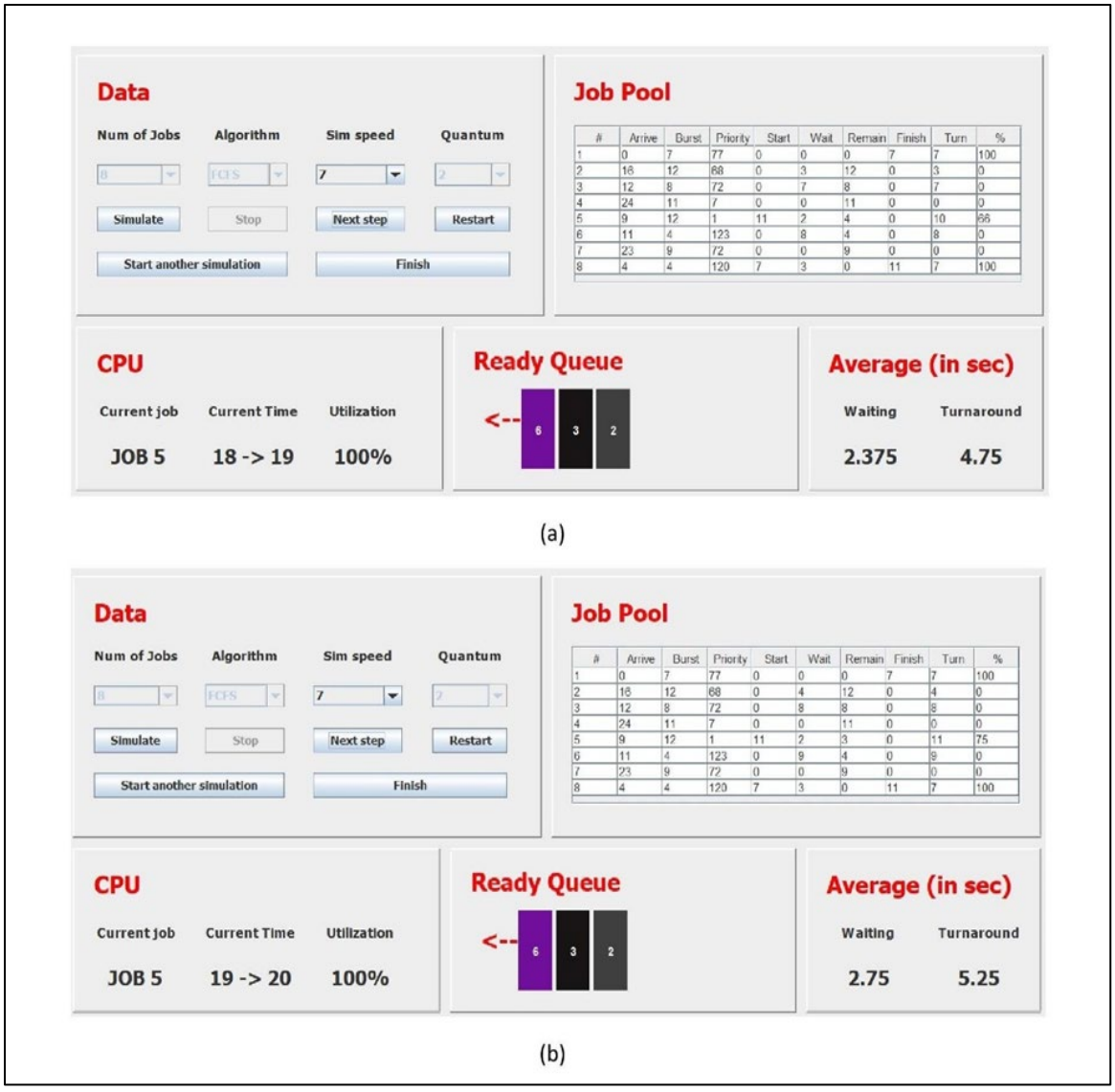


Figure 22. Calculation of waiting time and turnaround time (in seconds) at each step

Figure 23. Calculation of CPU utilization at each step

The simulation also records changes in the ready queue, thus reflecting the real-time process execution and verifying the value of these parameters in enhancing system performance, as is evident in Figure 24. Figure 24(a) illustrates the ready queue at time 11-12 seconds, containing jobs 5, 3, and 6, with Job 1 being executed. Figure 24(b) shows the ready queue at time 12-13 seconds, with jobs 3, 6, and 2 present and Job 5 being executed.

Despite all these achievements, the simulator has trouble accurately capturing the overheads involved with context switching and process synchronization, which can have a deleterious impact on the effectiveness and responsiveness of scheduling algorithms. Future developments should focus on a more subtle capture of these overheads to improve the accuracy of the simulation.

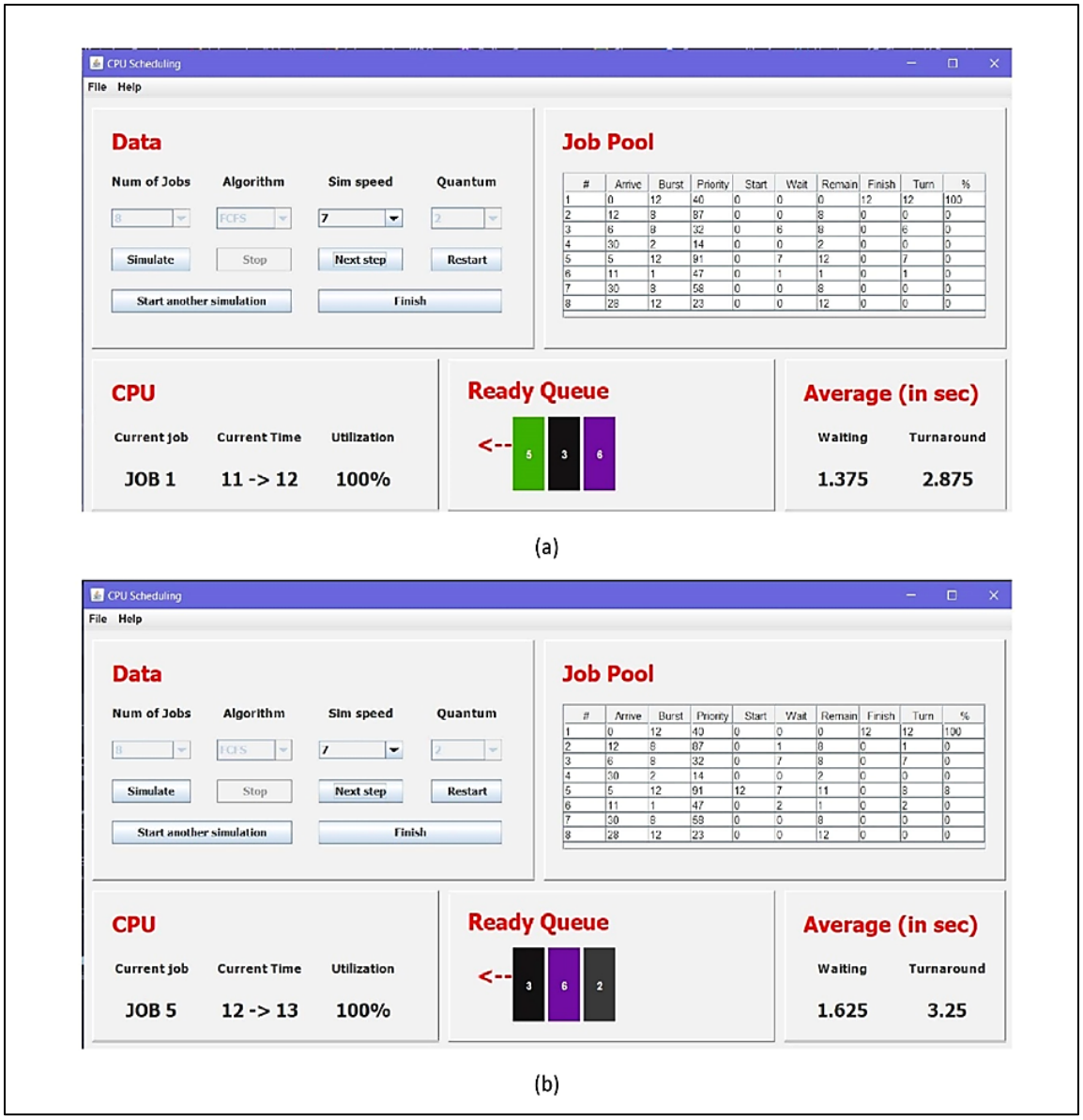


Figure 24. Ready queue for two simultaneous steps

BANKER'S ALGORITHM SIMULATION

The Banker's Algorithm Simulator, therefore, shows clearly how resources are being utilized, and deadlocks prevented using a series of simulations of scenarios using different allocation matrices. This is in consonance with Dijkstra's theoretical model, which puts forward the central role of safe resource allocation as the source of system stability.

The key characteristic of the simulator is its ability to handle faulty resource requests in addition to hazardous sequences, which offer more useful insights into realistic resource management challenges. This feature leads to further research on the administration of resources and the necessity of ensuring suitable methods for error handling.

However, the utility of the simulation might get lost in the ability it has to serve sophisticated scenarios, including various types of resources and shifting request patterns. As such, future work should

focus on developing stronger functionalities toward error handling and resource management as it would better reflect real-world scenarios.

DISK SCHEDULING SIMULATION

The Disk Scheduling Simulator gives a complete investigation of several disk scheduling methods, including FCFS, Shortest Seek Time First (SSTF), SCAN, C-SCAN, LOOK, and C-LOOK. The simulation successfully explains the trade-offs involved in disk access optimization by showing disk arm movements and comparing seek times across different algorithms. The extensive seek-time calculations and comparative analysis offered coincide with research that supports the use of simulation tools for analyzing disk scheduling options. However, the correct representation of real-time disk access behavior and hardware differences are still problems. Future development should focus on furthering in-depth evaluation criteria and various simulation scenarios so that the model is more relevant and accurate.

The operating system e-learning tool is, in general, a giant step in educational technology and provides vital materials for teaching and learning fundamental concepts on operating systems. The virtual lab does contribute to understanding and actualizing methodologies of the operating system very well, apart from other research showing the beneficial aspect of simulation-based learning. Future efforts should be made to perfect the simulation process to strengthen the pedagogic usefulness of simulations.

CONCLUSION AND FUTURE SCOPE

The e-learning tool offers detailed coverage of basic operating system concepts - simulations on process scheduling, disk scheduling, and Banker's Algorithm, among others. The set of various process scheduling algorithms – such as FCFS (First Come, First Serve), SJF (Shortest Job First), Priority – in both pre-emptive and non-pre-emptive variants, Round Robin, and SRTF (Shortest Remaining Time First) allows the students to gain proficiency in process management methods along with understanding the pros and cons which are inherent with each. This framework, like that of disk scheduling algorithms such as FCFS, SSTF, SCAN, CSCAN, LOOK, and CLOOK, enables the students to examine the impacts of changing disk access patterns on the performance to maximize the operations of the disks. Moreover, the experiment on the Banker's Algorithm presents the students with the possibility to expand their understanding of resource allocation as well as the procedure to avoid deadlock, which is one of the needs for system stability and security. This report focuses on the successful implementation and setting up of the learning tool, which contains thirteen separate experiments.

In a word, an e-learning tool for operating systems is a veritable giant step in educational training, remote learning, and developing skills. It reenergizes approaches to teaching through experience, thereby engendering a better understanding of the concepts behind operating systems. Additionally, the laboratory provides the advantage of remote access, thereby expanding educational diversity and functioning as an excellent instrument for bolstering system management abilities and professional competence. The topic of OS simulations has significant promise for future growth. Areas to be extended could be visualized with the simulation of additional algorithms: File Management, Memory Allocation Techniques, and Process Synchronization schemes, including Binary and Counting Semaphores.

The scope for simulations involving operating systems in the future would be with development and growth. The Process scheduling simulator can include a variety of overheads associated with context switching and scheduling algorithms. Disk Scheduling simulators can include evaluation metrics related to fairness and starvation avoidance. Banker's Algorithm simulations can incorporate error-handling scenarios of resource exhaustion, which will explain the behavior of a real-time system to stu-

dents. Similarly, systems using simulations of other algorithms like File Management, Memory Allocation Techniques, and Process Synchronization algorithms like Binary and Counting Semaphores can also be designed.

REFERENCES

Abd El-Haleem, A. M., Eid, M. M., Elmesalawy, M. M., & Hosny, H. A. H. (2022). A generic AI-based technique for assessing student performance in conducting online virtual and remote-controlled laboratories. *IEEE Access*, *10*, 128046-128065. https://doi.org/10.1109/ACCESS.2022.3227505

Altalbe, A. A. (2019). Performance impact of simulation-based virtual laboratory on engineering students: A case study of Australia virtual system. *IEEE Access*, *7*, 177387-177396. https://doi.org/10.1109/ACCESS.2019.2957726

Anish-U. (n.d.). *Banker's algorithm in JavaScript*. https://anish-u.github.io/Bankers-Algorithm-Js/

AssistedCoding. (n.d.-a). *Disk scheduling solver*. https://solver.assistedcoding.eu/disk_scheduling

AssistedCoding. (n.d.-b). *Process scheduling solver*. https://solver.assistedcoding.eu/process_scheduling

Boonsuen. (n.d.). *Process scheduling solver*. https://process-scheduling-solver.boonsuen.com/

Choudhary, S. (n.d.). *Process scheduling solver*. https://shubhamchoudharyshubh.github.io/process-scheduling-solver/

CPU Scheduling Sim. (n.d.). *CPU Scheduling Simulator* https://cpu-scheduling-sim.netlify.app/

Dahan, D., Zarubinsky, M., Liang, Y., Golani, O., & Shtaif, M. (2022). Universal virtual lab: A fast and accurate simulation tool for wideband nonlinear DWDM systems. *Journal of Lightwave Technology*, *40*(8), 2441-2455. https://doi.org/10.1109/JLT.2022.3141447

Do Hoang, H., Duy, P. T., & Pham, V. H. (2022). A method for flexible definition and automatic implementation of laboratory environment in online training platforms. In *2022 RIVF International Conference on Computing and Communication Technologies (RIVF)* (pp. 452-457). IEEE. https://doi.org/10.1109/RIVF55975.2022.10013872

Frady, K. (2022). Use of virtual labs to support demand-oriented engineering pedagogy in engineering technology and vocational education training programmes: A systematic review of the literature. *European Journal of Engineering Education*, *48*(5), 822-841. https://doi.org/10.1080/03043797.2022.2141610

Gamo, J. (2019). Assessing a virtual laboratory in optics as a complement to on-site teaching. *IEEE Transactions on Education*, *62*(2), 119-126. https://doi.org/10.1109/TE.2018.2871617

Garcia, M., Quiroga, J., & Ortin, F. (2021). An infrastructure to deliver synchronous remote programming labs. *IEEE Transactions On Learning Technologies*, *14*(2), 161-172. https://doi.org/10.1109/TLT.2021.3063298

Genkov, D., & Slavov, M. (2021). Implementation of a virtual laboratory for computer oriented disciplines. *Proceedings of the 29th Telecommunications Forum, Belgrade, Serbia*, 1-4. https://doi.org/10.1109/TELFOR52709.2021.9653245

Holovnia, O. S., & Oleksiuk, V. (2022). Selecting cloud computing software for a virtual online laboratory supporting the Operating Systems course. *CTE Workshop Proceedings*, *9*, 216-227. https://doi.org/10.55056/cte.116

Huang, H., & Song, L. (2023). DOSP: Distributed computing-based operating systems labs platform. In *2023 15th International Conference on Computer Research and Development (ICCRD)* (pp. 143-150). IEEE. https://doi.org/10.1109/ICCRD56364.2023.10080655

Jangid, B. (n.d.). *Banker's algorithm simulator*. https://jangidbhanu.github.io/BankersAlgorithm/

Kanwar Adnan. (n.d.). *Banker's algorithm simulator*. https://kanwaradnan.github.io/Bankers-Algorithm-Simulator/

Khushali. (n.d.). *Disk scheduling algorithms*. https://khushalip.github.io/OS-lab/diskAlgo/disk.html

Kleine, K., & Pessot, E. (2023). Virtualising labs in engineering education: A typology for structure and development. *Higher Education Research & Development*, *43*(1), 119–133. https://doi.org/10.1080/07294360.2023.2228227

Luse, A., Brown, A., & Rursch, J. (2021). Instruction in 802.11 technology in online virtual labs. *IEEE Transactions on Education*, *64*(1), 12-17. https://doi.org/10.1109/TE.2020.2998701

Makaroff, D. (n.d.). *Disk scheduling simulator.* https://www.cs.usask.ca/faculty/makaroff/cgi-bin/disk_sched.pl

Martin-Villalba, C., & Urquia, A. (2022). An approach to develop collaborative virtual labs in Modelica. *IEEE Access*, *10*, 58938-58949. https://doi.org/10.1109/ACCESS.2022.3179712

May, D., Morkos, B., Jackson, A., Hunsu, N. J., Ingalls, A., & Beyette, F. (2022). Rapid transition of traditionally hands-on labs to online instruction in engineering courses. *European Journal of Engineering Education*, *48*(5), 842-860. https://doi.org/10.1080/03043797.2022.2046707

McGarr, O. (2020). The use of virtual simulations in teacher education to develop pre-service teachers' behaviour and classroom management skills: Implications for reflective practice. *Journal of Education for Teaching*, *47*(2), 274-286. https://doi.org/10.1080/02607476.2020.1724654

Panasiuk, O., Akimova, L., Kuznietsova, O., & Panasiuk, I. (2021). Virtual laboratories for engineering education. *Proceedings of the 11th International Conference on Advanced Computer Information Technologies, Deggendorf, Germany,* 637-641. https://doi.org/10.1109/ACIT52158.2021.9548567

Pisqre. (n.d.-a). Banker's algorithm calculator. https://calculator.pisqre.com/bankers

Pisqre. (n.d.-b). Calculator for various algorithms. https://calculator.pisqre.com/

Rassudov, L., & Korunets, A. (2022). Virtual labs: An effective engineering education tool for remote learning and not only. *Proceedings of the 29th International Workshop on Electric Drives: Advances in Power Electronics for Electric Drives, Moscow, Russian Federation* (pp. 1-4). IEEE. https://doi.org/10.1109/IWED54598.2022.9722375

Reginald, G. (2023). Teaching and learning using virtual labs: Investigating the effects on students' self-regulation. *Cogent Education, 10*(1). https://doi.org/10.1080/2331186X.2023.2172308

Sáenz, J., de la Torre, L., Chacón, J., & Dormido, S. (2021). A study of strategies for developing online laboratories. *IEEE Transactions on Learning Technologies, 14*(6), 777-787. https://doi.org/10.1109/TLT.2022.3145807

Salas, R. P., & Ho, J. (2021). A remote/virtual robotics lab. *Proceedings of the IEEE Frontiers in Education Conference, Lincoln, NE, USA,* 1-4, https://doi.org/10.1109/FIE49875.2021.9637340

SeekTime. (n.d.). Disk Scheduler Visualisation. https://www.seektime.app/

Vayadande, K., Sheth, P., Pawal, D., Pathak, A., Paralkar, K., & Patil, S. (2023). Simulation of CPU scheduling algorithms for efficient execution of processes. *Proceedings of the International Conference for Advancement in Technology, Goa, India,* 1-6. https://doi.org/10.1109/ICONAT57137.2023.10080113

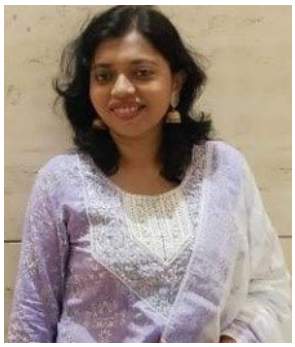
Wang, L., Zhen, Z., Wo, T., Jiang, B., Sun, H., & Long, X. (2020). A scalable operating system experiment platform supporting learning behavior analysis. *IEEE Transactions on Education*, *63*(3), 232-239. https://doi.org/10.1109/TE.2020.2975556

Wang, Y., Li, M., Dai, H., Kent, K. B., Ye, K., & Xu, C. (2022). Deadlock avoidance algorithms for recursion-tree modeled requests in parallel executions. *IEEE Transactions on Computers*, *71*(9), 2073-2087. https://doi.org/10.1109/TC.2021.3122843

XiaoLing, Y. (2019). A deadlock prevention algorithm for the two-phase locking protocol based on Petri Net. *Proceedings of the 2019 6th International Conference on Systems and Informatics (ICSAI)* (pp. 889-892). IEEE. https://doi.org/10.1109/ICSAI48974.2019.9010538

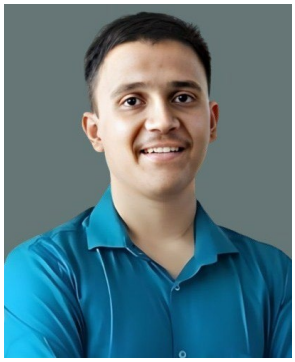
Zapata-Rivera, L. F., & Aranzazu-Suescun, C. (2020). Enhanced virtual laboratory experience for wireless networks planning learning. IEEE *Revista Iberoamericana de Tecnologias del Aprendizaje, 15*(2), 105-112. https://doi.org/10.1109/RITA.2020.2987725

AUTHORS



Dr Jyoti Wadmare is an Assistant Professor in the Computer Engineering Department at KJSIT. She has teaching experience of 17 years with an AI background. Her major domain of interest is the conjunction of AI and Computer Vision. Testimonials of work include many conference presentations and articles published that state advancement in this area by her, and she has filed a patent and acquired four copyrights.

Dakshita Kolte is studying for a B.Tech in Computer Engineering at KJSIT. She has developed Artificial Intelligence Machine Learning solutions that combine Artificial Intelligence (AI) and web-based technologies. She has a track record for participating in some of the most renowned competitions, such as Mastek Project Deep Blue, Aavishkar, and Creative Ideas and Innovations in Action. She has four copyrights for her work.



Kapil Bhatia, who is working on artificial intelligence and machine learning, has opted for a B. Tech in Computer Engineering at KJSIT. Kapil Bhatia is adept at developing solutions that are the interplay of web-based technologies, the Internet of Things, and artificial intelligence. His participation in well-known contests such as Aavishkar, Mastek Project Deep Blue, and Creative Ideas and Innovations in Action speak volumes about his exceptional expertise. He has also bagged four copyrights on his work.

Palak Desai is a third-year Computer Engineering student passionate about UI/UX design, front-end web development, and data analytics. She enjoys creating intuitive, beautiful user interfaces, bringing them to life with functional web development, and analyzing data to drive insights. She is dedicated to continuous learning and making impactful, user-centered solutions. She has been granted three copyrights for her work.

Ganesh Wadmare is an assistant professor in the Artificial Intelligence and Data Science department of KJSIT and a Ph.D. scholar at Savitribai Phule Pune University, with over 19 years of academic experience. Ganesh has extensive exposure and experience in the field of artificial intelligence and renewable sources of energy. He has published his research papers in both national and international conferences.