OSSP LAB PROJECT

ANALYZING AND VISUALIZING VARIOUS PROCESS SCHEDULING ALGORITHMS

PROJECT REPORT

Department of CSE & IT

Jaypee Institute of Information Technology, Noida

SUBMITTED BY:

Enrollment numbers	21103042	21103043	21103048	21103379
Names of students	Astha Raghuwanshi	Prerna	Princi Agrawal	Tanya Gupta
Batch	B2			

GENERAL INTRODUCTION

What is CPU Process Scheduling and Why is it Needed?

CPU Scheduling is a process that allows one process to use the CPU while another process is delayed (in standby) due to unavailability of any resources such as I / O etc, thus making full use of the CPU. The purpose of CPU Scheduling is to make the system more efficient, faster, and fairer. CPU Process scheduling is important because:

• The main function of the CPU scheduling is to ensure that whenever the CPU remains idle, the OS has at least selected one of the processes available in the ready-to-use line. • In Multiprogramming, if the long-term scheduler selects multiple I / O binding processes then most of the time, the CPU remains an idle.

• The function of an effective program is to improve resource utilization. If most operating systems change their status from performance to waiting then there may always be a chance of failure in the system.

• So in order to minimize this excess, the OS needs to schedule tasks in order to make full use of the CPU and avoid the possibility of deadlock.

CONCEPTS USED

CPU Scheduling Algorithms:

● First Come First Serve (FCFS): Tasks are executed in the order they arrive, like customers in a queue. It's simple but might cause delays for lengthy tasks if they arrive early.

● Shortest Job First (SJF): The task with the smallest execution time is chosen first. This minimizes waiting time but can be unpredictable if task lengths vary.

● Shortest Remaining Job First (SRTF): Similar to SJF, but tasks can be interrupted by shorter tasks. This reduces waiting time but adds complexity.

● Round Robin (RR): Tasks take turns to execute in fixed time slices. It's fair, but longer tasks might still wait a bit before getting their turn again.

● Priority Scheduling (Non-Preemptive and Preemptive): Tasks are assigned priorities, and the one with the highest priority goes first. Preemptive means tasks can be interrupted, non-preemptive does not allow interruptions.

Web Development:

● HTML for Structure: HTML (Hypertext Markup Language) provides the basic structure for web content, defining elements like headings, paragraphs, and lists.

● CSS for Styling: CSS (Cascading Style Sheets) styles HTML elements, controlling layout, colors, and fonts to make the web page visually appealing and user-friendly.

● JavaScript for Interactivity: JavaScript adds interactivity to the webpage. In this project, it's used to handle user inputs, dynamically update content, and make the application responsive.

FEATURES BUILT

Algorithm Selection: The user can choose from a variety of CPU scheduling algorithms, including FCFS, SJF, SRTF, Round Robin, Priority (Non-Preemptive), and Priority (Preemptive).

Interactive Process Input: Users can input process details such as Process ID, Priority, Arrival Time, and Burst Time through a user-friendly interface.

Dynamic Process Addition and Removal: The application allows users to dynamically add or remove processes, providing flexibility for experimentation.

Context Switching and Time Quantum: Users can specify context switch times and time quantum for algorithms that require them, such as Round Robin.

Priority Preference Toggle: Users can set priority preferences, toggling between "High" and "Low" priority.

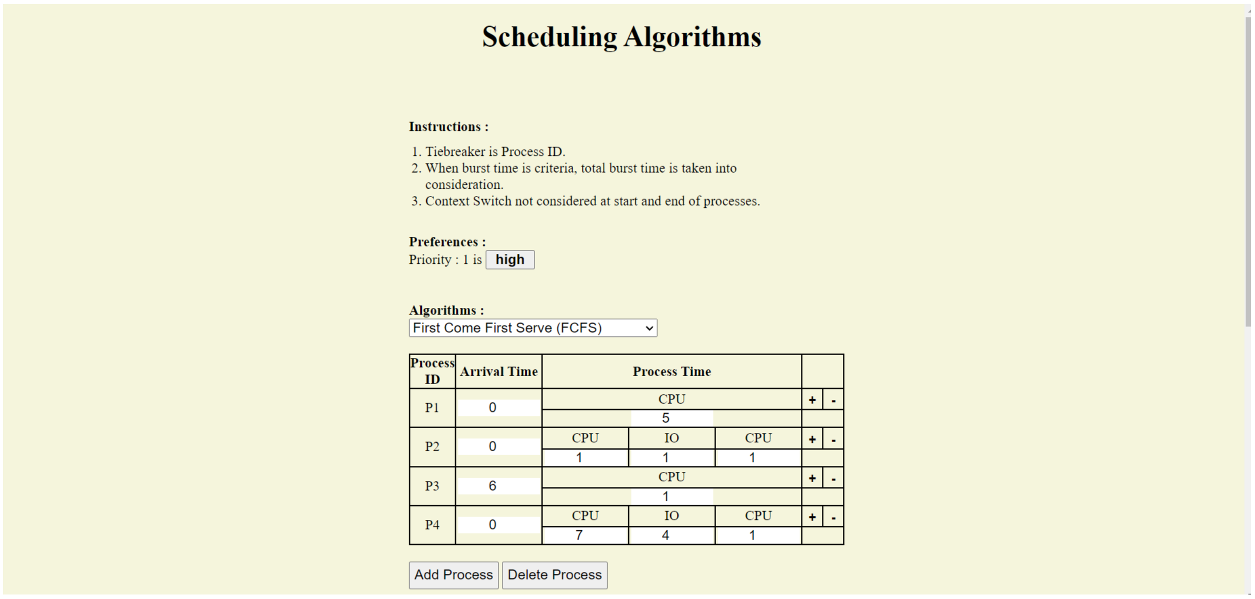
Clear Visualization: The application uses tables and charts to provide clear visualisations of the scheduling outcomes.

Gantt Chart: Gantt Chart Data Preparation: The Gantt chart data is prepared by iterating through the schedule generated during CPU scheduling.Different elements like processes, context switches, and idle time are represented with corresponding colours

Final Table: Creating the Table: A table is created to display detailed information for each process, including arrival time, total burst time, completion time, turn around time, waiting time, and response time.

Reset Functionality: The application includes a reset button to easily clear inputs and start afresh.

CODE OUTPUT



Scheduling Algorithms

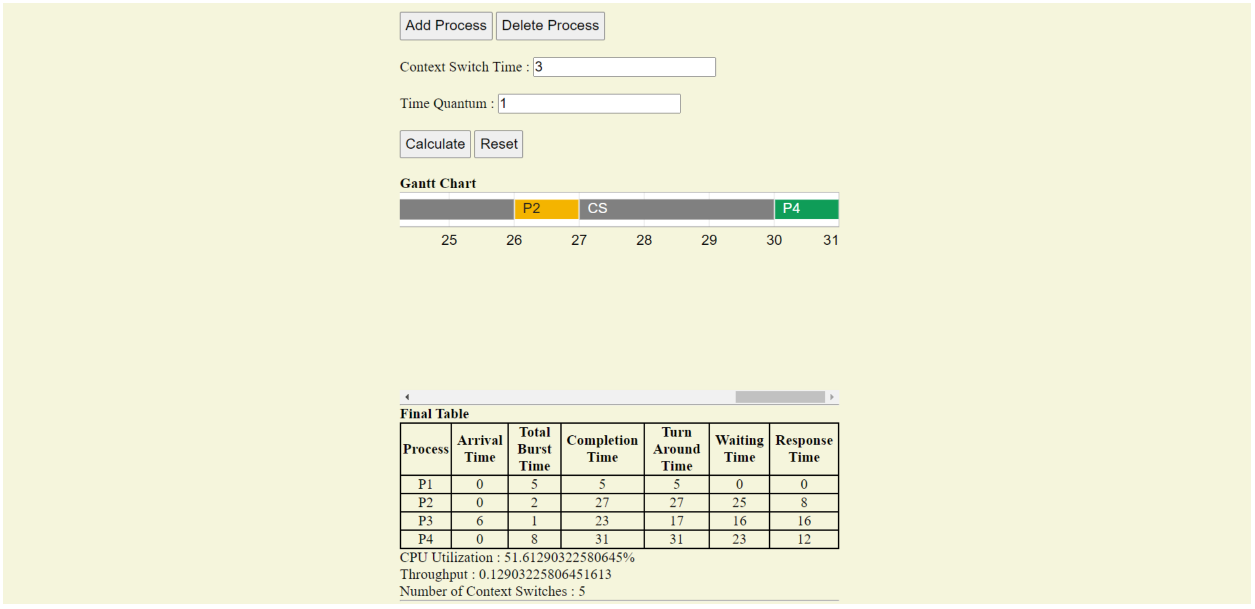
Instructions :

1. Tiebreaker is Process ID.
2. When burst time is criteria, total burst time is taken into consideration.
3. Context Switch not considered at start and end of processes.

Preferences :
Priority : 1 is high

Algorithms :
First Come First Serve (FCFS)

Process ID	Arrival Time	Process Time			
P1	0	CPU			+ -
		5			
P2	0	CPU	IO	CPU	+ -
		1	1	1	
P3	6	CPU			+ -
		1			
P4	0	CPU	IO	CPU	+ -
		7	4	1	



Context Switch Time : 3

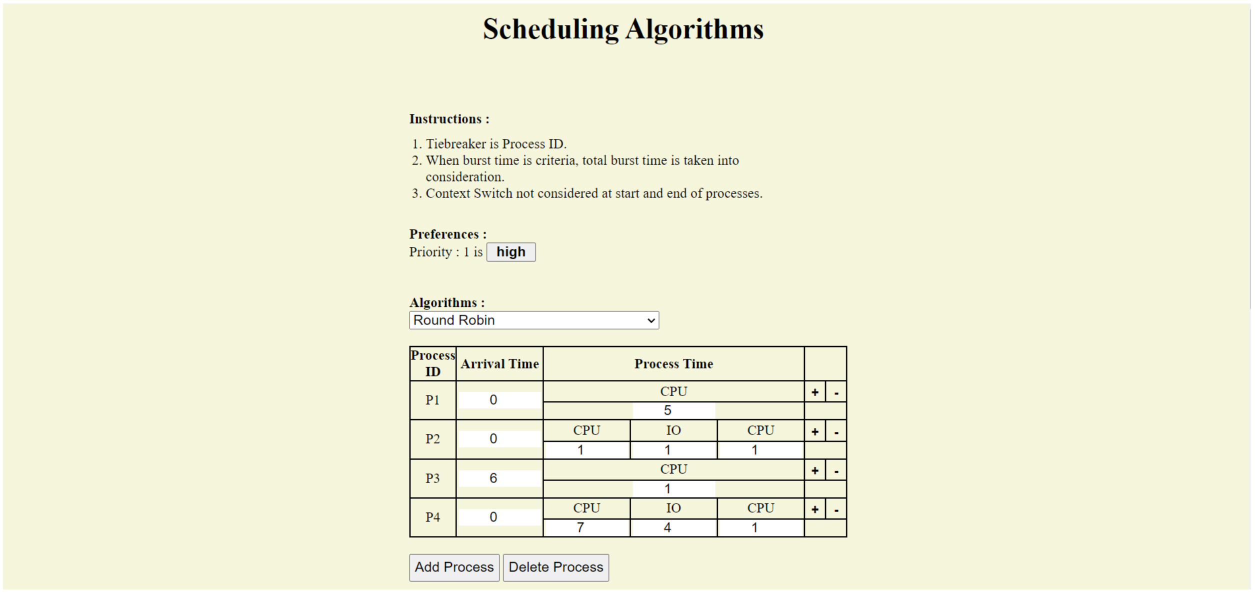
Time Quantum : 1

Gantt Chart

Final Table

Process	Arrival Time	Total Burst Time	Completion Time	Turn Around Time	Waiting Time	Response Time
P1	0	5	5	5	0	0
P2	0	2	27	27	25	8
P3	6	1	23	17	16	16
P4	0	8	31	31	23	12

CPU Utilization : 51.61290322580645%
Throughput : 0.12903225806451613
Number of Context Switches : 5



Scheduling Algorithms

Instructions :

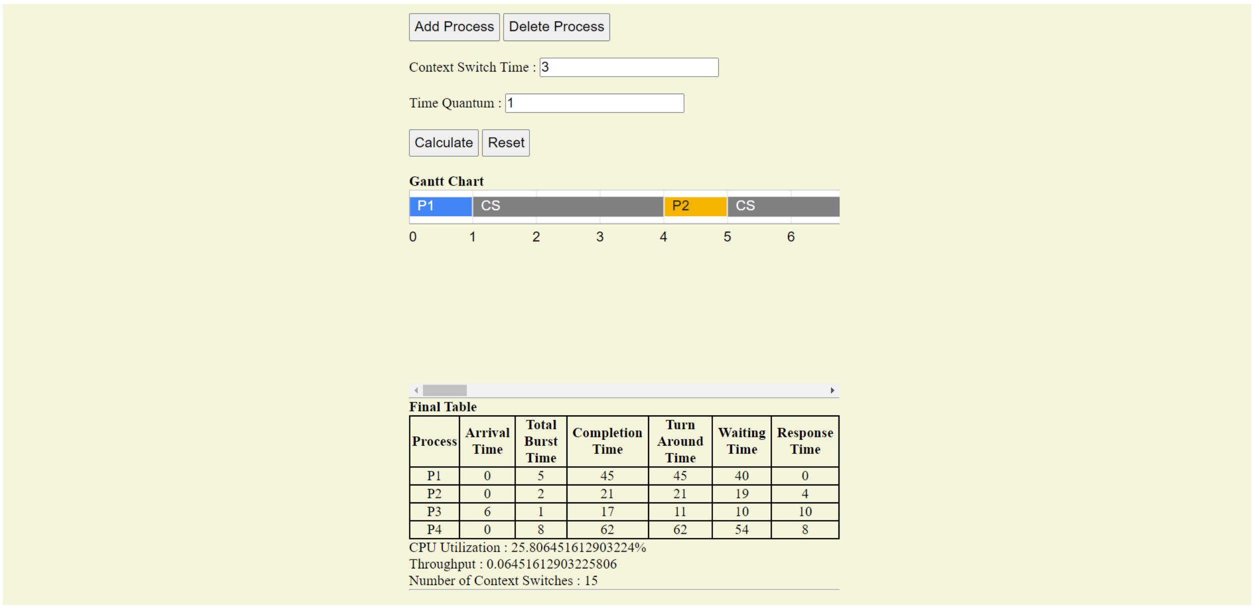
1. Tiebreaker is Process ID.
2. When burst time is criteria, total burst time is taken into consideration.
3. Context Switch not considered at start and end of processes.

Preferences :
Priority : 1 is high

Algorithms :
Round Robin

Process ID	Arrival Time	Process Time				
P1	0	CPU			+	-
		5				
P2	0	CPU	IO	CPU	+	-
		1	1	1		
P3	6	CPU			+	-
		1				
P4	0	CPU	IO	CPU	+	-
		7	4	1		

Add Process Delete Process



Add Process Delete Process

Context Switch Time : 3

Time Quantum : 1

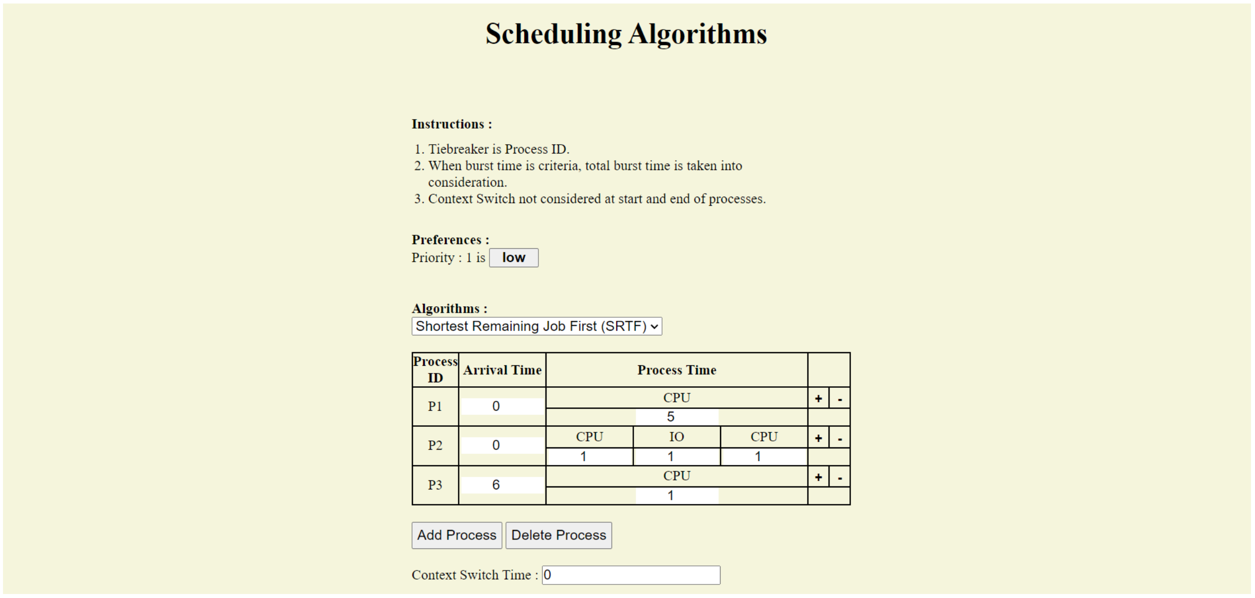
Calculate Reset

Gantt Chart

Final Table

Process	Arrival Time	Total Burst Time	Completion Time	Turn Around Time	Waiting Time	Response Time
P1	0	5	45	45	40	0
P2	0	2	21	21	19	4
P3	6	1	17	11	10	10
P4	0	8	62	62	54	8

CPU Utilization : 25.806451612903224%
Throughput : 0.06451612903225806
Number of Context Switches : 15



Scheduling Algorithms

Instructions :

1. Tiebreaker is Process ID.
2. When burst time is criteria, total burst time is taken into consideration.
3. Context Switch not considered at start and end of processes.

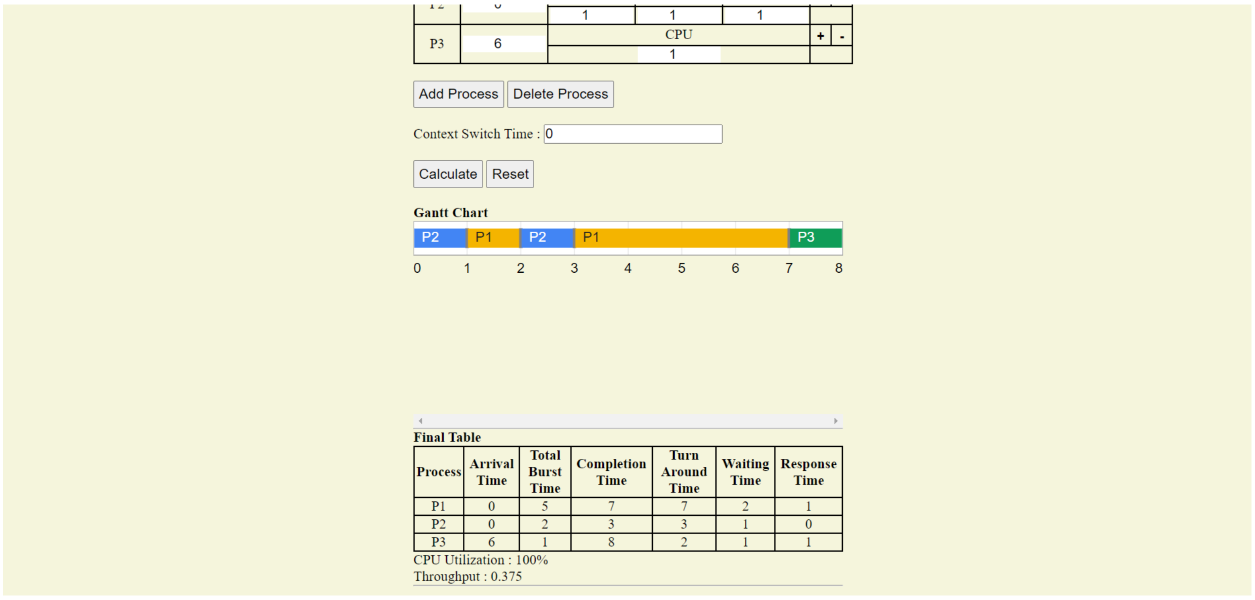
Preferences :
Priority : 1 is low

Algorithms :
Shortest Remaining Job First (SRTF)

Process ID	Arrival Time	Process Time			
P1	0	CPU			+ -
		5			
P2	0	CPU	IO	CPU	+ -
		1	1	1	
P3	6	CPU			+ -
		1			

Add Process Delete Process

Context Switch Time : 0



		1	1	1	
P3	6	CPU			+ -
		1			

Add Process Delete Process

Context Switch Time : 0

Calculate Reset

Gantt Chart

Final Table

Process	Arrival Time	Total Burst Time	Completion Time	Turn Around Time	Waiting Time	Response Time
P1	0	5	7	7	2	1
P2	0	2	3	3	1	0
P3	6	1	8	2	1	1

CPU Utilization : 100%
Throughput : 0.375

CONCLUSION

In conclusion, our project created a website to help people understand how computers manage tasks. The website lets users see different ways tasks are handled in real-time. This hands-on tool improves understanding and allows users to compare methods. The website is easy to use on any device, making it accessible for everyone. It also fits well with modern ways of learning online. In addition, our interactive tool engages users actively, letting them experiment with various scenarios and gain practical insights. By showcasing the strengths and weaknesses of different scheduling methods, it not only enhances theoretical knowledge but also sharpens problem-solving skills. Ultimately, our project aims to empower learners with not just theoretical knowledge but also practical skills, contributing to a deeper understanding of CPU scheduling algorithms in the realm of computer science.