Bhavishay Bharti
22115040
Pre-Final Year (EE), IIT Roorkee
bhavishay_b@ee.iitr.ac.in

CPU Scheduler

20th June 2024

OVERVIEW

A CPU Scheduler is a crucial component of an operating system responsible for managing the order in which programs are executed. It divides the CPU time between processes ensuring efficient resource utilization. It works on the principle of time sharing in the CPU which is a part of virtualization of CPU, virtualization being an important aspect of Operating Systems.

GOALS

1. Understanding various scheduling algorithms and their implementation details and best use cases.
2. Implementation of a fairly sophisticated scheduler which can schedule tasks when it is given the information of tasks.
3. Frontend should communicate the optimality of the decision made by the scheduler through various statistics.

BACKGROUND

Processes are basically stored programs loaded into memory at the time of execution.

There may be multiple processes waiting to be run on a modern day computer, not to mention different processes require different resources and different time on the CPU. CPU scheduling algorithms come into play when it is time to decide which process should be executed next on the CPU. The purpose of scheduling is to make the system more efficient, faster and fairer, also improving the response time of the system.

THE PROPOSED ALGORITHM

The mentioned algorithm is a combination of Round Robin Algorithm, Shortest Job First and Priority Algorithm.

WORKING

Step 1

Push all the processes into the ready queue, assuming all the processes arrived simultaneously (arrival time = 0). Calculate the time quantum using the formula mentioned below:

Ready Queue

P1	P2	P3	P4	P5	P6	P7	P8	P9	P10

$$\text{Time quantum} = \sqrt{\frac{\text{Sum of burst time} * \text{maximum burst time}}{\text{number of process in ready queue}}}$$

Step 2

Arrange all the processes on the basis of their burst time, the shortest process being ranked first. We will call this the burst time priority.

Process	Burst time	Burst time priority
P1	80	7
P2	60	4
P3	65	5
P4	120	10
P5	30	2
P6	90	8
P7	25	1
P8	40	3
P9	90	9
P10	75	6

Step 3

Now calculate the adjusted priority for each process, whose formula is ¾*priority + ¼ * Burst time Priority.

$$\text{Adjusted priority} = \frac{3}{4} * \text{priority rank} + \frac{1}{4} * \text{burst time priority}$$

Process	Burst time	Priority	Burst time priority	adjusted priority
P1	80	1	7	2.5
P2	60	2	4	2.5
P3	65	3	5	3.5
P4	120	4	10	5.5
P5	30	5	2	4.25
P6	90	6	8	6.5
P7	25	7	1	5.5
P8	40	8	3	6.75
P9	90	9	9	9
P10	75	10	6	9

Step 4

Using the adjusted priority, order all the processes again. Shortest process gets ranked first.

Process	Burst time	Priority	Burst time priority	adjusted priority	adjusted priority rank
P1	80	1	7	2.5	1
P2	60	2	4	2.5	2
P3	65	3	5	3.5	3
P4	120	4	10	5.5	5
P5	30	5	2	4.25	4
P6	90	6	8	6.5	7
P7	25	7	1	5.5	6
P8	40	8	3	6.75	8
P9	90	9	9	9	9
P10	75	10	6	9	10

Step 5

Make a new queue, and push the processes on the basis of their adjusted priority rank. Start executing the process using the round robin algorithm with the appropriate time quanta.If any process does not get executed completely (i.e. burst time of the process is

greater than the calculated time quanta), then push the process to the end of the ready queue and repeat the process again from step I for the remaining processes till the ready queue becomes empty.

Before first execution

Ready Queue

P1	P2	P3	P5	P4	P7	P6	P8	P9	P10

Time Quantum = 90

After execution

Ready Queue

P4

Time Quantum = 30

THE BACKEND

CGI

Common gateway Interface is an effective way to enable dynamic content generation and server-side processing. CGI serves as a bridge between the web server and C++ allowing user inputs from the HTML form to be processed by the C++ application.

REFERENCES

1. Introduction to scheduling Algorithms https://www.youtube.com/playlist?list=PLBlnK6fEyqRitWSE_AyyySWfhRgyA-rHk
2. Operating Systems: Three Easy Pieces by Arpaci-Dusseau (University of Madison-Wisconsin)