Team 04

**Software Requirement Specification (SRS)**

Introduction

This project reads the details of Jobs (Process Id, Arrival Time, Burst Time) and stores it in array of structures and writes it to file.

From that file we take the values for the process schedulers and compare the average wait time and select the best algorithm

A Process Scheduler schedules different processes to be assigned to the CPU based on particular scheduling algorithms. In this project we have used the following algorithms for process scheduling:

* First-Come, First-Served (FCFS) Scheduling
* Priority Scheduling
* Round Robin (RR) Scheduling
* Shortest Job First (SJF)

These algorithms are either non-preemptive or pre-emptive. Non-preemptive algorithms are designed so that once a process enters the running state, it cannot be pre-empted until it completes its allotted time, whereas the pre-emptive scheduling is based on priority where a scheduler may pre-empt a low priority running process anytime when a high priority process enters into a ready state.

**Scope of this project:**

In this project, we will learn about the most popular scheduling algorithms like First Come First Serve (FCFS), Shortest Job First (SJF), Priority Scheduler, and Round Robin (RR). We will talk about how these algorithms work and schedule the processes in the CPU. This project mainly focuses on the explanation of the scheduling algorithms, and some of their functions on coding level.

Contents

1. Chat Server-Read Jobs Input from User
2. First come first serve (FCFS)
3. Shortest Job First (SJF)
4. Round Robin Algorithm (RR)
5. Priority Scheduling
6. Comparing Scheduling Algorithms
7. **Chat Server – Read Jobs Input from User**
   1. Creation of socket

We create a socket connection between Client and Server with respect to port family and address.

* 1. Connection of Client to Server
  2. Process selection

1. Exit
2. FCFS
3. Priority
4. Round Robin
5. SJF
   1. After selecting a process, we give the input for (Pid, Burst Time, Arrival Time, Time Quantum, Priority) each process.
   2. Client reads all the values from the server for each process outcome
   3. Socket closes
6. **First come first serve (FCFS)**

FCFS is the easiest and simplest CPU scheduling algorithm. In this type of algorithm, processes which requests the CPU first get the CPU allocation first. This is managed with a FIFO queue.

* Input the processes along with their burst time (bt).
* Find waiting time for all processes.
* As first process that comes need not to wait so waiting time for process 1 will be 0 i.e. wt[0] = 0.
* Find waiting time for all other processes i.e., for process i -> wt[i] = bt[i-1] + wt[i-1].
* Find turnaround time = waiting\_time + burst\_time for all processes.
* Find average waiting time = total\_waiting\_time / no\_of\_processes.
* Similarly, find average turnaround time = total\_turn\_around\_time / no\_of\_processes.
  1. Takes input of all the data from the client and stores it into an array of structure (Pid, Arrival Time, Burst Time)
  2. Applies the algorithm to the inputs taken from the client side
  3. Returns the output (Turn Around Time, Waiting Time, Average Waiting Time) for all the processes

1. **Shortest Job First (SJF)**

Shortest Job First is an algorithm in which the process having the smallest execution time is chosen for the next execution. This scheduling method can be pre-emptive or non-preemptive. It significantly reduces the average waiting time for other processes awaiting execution. The full form of SJF is Shortest Job First.

Algorithm:

* Sort all the process according to the arrival time.
* Then select that process which has minimum arrival time and minimum Burst time.
* After completion of process make a pool of process which after till the completion of previous process and select that process among the pool which is having minimum Burst time.
  1. Takes input of all the data from the client (Pid, Arrival Time, Burst Time) and stores it into a structure of array
  2. Applies the algorithm for SJF on the data
  3. Returns the output (Turn Around Time, Waiting Time, Average Waiting Time) for all the processes

1. **Round Robin Algorithm (RR)**

The name of this algorithm comes from the round-robin principle, where each person gets an equal share of something in turns. It is the oldest, simplest scheduling algorithm, which is mostly used for multitasking.

In Round-robin scheduling, each ready task runs turn by turn only in a cyclic queue for a limited time slice. This algorithm also offers starvation free execution of processes

Algorithm:

* Ask number of processes, arrival time, CPU burst time, time quanta from the user
* Sort the processes in order of arrival time in ascending order
* Use simple FIFO queue
* Push the first process from the sorted list (step 2) into this queue
* Use an array to check if the process is in queue or not -> visited [ ]
* Keep track of the time using a variable -> current\_time
* If the process unit of time to the process that is at front in the queue and pop the process from the queue
* Give quantum unit of time to the process that is at the front of the queue and pop this process from the queue
* If the burst time of the process becomes 0, calculate CT, TAT, WT and RT for it
* If some processes have arrived when this process was executing, insert that arrived processes into the queue
* If the current process has burst time remaining, push the process into queue again
* If the queue is empty, pick the first process from the list that is not completed
* Keep doing this till all processes are completed
  1. Takes input of all the data from the client (Pid, Arrival Time, Burst Time) and stores it into a structure of array
  2. Applies the algorithm for Round Robin on all the data
  3. Returns the output (Turn Around Time, Waiting Time, Average Waiting Time) for all the processes

1. **Priority Scheduling**

Priority Scheduling is a method of scheduling processes that is based on priority. In this algorithm, the scheduler selects the tasks to work as per the priority (given by the user in this case).

The processes with higher priority should be carried out first, whereas jobs with equal priorities are carried out on a round-robin or FCFS basis. Priority depends upon memory requirements, time requirements, etc.

Algorithm:

* It is a scheduling algorithm that schedules the incoming processes on the basis of the priority
* Operating systems use it for performing batch processes
* If there exist two jobs / processes in the ready state (ready for execution) that have the same priority, then priority scheduling executed the processes on first come first serve basis. For every job that exists, we have a priority number assigned to it that indicates its priority level
* If the integer value of the priority number is low, it means that the process has higher priority. (Low number = high priority)
  1. Takes input of all the data from the client (Pid, Arrival Time, Burst Time, Priority) and stores it into a structure of array
  2. Applies the algorithm for Priority Scheduling on the data
  3. Returns the output (Turn Around Time, Waiting Time, Average Waiting Time) for all the processes

1. **Comparing Scheduling Algorithms**
   1. Client gives input of the process data (Pid, Burst Time, Arrival Time, Time Quantum, Priority)
   2. We compare the Average waiting time for each process and find out the best (most efficient) process