Junaid .A. Girkar

SAP ID: 60004190057

Shortest Job First Non_Preemptive Scheduling

AIM:
Write a program to demonstrate Shortest Job First Non-Preemptive scheduling with gantt diagram.

THEORY:
Shortest job first (SJF) or shortest job next, is a scheduling policy that selects the waiting process with the smallest execution time to execute next. SJN is a non-preemptive algorithm.

- Shortest Job first has the advantage of having a minimum average waiting time among all scheduling algorithms.
- It is a Greedy Algorithm.
- It may cause starvation if shorter processes keep coming. This problem can be solved using the concept of ageing.
- It is practically infeasible as Operating System may not know burst time and therefore may not sort them. While it is not possible to predict execution time, several methods can be used to estimate the execution time for a job, such as a weighted average of previous execution times. SJF can be used in specialized environments where accurate estimates of running time are available.

Algorithm:

1. Sort all the process according to the arrival time.
2. Then select that process which has minimum arrival time and minimum Burst time.

3. 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.

Advantages :

Here, are pros/benefits of Non-preemptive Scheduling method:

- Offers low scheduling overhead
- Tends to offer high throughput
- It is conceptually very simple method
- Less computational resources need for Scheduling

Disadvantages :

Here, are cons/drawback of Non-Preemptive Scheduling method:

- It can lead to starvation especially for those real-time tasks
- Bugs can cause a machine to freeze up
- It can make real-time and priority Scheduling difficult
- Poor response time for processes

CONCLUSION:

Thus, we were able to successfully use the SJF Non-Preemptive Scheduling Algorithm and understand its advantages and disadvantages.

Shortest Job First Preemptive Scheduling- 60004190057

AIM:
Write a program to demonstrate the Shortest Job First Preemptive Scheduling program with Gantt diagram.

THEORY:
In the Shortest Remaining Time First (SRTF) scheduling algorithm, the process with the smallest amount of time remaining until completion is selected to execute. Since the currently executing process is the one with the shortest amount of time remaining by definition, and since that time should only reduce as execution progresses, processes will always run until they complete or a new process is added that requires a smaller amount of time.

Advantages:

Here, are pros/benefits of Preemptive Scheduling method:

- Preemptive scheduling method is more robust, approach so one process cannot monopolize the CPU
- Choice of running task reconsidered after each interruption.
- Each event cause interruption of running tasks
- The OS makes sure that CPU usage is the same by all running process.
- In this, the usage of CPU is the same, i.e., all the running processes will make use of CPU equally.
- This scheduling method also improvises the average response time.
- Preemptive Scheduling is beneficial when we use it for the multi-programming environment.

Disadvantages :

Here, are cons/drawback of Preemptive Scheduling method:

- Need limited computational resources for Scheduling
- Takes a higher time by the scheduler to suspend the running task, switch the context, and dispatch the new incoming task.
- The process which has low priority needs to wait for a longer time if some high priority processes arrive continuously.

CONCLUSION:

Thus, we were able to successfully use the SJF Preemptive Scheduling Algorithm and understand its advantages and disadvantages.

Preemptive Priority Scheduling- 60004190057

AIM:
Write a program to demonstrate Preemptive Priority Scheduling with gantt diagram

THEORY:

In this problem smaller numbers denote higher priority.

The following functions are used in the given code below:

void quicksort(process array[], low, high)– This function is used to arrange the processes in ascending order according to their arrival time.

int partition(process array[], int low, int high)– This function is used to partition the array for sorting.

void insert(process Heap[], process value, int *heapsize, int *currentTime)– It is used to include all the valid and eligible processes in the heap for execution. heapsize defines the number of processes in execution depending on the current time currentTime keeps record of the current CPU time.

void order(process Heap[], int *heapsize, int start)– It is used to reorder the heap according to priority if the processes after insertion of new process.

void extractminimum(process Heap[], int *heapsize, int *currentTime)– This function is used to find the process with highest priority from the heap. It also reorders the heap after extracting the highest priority process.

void scheduling(process Heap[], process array[], int n, int *heapsize, int *currentTime)– This function is responsible for executing the highest priority extracted from Heap[].

void process(process array[], int n)– This function is responsible for managing the entire execution of the processes as they arrive in the CPU according to their arrival time.

Advantages:

Here, are benefits/pros of using priority scheduling method:

- Easy to use scheduling method

- Processes are executed on the basis of priority so high priority does not need to wait for long which saves time
- This method provides a good mechanism where the relative important of each process may be precisely defined.
- Suitable for applications with fluctuating time and resource requirements.

Disadvantages:

Here, are cons/drawbacks of priority scheduling

- If the system eventually crashes, all low priority processes get lost.
- If high priority processes take lots of CPU time, then the lower priority processes may starve and will be postponed for an indefinite time.
- This scheduling algorithm may leave some low priority processes waiting indefinitely.
- A process will be blocked when it is ready to run but has to wait for the CPU because some other process is running currently.
- If a new higher priority process keeps on coming in the ready queue, then the process which is in the waiting state may need to wait for a long duration of time.

CONCLUSION:

Thus, we were able to successfully use the Priority Preemptive Scheduling Algorithm and understand its advantages and disadvantages.

Round Robin Scheduling- 60004190057

AIM:
Write a program to demonstrate Round Robin Scheduling with Gantt diagram.

THEORY:
Round Robin is a CPU scheduling algorithm where each process is assigned a fixed time slot in a cyclic way.

- It is simple, easy to implement, and starvation-free as all processes get fair share of CPU.
- One of the most commonly used technique in CPU scheduling as a core.
- It is preemptive as processes are assigned CPU only for a fixed slice of time at most.
- The disadvantage of it is more overhead of context switching.

ADVANTAGES:

- There is fairness since every process gets equal share of CPU.
- The newly created process is added to end of ready queue.
- A round-robin scheduler generally employs time-sharing, giving each job a time slot or quantum.
- While performing a round-robin scheduling,a particular time quantum is allocated to different jobs.
- Each process gets a chance to reschedule after a particular quantum time in this scheduling.

DISADVANTAGES:

- There is Larger waiting time and Response time.
- There is Low throughput.
- There is Context Switches.

- Gantt chart seems to come too big (if quantum time is less for scheduling.For Example:1 ms for big scheduling.)
- Time consuming scheduling for small quantums .

CONCLUSION:

Thus, we were able to successfully use the Round Robin Scheduling Algorithm and understand its advantages and disadvantages.