Faculty of Engineering & Technology Electrical & Computer Engineering Department

Operating Systems - ENCS3390

Task_2 results
Report

- Task 2: Simulating CPU Scheduling Algorithm's

Prepared by:
Name: Ody Shbayeh ID-Number:1201462

Instructor: Dr.Abd elsalam sayyad

Section: 2

Date : 6/1/2024

Abstract :

Scheduling algorithms are like task organizers for computers. This brief overview explains how they manage tasks, focusing on types like First-Come-First-Serve, Shortest Job Next, Round Robin, and Priority Scheduling, Shortest Reminder Time First. We'll also touch on real-time scheduling and how it balances responsiveness and predictability. This is a quick example into how computers handle tasks efficiently with scheduling algorithms.

Theory :

Scheduling algorithm and the way implementing them was ideally generated for optimizing the CPU handling the problems of starvation and optimization of the time consumed for the processes to be finish which leads to a faster performance for the computer and for the applications starting on it .

The FCFS :

The First Come First Served method for executing the processes in the CPU is depending on the first process came to the CPU as the name refers after the current process finish executing the first process was held in the queue transfers to the CPU for executing and so on till the processes finish .

The FCFS algorithm is pretty basic and doesn't hold many cases to optimize the scheduling of the CPU and may be unprofessional method for it .

but for some cases when all the processes are requesting the same amount of burst time the FCFS could be a good method for the CPU

The SJF :

The Shortest Job First method of processes scheduling for the CPU processes may be more flexible to deal with more than the FCFS since it depends on sorting the waiting queue's comparing the burst time of the processes when it enters the queue "only the processes that in the waiting queue"

This short improvement may be a huge difference in the performance of the CPU and lowers the time consumed for the CPU but it have some down sides also , the down side of it that if there a process that have a huge burst time it will never get executed since the other processes have less burst time than it which leads to a starvation for that process

The SRTF

The Shortest Remaining Time First for the scheduling algorithm may is very similar to the SJF for scheduling algorithms but it's a pre-empitive way of it it depends on the remaining time for the process and it considers the arrival time for a process and then compares the remaining time for it for the execution process

The down side of it is also when there's a long process and it wants to get executed then it may wait for a long period of time to finish since it will get skipped all the time .

The RR

The Round Robin scheduling algorithm is a method for equalization the deals for the processes among each other the method for it is pretty simple there's a time quantum for the CPU and the processes enter the CPU for the time quantum if the burst time is greater than the time quantum then the process get's executed and then transferred to the waiting queue and the next process goes and so on.

The other case for the RR is when the process burst time is lower than the time quantum then it gets executed and finished and then the next process enters the CPU to get executed for a new time quantum

the down side of this scheduling algorithm is if a process is too long then it will finish after some period of time but the waiting avg and the turn around time will be some kind of high which may be inconvenient for the processes in the CPU

The Priority Scheduling

The priority scheduling may be one of the most used scheduling algorithm in the current OS because it have a good approach to what we need by solving the problems for the scheduling algorithms the basic algorithm for it is

The processes gets sorted depending on the priority of it next of that if two processes have the same priority they get sorted depending on the burst time if they have the burst time if they have the same burst time it depends on the arrival time of the process to the queue

Procedure:

to simulate the scheduling algorithm I wrote a simple java program that have three classes

Process Class

The process class defines what the process have and a overrided method to print the descriptions of the process

like the process ID , Arrival time , burst time , come back time , priority

for the processes that the user enters in the main method

CPUScheduler Class

This class contains the core of my project which are the scheduling methods and the way I implemented them using the explanation from the slides and some help from the youtube to get a better results

Schedulerprogram Class

This class is the Driver for the previous classes I talked about which contains the UI code along with the INPUT details mechanism and finally the calling of the functions and a print method to print the avg's for the processes

The Results I got from the program I wrote in java came as following

The FCFS :

```
for better running of the program try to run each method separatly running them sequentially like 1.2.3.4 may cause memory drop
Choose a scheduling algorithm:
1. FCFS
2. SJF
3. Round Robin
4. Priority scheduling
5. SRTF
6. exit the program
1
scheduling algorithm results :

gantt chart: [P1, P1, P1, P1, P1, P1, P1, P1, P1, P1, P2, P2, P2, P2, P2, P2, P2, P2, P2, P3, P3, P3, P3, P3, P3, P3, P3, P3, P3, P3, P3, P3, P3, P4,

average turnaround time: 30.857142857142858
average waiting time: 23.142857142857142
```

The SJF :

```
for better running of the program try to run each method separatly running them sequentially like 1.2.3.4 may cause memory drop
Choose a scheduling algorithm:
1. FCFS
2. SJF
3. Round Robin
4. Priority scheduling
5. SRTF
6. exit the program
2
scheduling algorithm results :

gantt chart: [P1, P1, P1, P1, P1, P1, P1, P1, P1, P1, P6, P6, P6, P6, P5, P5, P5, P5, P5, P7, P7, P7, P7, P7, P7, P4, P4, P4, P4, P4, P4, P4, P2,

average turnaround time: 23.571428571428573
average waiting time: 15.857142857142858
```

The SRTF :

```
for better running of the program try to run each method separatly running them sequentially like 1.2.3.4 may cause memory drop
Choose a scheduling algorithm:
1. FCFS
2. SJF
3. Round Robin
4. Priority scheduling
5. SRTF
6. exit the program
5
scheduling algorithm results :

gantt chart: [P1, P2, P2, P2, P2, P2, P2, P2, P2, P6, P6, P6, P6, P5, P5, P5, P5, P5, P7, P7, P7, P7, P7, P7, P4, P4, P4, P4, P4, P4, P4, P1, P1,

average turnaround time: 22.857142857142858
average waiting time: 15.142857142857142
```

The RR :
with Q = 5

```
for better running of the program try to run each method separatly running them sequentially like 1.2.3.4 may cause memory drop
Choose a scheduling algorithm:
1. FCFS
2. SJF
3. Round Robin
4. Priority scheduling
5. SRTF
6. exit the program
3
enter time quantum for Round Robin: 5
scheduling algorithm results :

gantt chart: [P1, P1, P1, P1, P1, P2, P2, P2, P2, P2, P3, P3, P3, P3, P3, P4, P4, P4, P4, P4, P5, P5, P5, P5, P5, P6, P6, P6, P6, P7, P7, P7, P7,

average turnaround time: 37.0
average waiting time: 29.285714285714285
```

The priority pre-empitive :

```
for better running of the program try to run each method separatly running them sequentially like 1.2.3.4 may cause memory drop
Choose a scheduling algorithm:
1. FCFS
2. SJF
3. Round Robin
4. Priority scheduling
5. SRTF
6. exit the program
4
choose priority scheduling method:
1. Preemptive
2. Non-Preemptive
please consider running the non-preemptive case alone because it uses a lot of memory and may cause crashes
1
scheduling algorithm results :

gantt chart: [P1, P2, P2, P2, P4, P4, P4, P5, P4, P5, P4, P5, P4, P5, P4, P5, P6, P6, P6, P6, P2, P7, P2, P7, P2, P2, P2, P7, P7, P7, P7, P1, P3,

average turnaround time: 25.142857142857142
average waiting time: 17.428571428571427
```

The priority non-pre-empitive :

```
for better running of the program try to run each method separatly running them sequentially like 1.2.3.4 may cause memory drop
Choose a scheduling algorithm:
1. FCFS
2. SJF
3. Round Robin
4. Priority scheduling
5. SRTF
6. exit the program
4
choose priority scheduling method:
1. Preemptive
2. Non-Preemptive
please consider running the non-preemptive case alone because it uses a lot of memory and may cause crashes
2
scheduling algorithm results :

gantt chart: [P1, P5, P4, P6, P2, P7, P3]

average turnaround time: 24.571428571428573
average waiting time: 16.857142857142858
```

**note I had some problems with the memory distribution in the code so it may crashes for some times but it worked as expected

The problem was when I try to run multiple cases sequentially it fills the memory and the program crashes and for the lack of time I couldn't fix it .

Also I wish I could wrote more in the report but the lack of time and the pressure from the projects led to this much greets