HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY, VNU HCM
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

OPERATING SYSTEM

Lab 7: CPU Scheduling

Pham Minh Tuan - MSSV: 1752595

1 Exercises

Summarise:

- Turnaround time = Time completion - Arrival time
- Waiting time = Turnaround time - Burst time
- Response time = First-run time - Arrival time

1.1 Questions:

Q1: Suppose that the following processes arrive for execution at the time instants indicated. Each process will run for the amount of time listed. Assume that the scheduler uses non-preemptive scheduling.

Process	Arrival Time	Burst Time
P1	0.0	8
P2	0.4	4
P3	1.0	1

a Calculate the average turnaround, waiting and response time for these processes with the FCFS scheduling algorithm?

Answer:

- t = 0.0 : P1
- t = 8.0 : P2
- t = 12.0 : P3
- t = 13.0 : finish

Turnaround time:

- T1 = 8.0 - 0.0 = 8.0(s)
- T2 = 12.0 - 0.4 = 11.6(s)
- T3 = 13.0 - 1.0 = 12.0(s)
- Average turnaround time = $\frac{8.0 + 11.6 + 12.0}{3} \approx 10.53(s)$

Waiting Time:

- T1 = 0(s)
- T2 = 11.6 - 4 = 7.6(s)
- T3 = 12.0 - 1 = 11.0(s)
- Average waiting time = $\frac{0 + 7.6 + 11.0}{3} = 6.2(s)$

Response time:

- T1 = 0(s)

- T2 = 8.0 - 0.4 = 7.6(s)
- T3 = 12.0 - 1.0 = 11.0(s)
- Average response time = $\frac{0 + 7.6 + 11.0}{3} = 6.2(s)$

b What is the average turnaround, waiting and response time for these processes with the SJF scheduling algorithm?

Answer:

- t = 0.0 : P1
- t = 8.0 : P3
- t = 9.0 : P2
- t = 13.0 : finish

Turnaround time:

- T1 = 8.0 - 0.0 = 8.0(s)
- T2 = 13.0 - 0.4 = 12.6(s)
- T3 = 9.0 - 1.0 = 8.0(s)
- Average turnaround time = $\frac{8.0 + 12.6 + 8.0}{3} \approx 9.53(s)$

Waiting Time:

- T1 = 0(s)
- T2 = 12.6 - 4 = 8.6(s)
- T3 = 8.0 - 1 = 7.0(s)
- Average waiting time = $\frac{0 + 8.6 + 7.0}{3} = 5.2(s)$

Response time:

- T1 = 0(s)
- T2 = 9.0 - 0.4 = 8.6(s)
- T3 = 8.0 - 1.0 = 7.0(s)
- Average response time = $\frac{0 + 8.6 + 7.0}{3} = 5.2(s)$

c The SJF algorithm is supposed to improve performance, but notice that we chose to run process P1 at time 0 because we did not know that two shorter processes would arrive soon. Compute what the average turnaround time will be if the CPU is left idle for the first 1 unit and then SJF scheduling is used. Remember that processes P1 and P2 are waiting during this idle time, so their waiting time may increase. This algorithm could be called future-knowledge scheduling.

Answer:

- t = 0.0 : idle
- t = 1.0 : P3
- t = 2.0 : P2
- t = 6.0 : P1

- t = 13.0 : finish

Turnaround time:

- T1 = 13.0 - 0.0 = 13.0(s)
- T2 = 6.0 - 0.4 = 5.6(s)
- T3 = 2.0 - 1.0 = 1.0(s)
- Average turnaround time = $\frac{13.0 + 5.6 + 1.0}{3} \approx 6.53(s)$

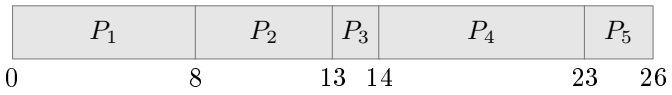
→ The average turnaround time when let the CPU idle is less than 3 seconds.

Q2: Consider the following set of processes, with the length of the CPU burst given in milliseconds:

Process	Burst Time	Priority
P1	8	4
P2	6	1
P3	1	2
P4	9	2
P5	3	3

The processes are assumed to have arrived in the order P1, P2, P3, P4, P5, all at time 0. Draw four Gantt charts that illustrate the execution of these processes using the following scheduling algorithms: FCFS, SJF, non-preemptive priority (a smaller priority number implies a higher priority), and RR (quantum = 1). Calculate the average waiting time and turnaround time of each scheduling algorithm.

1. FCFS:



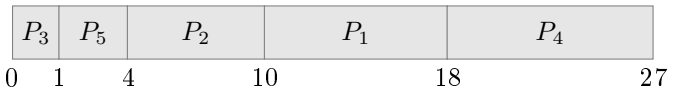
Turnaround time:

- T1 = 8 - 0 = 8(ms)
- T2 = 13 - 0 = 13(ms)
- T3 = 14 - 0 = 14(ms)
- T4 = 23 - 0 = 23(ms)
- T5 = 26 - 0 = 26(ms)
- Average turnaround time = $\frac{8 + 13 + 14 + 23 + 26}{5} = 16.8(s)$

Waiting Time:

- T1 = 0(s)
- T2 = 13 - 6 = 7(ms)
- T3 = 14 - 1 = 12(ms)
- T4 = 23 - 9 = 14(ms)
- T5 = 26 - 3 = 23(ms)
- Average waiting time = $\frac{0 + 7 + 12 + 14 + 23}{5} = 11.2(s)$

2. SJF:



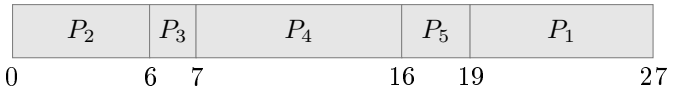
Turnaround time:

- T1 = 18 - 0 = 18(ms)
- T2 = 10 - 0 = 10(ms)
- T3 = 1 - 0 = 1(ms)
- T4 = 27 - 0 = 27(ms)
- T5 = 4 - 0 = 4(ms)
- Average turnaround time = $\frac{18+10+1+27+4}{5} = 12(s)$

Waiting Time:

- T1 = 18 - 8 = 10(s)
- T2 = 10 - 6 = 4(ms)
- T3 = 1 - 1 = 0(ms)
- T4 = 27 - 9 = 18(ms)
- T5 = 4 - 3 = 1(ms)
- Average waiting time = $\frac{10+4+0+18+1}{5} = 6.6(s)$

3. Non-preemptive priority:



Turnaround time:

- T1 = 27 - 0 = 27(ms)
- T2 = 6 - 0 = 6(ms)
- T3 = 7 - 0 = 7(ms)
- T4 = 16 - 0 = 16(ms)
- T5 = 19 - 0 = 19(ms)
- Average turnaround time = $\frac{27+6+7+16+19}{5} = 15(s)$

Waiting Time:

- T1 = 27 - 8 = 19(s)
- T2 = 6 - 6 = 0(ms)
- T3 = 7 - 1 = 6(ms)
- T4 = 16 - 9 = 7(ms)
- T5 = 19 - 3 = 16(ms)
- Average waiting time = $\frac{19+0+6+7+16}{5} = 9,6(s)$

4. RR(quantum = 1)

P_2	P_3	P_4	P_5	P_1

0 1 2 3 4 5

P_2	P_4	P_5	P_1

5 6 7 8 9

P_2	P_4	P_5	P_1

9 10 11 12 13

P_2	P_4	P_1

13 14 15 16

P_2	P_4	P_1

16 17 18 19

P_2	P_4	P_1

19 20 21 22

P_4	P_1

22 23 24

P_4	P_1

24 25 26

P_4

26 27

Turnaround time:

- T1 = 26 - 0 = 26(ms)
- T2 = 20 - 0 = 20(ms)
- T3 = 2 - 0 = 2(ms)
- T4 = 27 - 0 = 27(ms)
- T5 = 12 - 0 = 12(ms)
- Average turnaround time = $\frac{25+20+0+27+12}{5} = 16.8(s)$

Waiting Time:

- T1 = 26 - 8 = 18(s)
- T2 = 20 - 6 = 14(ms)
- T3 = 2 - 1 = 1(ms)
- T4 = 27 - 9 = 18(ms)
- T5 = 12 - 3 = 9(ms)
- Average waiting time = $\frac{18+14+1+18+9}{5} = 12(s)$

Q3:

a Describe the trade-off of increasing and decreasing the time quantum in RR.
Answer:

- Each process is assigned a time slice for its completion. The time slice (or time quantum) is required to complete a process and the time spent for each task or process is the same.
- The trade-off is that since there are long time processes and less time process, and each of them is gained a unchanged time slice, so there are chance that less time process will finish before the end of time slice.
- So, we need to choose the suitable time slice in case of the long delay in completion tasks or processes in overall

b Analyze the advantages and disadvantages of FCFS, SJF, priority and RR scheduling algorithms. Specify on which occasion should we use each of those algorithms.

(a) FCFS:
   i. Advantages:
      - It does not require complex logic, simple and easy to implement.
      - Every process will get a chance to run, no starvation.
   ii. Disadvantages:
      - No option for pre-empty.
      - If a process executes for a long time, the processes in the back of the queue will have to wait for a long time before they get a chance to be executed.

(b) SJF:
   i. Advantages:
      - Short processes are executed first and then followed by longer processes.
      - The throughput is increased because more processes can be executed in less amount of time.
   ii. Disadvantages:
      - The time taken by a process must be known by the CPU beforehand.
      - Long processes have long waiting time, starvation occur.

(c) Priority Scheduling:
   i. Advantages:
      - The priority of a process can be selected based on memory requirement, time requirement or user preference.
   ii. Disadvantages:
      - A second scheduling algorithm is required to schedule the processes in case they have same priority.
      - In preemptive priority scheduling, a higher priority process can execute ahead of an already executing lower priority process. If lower priority process keeps waiting for higher priority processes, starvation occurs.

(d) RR:
   i. Advantages:
      - Each process is served by the CPU for a fixed time quantum.
      - Starvation doesn't occur because for each round robin cycle, every process is given a fixed time to execute. No process is left behind.
   ii. Disadvantages:
      - The throughput in RR largely depends on the choice of the length of the time quantum. If time quantum is longer than needed, it tends to exhibit the same behavior as FCFS.

- If time quantum is shorter than needed, the number of times that CPU switches from one process to another process, increases. This leads to decrease in CPU efficiency.

In which situations we should use which CPU Scheduling:

(a) FCFS: The incoming processes are short and there is no need for the processes to execute in a specific order.

(b) SJF: If each process is already known time to complete, and burst time of each process is short to avoid the starvation.

(c) Priority: The processes are a mix of user based and kernel based processes since kernel based processes have higher priority when compared to user based processes in general.

(d) RR: The processes are a mix of long and short processes and the task will only be completed if all the processes are executed successfully in a given time.