ADVANCED OPERATING SYSTEMS

CSEN383

PROJECT-2 (Process Scheduling)

Group – 3

-Aman Jain, Amarnath S Kaushik, Daryl Ho, Meghana Kale, Sai Kiran Jasti

Structs:

1. **Process**
2. **Seeds**
3. **PriorityQueue**

Functions:

1. **void generateProcesses(Process processes[], int count, int seed)**
   This function takes an array of processes (Struct Process), their count and a seed generated by the nest function. This function updates the process details like arrival time, run time in the Process struct.

2. **void sortProcesses(Process processes[], int count)**
   This function sorts the generated processes based on their arrival time.

3. **void seedGenerator(Seeds bestSeeds[])**
   This function is responsible for choosing the best seeds in the range 1 to 10000, which leads to minimum CPU idle time and maximum job overlapping. This function gives us a list of 5 such seeds for our workloads.

Workloads:

Run #1:

Process ID	Arrival Time	Run Time	Priority
I	0	10	3
J	5	4	2
B	9	8	1
C	12	1	4
A	20	2	3
H	24	6	3
D	25	8	2
E	28	9	1
G	33	9	4
F	40	10	3

Total No. of Processes : 10

Run #2:

Process ID	Arrival Time	Run Time	Priority
B	3	4	4
F	14	8	3
I	17	9	1
E	23	4	1
A	26	6	3
H	27	9	4
J	32	7	3
C	38	7	3
G	45	6	3
D	48	7	3

Total No. of Processes : 10

Run #3:

Process ID	Arrival Time	Run Time	Priority
A	2	6	3
B	7	9	4
I	10	10	1
H	14	5	1
C	21	7	1
D	25	10	4
G	26	4	3
J	30	5	2
E	34	6	3
F	47	2	4

Total No. of Processes : 10

Run #4:

Process ID	Arrival Time	Run Time	Priority
J	0	5	3
F	6	6	2
G	19	7	1
B	21	4	1
D	25	9	1
I	34	8	1
A	43	7	2
E	47	6	2
C	53	1	1
H	56	2	1

Total No. of Processes : 10

Run #5:

Process ID	Arrival Time	Run Time	Priority
D	1	7	2
C	7	8	4
G	9	6	2
E	15	2	2
F	21	8	2
B	21	5	3
J	29	9	4
I	38	2	3
H	45	6	4
A	51	7	3

Total No. of Processes : 10

Algorithm's Averaged Output After 5 Runs:

1. **First Come First Served**

Average Quantity	Quanta
Turn-Around Time	14.44
Waiting Time	8.18
Response Time	8.18

2. **Shorted Job First**

Average Quantity	Quanta
Turn-Around Time	12.90
Waiting Time	6.64
Response Time	6.64

3. Shortest Remaining Time:

Average Quantity	Quanta
Turn-Around Time	12.30
Waiting Time	6.04
Response Time	11.60

4. Round Robin:

Average Quantity	Quanta
Turn-Around Time	18.84
Waiting Time	12.58
Response Time	2.04

5. High Priority First Preemptive:

Average Quantity	Quanta
Turn-Around Time	16.06
Waiting Time	9.80
Response Time	4.06

6. High Priority First Non-Preemptive:

Average Quantity	Quanta
Turn-Around Time	13.13
Waiting Time	7.04
Response Time	7.19

Conclusion:

The results from the project demonstrate the varying efficiencies of different process scheduling algorithms. Among the algorithms analyzed, **Round Robin** stands out as the most effective for real-time and multitasking scenarios. Its lower response time (2.04) ensures that all processes receive CPU attention quickly, which is particularly important in time-sharing systems. While its turn-around time (18.84) and waiting time (12.58) are slightly higher than other algorithms, this trade-off is acceptable for systems prioritizing fairness and responsiveness.

In contrast, algorithms like **Shortest Job First (SJF)** and **Shortest Remaining Time (SRT)** exhibit better turn-around and waiting times due to their focus on shorter processes. However, they lack the fairness and responsiveness required for diverse workloads. Similarly, **First Come First Served (FCFS)** is simple to implement but suffers from significant waiting time due to the "convoy effect."

The preemptive and non-preemptive variants of **High Priority First (HPF)** offer balanced performance but can lead to starvation for low-priority processes, making them less favorable for systems requiring equal process treatment.

In conclusion, the **Round Robin scheduling algorithm** strikes an optimal balance between fairness and responsiveness, making it an ideal choice for environments where equitable process handling and minimal response times are critical.