CS342 Project 2 Report

Ahmet Kaan Uğuralp

Comparison of each scheduling algorithm based on different burst times:

Parameters: N=4, Bcount=30, minB=900, avgB=1000, minA=100, avgA=250

Total Waiting Time (second) of each burst for each W Thread

	Thread 1	Thread 2	Thread 3	Thread 4	Total
FCFS	7289	3960	2828	3370	12938
SJF	4071	3612	2717	2568	12970
PRIO	655	2226	3922	5327	12132
VRUNTIME	1810	3212	3488	3573	12084

Scheduling Algorithm	Total Execution Time (second)
FCFS	232
SJF	231
PRIO	223
VRUNTIME	213

Parameters: N=4, Bcount=30, minB=400, avgB=500, minA=100, avgA=250

Total Waiting Time (second) of each burst for each W Thread:

	Thread 1	Thread 2	Thread 3	Thread 4	Total
FCFS	1533	1820	1351	1113	5819
SJF	760	588	1699	2169	5218
PRIO	284	1020	1775	2605	5685
VRUNTIME	1587	1307	545	2001	5441

Scheduling Algorithm	Total Execution Time (second)
FCFS	110
SJF	105
PRIO	104
VRUNTIME	101

Parameters: N=4, Bcount=30, minB=100, avgB=200, minA=100, avgA=250

Total Waiting Time

	Thread 1	Thread 2	Thread 3	Thread 4	Total
FCFS	349	351	357	341	1400
SJF	743	233	68	388	1433
PRIO	129	188	465	747	1433
VRUNTIME	39	571	347	776	1734

Scheduling Algorithm	Total Execution Time (second)
FCFS	33
SJF	35
PRIO	32
VRUNTIME	39

When we compare the tables above, we can see that as the average burst time increases, the performance difference between each scheduling algorithm becomes more distinct and we can observe the performance ranking as: VRUNTIME > PRIO > SJF > FCFS.

Comparison of each scheduling algorithm based on different waiting times:

Parameters: N=4, Bcount=30, minB=100, avgB=200, minA=100, avgA=250

Total Waiting Time-2

	Thread 1	Thread 2	Thread 3	Thread 4	Total
FCFS	349	351	357	341	1400
SJF	743	233	68	388	1433
PRIO	129	188	465	747	1433
VRUNTIME	39	571	347	776	1734

Scheduling Algorithm	Total Execution Time (second)
FCFS	33
SJF	35
PRIO	32
VRUNTIME	39

Parameters: N=4, Bcount=30, minB=100, avgB=200, minA=400, avgA=500

Total Waiting Time (second) of each burst for each W Thread.

	Thread 1	Thread 2	Thread 3	Thread 4	Total
FCFS	125	121	106	117	470
SJF	179	204	86	7	315
PRIO	6	21	168	571	767
VRUNTIME	163	335	393	96	687

Scheduling Algorithm	Total Execution Time (second)
FCFS	36
SJF	34
PRIO	36
VRUNTIME	39

Parameters: N=4, Bcount=30, minB=100, avgB=200, minA=900, avgA=1000

Total Waiting Time-1

	Thread 1	Thread 2	Thread 3	Thread 4	Total
FCFS	7	7	11	6	32
SJF	3	4	3	2	14
PRIO	3	5	5	5	19
VRUNTIME	3	7	5	6	22

Scheduling Algorithm	Total Execution Time (second)
FCFS	60
SJF	64
PRIO	56
VRUNTIME	60

We can observe from these tables that, as the average waiting time increases, significance of the scheduling algorithm that we use decreases, because the server thread is able to keep up with the worker threads more easily.

Comparison of each scheduling algorithm based on total thread amount:

Parameters: N=10, Bcount=30, minB=900, avgB=1000, minA=100, avgA=250

	Threa d 1	Threa d 2	Threa d 3	Threa d 4	Threa d 5	Threa d 6	Threa d 7	Threa d 8	Threa d 9	Threa d 10	Total
FCFS	7450	8740	10306	7852	9606	8401	7969	10758	8246	6962	86294
SJF	10525	3730	10582	2733	7601	9244	13075	11136	7993	7534	84159
PRIO	723	2496	4309	6031	7832	9557	11122	12704	13305	15816	85000
VRUNTIME	2571	1620	5176	5674	5148	9160	9429	14200	11218	11924	76124

Scheduling Algorithm	Total Execution Time (second)
FCFS	588
SJF	597
PRIO	559
VRUNTIME	526

Parameters: N=4, Bcount=30, minB=900, avgB=1000, minA=100, avgA=250

Total Waiting Time (second) of each burst for each W Thread:-1-1

	Thread 1	Thread 2	Thread 3	Thread 4	Total
FCFS	7289	3960	2828	3370	12938
SJF	4071	3612	2717	2568	12970
PRIO	655	2226	3922	5327	12132
VRUNTIME	1810	3212	3488	3573	12084

Scheduling Algorithm	Total Execution Time (second)
FCFS	232
SJF	231
PRIO	223
VRUNTIME	213

Performance ranking of the algorithms are as follows: VRUNTIME > PRIO > FCFS > SJF in the first sample with 10 threads, and it is VRUNTIME > PRIO > SJF > FCFS in the second sample with 4 threads. Performance of the SJF and FCFS algorithms are very similar in our sample with 4 threads.

As we increase the thread amount, the performance difference between each algorithm increase as our runqueue gets more busy because all of our threads are generating bursts at the same time. Therefore, our scheduling algorithms become more influential on the total execution time.