Murat Sevinç
21702603
CS 342 - Project 2
Report

Introduction

This report compares different scheduling algorithms (FCFS, SJF, PRIO and VRUNTIME) with randomly and exponentially distributed burst and waiting times.

In experiment 1, algorithms will be compared on 3 certain average burst and waiting values. Then, each algorithm will be compared on several different average and minimum values. Figures and tables will be provided to show the differences.

Experiment 1

First, we will take several min burst & waiting values when thread count is 4 and burst count is 10. This will be applied also when thread count is 8 and burst count is 10. To be more consistent and accurate, results will be averaged from 3 trials.

Data:

Thread Count: 4
Burst Count: 10

Time (ms)	FCFS	SJF	PRIO	VRUNTIME
Avg: 500 Min: 100	286	682	236	622
Avg: 1000 Min: 250	542	1055	538	1066
Avg: 2000 Min: 1000	1491	2055	1492	2872

Table 1: Time result of algorithms when thread count is 4 and burst count is 10

Thread Count: 8

Burst Count: 10

Time (ms)	FCFS	SJF	PRIO	VRUNTIME
Avg: 500 Min: 100	1820	2589	2056	2502
Avg: 1000 Min: 250	3053	4728	3209	4549
Avg: 2000 Min: 1000	7839	9852	7641	10899

Table 2: Time result of algorithms when thread count is 8 and burst count is 10

Results

- From table 1 and table 2, it can be said that the **best algorithms** (with low waiting time) are **FCFS and PRIO**. In both 6 experiments, they are nearly equal and the lowest.

- SJF is **slightly better** than VRUNTIME.

- **VRUNTIME** has the **longest waiting time** among these 4 scheduling algorithms when thread count and burst count is low. In experiment 5, VRUNTIME becomes the best algorithm for large amounts of threads and bursts.

- When thread count increases and burst count stable, the difference is not that much anymore.

- When **thread count increases**, the waiting time for FCFS and PRIO **increased by 6-7 times**. Whereas SJF and VRUNTIME **increased 3-4 times**. This lessens the gap between the algorithms.

- By this conclusion, it can be said that for **small** number of **threads**, **PRIO and FCFS is better**, yet, for **large** number of **threads**, **SJF and VRUNTIME is better.**

Experiment 2

This experiment is to examine the behavior of the FCFS algorithm on several thread and burst counts. The experiment is done when average is 500 and min is 100.

Amounts	FCFS waiting time (ms)
Thread: 5 Burst: 10	522
Thread: 10 Burst: 10	3465
Thread: 5 Burst: 20	2573
Thread: 10 Burst: 20	12760

Table 3: FCFS behavior when minimum waiting&burst time is 100 and average waiting&burst time is 500

Results

- Both thread count and burst count affects and increases the waiting time.

- When **burst count doubles**, waiting time increases by 5 times. So the average waiting time **for each thread increased by 5 times**.

- When **thread count doubles**, waiting time increases by 7 times but the average waiting time **for each thread increases by 3.5 times**.

Experiment 3

This experiment is to examine the behavior of the SJF algorithm on several thread and burst counts. The experiment is done when average is 500 and min is 100.

Amounts	SJF waiting time (ms)
Thread: 5 Burst: 10	873
Thread: 10 Burst: 10	4002
Thread: 5 Burst: 20	3826
Thread: 10 Burst: 20	16574

Table 4: SJF behavior when minimum waiting&burst time is 100 and average waiting&burst time is 500

Results

- Both thread count and burst count affects and increases the waiting time.

- When **burst count doubles**, waiting time increases by 4 times. So the average waiting time **for each thread increased by 4 times**.

- For large numbers of bursts, SJF is better than FCFS.

- When **thread count doubles**, waiting time increases by 4.5 times but the average waiting time **for each thread increases by 2.2 times**.

- This proves that **SFJ is better than FCFS for large amounts of threads**.

Experiment 4

This experiment is to examine the behavior of the PRIO algorithm on several thread and burst counts. The experiment is done when average is 500 and min is 100.

Amounts	PRIO waiting time (ms)
Thread: 5 Burst: 10	605
Thread: 10 Burst: 10	3137
Thread: 5 Burst: 20	2715
Thread: 10 Burst: 20	13286

Table 5: PRIO behavior when minimum waiting&burst time is 100 and average waiting&burst time is 500

Results

- Both thread count and burst count affects and increases the waiting time.

- When **burst count doubles**, waiting time increases by 4.5 times. So the average waiting time **for each thread increased by 4.5 times**.

- For large numbers of bursts, PRIO is better than FCFS but worse than SJF.

- When **thread count doubles**, waiting time increases by 5 times but the average waiting time **for each thread increases by 2.5 times**.

- This indicates that for large amounts of threads, PRIO is better than FCFS but slightly worse than SJF.

Experiment 5

This experiment is to examine the behavior of the VRUNTIME algorithm on several thread and burst counts. The experiment is done when average is 500 and min is 100.

Amounts	VRUNTIME waiting time (ms)
Thread: 5 Burst: 10	1078
Thread: 10 Burst: 10	4632
Thread: 5 Burst: 20	3989
Thread: 10 Burst: 20	15591

Table 6: VRUNTIME behavior when minimum waiting & burst time is 100 and average waiting & burst time is 500

Results

- Both thread count and burst count affects and increases the waiting time.

- When **burst count doubles**, waiting time increases by 3.6 times. So the average waiting time **for each thread increased by 3.6 times**.

- This is the **best algorithm for large numbers of bursts**, since it has the lowest ratio.

- When **thread count doubles**, waiting time increases by 4.2 times but the average waiting time **for each thread increases by 2.1 times**.

- This indicates that **for large amounts of threads**, VRUNTIME is the **best algorithm** among these for algorithms.