CS 342 - OPERATING SYSTEMS

PROJECT 2

ABDULLAH CAN ALPAY

21702686 - SECTION 3

28.03.2021

In this project, we consider how to proceed in the multi-thread systems where multiple threads try to access and modify a certain memory space. With the use of pthread_mutex_lock and pthread_mutex_unlock methods, we learn to lock the access while someone is modifying. After that, we implement several scheduler algorithms to deal with scheduling.

N, AVGB, AVGA, MINB, MINA VALUES

We consider the following values:

1) n, minB, avgB, minA, avgB => 3 100 200 1000 1500
2) n, minB, avgB, minA, avgB => 4 100 300 100 150
3) n, minB, avgB, minA, avgB => 4 500 900 1000 1500
4) n, minB, avgB, minA, avgB => 3 100 100 10000 15000

Firstly, when we look at the outputs for 1 and 2 we understand the importance of minA and avgA. A represents the sleep time for each thread after creating one job. The sleep time is defined by the formula: (-avgA) * log(1 - (rand()/(RAND_MAX + 1.))) which generates an exponentially distributed random value. This random variable must be bigger than the minA. This means that in the first execution, threads will sleep more than the second execution. This will allow the schedule thread to execute jobs while worker threads are sleeping. This will result in a shorter waiting time.

When we look at the second execution, however, we see that jobs will be created faster than the schedule thread's job executions. Thus, jobs in the rq linked list will be accumulated (lots of jobs enters the queue) and increase the waiting time of the jobs. In the second execution schedule thread cannot keep up with the worker threads speed, hence, lots of jobs will wait for a long time. Similar comparisons can also be obtained from 1-4, 2-4 and 3-4.

Furthermore, when we compare the first execution with the third execution, we understand the importance of the minB and avgB. In the third execution, the scheduler thread will spend a lot of time to complete the job. Thus, while the scheduler working on a job, the worker threads will add new jobs to the linked list. Hence, waiting time will increase. The job's burst time is found by the similar formula: (-avgA) * log(1 - (rand()/(RAND_MAX + 1.))) which will generate a exponentially distributed random value.

To sum up, for lower minA, avgA; or higher minB, avgB values, the waiting time for the threads will increase. We take waiting time independently; thus, more different thread jobs mean more waiting in the queue.

./schedule 3 75 100 200 1000 1500 FCFS

```
FINISHED
For thread 1: Average waiting time is: 57.9867ms
4349 75
For thread 2: Average waiting time is: 83.5600ms
6267 75
For thread 3: Average waiting time is: 64.1067ms
4808 75
```

./schedule 3 75 100 200 1000 1500 SJF

```
FINISHED
For thread 1: Average waiting time is: 52.5467ms
3941 75
For thread 2: Average waiting time is: 96.3467ms
7226 75
For thread 3: Average waiting time is: 55.9067ms
4193 75
```

./schedule 3 75 100 200 1000 1500 PRIO

```
FINISHED
For thread 1: Average waiting time is: 61.2000ms
4590 75
For thread 2: Average waiting time is: 38.4267ms
2882 75
For thread 3: Average waiting time is: 74.0933ms
5557 75
```

./schedule 3 75 100 200 1000 1500 VRUNTIME

```
VRUNTIME For Thread1 is 21964.000000
VRUNTIME For Thread2 is 32334.900000
VRUNTIME For Thread3 is 40540.800000
FINISHED
For thread 1: Average waiting time is: 44.0933ms
3307 75
For thread 2: Average waiting time is: 42.0533ms
3154 75
For thread 3: Average waiting time is: 80.1333ms
6010 75
```

./schedule 4 50 100 300 100 150 FCFS

```
FINISHED
For thread 1: Average waiting time is: 33050.9400ms
1652547 50
For thread 2: Average waiting time is: 35119.9400ms
1755997 50
For thread 3: Average waiting time is: 30986.0600ms
1549303 50
For thread 4: Average waiting time is: 35041.8800ms
1752094 50
```

./schedule 4 50 100 300 100 150 SJF

```
FINISHED
For thread 1: Average waiting time is: 16232.3000ms
811615 50
For thread 2: Average waiting time is: 18613.6800ms
930684 50
For thread 3: Average waiting time is: 22084.4600ms
1104223 50
For thread 4: Average waiting time is: 18505.1600ms
925258 50
```

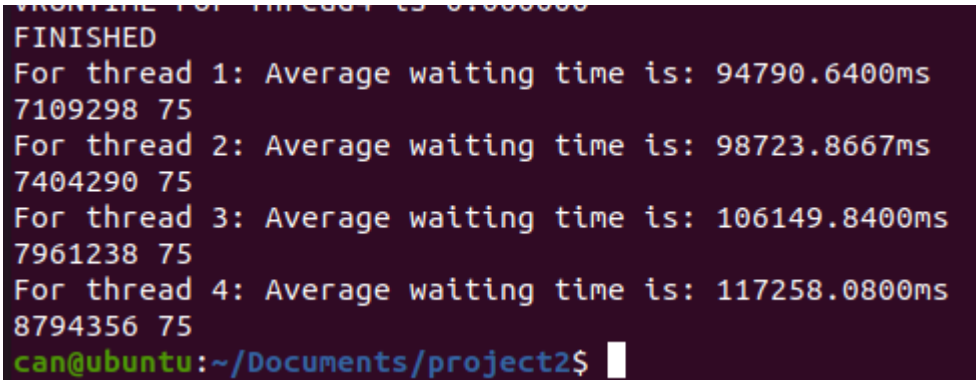
./schedule 4 50 100 300 100 150 PRIO

```
FINISHED
For thread 1: Average waiting time is: 5791.0600ms
289553 50
For thread 2: Average waiting time is: 27611.3000ms
1380565 50
For thread 3: Average waiting time is: 49396.9800ms
2469849 50
For thread 4: Average waiting time is: 72599.2200ms
3629961 50
```

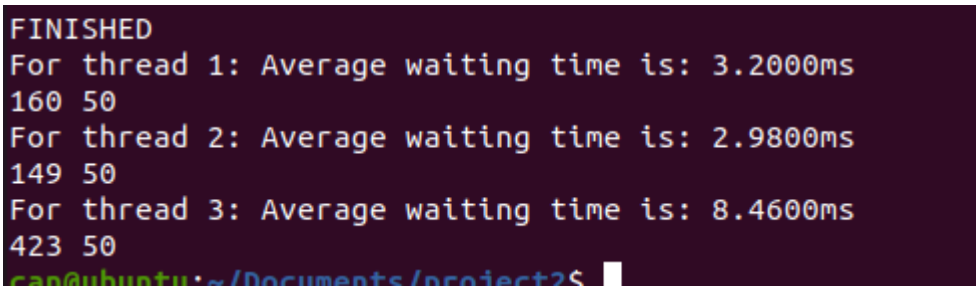
./schedule 4 50 100 300 100 150 VRUNTIME

```
VRUNTIME For Thread1 is 18182.000000
VRUNTIME For Thread2 is 19799.000000
VRUNTIME For Thread3 is 30352.000000
VRUNTIME For Thread4 is 41750.600000
FINISHED
For thread 1: Average waiting time is: 22094.7000ms
1104735 50
For thread 2: Average waiting time is: 25198.2000ms
1259910 50
For thread 3: Average waiting time is: 31861.9800ms
1593099 50
For thread 4: Average waiting time is: 43290.2200ms
2164511 50
```

./schedule 4 75 500 900 1000 1500 FCFS



./schedule 3 50 100 100 10000 15000 SJF



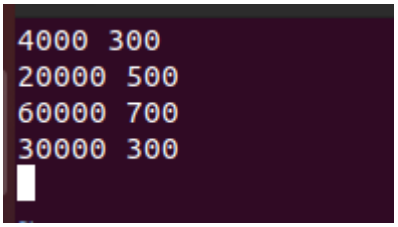
FCFS, SJF, PRIO and VRUNTIME

When we look at the executions we can conclude the following findings:

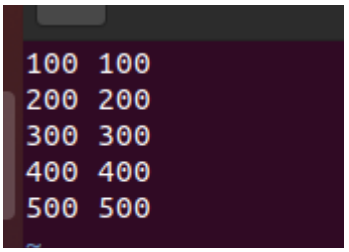
- In FCFS, the fastest thread will have the lowest waiting time.
- In SJF, it will depend on the job's burst length. Thus, some thread might wait longer than other threads.
- In PRIO, it is expected that waiting times are: threadn > thread(n-1) > ….>thread1
- In VRUNTIME, all threads will have similar waiting times. VRUNTIME tries to run every thread equally; if one thread worked for a long time, it will wait for other threads to catch it. Thus, VRUNTIME will give the most equally distributed waiting times. It can be said that the average waiting time of the threads is around any of the thread wait times. However, it should be noted that the waiting times depend on the burst lengths and worker thread's sleep time. If one job will take a lot of time the other waiting time will increase, comparing to the currently selected thread.

In the following, the file read example is considered. In this example, it can be seen that threads with the highest sleep time (A value) will not wait for a long time. It is because the thread inserts the first job after every thread executed. Additionally, since thread one creates the longest jobs, PRIO will cause threads to wait for more. With SJF, we reduce the waiting time of thread 2 significantly because thread 2 creates the shortest jobs but waits for other jobs inserted previously. When we look at the VRUNTIME we see that threads waiting times come closer to the average waiting time (do not consider thread 4, since it inserts its first job after other threads finished).

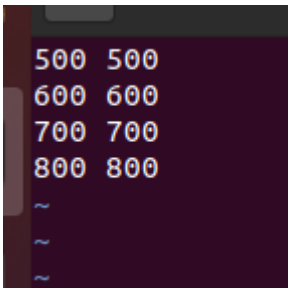
infile-1.txt



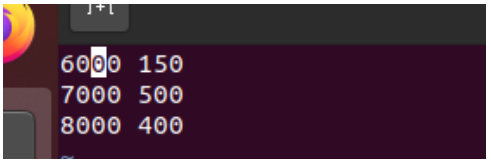
infile-2.txt



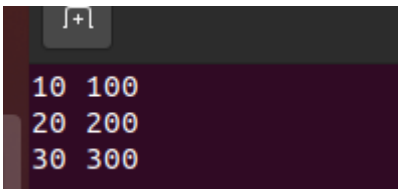
infile-3.txt



infile-4.txt



infile-5.txt



./schedule 5 FCFS -f infile

```
FINISHED
For thread 1: Average waiting time is: 181.2500ms
725 4
For thread 2: Average waiting time is: 738.6000ms
3693 5
For thread 3: Average waiting time is: 1047.2500ms
4189 4
For thread 4: Average waiting time is: 0.0000ms
0 3
For thread 5: Average waiting time is: 112.3333ms
337 3
can@ubuntu:~/Documents/project2$
```

./schedule 5 SJF -f infile

```
FINISHED
For thread 1: Average waiting time is: 183.5000ms
734 4
For thread 2: Average waiting time is: 262.8000ms
1314 5
For thread 3: Average waiting time is: 1352.5000ms
5410 4
For thread 4: Average waiting time is: 0.0000ms
0 3
For thread 5: Average waiting time is: 248.3333ms
745 3
```

./schedule 5 PRIO -f infile

```
FINISHED
For thread 1: Average waiting time is: 105.2500ms
421 4
For thread 2: Average waiting time is: 172.0000ms
860 5
For thread 3: Average waiting time is: 942.7500ms
3771 4
For thread 4: Average waiting time is: 0.3333ms
1 3
For thread 5: Average waiting time is: 1614.0000ms
4842 3
can@ubuntu:~/Documents/project2$
```

./schedule 5 VRUNTIME -f infile

```
VRUNTIME For Thread1 is 1800.000000
VRUNTIME For Thread2 is 1950.000000
VRUNTIME For Thread3 is 4160.000000
VRUNTIME For Thread4 is 1995.000000
VRUNTIME For Thread5 is 1320.000000
FINISHED
For thread 1: Average waiting time is: 182.7500ms
731 4
For thread 2: Average waiting time is: 519.6000ms
2598 5
For thread 3: Average waiting time is: 975.0000ms
3900 4
For thread 4: Average waiting time is: 0.0000ms
0 3
For thread 5: Average waiting time is: 515.6667ms
1547 3
can@ubuntu:~/Documents/project2$
```