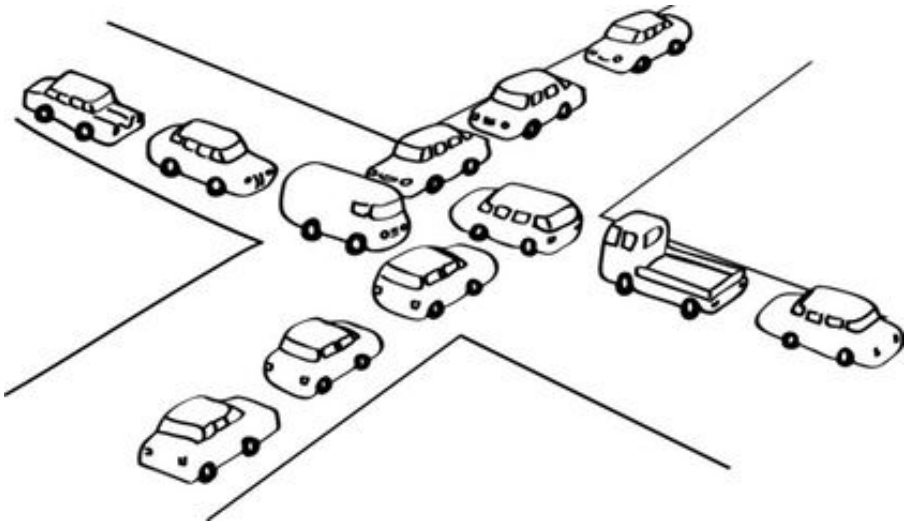
Scheduling and Deadlock Avoidance

-- Operating System Programming Project



Percy (Jiaxuan) Pan
Qun Cheng

05.18.2018

INTRODUCTION

In this project, we designed and implemented a program that simulates the job scheduling and CPU scheduling of an operating system, using c++ as our programming language.

We used Shortest Job First and First In First Out methods in job scheduling, and Round Robin method in process scheduling. Also, we used the Banker's Algorithm to implement a deadlock avoidance.

In the whole project, we separated our code into different classes and organized them by using .hpp and .cpp files.

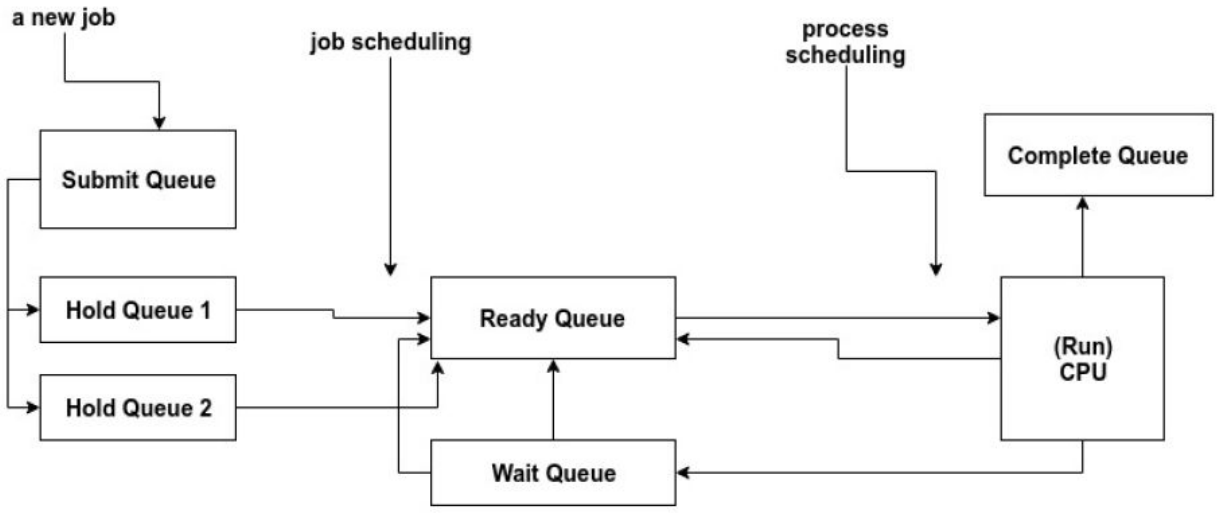
Assumptions

1. We assume a context switch will take zero time.
2. In this program, all jobs only need one type of device(resource). So in Banker's algorithm, we only consider one type of resources instead of a resource array.
3. We only consider single type of resources, and the processes in Ready Queue can only be executed one by one. In no circumstances should two or more processes go to CPU at same time.
4. The only constraints to move from one of the Hold Queues to the Ready Queue are main memory and devices.
5. If jobs have same runtime and same priority, use First In First Out (FIFO) scheduling.
6. A job only requests devices when it is running on the CPU. Therefore, every time the system notices a request input, it will check if the job ID is the running job ID on CPU. If not, the system ignores this request.
7. We handled all internal events before external. (Internal events include execution completion and quantum interruption; External event includes the arrival of a new job.) Also, there will never be two external events at the same time.

Design Approach

First, we established a system configuration based on the first command line of the input file, which contains a start time for the simulator, total main memory, total serial devices, and a time quantum (denoted as C, M, S, and Q respectively).

The following diagram describes a graphic view of the simulator which illustrates job and process transitions. We used different queues to contain jobs with different statuses. All queues are implemented as sorted linked lists and each of them has their own class to store jobs based on different requirements.



Job Scheduling

As the diagram shown above, we created three queues to contain all the upcoming jobs: Hold Queue1, Hold Queue2, and Ready Queue.

When a new job arrives, it comes with its arrival time, job ID, required main memory, max demand of devices(resources), runtime, and priority number (denoted as A, J, M, D, R, and P respectively in input file). Then one of three conditions may happen:

1. If there is not enough total main memory or total number of devices in the system for the job, the job is rejected and never gets to any of the Hold Queues.
2. If there is not enough available main memory for the job, the job is put in one of the Hold Queues, based on its priority which is already given when the job arrives (Job with a priority of 1, representing the highest priority, goes to Hold Queue1. Similarly, Hold Queue2 contains all jobs with a priority of 2). All jobs in Hold Queues will wait for enough available main memory to get to Ready Queue.

   We used different job scheduling methods to determine job sequence in the two Hold Queues: Hold Queue1 contains all jobs in Shortest Job First(SJF) order; Hold Queue2 contains all jobs in First In First Out(FIFO) order. In other words, every time the system pushes a new job with priority 1 into Hold Queue1, we will check

its runtime. a job with shorter runtime will be put before the job with longer runtime. In Hold Queue2, all jobs will be ordered based on their arrival time(FIFO).

3. If there is enough main memory for the job, then a process will be created and pushed to Ready Queue. At the meantime, the required memory will be allocated to the process.

When the system finishes the job scheduling and notices Ready Queue is not empty, processes in Ready Queue go to CPU to start process scheduling.

Process Scheduling

We used Round Robin method in process scheduling. Three queues are used in this part: Ready Queue, Complete Queue which contains all finished jobs, and Wait Queue which contains all jobs waiting for more main devices(resources) to be released.

When Ready Queue is not empty, processes stored there will be pushed into CPU one by one and start to execute. If the current running job finishes during one time quantum, the job then will be pushed into Complete Queue and release its allocated memory and devices to system. In this case, we run Banker's algorithm to check if any jobs in Wait Queue and Hold Queues can be pushed into Ready Queue. At the same time, a new job in Ready Queue can get to CPU and start execution.

When one time slice(quantum) ends, the current job in CPU will terminate and go back to the end of the Ready Queue.

Request/Release for Devices

During process scheduling, quantum interruption will happen when the system notices a device request or a device release of a certain job(Internal events). When a job requests for devices, the system first checks its job ID. The quantum is interrupted to process request only if the job ID matches the running job ID on CPU. If the two IDs are matched to each other, current job terminates and the system runs Banker's algorithm to determine if the request can be granted. If request is granted, process goes to the end of the Ready Queue and the requested devices are allocated to the job, otherwise the job goes to Wait Queue.

Similarly, when a running job releases some devices, available devices in system increases. In this case, Banker's algorithm will be run again to check if any jobs in Wait Queue and Hold Queues can be taken off and go to Ready Queue. We check Wait Queue

first and then Hold Queues.

Thus, devices released would only happen at the 1) Release process, 2) Job completed. The banker's algorithm needs to be used when devices request/release happened.

Simulator

We created Run.cpp file which contains main() method to run our simulator. To start the simulator, an input file will be requested and eventually, an output file will result.

Input

The input stream to the program describes a set of commands, including system configuration, job arrival, request for devices, release for devices and certain time nodes to display system status.

Test Case 1:

```
C 1 M=200 S=12 Q=4
A 3 J=1 M=120 S=10 R=10 P=1
A 4 J=2 M=70 S=3 R=12 P=2
Q 6 J=1 D=10
Q 7 J=2 D=3
D 8
L 9 J=1 D=4
A 10 J=3 M=10 S=8 R=4 P=1
D 11
D 20
```

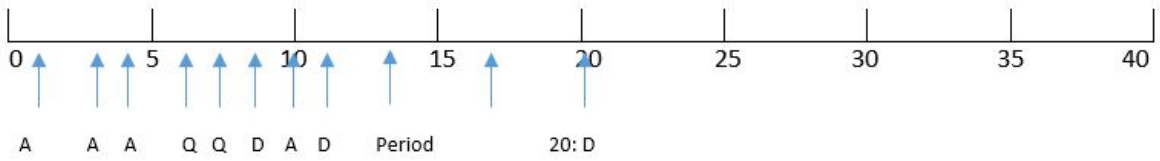
Test Case 2:

```
C 1 M=200 S=11 Q=4
A 3 J=1 M=120 S=6 R=20 P=1
A 4 J=2 M=70 S=6 R=20 P=2
A 5 J=3 M=10 S=10 R=20 P=1
Q 6 J=1 D=1
Q 7 J=2 D=1
Q 8 J=3 D=4
Q 9 J=1 D=1
Q 10 J=2 D=1
L 11 J=3 D=1
D 12
```

Test Case 1:

```
1 C 1 M=200 S=12 Q=4
2 A 3 J=1 M=120 S=10 R=10 P=1
3 A 4 J=2 M=70 S=3 R=12 P=2
4 Q 6 J=1 D=10
5 Q 7 J=2 D=3
6 D 8
7 L 9 J=1 D=4
8 A 10 J=3 M=10 S=8 R=4 P=1
9 D 11
10 D 20
```

Step Analysis



Time	Happening
1	System info comes
3	Job 1 comes, goes into ready queue, then, goes into cpu immediately. Quantum finish at 7. Job 1's start time is 3.
4	Job 2 comes, goes into ready queue, because cpu is not in idle time, job 1 is running
6	Request comes, request accepted. Job 1 get into ready queue, quantum interrupted. Job 2 goes into cpu. Quantum finished at 10. Job 2's start time is 6. Job 1's remaining time is 10-3=7. System's devices available = 12 - 10 = 2
7	Request comes, request accepted. Job 2 get into wait queue (lack devices), quantum interrupted. Job 1 goes into cpu. Quantum finished at 11. Job 2's remaining time is 11, devices request = 3
8	Print comes, Wait queue: job2 {remaining time: 11} cpu: job1 {remaining time: 6}

9	Release comes, release accepted. Quantum interrupted, job 1 goes into ready queue. Job 2 goes into ready Queue because of enough devices. Job 1 goes into cpu, Quantum finished at 13. Job 1's devices need: 7-2 = 5, system's available devices = 2 + 4 - 3 = 1
10	Job 3 comes. It goes into ready queue.
11	Print comes, Job1 which is in the cpu runs 2. Ready queue: Job2 {remaining time = 11}; job3 { remaining time = 4} Cpu: Job1 {remaining time = 3}
13	Quantum ended, Job 1 goes into ready queue. Remaining time = 3 - 2 = 1 Job 2 goes into cpu.
17	Quantum ended, Job 2 goes into ready queue. Remaining time =11 - 4 = 7 Job 3 goes into cpu. Start time = 17
20	Print comes, Ready queue: Job 1 {1} ; Job 2 {7}; Cpu :Job 3 {4}
21	**Job 3 finished goes into completed queue. Finish time = 21.** Job 1 goes into cpu.
22	**Job 1 finished, finished time =22** Job 2 goes into cpu
26	Quantum ended, job 2 goes into ready queue. Remaining time = 3 Job 2 goes into cpu
29	**Job 2 finished, finished time =26**

Programing Output

By defining print functions for each queue, we display the system status with a readable format.

For the list of jobs that have already entered the system, we provide:

1) their state(e.g. Running on the CPU, in the Hold Queue, or Complete Queue), the remaining service time for unfinished jobs, and the turnaround time and weighted turnaround time for finished jobs.
2) The contents of each queue.
3) The system turnaround time and system weighted turnaround at the last display.

Note: in this process scheduling, the running job in CPU will terminate and go back to Ready Queue; i.e. quantum interruption happens. At the same time, the first job in Ready Queue will go to CPU and start execute.

```
----- Current Time: 8 -----
******** System Infomation *********
Start Time: 1
Main Memory: 200
Device Number: 12
Quantum: 4
Available Devices: 2
Available Memory: 10

******** Hold Queue 1 ********
******** Hold Queue 2 ********
******** Ready Queue ********
******** Wait Queue ********
Job2
Arribal Time: 4
Total run time: 12
Priority: 2
Devices allocated: 0
Devices requested: 3
Start time: 6
Finish time -1
Turnaround time: -1
Remaining time: 11
Weighted turnaround time: -1

******** Completed Queue ********
******** CPU Running ********
Job1
Arribal Time: 3
Total run time: 10
Priority: 1
Devices allocated: 10
Devices requested: 10
Start time: 3
Finish time -1
Turnaround time: -1
Remaining time: 6
Weighted turnaround time: -1



-------- Get A release --------
ID: 1 max: 10 Allo: 6 Need: 4
ID: 2 max: 3 Allo: 0 Need: 3
System Available devices: 6
```

```
----- Current Time: 11 -----
******** System Infomation *********
Start Time: 1
Main Memory: 200
Device Number: 12
Quantum: 4
Available Devices: 3
Available Memory: 0

******** Hold Queue 1 ********
******** Hold Queue 2 ********
******** Ready Queue ********
Job2
Arribal Time: 4
Total run time: 12
Priority: 2
Devices allocated: 3
Devices requested: 3
Start time: 6
Finish time -1
Turnaround time: -1
Remaining time: 11
Weighted turnaround time: -1


Job3
Arribal Time: 10
Total run time: 4
Priority: 1
Devices allocated: 0
Devices requested: 0
Start time: -1
Finish time -1
Turnaround time: -1
Remaining time: 4
Weighted turnaround time: -1


******** Wait Queue ********
******** Completed Queue ********
******** CPU Running ********
Job1
Arribal Time: 3
Total run time: 10
Priority: 1
Devices allocated: 6
Devices requested: 6
Start time: 3
Finish time -1
Turnaround time: -1
Remaining time: 3
Weighted turnaround time: -1
```

```
----- Current Time: 20 -----
******** System Infomation *********
Start Time: 1
Main Memory: 200
Device Number: 12
Quantum: 4
Available Devices: 3
Available Memory: 0

******** Hold Queue 1 ********
******** Hold Queue 2 ********
******** Ready Queue ********
Job1
Arribal Time: 3
Total run time: 10
Priority: 1
Devices allocated: 6
Devices requested: 6
Start time: 3
Finish time -1
Turnaround time: -1
Remaining time: 1
Weighted turnaround time: -1


Job2
Arribal Time: 4
Total run time: 12
Priority: 2
Devices allocated: 3
Devices requested: 3
Start time: 6
Finish time -1
Turnaround time: -1
Remaining time: 7
Weighted turnaround time: -1


******** Wait Queue ********
******** Completed Queue ********
******** CPU Running ********
Job3
Arribal Time: 10
Total run time: 4
Priority: 1
Devices allocated: 0
Devices requested: 0
Start time: 17
Finish time -1
Turnaround time: -1
Remaining time: 1
Weighted turnaround time: -1
```

```
----- Current Time: 9999 -----
******** System Infomation *********
Start Time: 1
Main Memory: 200
Device Number: 12
Quantum: 4
Available Devices: 12
Available Memory: 0

******** Hold Queue 1 ********
******** Hold Queue 2 ********
******** Ready Queue ********
******** Wait Queue ********
******** Completed Queue ********
Job3
Arribal Time: 10
Total run time: 4
Priority: 1
Devices allocated: 0
Devices requested: 0
Start time: 17
Finish time 21
Turnaround time: 11
Remaining time: 0
Weighted turnaround time: 2.75

Job1
Arribal Time: 3
Total run time: 10
Priority: 1
Devices allocated: 0
Devices requested: 6
Start time: 3
Finish time 22
Turnaround time: 19
Remaining time: 0
Weighted turnaround time: 1.9

Job2
Arribal Time: 4
Total run time: 12
Priority: 2
Devices allocated: 0
Devices requested: 3
Start time: 6
Finish time 29
Turnaround time: 25
Remaining time: 0
Weighted turnaround time: 2.08333
```

Test Case 2:

```
1 C 1 M=200 S=11 Q=4
2 A 3 J=1 M=120 S=6 R=20 P=1
3 A 4 J=2 M=70 S=6 R=20 P=2
4 A 5 J=3 M=10 S=10 R=20 P=1
5 Q 6 J=1 D=1
6 Q 7 J=2 D=1
7 Q 8 J=3 D=4
8 Q 9 J=1 D=1
9 Q 10 J=2 D=1
10 L 11 J=3 D=1
11 D 12
```

Step Analysis - Important Occurs

At time 10, the request of 1 devices for job 2 is rejected because of unsafe system result, although the requested device number is smaller than current available devices.

At time 11, the release device cause increased system available devices, which makes job 2 go back to ready queue from wait queue by using banker's algorithm.

Programing Output

```
-------- currentTime: --------6

-------- Get A request --------
ID: 2 max: 6 Allo: 0 Need: 6
ID: 3 max: 10 Allo: 0 Need: 10
ID: 1 max: 6 Allo: 1 Need: 5
System Available devices: 10

-------- currentTime: --------7

-------- Get A request --------
ID: 3 max: 10 Allo: 0 Need: 10
ID: 1 max: 6 Allo: 1 Need: 5
ID: 2 max: 6 Allo: 1 Need: 5
System Available devices: 9

-------- currentTime: --------8

-------- Get A request --------
ID: 1 max: 6 Allo: 1 Need: 5
ID: 2 max: 6 Allo: 1 Need: 5
ID: 3 max: 10 Allo: 4 Need: 6
System Available devices: 5

-------- currentTime: --------9

-------- Get A request --------
ID: 2 max: 6 Allo: 1 Need: 5
ID: 3 max: 10 Allo: 4 Need: 6
ID: 1 max: 6 Allo: 2 Need: 4
System Available devices: 4

-------- currentTime: --------10

-------- Get A request --------
ID: 3 max: 10 Allo: 4 Need: 6
ID: 1 max: 6 Allo: 2 Need: 4

ID: 2 max: 6 Allo: 2 Need: 4
System Available devices: 3
System is not in safe state

-------- Get A release --------
ID: 1 max: 6 Allo: 2 Need: 4
ID: 3 max: 10 Allo: 3 Need: 7
ID: 2 max: 6 Allo: 0 Need: 6
System Available devices: 6
```

```
----- Current Time: 12 -----
******** System Infomation *********
Start Time: 1
Main Memory: 200
Device Number: 11
Quantum: 4
Available Devices: 4
Available Memory: 0

******** Hold Queue 1 ********
******** Hold Queue 2 ********
******** Ready Queue ********
Job3
Arribal Time: 5
Total run time: 20
Priority: 1
Devices allocated: 3
Devices requested: 3
Start time: 7
Finish time -1
Turnaround time: -1
Remaining time: 18
Weighted turnaround time: -1


Job2
Arribal Time: 4
Total run time: 20
Priority: 2
Devices allocated: 2
Devices requested: 2
Start time: 6
Finish time -1
Turnaround time: -1
Remaining time: 18
Weighted turnaround time: -1


******** Wait Queue ********
******** Completed Queue ********
******** CPU Running ********
Job1
Arribal Time: 3
Total run time: 20
Priority: 1
Devices allocated: 2
Devices requested: 2
Start time: 3
Finish time -1
Turnaround time: -1
Remaining time: 15
Weighted turnaround time: -1
```

```
----- Current Time: 9999 -----
******** System Infomation *********
Start Time: 1
Main Memory: 200
Device Number: 11
Quantum: 4
Available Devices: 11
Available Memory: 0

******** Hold Queue 1 ********
******** Hold Queue 2 ********
******** Ready Queue ********
******** Wait Queue ********
******** Completed Queue ********
Job1
Arribal Time: 3
Total run time: 20
Priority: 1
Devices allocated: 0
Devices requested: 2
Start time: 3
Finish time 51
Turnaround time: 48
Remaining time: 0
Weighted turnaround time: 2.4

Job3
Arribal Time: 5
Total run time: 20
Priority: 1
Devices allocated: 0
Devices requested: 3
Start time: 7
Finish time 61
Turnaround time: 56
Remaining time: 0
Weighted turnaround time: 2.8

Job2
Arribal Time: 4
Total run time: 20
Priority: 2
Devices allocated: 0
Devices requested: 2
Start time: 6
Finish time 63
Turnaround time: 59
Remaining time: 0
Weighted turnaround time: 2.95
```

REFERENCES

1. Banker's algorithm - https://www.geeksforgeeks.org/program-bankers-algorithm-set-1-safety-algorithm/