OS2021: Challenge #3: SCHSIM

Jordi Mateo Fornés `jordi.mateo@udl.cat`

University of Lleida (Campus Igualada -UdL) — **From:** November 18, 2020; **To**: January 10, 2021

Scope

One of the many responsibilities of a timesharing operating system is to provide each running program with its "fair" share of the CPU time. We have studied different algorithms to determine when a process should be allowed to run.

Objectives

- To develop graphical applications using JAVA.
- To improve your understanding of a typical process scheduling algorithm.
- To apply programming knowledge learned in other courses.
- To have fun!

Instructions

- This challenge can be delivered in groups between 2 or 3 students.
- The challenge must be delivered on GitHub and present in the Campus Virtual the link, with a summary of the tasks performed by each member.
- There are several different ways to approach these problems. It is your job to analyze them from an engineering point-of-view, determine the trade-offs, and to explain the implementation you select in the (GitHub repo).
- Do not underestimate the importance of the write-up. Your project grade depends significantly on how well you understood what you were doing, and the write-up is the best way for you to demonstrate that understanding.
- Create a READEME.md with the project information about author, workflow, design, implementation, files.
- Append to the README.md and answer the following questions: What have I learned from doing the challenge? How did I learn that? What has allowed me to improve? Why did it help me? Why can it serve me?

1 Obtain simulation parameters — (1.5 points)

The programs is required to be configured using all the information you get when you read a problem statement, you can use a form, a text file or a combination to collect this information:

1. Nº of cpus;
2. Nº of jobs with:

- Arrival time.
- Job Burst, for example: (3CPU, 2E/S, 1CPU).

3. Mode: Preemptive or Non-Preemptive
4. Algorithm: FIFO, SJF, Round Robin, Priorities.

2 Algorithm implementation — (3 points)

The simulator needs to implement the logic of different algorithms.

1. FIFO 0.5 points
2. SJF 0.5 points
3. Priorities 0.5 points
4. Round Robin 1 points
5. Design patterns, inheritance and polymorphism 0.25 points
6. Unit testing 0.25 points

3 Simulator — (1 points)

The simulator can be executed using two modes. The first one runs all the simulation and the second mode is a step-by-step mode. Where the simulation is completing the Grant diagram step-by-step.

4 Results presentation — (2.5 points)

The simulator must show a Grant diagram similar to the ones we make in class when we solved the problem, using the same notation to show the evolution (.,E,W,P,F). Moreover, we want to get a report on the main metrics.

1. Metrics 1.25 points
2. Grant Diagram 1.25 points

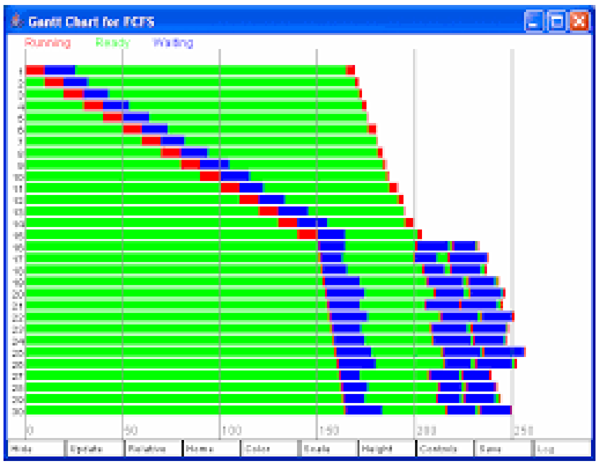
Extra points

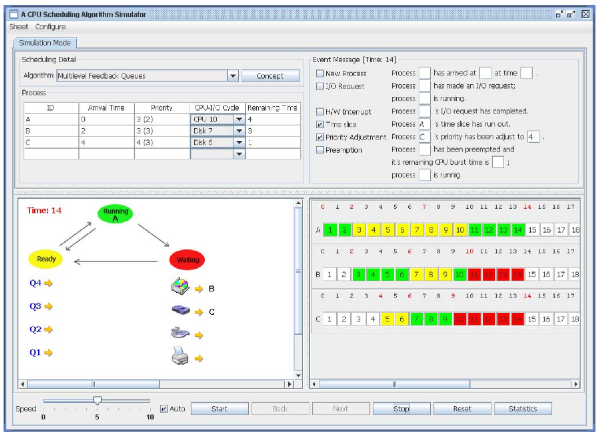
1. UX and usability. 1 points
2. Multiple CPUs. 2 points
3. Use graphs to plot system performance parameter(s) on a vertical axis vs time on the horizontal axis: CPU utilization vs time, Average Wait vs time, ... Refresh this graph as the simulation time progresses. 1 points
4. Experiment mode: run all the algorithm for the same input and make a comparative report. 1.5 points
5. Use a relational DB to read/write parameters and results. 3 points
6. Export results to PDF 0.5 points

Evaluation

1. I will focus on the design, creativity and implementation.
2. The evaluation will be a face-to-face or virtual delivery (presentation) where the groups will present their solutions, and the class or the teacher can ask them questions about the code or the design.
3. Members of a group may have different grades depending on the degree of comprehension in the face-to-face assessment.

Samples





Scheduling Detail

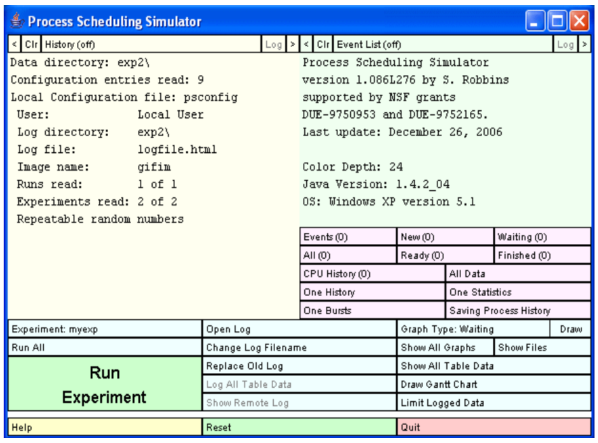
Algorithm: Multilevel Feedback Queues

Process

ID	Arrival Time	Priority	CPU-I/O Cycle	Remaining Time
A	0	3 (2)	CPU 10	4
B	2	3 (3)	Disk 7	3
C	4	4 (3)	Disk 6	1

Event Message [Time: 14]

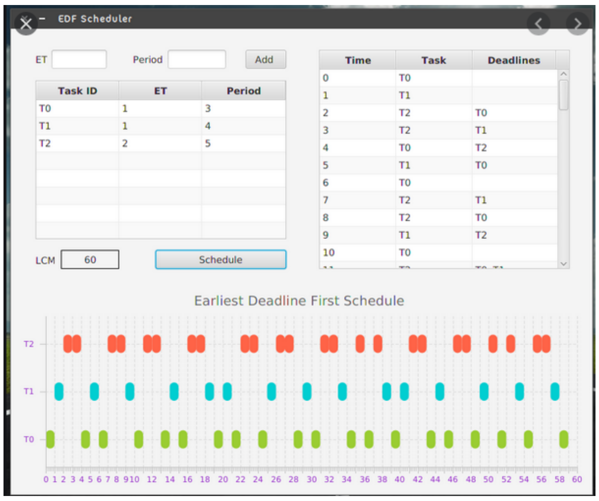
- [ ] New Process — Process ☐ has arrived at ☐ at time ☐ .
- [ ] I/O Request — Process ☐ has made an I/O request; process ☐ is running.
- [ ] H/W Interrupt — Process ☐ 's I/O request has completed.
- [x] Time slice — Process A 's time slice has run out.
- [x] Priority Adjustment — Process C 's priority has been adjust to 4 .
- [ ] Preemption — Process ☐ has been preempted and it's remaining CPU burst time is ☐ ; process ☐ is runnig.



```
Data directory: exp2\
Configuration entries read: 9
Local Configuration file: psconfig
 User:              Local User
 Log directory:     exp2\
 Log file:          logfile.html
 Image name:        gifim
 Runs read:         1 of 1
 Experiments read: 2 of 2
 Repeatable random numbers
```

```
Process Scheduling Simulator
version 1.086L276 by S. Robbins
supported by NSF grants
DUE-9750953 and DUE-9752165.
Last update: December 26, 2006

Color Depth: 24
Java Version: 1.4.2_04
OS: Windows XP version 5.1
```



Task ID	ET	Period
T0	1	3
T1	1	4
T2	2	5

Time	Task	Deadlines
0	T0	
1	T1	
2	T2	T0
3	T2	T1
4	T0	T2
5	T1	T0
6	T0	
7	T2	T1
8	T2	T0
9	T1	T2
10	T0	