CS475: Scheduler Simulation Code Review

# Questions to answer:

1. Who did you work with?
2. How is time defined?
3. How is a process modelled?
4. How are the different scheduling algorithms implemented?
5. How is multi-processor distribution accomplished?
6. Were there any bugs or issues that you fixed?
7. Explain any additional data structures.
8. What alternative data structures were considered? Why were they not chosen?
9. What alternative data structures would you have also considered?
10. Does the class hierarchy make sense? Are there other refactoring opportunities?
11. Is memory managed correctly (e.g. are pointers correctly deallocated)?
12. Are exceptions handled correctly?

|  |  |  |
| --- | --- | --- |
| Derek, Tyler, Colin | Scott | Bao, Chris, Cyruz |
| Alex, McKenna, Julia | Elisha, Geoff, Andrew |  |
| Aaron, LJ, Oscar | Zach, Zach, Drew |  |
| Kathleen, Andrew, Jarek | Marissa, Amanda, Kat |  |
| Brady, Kionte, Jason | Eli, Brodie, Clinton |  |

1. Alex Blair, McKenna Galle, and Julia Abbott reviewing Geoff, Elisha, and Andrew
2. Created an operating system class that controlled time through a scheduler. This value was passed to each algorithm.
3. Created a process object which is sent through the scheduler to control CPU ticks and execution.
4. Each algorithm is its own class and it inherits from the scheduler. That way time can be controlled for execution.
5. They used one function to run processes and used nested loops with a higher counter to simulate multiple core FCFS.
6. One of the major issues occurred in FCFS when one process finished and had a very short I/O burst, it was put back into the queue, but was also still in the execution vector. Their solution was to remove it from the vector when it got close to finishing not when it was finished. Also an issue was adapting the single runProcess function to all different algorithms and multiprocessor approaches.
7. Arrays, Vectors were used for execution. The Scheduler used a queue and a map was used for a process table. It would map to the pointer for each process.
8. A stack was considered to sort processes differently. But first in first out was needed so a queue was needed.
9. If utilizing the file system a heap may have been applicable.
10. Followed class hierarchy well from the UML. Everything works from the Scheduler class. One thing is that FCFS and SPN look very similar so they could probably be merged. Multicore’s number of cores is only used in the FCFS algorithm so this could be added to all other algorithms.
11. Pointers are properly deleted and memory is not leaked. Pointers are only used for processes and the scheduler.
12. Exceptions are handled through a try, catch block in main. Errors were fixed when they arose as opposed to try catch.