Priority Tasking

2/6/2022

Justin Davis

**Project** **Health**:

On track, though I haven’t done much since I didn’t know how the jobs would be loaded in. Most of it is just hypothetical right now.

**Summary**:

After being given information about how the jobs will be provided and with what information, I no longer need to worry about determining the priority based on memory, so I am just going to handle them based on the given priorities. For multiple jobs of the same priority, I plan on handling those with a basic Round Robin approach. I also would like to get a system in place for tracking how long jobs have been sitting in the lower ques, and if it’s greater than a set determined time their priority would get bumped up by one, to avoid them becoming stuck and never working.

**Key area 1: Loading and Prioritizing**

* Now that I know how the jobs are being provided and the information given, I need to get a working file reader that will split up the data as needed. I already have a .csv reader/parser from previous projects, so some minor adjustments to it is all that’s needed and it will work just fine. It will need to cut up each line of the .txt file provided, and store them into a job object that will have variables to hold the PID, ToA (Time of Arrival), ST (Service Time), and Priority, as well as adding a WT (Wait Time) to track how long a given job has been waiting in an inactive priority que.
* In terms of loading the data, the priorities are all provided so I just need to consider those when loading them into my que. I am planning on loading all of the jobs in at once, and then just delaying when they are put into the job que dependent on their “Time of Arrival” value. From there, a system will check for the first job with the highest priority, and then start working on that job.

**Key area 2: Round Robin**

* When there are multiple jobs in the same priority, I plan on implementing a Round Robin approach to dealing with the jobs. I’m considering reading in the ST of all the jobs in the same priority, and adjusting how much time the Round Robin provides to them based on the averaging of them. So, if there is more than one job in a que with ST’s of 5 and 15, it would attempt to give 10 units of work to each job until the que count drops to 1, at which point it would just work until another gets inserted into the priority. If a job finished in its workload allocation, then I would have it recalculate the provided workload based on the remaining jobs.
* This would be the area that I keep track of how long the program has been running (number of ticks if I can figure out how to track that). This tracking would also be used to determine when a job is loaded into the que.
* Since jobs will likely be added during one of the workloads that are being provided, I would have to make some kind of check after a given round of workload is finished to determine a new unit of workload to provide. So, if a job got added at work unit 7/10, the remaining 3 would finish, and then the system would recalculate based on the amount of work remaining for that job, and the new job in the que.

**Key area 3: Waiting Times**

* I am considering using a set time of 20 workload units for this waiting time. If a job has been sitting in a lower que for 20 units of work and hasn’t done anything, then it would get bumped up into the next level of priority. I have no clue if 20 is a good value for this, so it may change depending on how the ques end up looking when testing. I foresee this value getting increased.

**Blockers:**

As it stands, the only blocker for me was the input file and that has been remedied. Now I just need to get to working, which I may find some complications later but I don’t see anything right now that should be a blocker.

**Additional notes or highlights:**

Nothing to add right now.