COMP 304: Project 2
Spacecraft Control with Pthreads

Sinan Cem Erdoğan - 68912

Part I

In Part I structure of the program as follows:

Main function creates threads (jobs, Control Tower, Pad A, and Pad B), initializes shared time variables, queues, and mutexes. The loop in the main creates jobs (threads) at every t seconds according to the probabilities. To simulate, thread sleeps for t seconds at every iteration.

Landing, Launch, Emergency and Assembly functions/threads creates the job object, assign its variables, and enqueue them to queues according to its type.

First Land Job signals to Control Tower function/thread to start. It gives jobs to the fixed sized (1) Pad A and Pad B according to priorities of the jobs. Since it is a computer simulation, delays are not acceptable. For that reason, Control tower is ready to respond every moment of iteration.

Pad A and Pad B functions/threads takes sleeps for the size of the jobs assigned to simulate job is being processed. It sets the end and turnaround times of the jobs and then dequeue them.

Part II

Starvation for pieces waiting in the orbits may occur if more and more launch and assembly jobs keep coming. In this way, launchQueue and assembly Queues always have more than 3 job waiting and jobs in the landingQueue starve. To solve this, the priority for the landing pieces is increased gradually. If the first landing job waiting in the landingQueue waits for 16 or more seconds then the priority is given to the job.

In other words, if the following conditions holds;

(current Time – start Time) – request Time of first job in the queue) >= 16

then the priority will be given the landing pieces.

While the landing pieces waiting in the queue their priority will be increased gradually to solve starvation problem.

Part III

For implementation of Part III, the highest priority is given to the emergency job. If there is one or more emergency job, then both pads will be given to them. Note that no to complicate, there will be no preemptive actions. If the Pad A and Pad B working on another job when the emergency job/s arrive, the simulation lets them finish first then takes the emergency. Also note that because of this at some situations both emergency job is assigned to same pad. This does not affect the turnaround time of them, and this is expected behavior under these circumstances.

Note: To see how starvation implementations work you need to run the simulation with higher probability (p = 0.6 etc.).

Compilation: gcc -o project2 project_2.c -lpthread **Run:** ./project2