Novak, Kevin

Torres, Jorge

CS 433

November 3, 2013

Assignment 3 report

The included files in **novak009\_n\_torre129** are:

* Makefile - Our makefile
* Assignment3.o - The executable, as generated by the Makefile
* client.cpp - The client file
* myclasses.h - Header file containing classes for events, processes, the simulation itself, and the processor
* simulation.cpp - Where the event simulation takes place
* random.h - The header for the random number generator
* random.c - The number generator

To run our program, type **make** and that will execute the Makefile. The result will be **Assignment3.o**. To execute the program, type **./Assignment3.o** <**1** for FCFS or **2** for SJF> <**#** of processes to run>. For example, to do FCFS with 20 processes, type **./Assignment3.o 1 20**

This program simulates events on a CPU. First, it initializes a bunch of events in a priority queue, and then there is a while loop that goes until either time runs out or the jobs all finish. Within the loop, it looks at the type of every event in order of priority (either by the arrival time, for FCFS, or duration, for SJF) and determines what to do with it.

* New process events mean that a process needs to be generated and put into another priority queue, and then a CPU Burst Completion event needs to be set up.
* CPU Burst completion events mean that either a process is completed, in which case we print information about the process, or that an IO burst is needed, in which case we generate an IO Burst Completion event.
* IO Burst completion events mean that a CPU Burst is needed, so a CPU Burst Completion event is generated.

Once either time has run out, or all jobs have finished, some general statistics are printed.

Number of jobs completed for specific algorithms and load levels:

|  |  |  |
| --- | --- | --- |
| Load level \ Algorithm | FCFS | SJF |
| 10 jobs | 4 | 5 |
| 20 jobs | 7 | 11 |
| 100 jobs | 8 | 23 |

The difference between FCFS and SJF with lower loads could be chalked up to random chance, but with larger loads, FCFS levels off while SJF soars in efficiency.

SJF is always looking for the quickest thing to complete, so when it comes to any kind of turnaround statistics, it beats FCFS.