Programing Assignment 2 Report

Files included are README, pa2.cpp and this report. README tells you how to compile and run pa2.cpp to create a run of the simulator. “pa2.cpp” is the workload simulator that runs up to 10000 processes using a given lambda for arrival times and a given service time, which then prints out the Average Turnaround Time, Throughput, Utilization, and average processes in ready queue. Each completion of “pa2.cpp” counts as a run for this project. This report will share the results of 21 runs of “pa2.cpp” with lambda ranging from 10-30 and service time being 0.04s.

The program “pa2.cpp” uses a linked list to create an event queue that tracks events such as arrivals and departures. Upon each event the program saves the state of the current process and creates a new event depending on which event occurred. The event is then deleted from the queue and the next event arrives. Upon each arrival event the arrival time is recorded into its own linked list and the next arrival event is created if the server is not idle. If the server is idle, then it schedules the next departure event. For each departure event the server calculates the turnaround for the process and removes the first arrival time from the arrival time queue. The departure also creates the next departure event unless the ready queue count is 0. The process states are then averaged out upon 10000 departure events, resulting in the following output:

Graphical user interface, text, application

Description automatically generated

The program also uses a seed to make the results of the experiment replicable.

Results of the 21 runs are as follows:

What these results show is that the simulated system becomes overloaded around the lambda = 25. Which causes the throughput to cap out at 25 processes/sec and causes both the turnaround time and ready queue to explode in size. We can also notice that at 25 and after 25 Utilization is around 1.0 or 100%. This also indicates that if we keep going beyond lambda = 25 we will continue to get the same trends.