OS Process Scheduler Simulation - Individual Writeup

Adam Cassidy | Operating Systems Spring 2022

Summary:

Overall in this project I took a lot more time organizing, thinking ahead, and pre planning before actually writing any code. I had taken approaches similar to this before, but never to the extreme that I did here. What I found was that when it came to actually implementing my design, I ended up tossing much of what I had come up with anyway. While it was helpful to have a guideline to follow just from thinking out the simulation, I definitely spent too much time trying to make it perfect in theory before diving in.

Findings:

I found that in the low-core count tests, the Round Robin Low method was really fast in average turnaround and response times. The only statistic that it really lost during the 4-core tests was in wait time in IO queues. Round Robin short had the highest Throughput, second-highest Speedup, second highest CPU Utilization, fastest Average Process Turnaround and fastest Average Process Response.

When the core count was increased, the story got a bit different. While CPU Utilization went down across the board into the 20% range, Round Robin Low now had the Lowest CPU Utilization with Round Robin Medium taking the lead. In fact, for higher core count processors, it seemed that Round Robin Medium was the best option. It had the fastest Average Process Turnaround, fastest Average Process Wait, and second fastest Average Process Response. It also had the highest throughput compared to the other algorithms, however they were all very close in value.

When burst times were increased or decreased, it seemed that throughput, turnaround time, response time, and idle time all also either increased or decreased accordingly.

The best scenario that I found to use (gauged purely by throughput in processes / ms) using burst times of 30-60 ms and keeping CPU Utilization above 45% (because 300 core processor would be dumb) was as follows:

- Processor count: 16
- Algorithm used: Round Robin Medium
- RRTIME (round robin reset time) : 40 ms

Using Round Robin Low ended up with CPU Utilization below 45%, and First Come First Serve had a poorer throughput and CPU Utilization below 45%.

System Assessment:

I think that overall the design is interesting but lacks sufficient implementation, especially since priorities of processes are fixed from the beginning. It seems that choosing high priority processes to have a claim to half of the cores if needed is very arbitrary, however the reason behind that system is notable.

This system ensures that at least one process from each type of priority is going to be assigned and worked through. This makes it impossible for a process to be left in a queue indefinitely by higher priority processes, but also giving more processing power and privilege to the higher priority processes.

Role(s) in Project:

Designer, abstractor, thinker, rethinker, graphic designer, implementer, tester, snack grabber, coder, reviewer, and CEO.