High Level CPU Scheduler

Robert Jenko – g65n539

Kade Pitsch – t12r458

Alex Ekstrom – g42x664

Operating Systems

Montana State University

November 12, 2020

Introduction

Every Operating System in existence involves a wide variety of scheduling algorithms with some algorithms certainly being better than others. Our group wanted to focus our project on learning these algorithms, the positives and negatives of each of these algorithms, and how to implement them. Our final project is a web-page written in Django(a python framework library), JavaScript and HTML/CSS. The algorithms we chose to represent are First Come First Serve, Priority Based, Shortest Job First, Round Robin and Multi-Level Queue. These are all implemented in javascript that run on the client side when they access our web-page. The algorithms will be represented at a high-level with some simple(looking) animations in a HTML canvas element
We wanted our focus to be on the end user and how they can play with an algorithm and run it, being able to see at a very high-level what is going on and how things are being distributed.

Approach

We knew from the beginning that we did not want to implement something in a low-level language and deal with boring terminal output, but we also wanted a challenge. None of us have built a project in Javascript or done extensive work with the Django framework so it was a learning experience for all of us We definitely took an optimistic approach to all of this as we could not find a site to reference or anything that really suited our needs for animation packages so we started from the ground up with the basic Django frame-work and just kept (git) pushing, that being said the javascript file came out to be just short of 1,000 lines of code.

Background

Here is the information on each of the CPU scheduling algorithms and some of the pros and cons that using each entails.

First Come First Serve

The first come first serve (FCFS), sometimes referred to as the first-in, first-out (FIFO) scheduling algorithm is the simplest scheduling algorithm. When each incoming process becomes ready, it joins the ready queue and waits until the CPU is ready for the process to be executed. When the process that is currently running in the CPU comes to an end, the process that has been waiting in the queue the longest is selected next to run. FCFS performs much better for large processes. The main benefit of FCFS is it is very simple to implement and understand. However, there are many drawbacks to FCFS such as it tends to favor processor-bound processes over I/O-bound processes which can create a very inefficient use between the processor and the I/O devices. Perhaps the biggest drawback to FCFS is if small processes arrive after a large process arrives, the small processes have to wait to be helped until the large process is finished executing. This is why FCFS performs much better for large processes than small processes.

Round Robin

The round robin (RR) scheduling algorithm is a very straightforward way to prevent short jobs from suffering like they do with the FCFS algorithm. It does this by allowing each process to run for an equal amount of time. An amount of time is specified ahead of any process running; this is referred to as a clock interrupt. The order in which processes are selected to run is based on a FCFS basis. When a process is running, it is allowed to run for the specified time. If the process finishes executing before the time is up, the next process begins. However, if the process does not finish before the time is up, the process is placed back in the ready queue and the next ready process begins executing.

The main benefit of RR is it allows each process to run for an equal amount of time which allows small processes to cycle through the processor quickly. The drawback of RR is it is not efficient since it only allows large processes to run for a selected amount of time which will result in slow average turnaround time for every process.

Shortest Job First

The shortest job first (SJF), which is also referred to as the shortest process next, is another process that prevents small processes from suffering like they do with the FCFS algorithm. It does this by selecting which process is allowed to run based on the shortest expected processing time of the processes in the ready queue. The main benefit of the SJB algorithm is small jobs receive the priority which results in the overall performance to significantly improve in terms of response time. The drawbacks of SJF is even though the response time is significantly improved, the variability of response time is increased especially for large processes. An additional drawback is that the SJF needs to know the estimated processing time of each job.

Priority Scheduling

The priority scheduling (PS) algorithm executes processes based upon the order in which the programmer wants the processes to execute. The programmer will assign each process a number based upon which process they want running before the next one. When the CPU is free, it will choose the process with the highest priority to execute. The benefit of PS is to allow the most important processes to execute before the less important processes regardless of when it arrived at the ready queue or its estimated execution time. The drawback to PS is the lack of efficiency it possesses, however, thankfully this algorithm is used based on priority of the processes and not used based on efficiency.

Multi-Level

The multi-level (ML) algorithm combines multiple other scheduling algorithms to execute processes from multiple queues. The ML uses a priority like scheme to choose which queue will execute before other queues. The highest level queue will be executed first, unless it is empty, in which the next highest queue is executed instead. In our simulation, the first level queue with the highest priority processes uses RR scheduling to execute the processes. The second level queue uses SJF scheduling, and the lowest level queue uses PS. The benefit of using the ML algorithm is the programmer can decide which scheduling algorithm they will use to execute each queue. The drawback to the ML algorithm is some processes might starve for the CPU if some of the higher priority queues are not becoming empty.

Results

Our website takes in an algorithm option like first come first serve and we run that through our javascript function(s) and will animate it accordingly. We have different animation techniques for different algorithms although some of them are the same. While we are animating each job we are also recording each job and how long it was running, the burst time and more. Then after we get done with all the animation and recording each job time we output that information to an HTML table to be nice and human readable. We achieved what we had in mind for the end result although we would like to flesh out the idea more and add to it. More algorithms, better animations, more interactivity...etc. we have several ideas in mind and maybe we will continue to work on this.

Summary

After completing this proof of concept, our group was able to gain hands-on experience of what it takes to program a very important part of an operating system. Our group was able to gain a substantial amount of knowledge of different CPU scheduling algorithms.

After getting through the site we Most importantly, implementing these algorithms allowed us to visually see how these algorithms are structured and executed. For possible future work we plan on publishing our website through an EC2 instance with AWS. Once that is done we would also like to implement more scheduling algorithms and perhaps add more simulations involving other topics related to operating systems such as how threading works .

Resources

Stallings, William. "Scheduling Algorithms." *Operating Systems Internals and Design Principles*, Ninth Edition, Pearson, 2018, pp. 430-452.

https://docs.djangoproject.com/en/3.1/

https://www.youtube.com

https://stackoverflow.com