Author: Mohamed Eladl
Date: March 21, 2020

CPU Scheduling:

CPU scheduling allows for operating systems that support multiprogramming. This is achieved by context switching between processes to increase CPU productivity. Multiprogramming increases CPU utilization by organizing processes so that the CPU always has one to execute. CPU scheduling was designed to make systems efficient and impartial to the processes that needed system resources. Before multiprogramming environments existed, the CPU would only focus on one process at a time. With CPU scheduling, idle time is shortened because the operating system chooses a process to utilize the CPU from memory while another process is in waiting or execution state. This is known as the short-term scheduler.

CPU Scheduling consists of:

> ***CPU Utilization*** - using CPU as much as possible
>
> ***Throughput*** - number of processes completed execution in a given time period
>
> ***Turnaround time*** - the amount of time to execute a specific process
>
> ***Wait time*** – amount of time a process has been in the ready queue
>
> ***Response time*** – amount of time it takes for a process request to receive an initial response

While adhering to these criterion, different scheduling algorithms were developed to maximize CPU utilization. They include:

1. First Come First Serve
2. Shortest Job First
3. Priority Scheduling
4. Round Robin
5. Multilevel Queue Scheduling
6. Multilevel Feedback Queue Scheduling

Round Robin:

Round Robin is one of several scheduling algorithms used to maximize CPU utilization. This algorithm is preemptive and similar to the First Come First Serve scheduling algorithm. In Round Robin each process is given a set period of time to complete its execution. This is called the time quantum. After the time quantum has ended, the process will be preempted and added to the back of the ready queue. Each process has a burst time that decreases as the process is in execution and if it is not completed within the time quantum, the process gets preempted and goes through the cycle again. When the counter is at 0, an interrupt is generated to stop the execution of the process and the schedule the next process. The state of that process must be saved before moving to the next process. This is where context switching occurs. However, context switching comes with overhead even though average response times are improved. There can be different time quantum set by the operating system.

Code Execution Instructions:

1. Copy the path of the folder with the files
2. Open command prompt.
3. Change directories to the path of your folder.
4. Convert the java files to bytecode class files using the “javac” command.
5. In the zip file you will find in Running File a csv file containing processes with their ID, arrive time, and burst time. Run the command

    java RoundRobin processes.csv [Time Quantum]
    java RoundRobin processes.csv 3

Analysis:

Based on the output of this scheduler, the larger the time quantum, the shorter the wait time and turnaround time was. The smaller the time quantum, the more wait time there was and more turnaround time. When the time quantum was 1, the turnaround time was 5.25 and the wait time was 2.75 roughly. If we look at the largest time quantum, 10, the turnaround time was 4.5 and wait time was 2.0. With these times, you can clearly see that more context switches are needed to complete the processes requests with the smaller time quantum. With the larger time quantum, the process has more time to execute, requiring less context switches when the quantum has elapsed.

As the time quantum got larger the results started to become similar until they were exactly the same. This started at around quantum 5. The turnaround time, average wait time, number of context switches, CPU utilization, and throughput were all the same. Prior to time quantum 5 there were minor differences in context switches or time averages between time quantum 4 and

their context switches. Seeing that the number of context switches plateau's at time quantum 5, it would be best to set the scheduler here. There is no difference between time quantum 5 and time quantum 10 because of the processes in the csv file. If the processes and their corelating arrive and burst times were different, the time quantum would need to be adjusted for efficiency.