Shiraz university

January 20, 2020

Project Report

CPU scheduling

Purpose of this project is to implement the algorithms of CPU scheduling. We have chosen python as our programming language.

There is a main function to call our functions.

At first, we should have a method to read inputs. Our inputs are in a csv file, there is a library that reads csv files, its name is csv. So we implement a read function using csv, this function reads csv file by row. First row is title so we skip this row. After that every other row is our input, first column is process id, second is arrival time and the last one is burst time.

When we got our inputs, we should new a process from process class. Constructor of this class take process id, arrival time and burst time. We need to know when a process is finished and when it is started, so we have these attributes too. We make a list of processes to pass to our algorithms.

Our first algorithms is FCFS(first come first serve). How it works? There is a queue, each process with lower arrival time will place in higher position in queue and will serve first. To implement this algorithm, we need a list as our queue that is ordered by process arrival time. Then we serve the process in the list by their indexes. If the CPU is idle and there is no process in that time we should set the CPU time to next arrival time.

R. R (round robin). this algorithm has a quantum time, each process have just as much as quantum time to execute. If the process didn't finish in that time we should take that process to the end of the queue. So we need to what processes are in the ready queue and what processes didn't even arrive yet. We use CPU time to know which process is arrived. Also, we need to know which processes has finished so we need a list to show which process has entered to ready queue and now is not in this queue.

For SPN algorithm we decided to have a class. In this class, there's a method that chooses a process with the least CPU burst as the next process to be executed. A while loop starts until all processes are executed. In each loop, after the process is done, the mentioned method is called again to determine the next process.

Since SRT algorithm is a preemptive version of SPN, we used the same way for implementation. However in this algorithm, after each millisecond elapsed, the method checks whether there is a process with less remaining time than the one executing currently or not. If so, a context switch will occur and the other process starts executing.

Outputs. When algorithms work has finished we should now produce output. So, what do we need? A. W. T (average waiting time), A. R. T (average response time), A. T. T (average turnaround time), CPU utilization and through put. Waiting time is the time each process waits in the queue to serve for the first time so we should sub start time from arrival time to have waiting time. Response time is the time each process finishes. Turnaround time is the time each process takes to finish so we have to sub end time from arrival time. CPU utilization is how much CPU is used in specific time so we have to divide burst time of all processes from last process end time. Through put is how many processes is finished in a specific time so we have to divide number of processes from last process end time.

Print output. We want to write our outputs to a csv file so again we use csv library provided by python. It has a write method that we use easily to write in a csv file.