Report of Assignment 3

Name: Runyu Xu

Date: March 29, 2018

Course: CS433 Operating Systems

Assignment: 3 CPU scheduling

Description:

This project is a simple Discrete Event Simulation (DES) to analyze different CPU scheduling algorithms. In my program, it will run for a total 5 minutes and report the result of the simulation with FCFS and SJF algorithms.

Implementation:

In my project, there are two Object class Process and Event. Process class will create processes which will be executed. Event class has four types of event and implement Process class. In each event, it will have a process and startTime.

Four types of event:

1) Process Arrival: This event represents the arrival of a new process in the ready queue. When handled, new processes are pushed into the ready queue and the CPU scheduler checks if the CPU is idle, then dispatches a process.

2) CPU Burst Completion: This event represents the completion of a processes burst time. When handled, it checks if total duration time of a process is end. If completed, it removes the process from the CPU. If not completed, it generates an I/O burst time and creates an I/O completion event.

3) I/O Completion: This event represents the completion of a processes I/O burst time. When handled, it generates a CPU burst time for the process and creates a CPU Burst Completion event.

4) Timer Completion: This event is used with preemptive scheduling algorithms for when a process's burst time exceeds the maximum time slice allowed for that process. When handled, it moves the current job from the CPU back to the ready queue and updates the remaining CPU burst time of the event.

The other parts in my project are the simulations. I create a general simulation, and both FCFS and SJF are its descendants.

The difference between FCFS and SJF classes is that FCFS uses a general queue to hold all processes in order by their coming time, while SJF uses a priority queue based on the CPU burst time of process.

Analysis:

FCFS scheduling

	10 processes	**20 processes**	**50 processes**
Completed Jobs	**6**	**7**	**3**
CPU Utilization	**59.23 %**	**93.98 %**	**98.66 %**
Throughput	**0.002 jobs/sec**	**0.001 jobs/sec**	**0.000 jobs/sec**
Avg turnaround	**119.80 sec**	**365.79 sec**	**340.71 sec**
Avg waiting time	**77.20 sec**	**117.58 sec**	**148.94 sec**

SJF scheduling

	10 processes	**20 processes**	**50 processes**
Completed Jobs	**6**	**10**	**14**
CPU Utilization	**59.07 %**	**93.81 %**	**98.65 %**
Throughput	**0.002 jobs/sec**	**0.002 jobs/sec**	**0.001 jobs/sec**
Avg turnaround	**117.77 sec**	**195.85 sec**	**168.19 sec**
Avg waiting time	**77.13 sec**	**82.60 sec**	**105.92 sec**

According to these two result forms, I find that in general, SJF scheduling algorithm has better performance than FCFS scheduling algorithm.

With very few processes (which is like the case 10 processes), SJF and FCFS have almost the same performance, SJF is a little better with average turnaround time and average waiting time than FCFS.

After adding more and more processes into the simulation in the same time duration (case 20 processes and case 50 processes), we can easily see that, with the nearly CPU utilization, SJF can finish more jobs with higher throughput, lower average turnaround time and average waiting time.

Included Files:

Header:

- event.h

- process.h

- simulation.h

- FCFS.h

- SJF.h

- random.h

Implementation:

- event.cpp
- process.cpp
- simulation.cpp
- FCFS.cpp
- SJF.cpp
- random.cpp
- main.cpp

Makefile:

- Makefile

Use make all command to execute the Makefile

And then type: ./cpusim.exe to run the program