CS376 – Weekly Assignment Covering Week 10

Dalton Rothenberger

1. Compare and contrast non-pre-emptive and pre-emptive scheduling. Include in your discussion: the complexity of the operating system, overhead involved, benefits of each. Be thorough.
    a. Pre-emptive scheduling is when a process in the CPU can be taken out before it is done or errors. What this means is that a process can switch from running state to the ready stat or when a process switches from the waiting state to the ready state. Non-pre-emptive cannot be taken out of the CPU until completion or when it is waiting. This is when a process switches from the running state to the waiting state or when a process terminates. Pre-emptive scheduling is more complex because it involves swapping in and out processes frequently whereas non-pre-emptive only swaps when the process is complete or from running to waiting. Pre-emptive has more overhead because of switching processes whereas non-pre-emptive has less overhead of swapping processes around. With pre-emptive if a high priority process arrives then it can be swapped over and addressed but non-pre-emptive cannot do this. In non-pre-emptive larger burst processes can cause small burst processes to starve.
2. Describe what happens when a context switch occurs. What exactly is happening and why? What data structures are involved?
    a. A context switch is when the CPU changes from one task to another while ensuring that the tasks do not conflict. The data structures involved are the PCB and the process table. The PCB of the current running process gets swapped with the PCB of the process that is being put in which is pulled from the process table. The PCB of the current process is put into the process table. This is done because the context of the process needs to be saved away with it so that when the process resumes it can continue where it left off.
3. Consider the following processes and their burst times. Compute the turnaround and wait time for SJF, RR (tick=3), and FCFS algorithms. Which algorithm gives the shortest average turnaround and wait times? Draw a Gantt chart depicting the execution of each the execution for each scheduling algorithm. Assume they arrive in order A, B, C, D, E, F G all at time 0

SJF RR FCFS

Process	Turnaround	Wait time		Process	Turnaround	Wait time		Process	Turnaround	Wait time
A	29	19		A	47	37		A	10	0
B	10	4		B	25	19		B	16	10
C	2	0		C	8	6		C	18	16
D	41	29		D	50	38		D	30	18
E	86	41		E	86	41		E	75	30
F	4	2		F	16	14		F	77	75
G	19	10		G	46	37		G	86	77

Algo	**Avg Turnaround**	**Avg Wait time**
SJF	27.29	15
RR	39.71	27.43
FCFS	44.57	32.29

SJF gave the shortest average turnaround time and shortest average wait time.

Gantt Charts

SJF

C	F	B	G	A	D	E
0 2	4	10	19	29	41	86

RR

A	B	C	D	E	F	G	A	B	D	E	G	A	D	E	G	A	D	E
0 3	6	8	11	14	16	19	22	25	28	31	34	37	40	43	46	47	50	86

FCFS

A	B	C	D	E	F	G
0 10	16	18	30	75	77	86

Process	Burst
A	10
B	6
C	2
D	12
E	45
F	2
G	9

4. Give a detailed algorithm that will implement SJF scheduling algorithm with ageing of processes.
    a. Loop while processes:
        i. Sort current processes so in ascending order, so that the shortest burst is first, and the longest burst is last
        ii. Pick the first process and run it.
        iii. Remove the first process from the data structure
        iv. Decrement the burst durations of remaining processes by 1 (Note: this does not change the run duration, but the duration used when sorting)
        v. Accept new process arrivals into the process data structure