University of Calgary

CPSC 457 - Principles of Operating Systems

Professor: Dr. Pavol Federl

Assignment #3

Multi-Threading and Scheduling

Evan Loughlin

Student ID: 00503393

Tutorial Section: T04

TA: Sina Keshvadi (AKA: John Cena)

March 12, 2018

Q1

Suppose a program spends 90% of time doing I/O. Calculate the CPU utilization when running 5 copies of such a program. Assume one single-core CPU, and all I/O operations are parallel.

The CPU utilization is taken as one minus the probability that all processes are simultaneously waiting for I/O.

$$\text{CPU Utilization} = 1 - p^n = 1 - (0.9)^5 = 0.41 = 41\%$$

Q2

Consider the following set of processes (all times given in seconds):

Process	Arrival Time	Burst Time	Completion Time	Turnaround Time	Waiting Time
P1	0	10	14	14	4
P2	1	1	2	1	0
P3	3	2	5	2	0
P4	5	1	6	1	0
P5	9	5	19	10	5

SRTN Algorithm : Shortest Remaining Time Next algorithm. With this algorithm, the scheduler always chooses the process whose remaining run time is the shortest. Here, the time has to be known in advance. When a new job arrives, its total time is compared to the current process' remaining time. If the new job needs less time to finish than the current process, the current process is suspended and the new job started. This scheme allows new jobs to get good service. In this example, ties are broken using FCFS.

P_1	P_2	P_1	P_3	P_4	P_1	P_5

0 1 2 3 5 6 14 19

Average wait time:

$$\text{Waiting Time} = \text{Turnaround Time - Burst Time}$$
$$\text{Turnaround Time} = \text{Completion Time - Arrival Time}$$

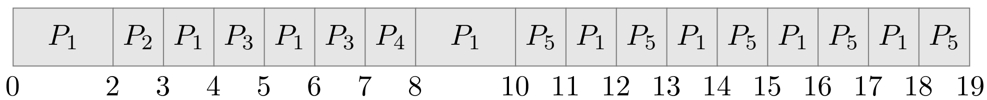
From the above table, the average wait time is $(4 + 0 + 0 + 0 + 5)/5 = 1.8s$.

Q3

Gantt Chart illustrating the execution of these processes (from Q2) using the RR scheduling algorithm with 1sec quantum. FCFS is used to break ties.

The RR technique involves tracking a priority queue as well as a quantum (the amount of time any given process will run for). For this example, ties are broken using FCFS.

Process	Arrival Time	Burst Time	Completion Time	Turnaround Time	Waiting Time
P1	0	10	18	18	8
P2	1	1	3	2	1
P3	3	2	7	4	2
P4	5	1	8	3	2
P5	9	5	19	10	5



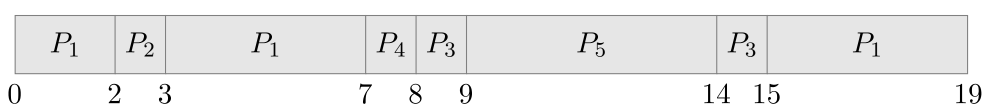
Number of context switches: 13
Average wait time = $(8 + 1 + 2 + 2 + 5)/5 = 3.6s$

Q4

Same as Q2 except:

- P1 and P5 are high-priority processes (RR, Q=2s)
- P2 and P4 are medium priority processes (RR, Q=4s)
- P3 is a low priority process (FCFS)

Process	Arrival Time	Burst Time	Completion Time	Turnaround Time	Waiting Time
P1	0	10	19	19	9
P2	1	1	3	2	1
P3	3	2	15	12	10
P4	5	1	8	3	2
P5	9	5	14	5	0



Average wait time = $(22)/5 = 4.4s$

Q5

Refer to count.cpp

Q6

Time testing of count.cpp:

Test file: medium.txt

# Threads	Observed Timing	Observed Speedup Compared to Original	Expected Speedup
Original Program	?	?	1.0
1	?	?	1.0
2	?	?	2.0
3	?	?	3.0
4	?	?	4.0
8	?	?	8.0
16	?	?	16.0

Test file: hard.txt

# Threads	Observed Timing	Observed Speedup Compared to Original	Expected Speedup
Original Program	?	?	1.0
1	?	?	1.0
2	?	?	2.0
3	?	?	3.0
4	?	?	4.0
8	?	?	8.0
16	?	?	16.0

Test file: hard2.txt

# Threads	Observed Timing	Observed Speedup Compared to Original	Expected Speedup
Original Program	?	?	1.0
1	?	?	1.0
2	?	?	2.0
3	?	?	3.0
4	?	?	4.0
8	?	?	8.0
16	?	?	16.0

Q7

Refer to scheduler.cpp