CIS 415 – Operating Systems

Homework Assignment 2

Textbook Questions (48 points)

1. A system with two dual-core processors has four processors available for scheduling. A CPU-intensive application is running on this system. All input is performed at program start-up, when a single file must be opened. Similarly, all output is performed just before the program terminates, when the program results must be written to a single file. Between start-up and termination, the program is entirely CPU-bound. Your task is to improve the performance of this application by multithreading it. The application runs on a system that uses the one-to-one threading model (each user thread maps to a kernel thread).
    a. How many threads will you create to perform the input and output? Explain. (3 points)
    It only makes sense to create as many threads as there are blocking system calls, as the threads will be spent blocking. Creating additional threads provides no benefit. Thus, it makes sense to create a single thread for input and a single thread for output.
    b. How many threads will you create for the CPU-intensive portion of the application? Explain. (3 points)
    Four. There should be as many threads as there are processing cores. Fewer would be a waste of processing resources, and any number > 4 would be scheduled on the 4 cores.

2. Consider a multicore system and a multithreaded program written using the many-to-many threading model. Let the number of user-level threads in the program be greater than the number of processing cores in the system. Discuss the performance implications of the following scenarios. (9 points)
    a. The number of kernel threads allocated to the program is less than the number of processing cores.
    b. The number of kernel threads allocated to the program is equal to the number of processing cores.
    c. The number of kernel threads allocated to the program is greater than the number of processing cores but is still less than the number of user-level threads.
    When the number of kernel threads is less than the number of processors, then some of the processors would remain idle since the scheduler maps only kernel threads to processors and not user-level threads to processors. When the number of kernel threads is exactly equal to the number of processors, then it is possible that all of the processors might be utilized simultaneously. However,when a kernel-thread blocks inside the kernel (due to a page fault or while invoking system calls), the corresponding processor would remain idle. When there are more kernel threads than processors, a blocked kernel thread could be swapped out in favor of another kernel thread that is ready to execute, thereby increasing the utilization of the multiprocessor system.

3. Suppose that a short-term CPU scheduling algorithm favors those processes that have used the least processor time in the recent past. Why will this algorithm favor I/O-bound processes and yet not permanently starve CPU-bound programs? (6 points)

It will favor the I/O-bound programs because of the relatively short CPU bursts requested by them; however, the CPU-bound programs will not starve because the I/O-bound programs will relinquish the CPU relatively often to do their I/O

4. Discuss how the following pairs of scheduling criteria conflict in certain settings (9 points):
   a. CPU utilization and response time
      CPU utilization is increased if the overheads associated with context switching is minimized. The context switching overheads could be lowered by performing context switches infrequently. This could, however, result in increasing the response time for processes.
   b. Average turnaround time and maximum waiting time
      Average turnaround time is minimized by executing the shortest tasks first. Such a scheduling policy could, however, starve long-running tasks and thereby increase their waiting time.
   c. I/O device utilization and CPU utilization
      CPU utilization is maximized by running long-running CPU-bound tasks without performing context switches. I/O device utilization is maximized by scheduling I/O-bound jobs as soon as they become ready to run, thereby incurring the overheads of context switches.

5. [18 points]
   Suppose that a disk drive has 5,000 cylinders, numbered 0 to 4999. The drive is currently serving a request at cylinder 2150, and the previous request was at cylinder 1805. The queue of pending requests, in FIFO order, is:

   2069, 1212, 2296, 2800, 544, 1618, 356, 1523, 4965, 3681

   Starting from the current head position, what is the total distance (in cylinders) that the disk arm moves to satisfy all the pending requests for each of the following disk-scheduling algorithms?
   a. FCFS

current	next	cylinders
2150	2069	81
2069	1212	857
1212	2296	1084
2296	2800	504
2800	544	2256
544	1618	1074
1618	356	1262
356	1523	1167
1523	4965	3442
4965	3681	1284
TOTAL		13,011

b. SSTF

current	next	cylinders
2150	2069	81
2069	2296	227
2296	2800	504
2800	3681	881
3681	4965	1284
4965	1618	3347
1618	1523	95
1523	1212	311
1212	544	668
544	356	188
TOTAL		7,586

c. SCAN

current	next	cylinders
2150	2296	146
2296	2800	504
2800	3681	881
3681	4965	1284
4965	4999	34
4999	2069	2930
2069	1618	451
1618	1523	95
1523	1212	311
1212	544	668
544	356	188
TOTAL		7,492

shorthand is (4999-2150) + (4999-356) = 2849 + 4643 = 7,492

d. LOOK

current	next	cylinders
2150	2296	146
2296	2800	504
2800	3681	881
3681	4965	1284
4965	2069	2896

2069	1618	451
1618	1523	95
1523	1212	311
1212	544	668
544	356	188
	TOTAL	7,424

shorthand is (4965-2150) + (4965-356) = 2815 + 4609 = 7,424

e. C-SCAN

current	next	cylinders
2150	2296	146
2296	2800	504
2800	3681	881
3681	4965	1284
4965	4999	34
4999	0	4999
0	356	356
356	544	188
544	1212	668
1212	1523	311
1523	1618	95
1618	2069	451
	TOTAL	9,918

shorthand is (4999-2150) + 4999 + (2069-0) = 2849 + 4999 + 2069 = 9,917

f. C-LOOK

current	next	cylinders
2150	2296	146
2296	2800	504
2800	3681	881
3681	4965	1284
4965	356	4609
356	544	188
544	1212	668
1212	1523	311
1523	1618	95
1618	2069	451
	TOTAL	9,137

shorthand is (4965-2150) + (4965-356) + (2069 - 356) = 2815 + 4609 + 1713 = 9,137