Operating Systems–II: CS3523
Spring 2022
Theory Assignment 1: Chapter 6, 7 from the book

Submission Date: 21/02/2022, 9:00 PM

1. Consider the atomic ‘increment’ function discussed in the book (page 270 of the pdf) using hardware instruction ‘compare_and_swap’. As of now, there is no guarantee that a thread invokind this function will terminate. So, can you develop an ‘atomic’ increment function that will eventually terminate?

2. In the class we discussed the solution to the reader-writers problem using semaphores. We saw that this solution can cause the writer threads to starve. Please develop an alternative solution in which neither the reader nor the writers will starve. For this you can assume that the underlying semaphore queue is fair.

3. Exercise 6.11 from the book.

4. Exercise 6.12 from the book.

6.11 one approach for using compare and swap() for implementing a spin-lock is as follows:

```
void lock spinlock(int *lock) {
while (compare and swap(lock, 0, 1) != 0)
; /* spin */
}
```

A suggested alternative approach is to use the “compare and compare-and-swap” idiom, which checks the status of the lock before invoking the compare and swap() operation. (The rationale behind this approach is to invoke compare and swap()only if the lock is currently available.) This strategy is shown below:

```
void lock spinlock(int *lock) {
{
while (true) {
if (*lock == 0) {
/* lock appears to be available */
if (!compare and swap(lock, 0, 1))
break;
}
}
}
```

Does this “compare and compare-and-swap” idiom work appropriately for implementing spinlocks? If so, explain. If not, illustrate how the integrity of the lock is compromised.

6.12 Some semaphore implementations provide a function getValue() that returns the current value of a semaphore. This function may, for instance, be invoked prior to calling wait() so that a process will only call wait() if the value of the semaphore is > 0, thereby preventing blocking while waiting for the semaphore. For example:

```
if (getValue(&sem) > 0)
wait(&sem);
```

Many developers argue against such a function and discourage its use. Describe a potential problem that could occur when using the function getValue() in this scenario