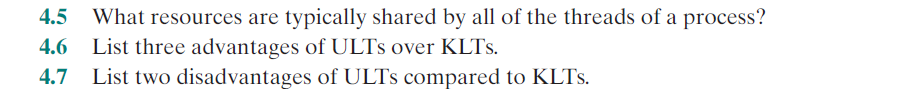
Gaurav Haral

Chapter 4

Review questions



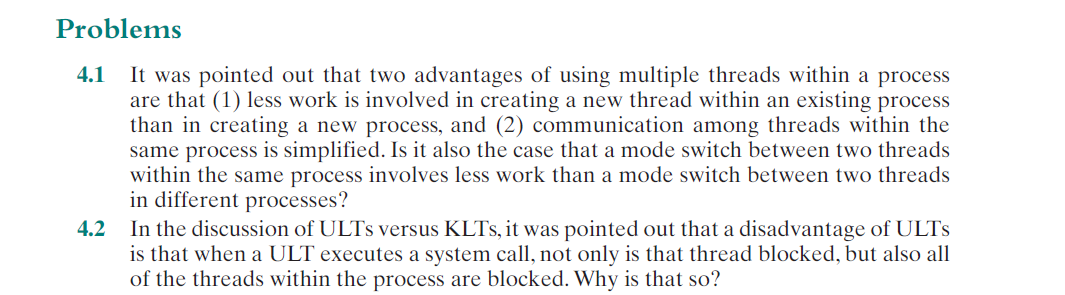
Ans 4.5 There are some resources shared by different threads of the same process while some are not. The threads share the address space, file, global variables. But each threads has its own stack of its copy of registers.

Ans 4.6

1. Thread switching does not require kernel mode privileges because all of the thread management data structures are within the user address space of a single process. Therefore, the process does not switch to the kernel mode to do thread management. This saves the overhead of two mode switches (user to kernel; kernel back to user).  
      
   2. Scheduling can be application specific. One application may profit most from a simple round based scheduling algorithm, while another might profit from a priority-based scheduling algorithm. The scheduling algorithm can be tailored to the application without disturbing the underlying OS scheduler.  
      
   3. ULTs can run on any OS.

Ans 4.7 1) When user level thread executes a system call, not only the particular thread blocked all of the threads in the process are blocked.

2) in user level threads, only a single thread within a process can execute at a time, so multiprocessing cant be implement.



Ans 4.1 Yes, because more state information must be saved to switch from one process to another.

Advantages of two threads communicating in same process are:

1. threads have read privileges of files from other threads in same process.
2. Less time is needed to create, switch and terminate threads in same process than other processes.

4.2

a disadvantage of ULTs is that when a ULT executes a system call, not only is that thread blocked, but also all of the threads within the process are blocked.

Because the user process functions separately from Kernel processes. It is a kind of thread structure process, So it is not visible to the OS/kernel, which schedules on the basis of processes.