King Saud University
Department of Computer Science
CSC227: Operating Systems
Tutorial No. 4

Q 1) Give programming examples in which multithreading provides better performance than a single-threaded solution.

Answer

1) A web server that services each request in a separate thread.
2) A web browser with separate threads playing sound, downloading a file, collecting user input, etc.
3) A word processor with a thread to save, at regular intervals, the file that is currently being executed, and another thread as a spell-checker, etc.
4) An application such as matrix multiplication where each row of the matrix product is evaluated, in parallel, by a different thread.

Q 2) What is the difference between a process and a thread? Which one consumes more resources?

Answer

A process defines the address space, text, resources, etc. A thread defines a single sequential execution stream within a process. Threads are bound to a single process. Each process may have multiple threads of control within it.
It is much easier to communicate between threads than between processes. It is easy for threads to accidentally disrupt each other since they share the entire address space.
Process consumes more resources.

Q 3) Context switching between kernel threads typically requires saving some items in its TCB. Mention two such items.

Answer

Thread's register values; Thread's stack pointer; State of the thread; Program counter; Pointer to the PCB of the process that the thread lives on.

Q 4) Assume that the OS implements Many-to-Many multithreading model. What is the minimum number of kernel threads required to achieve better concurrency than in the Many-to-One model? Why?

Answer

At least two kernel threads are required to achieve better concurrency in the many-to many model than in the Many-to-One model. With more than two kernel threads, CPU can still schedule other threads for execution when one threads is performing a blocking system calls.

Q 5) Consider two scenarios: a) two user threads are mapped into one kernel thread, and b) each of the two user threads is mapped into a unique kernel thread. Which achieves better concurrency in execution? Why?

Answer

The b) achieves better concurrency since two threads can execute concurrently.

In a), when one thread is blocked or in waiting state, the other thread cannot be scheduled to run.

Q 6) Can a multithreaded solution using multiple user-level threads achieve better performance on a multiprocessor system than on a single-processor system?

Answer

A multithreaded system consisting of multiple user-level threads mapped to one kernel thread cannot make use of the different processors in a multiprocessor system. Consequently, there is no performance benefit associated with this solution. The multithreaded solution could be faster if the multiple user-level threads are mapped to different kernel threads.