CSL862 Minor 2
Advanced Topics in Operating Systems

6 October 2012 Max. Marks: 20

Scheduler Activations

1. One way to implement kernel-level threads functionality with user-level threads performance is to treat each kernel-level thread as a separate virtual CPU and let the user-level thread scheduler schedule on these CPUs. Describe what are some problems with this model. Each problem should be described briefly. [3.5]

2. In scheduler activations, the kernel need not obtain permission in advance from an address space to steal its processor. This is in contrast with the exokernel approach, where the kernel explicitly obtains permission from a process before pre-empting its processor. Contrast the two approaches in how they handle a process executing inside a critical section. Compare on both correctness/security and performance. Discuss all the interesting tradeoffs briefly. [3.5]

Transactional Memory

3. In the Herlihy '93 paper (first paper on transactional memory), an orphan is a transaction that continues to execute after it has been aborted. Why do they continue to execute a transaction even after it has aborted? What are some things a programmer should be careful about while executing an orphan transaction? How? [3]

4. Look at the “Counting benchmark” in Herlihy '93 paper. Implement the same program using TCC (Transactional Memory Coherence and Consistency). [2]

5. Explain how nested locks are handled in Speculative/Hardware Lock Elision. [2]

6. Why do you think that Intel's Hardware Lock Elision mode in Haswell architecture does not support eliding recursive locks. [2]

7. The Haswell architecture does not implement “Virtual Transactional Memory” as discussed in the Rajwar '05 paper. What would be some important differences in the programming model if VirtualTM was supported? [2]

8. In Virtual TM, the transaction address data table (XADT) is a global data structure. i.e., common to all transactions sharing the address space. Why? [2]