6.033 Quiz 1 Review Session

March 7, 2007

Virtualization

Performance

Client/server organization within a computer using virtualization

- abstractions for virtualizing computers
  - threads
  - VM
  - send/receive (bounded buffers)
- emulation and virtual machines (VMMs)

Virtual links using send, receive, and a bounded buffer

- sequence coordination (producer and consumer)
- one-writer principle (assuming atomic writes)
- race conditions
- isolation, locks, how to implement locks
- deadlock, livelock
- alibis: justify unprotected sharing

Enforcing modularity with domains

- domains and sharing (permissions)
- kernel vs. user mode
- gates and supervisor call instructions
- monolithic kernel vs. microkernel

Virtualizing memory

- virtual addrs and virtual addr spaces
- page maps
  - page/block numbers + offsets
  - page map address register
- kernel and addr spaces
- hardware vs. software; TLBs

Virtualizing CPUs with threads

- time-sharing
- preemptive scheduling vs. cooperative multitasking
- interrupts/exceptions
- layered threads: kernel vs. user

Thread primitives for sequence coordination

- interface: wait, notify
- semaphores, condition vars
- polling vs. interrupts

Designing for performance

- Bottlenecks: due to limits, sharing
- Challenges: consider tech improvements (brute force); maintain simplicity
- Performance metrics: capacity, utilization, overhead, useful work; latency; throughput
- Law of diminishing returns (optimizing stages) vs. iterative approach (holistic)
- Reduce latency by exploiting workload properties (fast and slow paths, e.g. caches)
- Improve throughput via concurrency

Designing for performance

- Queuing: exponentially distributed inter-arrival times
  - Service time, offered load
  - Overload: load shedding, bounded buffers, self-pacing (feedback), quotas
- Anti-bottlenecks: batching, dallying, speculation

Multilevel memories

- Memory characterization: capacity, average random latency, cost, cell size, throughput
- Management: automatic; virtualize the read/write interface (VMs with single-level stores, mem-mapped files, COW, lazy zeroing, network shared mem)
- Add resident bit to page table; use missing page exception
- Cache vs. VM: what the client names (primary vs. secondary mem)
- Reference locality (temporal, spatial); working sets
- Policies: bring-in (on-demand), eviction (FIFO, LRU, MRU, clock, rand)

Scheduling

- Turn-around time, response time, waiting time
- Policies: FCFS, SJF (EWMA), RR, priority, RT