COT 4600 Operating Systems Fall 2009

Dan C. Marinescu

Office: HEC 439 B

Office hours: Tu-Th 3:00-4:00 PM

Lecture 22

- Attention: due dates for the project
  - Phase 3 – due today
  - Phase 4 – due Tuesday November 24
  - Final exam – Thursday December 10 4-6:50 PM
- Last time:
  - Implementation of AWAIT, ADVANCE, TICKET, and READ
  - Polling and interrupts
  - Evolution of the Intel x86 architecture
  - Virtual Machines
- Today:
  - Performance Metrics (Chapter 5)
  - Random variables
  - Elements of queuing theory
- Next Time:
  - I/O bottleneck

Performance metrics

- Wide range, sometimes correlated, other times with contradictory goals :
  - Throughput, utilization, waiting time, fairness
  - Latency (time in system)
  - Capacity
  - Reliability as a ultimate measure of performance
- Some measures of performance reflect physical limitations: capacity, bandwidth (CPU, memory, communication channel), communication latency.
- Often measures of performance reflect system organization and policies such as scheduling priorities.
- Resource sharing is an enduring problem; recall that one of the means for virtualization is multiplexing physical resources.
  - The workload can be characterized statistically
  - Queuing Theory can be used for analytical performance evaluation.

System design for performance

- When you have a clear idea of the design, simulate the system before actually implementing it.
- Identify the bottlenecks.
  - Identify those bottlenecks likely to be removed naturally by the technologies expected to be embedded in your system.
  - Keep in mind that removing one bottleneck exposes the next.
- Concurrency helps a lot both in hardware and in software.
  - in hardware implies multiple execution units
    - Pipelining → multiple instructions are executed concurrently
    - Multiple exaction units in a processor: integer, floating point, pixels
    - Graphics Processors – geometric engines.
    - Multi-processor system
    - Multi-core processors
    - Paradigm: SIMD (Single instruction multiple data), MIMD (Multiple Instructions Multiple Data.

System design for performance (cont'd)

- in software → complicates writing and debugging programs. SPMD (Same Program Multiple data) paradigm

- Design a well balanced system:
  - The bandwidth of individual sub-systems should be as close as poosible
  - The execution time of pipeline stages as close as poosible

Resource sharing -queuing

- See class notes.