



Midterm Review

CS 571: *Operating Systems* (Spring 2021)
Lecture 6

Yue Cheng

Midterm

- Monday, March 10, 7:20pm – 9:20pm
 - 120 min, open book, open notes
- Covering topics from Lec 1 to Lec 5
 - Process abstraction
 - CPU virtualization:
 - FIFO, SJF, RR, Priority, MLFQ
 - Memory virtualization:
 - Paging
 - Cache replacement policies

Midterm (cont.)

- The exam sheet will be available on Blackboard (under “Assignment”) at 7:20pm
- You may work directly on the Word document
 - Or, you may print it out and write on printed papers – make sure to scan to pdf with visible resolution
 - Convert it to pdf for submission
- Submission closes at 9:20pm, please make sure to submit before the deadline

Process Creation in Linux

- System call `fork()`
 - The return value of `fork()`
- Process tree

CPU Scheduling Policies

- FIFO
 - How it works?
 - Its inherent issues (why we need SJF)?
- SJF
 - How it works?
 - Any limitations (why we need STCF)?
- STCF (preemptive SJF)
 - How it works? How it solves SJF's limitations?

Various Metrics

- Average waiting time
- Average turnaround time
- How to calculate each metric under a specific schedule (Gantt chart)

CPU Scheduling Policies (cont.)

- RR
 - How it works?
 - Why it is needed (compared to SJF & STCF)?
 - The turnaround time vs. response time tradeoff
 - Impact of quantum tuning on turnaround time
- Priority
 - How it works?
 - Problems of Priority scheduling and solution?
- MLFQ
 - How it works?
 - Rules that were discussed in lecture. Which rule solves what problem?

Memory Virtualization: Paging

- Virtual addresses and physical addresses
 - VPN, PFN, page offset
 - Virtual address = VPN | offset
- Virtual to physical address translation
 - Linear **page table**: using VPN as index of array
- TLB mechanism
 - A hardware cache to accelerate address translation

Advanced Page Tables

- Approach 1: Linear inverted page table
 - Whole system maintains only one PT
 - Performs a whole-table linear search using pid+VPN to get the index
- Approach 2: Hash inverted page table
 - Uses hashing to reduce the time complexity from $O(N)$ to $O(1)$
- Approach 3: Multi-level page table
 - Uses hierarchy to minimize the memory usage

Cache Replacement Policies

- FIFO
 - Why it might work? Maybe the one brought in the longest ago is one we are not using now
 - Why it might not work? No real info to tell if it's being used or not
- Random
 - Sometimes non intelligence is better
- OPT
 - Assume we know about the future
 - Not practical in real cases: **offline** policy
 - However, can be used as a **best case baseline** for comparison purpose
- LRU
 - Intuition: we can't look into the future, but let's look at past experience to make a good guess
 - Our "bet" is that pages used recently are ones which will be used again (**principle of locality**)

Cache Performance Modeling and Locality Properties

- AMAT under different assumptions
- Spatial locality
 - Access to a single byte on disk brings in the whole page
- Temporal locality
 - Repetitive accesses to the same data

Good Luck!