CS 423
Operating System Design: Midterm Review

Professor Adam Bates
Spring 2018
Goals for Today

- **Learning Objective:**
  - Review material, and also my strategies for writing midterm questions
- **Announcements, etc:**
  - MP1 Grades Released*
    - Average score — 87%

* Plagiarism checking still in-progress

Reminder: Please put away devices at the start of class
In-Class on March 7th.
- i.e., 50 minutes

Scantron Multiple choice
- bring pencils!

20-30 Questions

Openbook: Textbooks, paper notes, printed sheets allowed. No electronic devices permitted (or necessary)!

Content: All lecture and text material covered prior to March 5th (i.e., up to and including memory)
Which of the following is not a good reason for increasing the size of a system’s page frames?

- Improves memory utilization/efficiency
- Decreases memory footprint of virtual memory management
- Improves disk utilization/efficiency
Which of the following is not a good reason for increasing the size of a system’s page frames?

- Improves memory utilization/efficiency
- Decreases memory footprint of virtual memory management
- Improves disk utilization/efficiency
Which of the following is not a good reason for increasing the size of a system’s page frames?

- Less Fragmentation
- Smaller Page Table
- Better to transfer more data per disk seek
Page Size Considerations

- Small pages
  - Reason:
    - Locality of reference tends to be small (256)
    - Less fragmentation
  - Problem: require large page tables

- Large pages
  - Reason
    - Small page table
    - I/O transfers have high seek time, so better to transfer more data per seek
  - Problem: Internal fragmentation, needless caching
Which of the following is not a good reason for increasing the size of a system’s page frames?

- Less Fragmentation
- Smaller Page Table
- Better to transfer more data per disk seek
Which of the following is not a good reason for increasing the size of a system’s page frames?

- Improves memory utilization/efficiency
- Decreases memory footprint of virtual memory management
- Improves disk utilization/efficiency
• Definition: The priority ceiling of a semaphore is the highest priority of any task that can lock it
• A task that requests a lock $R_k$ is denied if its priority is not higher than the highest priority ceiling of all semaphores currently locked by other tasks (say it belongs to semaphore $R_h$)  
  • The task is said to be blocked by the task holding lock $R_h$
• A task inherits the priority of the top higher-priority task it is blocking
Priority Ceiling Protocol:

- Need **Blue** but Priority is lower than **Red** ceiling
- Need **Yellow** but Priority is lower than **Red** ceiling
- Needs **Red**, waits for 1 critical section to complete.

Done
On a 32 bit system we have $2^{32}$ B virtual address space
- i.e., a 32 bit register can store $2^{32}$ values

Page size (range) is $2^n$ (e.g., 512 B, 1 KB, 2 KB, 4 KB...)

Given a page size, how many pages are needed?
- e.g., If 4 KB pages ($2^{12}$ B), then $2^{32}/2^{12} = \ldots$
  - $2^{20}$ pages required to represent the address space

But! each page entry takes more than 1 Byte of space to represent.

- Suppose page size is 4 bytes (Why?)
  - $(2 \times 2) \times 2^{20} = 4$ MB of space required to represent our page table in physical memory.
Completely Fair Scheduler

- Merged into the 2.6.23 release of the Linux kernel and is the default scheduler.
- Scheduler maintains a red-black tree where nodes are ordered according to received virtual execution time.
- Node with smallest virtual received execution time is picked next.
- Priorities determine accumulation rate of virtual execution time:
  - Higher priority → slower accumulation rate.
Completely Fair Scheduler

- Merged into the 2.6.23 release of the Linux kernel and is the default scheduler.
- Scheduler maintains a red-black tree where nodes are ordered according to received virtual execution time.
- Node with smallest virtual received execution time is picked next.
- Priorities determine accumulation rate of virtual execution time.
  - Higher priority → slower accumulation rate

Property of CFS: If all task’s virtual clocks run at exactly the same speed, they will all get the same amount of time on the CPU.

How does CFS account for I/O-intensive tasks?
CFS dispenses with a run queue and instead maintains a time-ordered red-black tree. Why?

Benefits over run queue:
- O(1) access to leftmost node (lowest virtual time).
- O(log n) insert
- O(log n) delete
- self-balancing
With CFS active, tasks X, Y, and Z accumulate virtual execution time at a rate of 1, 2, and 3, respectively. What is the expected share of the CPU that each gets?

- X=17%, Y=33%, Z=50%
- X=55%, Y=27%, Z=18%
- X=50%, Y=33%, Z=17%
- X=18%, Y=27%, Z=55%
With CFS active, tasks X, Y, and Z accumulate virtual execution time at a rate of 1, 2, and 3, respectively. What is the expected share of the CPU that each gets?

- X=17%, Y=33%, Z=50%
- X=55%, Y=27%, Z=18%
- X=50%, Y=33%, Z=17%
- X=18%, Y=27%, Z=55%

“X should have twice as much CPU as Y, three times as much CPU as Z”
With CFS active, tasks X, Y, and Z accumulate virtual execution time at a rate of 1, 2, and 3, respectively. What is the expected share of the CPU that each gets?

- X=17%, Y=33%, Z=50%
- X=55%, Y=27%, Z=18%
- X=50%, Y=33%, Z=17%
- X=18%, Y=27%, Z=55%

Property of CFS: If all tasks’ virtual clocks run at the same speed, they all get the same amount of time on the CPU.

How does CFS account for I/O-intensive tasks?

- Priorities determine accumulation rate of virtual execution time
- Higher priority → slower accumulation rate

“X should have twice as much CPU as Y, three times as much CPU as Z”
With CFS active, tasks X, Y, and Z accumulate virtual execution time at a rate of 1, 2, and 3, respectively. What is the expected share of the CPU that each gets?

- X = 17%, Y = 33%, Z = 50%
- X = 55%, Y = 27%, Z = 18%
- X = 50%, Y = 33%, Z = 17%
- X = 18%, Y = 27%, Z = 55%

“X should have twice as much CPU as Y, three times as much CPU as Z”
Below are chronologically-ordered series of tasks with their completion time shown. Which sequence offers a pessimal (i.e., worst-case) average response time for FIFO scheduling?

- 1, 2, 3, 4
- 2, 2, 2, 2
- 3, 1, 3, 1
- 4, 3, 2, 1
More Q&A

ANY QUESTIONS?