TCSS 422: OPERATING SYSTEMS

Introduction to OS Schedulers

Wes J. Lloyd
School of Engineering and Technology
University of Washington - Tacoma

April 15, 2025 TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma

1

TEXT BOOK COUPON

- 10% off textbook code: **WRITING10** *(through Friday Apr 18)*
- https://www.lulu.com/shop/andrea-arpaci-dusseau-and-remzi-arpaci-dusseau/operating-systems-three-easy-pieces-hardcover-version-110/hardcover/product-15gjeeky.html?q=three+easy+pieces+operating+systems&page=1&pageSize=4
- With coupon textbook is only $35.78 + tax & shipping

April 15, 2025 TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma L5.2

2

TCSS 422 – OFFICE HRS – SPRING 2025

- **Office Hours plan for Spring (by Zoom):**
- **Monday 11:30am - 12:30p GTA Xinghan**
- **Tuesday 11:30am - 12:30p GTA Xinghan**
- **Wednesday 11:00am - 12:00p Instructor Wes**
- **Friday 12:00pm - 1:00p Wes or Xinghan**
- **Office hours this Friday April 18th**
  - Xinghan
- Instructor is available after class at 6pm in CP 229 each day

April 15, 2025 TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma L5.3

3

OBJECTIVES – 4/15

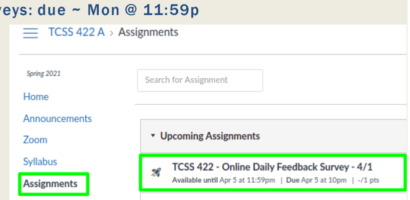
- Questions from 4/10
- Assignment 0
- C Tutorial - Pointers, Strings, Exec in C
- Chapter 6: Limited Direct Execution
- Chapter 7: Scheduling Introduction
  - Scheduling metrics
    - Turnaround time, Jain's Fairness Index, Response time
  - FIFO, SJF, STCF, RR schedulers
- Chapter 8: Multi-level Feedback Queue
  - MLFQ Scheduler
  - Job Starvation
  - Gaming the Scheduler
  - Examples

April 15, 2025 TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma L5.4

4

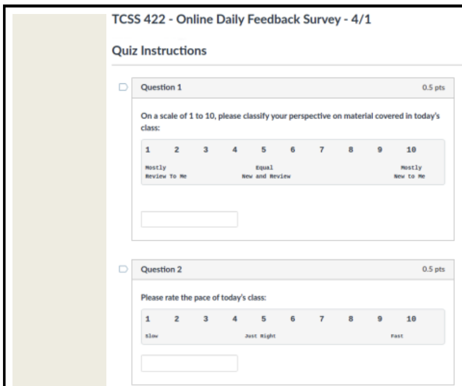
ONLINE DAILY FEEDBACK SURVEY

- Daily Feedback Quiz in Canvas – Available After Each Class
- Extra credit available for completing surveys ***ON TIME***
- Tuesday surveys: due by ~ Wed @ 11:59p
- Thursday surveys: due ~ Mon @ 11:59p



April 15, 2025 TCSS422: Computer Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma L5.5

5



April 15, 2025 TCSS422: Computer Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma L5.6

6

MATERIAL / PACE

- Please classify your perspective on material covered in today's class (48 of 63 respondents – 76.2%) :
- 1-mostly review, 5-equal new/review, 10-mostly new
- **Average – 6.47 (↓ - previous 6.64)**

- Please rate the pace of today's class:
- 1-slow, 5-just right, 10-fast
- **Average – 4.83 (↓ - previous 5.29)**

April 15, 2025 | TCSS422: Computer Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.7

7

FEEDBACK FROM 4/10

- ***How does the kernel keep track of process/PID pairs?***
- > what is the Linux kernel data structure that is used to describe a process ?

April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.8

8

FEEDBACK - 2

- ***Can you explain how the path works in detail?***
- ***I understand that the path is where a program will look and that it is a collection of folders, but how is this actually stored?***
- The PATH is an environment variable
- It is persistent for the lifetime of the shell
- BASH shell inherits the default system path from:
  - `/etc/environment`
- In your local bash resource file, you can modify the path var
  - `~/.bashrc`
  - "`~/`" is an alias for your home directory
  - `cd ~/` goes home…
- Append to path:
  - `export PATH=$PATH:/monkeys`

April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.9

9

FEEDBACK - 3

- ***How will the quizzes and midterm exam be structured? What type of questions can we expect on those assessments?***
- Poll EV questions provide samples
- Practice Quizzes (as Activities in Canvas) from examples
- Look for various sample problems starting in Chapter 7

April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.10

10

FEEDBACK FROM 4/4

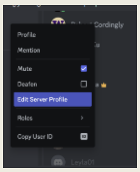
- ***Wait function parameter being null is still a bit unclear***
- There are two variants for the wait() API
- wait(), and waitpid()
  - See 'man 2 wait' for manual page
- wait() takes an optional integer as a 'flag'
- These flags provide instructions for how the API should behave
- If NULL, then there is no special behavior
- The manual page details specific behavior for various constants:
- **WIFEXITED** – return true if child terminated normally
- **WEXITSTATUS** – return exit status of child
- **WIFSIGNALED** – returns true if child process terminated by signal

April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.11

11

TCSS 422 DISCORD SERVER

- Please join the TCSS 422 A – Spring 2025 Discord Server
- https://discord.gg/H7PPZ5ArFW
- Under Edit Server Profile:
  Please update your 'Server Nickname' to your real name or UW NET ID
  THANK YOU



April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.12

12

OBJECTIVES – 4/15

- Questions from 4/10
- **Assignment 0**
- C Tutorial - Pointers, Strings, Exec in C
- Chapter 6: Limited Direct Execution
- Chapter 7: Scheduling Introduction
  - Scheduling metrics
    - Turnaround time, Jain's Fairness Index, Response time
  - FIFO, SJF, STCF, RR schedulers
- Chapter 8: Multi-level Feedback Queue
  - MLFQ Scheduler
  - Job Starvation
  - Gaming the Scheduler
  - Examples

April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.13

13

TCSS 422 – SET VMS

- School of Engineering and Technology hosted Ubuntu 24.04 VMs for TCSS 422 – Spring 2025 are created
- The VMs are on the UW Tacoma private network
- The instructor has requested instructions for off-campus access (i.e. VPN setup, etc.)

April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.14

14

OBJECTIVES – 4/15

- Questions from 4/10
- Assignment 0
- **C Tutorial - Pointers, Strings, Exec in C**
- Chapter 6: Limited Direct Execution
- Chapter 7: Scheduling Introduction
  - Scheduling metrics
    - Turnaround time, Jain's Fairness Index, Response time
  - FIFO, SJF, STCF, RR schedulers
- Chapter 8: Multi-level Feedback Queue
  - MLFQ Scheduler
  - Job Starvation
  - Gaming the Scheduler
  - Examples

April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.15

15

FINISH CHAPTER 5

- Switch to Lecture 4 Slides
- Slides L4.34 to L4.59 (thru system calls and traps)

April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.16

16

OBJECTIVES – 4/15

- Questions from 4/10
- Assignment 0
- C Tutorial - Pointers, Strings, Exec in C
- **Chapter 6: Limited Direct Execution**
- Chapter 7: Scheduling Introduction
  - Scheduling metrics
    - Turnaround time, Jain's Fairness Index, Response time
  - FIFO, SJF, STCF, RR schedulers
- Chapter 8: Multi-level Feedback Queue
  - MLFQ Scheduler
  - Job Starvation
  - Gaming the Scheduler
  - Examples

April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.17

17

CH. 6: LIMITED DIRECT EXECUTION

April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.18

18

CHAPTER 6

- Chapter 6: Limited Direct Execution
  - Direct execution
  - Limited direct execution
  - CPU modes
  - System calls and traps
  - Cooperative multi-tasking
  - Context switching and preemptive multi-tasking

19

MULTITASKING

- How/when should the OS regain control of the CPU to switch between processes?
- Cooperative multitasking (mostly pre 32-bit)
  - < Windows 95, Mac OSX
  - Opportunistic: running programs must give up control
    - User programs must call a special **yield** system call
    - When performing I/O
    - Illegal operations
  - (POLLEV)
    What problems could you for see with this approach?

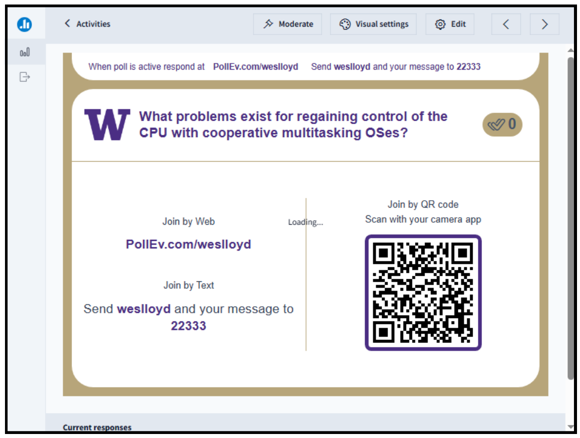
20

MULTITASKING

- How/when should the OS regain control of the CPU to switch between processes?
- Cooperative multitasking (mostly pre 32-bit)
  - < Windows 95, Mac OSX
  - Opportunistic: running programs must give up control
    - When performing I/O
    - Illegal operations
  - (POLLEV)
    What problems could you for see with this approach?

A process gets stuck in an infinite loop.
→ **Reboot the machine**

21



22

QUESTION: MULTITASKING

- What problems exist for regaining the control of the CPU with cooperative multitasking OSes?

23

MULTITASKING - 2

- Preemptive multitasking (32 & 64 bit OSes)
- >= Mac OSX, Windows 95+
- Timer interrupt
  - Raised at some regular interval (in ms)
  - Interrupt handling
    1. Current program is halted
    2. Program states are saved
    3. OS Interrupt handler is run (kernel mode)
- (PollEV) What is a good interval for the timer interrupt?

24

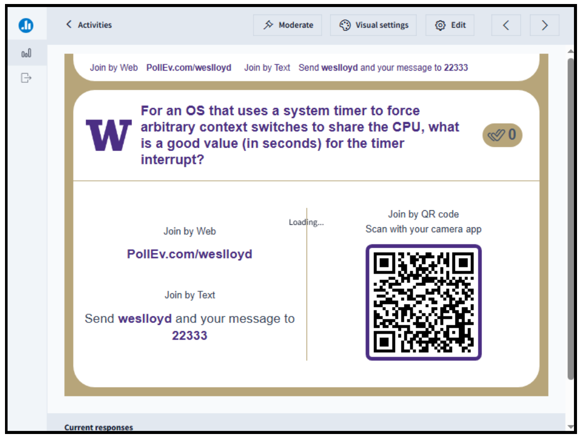
MULTITASKING - 2

- Preemptive multitasking (32 & 64 bit OSes)
- >= Mac OSX, Windows 95+
- Timer
  - Rais
  - Inter
    1. Current program is halted
    2. Program states are saved
    3. OS Interrupt handler is run (kernel mode)
- (PollEV) What is a good interval for the timer interrupt?

A **timer interrupt** gives OS the ability to run again on a CPU.

April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.25

25



26

QUESTION: TIME SLICE

- For an OS that uses a system timer to force arbitrary context switches to share the CPU, what is a good value (in seconds) for the timer interrupt?

April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.27

27

QUESTION: TIME SLICE

- For an OS that uses a system timer to force arbitrary context switches to share the CPU, what is a good value (in seconds) for the timer interrupt?
  - Typical time slice for process execution is **10 to 100 milliseconds**
  - Typical context switch overhead is (*switch between processes*) **0.01 milliseconds**
    - 0.1% of the time slice (1/1000th)

April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.28

28

CHAPTER 6

- Chapter 6: Limited Direct Execution
  - Direct execution
  - Limited direct execution
  - CPU modes
  - System calls and traps
  - Cooperative multi-tasking
  - **Context switching and preemptive multi-tasking**

April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.29

29

CONTEXT SWITCH

- Preemptive multitasking initiates "trap" into the OS code to determine:
- Whether to continue running the **current process**, or switch to a **different one**.
- If the decision is made to switch, the OS performs a context switch swapping out the current process for a new one.

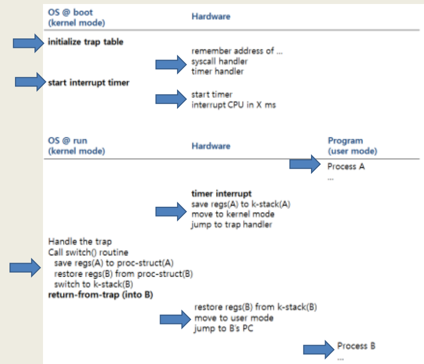
April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.30

30

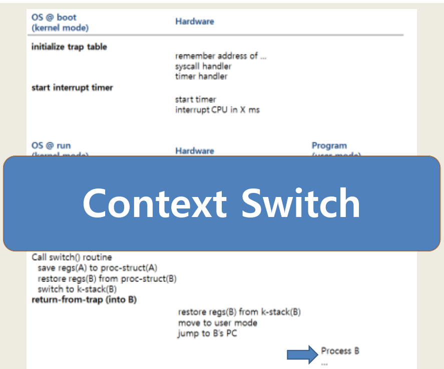
CONTEXT SWITCH - 2

1. Save register values of the current process to its kernel stack
   - General purpose registers
   - PC: program counter (instruction pointer)
   - kernel stack pointer
2. Restore soon-to-be-executing process from its kernel stack
3. Switch to the kernel stack for the soon-to-be-executing process

31



32



33

INTERRUPTED INTERRUPTS

- What happens if during an interrupt (trap to kernel mode), another interrupt occurs?
- Linux
  - < 2.6 kernel: non-preemptive kernel
  - >= 2.6 kernel: preemptive kernel

34

PREEMPTIVE KERNEL

- Use "locks" as markers of regions of non-preemptibility (non-maskable interrupt)
- Preemption counter (`preempt_count`)
  - begins at zero
  - increments for each lock acquired (not safe to preempt)
  - decrements when locks are released
- Interrupt can be interrupted when `preempt_count=0`
  - It is safe to preempt (maskable interrupt)
  - the interrupt is more important

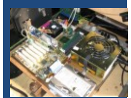
35

OBJECTIVES – 4/15

- Questions from 4/10
- Assignment 0
- C Tutorial - Pointers, Strings, Exec in C
- Chapter 6: Limited Direct Execution
- **Chapter 7: Scheduling Introduction**
  - **Scheduling metrics**
    - **Turnaround time**, **Jain's Fairness Index**, Response time
  - FIFO, SJF, STCF, RR schedulers
- Chapter 8: Multi-level Feedback Queue
  - MLFQ Scheduler
  - Job Starvation
  - Gaming the Scheduler
  - Examples

36

CHAPTER 7- SCHEDULING: INTRODUCTION



April 15, 2025 TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma L5.37

37

SCHEDULING METRICS

- **Metrics**: A standard measure to quantify to what degree a system possesses some property. Metrics provide *repeatable* techniques to quantify and compare systems.
- **Measurements** are the numbers derived from the application of metrics
- Scheduling Metric #1: **Turnaround time**
- The time at which the job completes minus the time at which the job arrived in the system

$$T_{turnaround} = T_{completion} - T_{arrival}$$

- How is turnaround time different than execution time?

April 15, 2025 TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma L5.38

38

WE WILL RETURN AT ~4:55PM

April 15, 2025 TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma L5.39

39

SCHEDULING METRICS - 2

- Scheduling Metric #2: **Fairness**
  - Jain's fairness index
  - Quantifies if jobs receive a fair share of system resources

$$\mathcal{J}(x_1, x_2, \ldots, x_n) = \frac{(\sum_{i=1}^{n} x_i)^2}{n \cdot \sum_{i=1}^{n} x_i{}^2}$$

- n processes
- x_i is time share of each process
- worst case = 1/n
- best case = 1
- Consider n=3, worst case = .333, best case=1
- With n=3 and x_1=.2, x_2=.7, x_3=.1, fairness=.62
- With n=3 and x_1=.33, x_2=.33, x_3=.33, fairness=1

April 15, 2025 TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma L5.40

40

With n=3 and x_1=.2, x_2=.7, x_3=.1

$$\mathcal{J}(x_1, x_2, \ldots, x_n) = \frac{(\sum_{i=1}^{n} x_i)^2}{n \cdot \sum_{i=1}^{n} x_i{}^2}$$

April 15, 2025 TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma L5.41

41

With n=3 and x_1=.2, x_2=.7, x_3=.1

$$\mathcal{J}(x_1, x_2, \ldots, x_n) = \frac{(\sum_{i=1}^{n} x_i)^2}{n \cdot \sum_{i=1}^{n} x_i{}^2}$$

$$\frac{(.2 + .7 + .1)^2}{3 \cdot \Sigma\ (.2)^2 + (.7)^2 + (.1)^2} = \frac{1}{1.62} = .617$$

$\frac{1}{3}$ TO 1

$3 \cdot (.04 + .49 + .01)$

$3 \cdot (.54)$

1.62

April 15, 2025 TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma L5.42

42

With n=3 and x_1=.33, x_2=.33, x_3=.33

$$\mathcal{J}(x_1, x_2, \ldots, x_n) = \frac{(\sum_{i=1}^{n} x_i)^2}{n \cdot \sum_{i=1}^{n} x_i^2}$$

April 15, 2025 TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma L5.43

43

With n=3 and x_1=.33, x_2=.33, x_3=.33

$$\mathcal{J}(x_1, x_2, \ldots, x_n) = \frac{(\sum_{i=1}^{n} x_i)^2}{n \cdot \sum_{i=1}^{n} x_i^2}$$

.33 + .33 + .33 → 1 / .9801 → 1

3 · Σ (.33)² + (.33)² + (.33)²

3 · (.1089 + .1089 + .1089)

3 · (.3267)

3 · ((.333)² + ...)

.9801

.998001   1 / .998 → 1.002

April 15, 2025 TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma L5.44

44

OBJECTIVES – 4/15

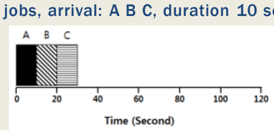
- Questions from 4/10
- Assignment 0
- C Tutorial - Pointers, Strings, Exec in C
- Chapter 6: Limited Direct Execution
- Chapter 7: Scheduling Introduction
  - Scheduling metrics
    - Turnaround time, Jain's Fairness Index, Response time
  - **FIFO**, SJF, STCF, RR schedulers
- Chapter 8: Multi-level Feedback Queue
  - MLFQ Scheduler
  - Job Starvation
  - Gaming the Scheduler
  - Examples

April 15, 2025 TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma L5.45

45

SCHEDULERS

- FIFO: first in, first out
  - Very simple, easy to implement
- Consider
  - 3 x 10sec jobs, arrival: A B C, duration 10 sec each



$$Average\ turnaround\ time = \frac{10 + 20 + 30}{3} = 20\ sec$$

April 15, 2025 TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma L5.46

46

OBJECTIVES – 4/15

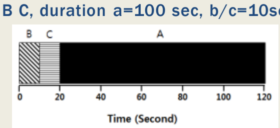
- Questions from 4/10
- Assignment 0
- C Tutorial - Pointers, Strings, Exec in C
- Chapter 6: Limited Direct Execution
- Chapter 7: Scheduling Introduction
  - Scheduling metrics
    - Turnaround time, Jain's Fairness Index, Response time
  - FIFO, **SJF**, STCF, RR schedulers
- Chapter 8: Multi-level Feedback Queue
  - MLFQ Scheduler
  - Job Starvation
  - Gaming the Scheduler
  - Examples

April 15, 2025 TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma L5.47

47

SJF: SHORTEST JOB FIRST

- Given that we know execution times in advance:
  - Run in order of duration, shortest to longest
  - Non preemptive scheduler
  - This is not realistic
  - Arrival: A B C, duration a=100 sec, b/c=10sec



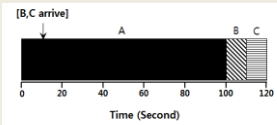
$$Average\ turnaround\ time = \frac{10 + 20 + 120}{3} = 50\ sec$$

April 15, 2025 TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma L5.48

48

SJF: WITH RANDOM ARRIVAL

- If jobs arrive at any time: duration a=100s, b/c=10s
- A @ t=0sec, B @ t=10sec, C @ t=10sec



$$\textit{Average turnaround time} = \frac{100 + (110 - 10) + (120 - 10)}{3} = 103.33\ sec$$

49

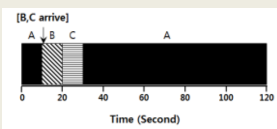
OBJECTIVES – 4/15

- Questions from 4/10
- Assignment 0
- C Tutorial - Pointers, Strings, Exec in C
- Chapter 6: Limited Direct Execution
- Chapter 7: Scheduling Introduction
  - Scheduling metrics
    - Turnaround time, Jain's Fairness Index, Response time
  - FIFO, SJF, STCF, RR schedulers
- Chapter 8: Multi-level Feedback Queue
  - MLFQ Scheduler
  - Job Starvation
  - Gaming the Scheduler
  - Examples

50

STCF: SHORTEST TIME TO COMPLETION FIRST

- Consider: duration a=100sec, b/c=10sec
  - A_{len}=100 $A_{arrival}$=0
  - B_{len}=10, $B_{arrival}$=10, C_{len}=10, $C_{arrival}$=10



$$\textit{Average turnaround time} = \frac{(120 - 0) + (20 - 10) + (30 - 10)}{3} = 50\ sec$$

51

OBJECTIVES – 4/15

- Questions from 4/10
- Assignment 0
- C Tutorial - Pointers, Strings, Exec in C
- Chapter 6: Limited Direct Execution
- **Chapter 7: Scheduling Introduction**
  - **Scheduling metrics**
    - Turnaround time, Jain's Fairness Index, **Response time**
  - FIFO, SJF, STCF, RR schedulers
- Chapter 8: Multi-level Feedback Queue
  - MLFQ Scheduler
  - Job Starvation
  - Gaming the Scheduler
  - Examples

52

SCHEDULING METRICS - 3

- Scheduling Metric #3: **Response Time**
- Time from when job arrives until it starts execution

$$T_{response} = T_{firstrun} - T_{arrival}$$

- STCF, SJF, FIFO
  - can perform poorly with respect to response time

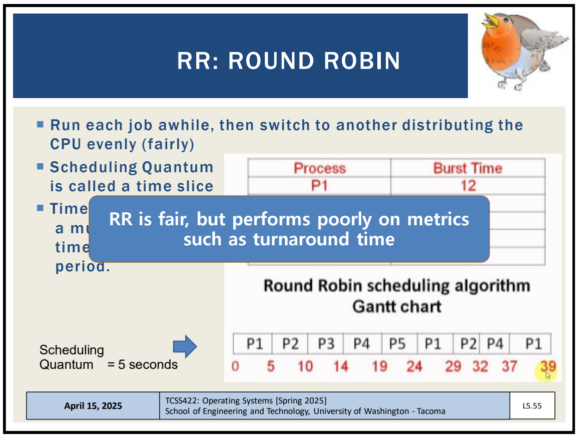
What scheduling algorithm(s) can help minimize response time?

53

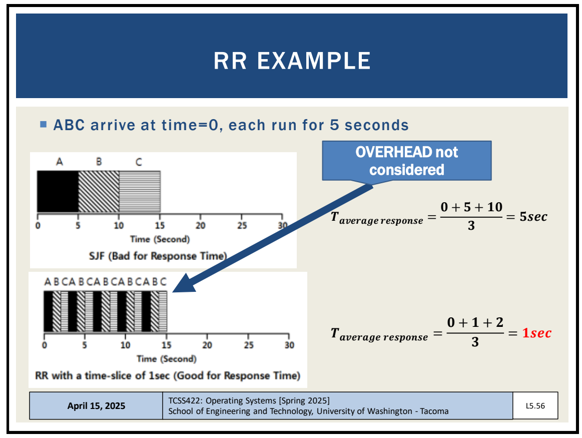
OBJECTIVES – 4/15

- Questions from 4/10
- Assignment 0
- C Tutorial - Pointers, Strings, Exec in C
- Chapter 6: Limited Direct Execution
- Chapter 7: Scheduling Introduction
  - Scheduling metrics
    - Turnaround time, Jain's Fairness Index, Response time
  - FIFO, SJF, STCF, RR schedulers
- Chapter 8: Multi-level Feedback Queue
  - MLFQ Scheduler
  - Job Starvation
  - Gaming the Scheduler
  - Examples

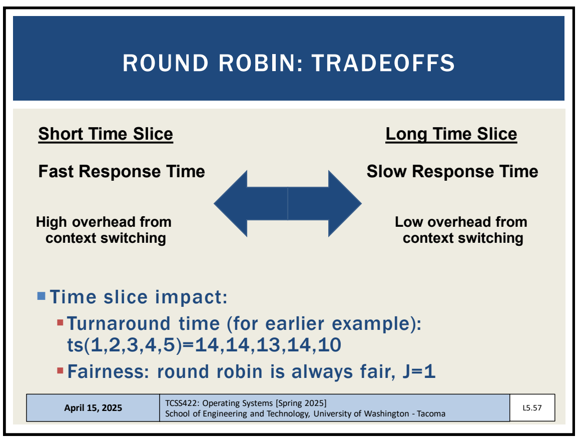
54



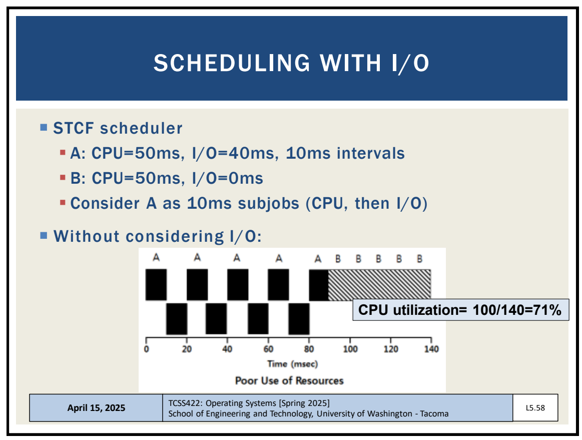
55



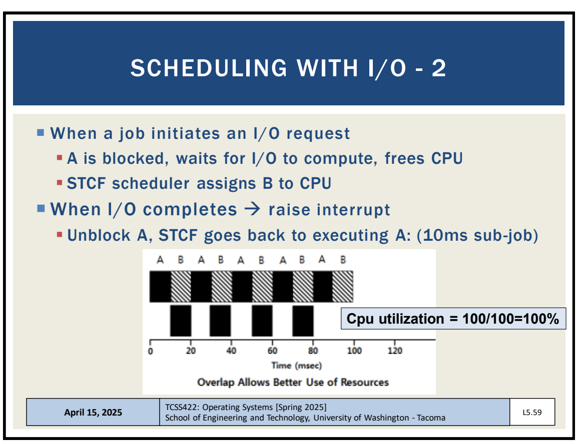
56



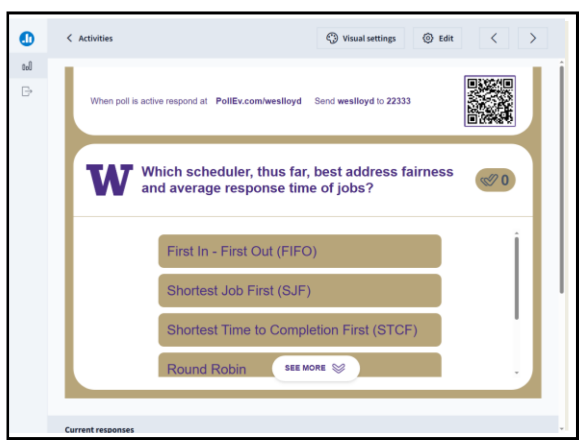
57



58



59



60

QUESTION: SCHEDULING FAIRNESS

- Which scheduler, this far, best addresses fairness and average response time of jobs?
- First In – First Out (FIFO)
- Shortest Job First (SJF)
- Shortest Time to Completion First (STCF)
- Round Robin (RR)
- None of the Above
- All of the Above

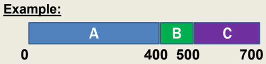
April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.61

61

SCHEDULING METRICS

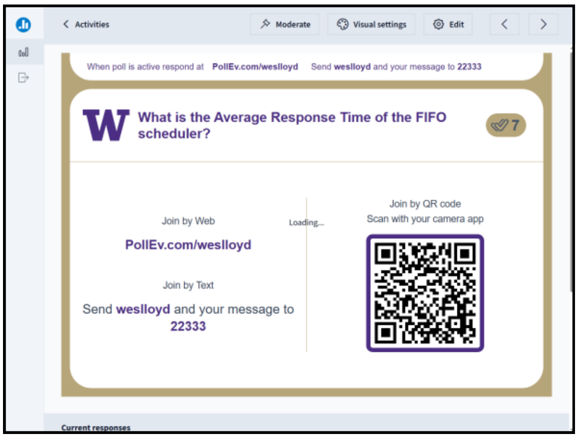
- Consider Three jobs (A, B, C) that require: $time_A$=400ms, $time_B$=100ms, and $time_C$=200ms
- All jobs arrive at time=0 in the sequence of A B C.
- Draw a scheduling graph to help compute the average response time (ART) and average turnaround time (ATT) scheduling metrics for the FIFO scheduler.

Example:

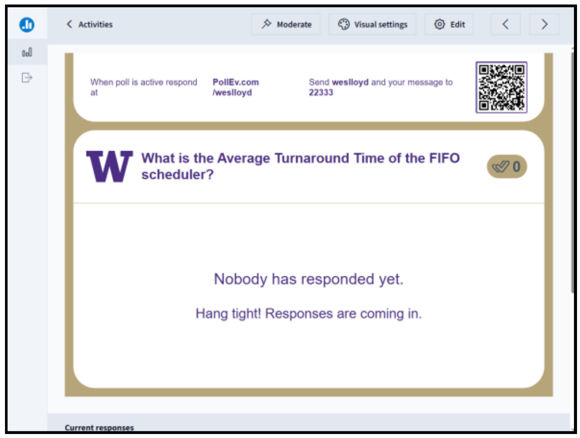


April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.62

62



63

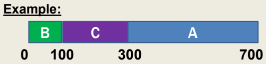


64

SCHEDULING METRICS

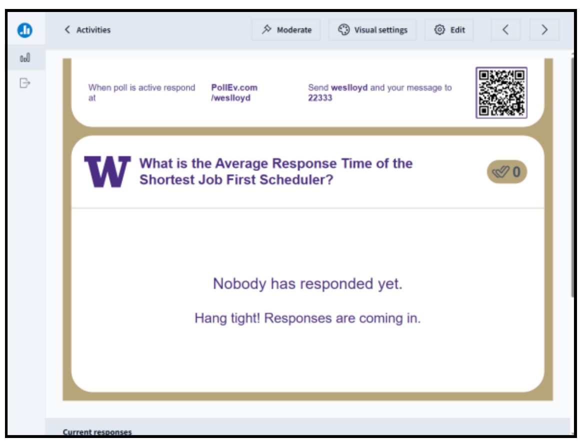
- Consider Three jobs (A, B, C) that require: $time_A$=400ms, $time_B$=100ms, and $time_C$=200ms
- All jobs arrive at time=0 in the sequence of A B C.
- Draw a scheduling graph to help compute the average response time (ART) and average turnaround time (ATT) scheduling metrics for the SJF scheduler.

Example:

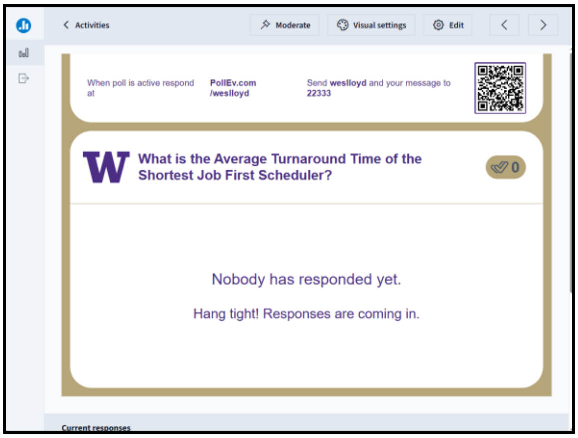


April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.65

65

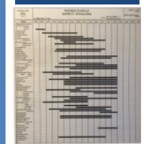


66



67

CHAPTER 8 – MULTI-LEVEL FEEDBACK QUEUE (MLFQ) SCHEDULER



April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.68

68

OBJECTIVES – 4/15

- Questions from 4/10
- Assignment 0
- C Tutorial - Pointers, Strings, Exec in C
- Chapter 6: Limited Direct Execution
- Chapter 7: Scheduling Introduction
  - Scheduling metrics
    - Turnaround time, Jain's Fairness Index, Response time
  - FIFO, SJF, STCF, RR schedulers
- Chapter 8: Multi-level Feedback Queue
  - **MLFQ Scheduler**
  - Job Starvation
  - Gaming the Scheduler
  - Examples

April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.69

69

MULTI-LEVEL FEEDBACK QUEUE

- Objectives:
  - Improve turnaround time: *Run shorter jobs first*
  - Minimize response time: *Important for interactive jobs (UI)*
- Achieve without a priori knowledge of job length

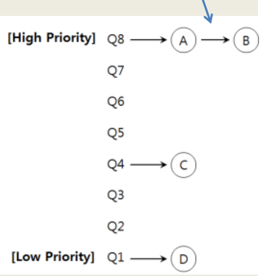
April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.70

70

MLFQ - 2

Round-Robin within a Queue

- Multiple job queues
- Adjust job priority based on observed behavior
- Interactive Jobs
  - Frequent I/O → keep priority high
  - Interactive jobs require fast response time (GUI/UI)
- Batch Jobs
  - Require long periods of CPU utilization
  - Keep priority low



April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.71

71

MLFQ: DETERMINING JOB PRIORITY

- New arriving jobs are placed into highest priority queue
- If a job uses its entire time slice, priority is reduced (↓)
  - Jobs appears CPU-bound ("batch" job), not interactive (GUI/UI)
- If a job relinquishes the CPU for I/O priority stays the same

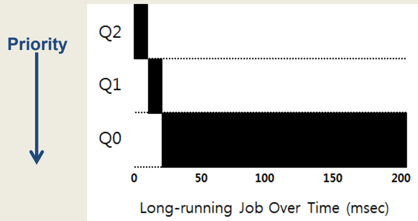
MLFQ approximates SJF

April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.72

72

MLFQ: LONG RUNNING JOB

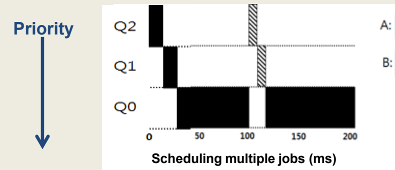
- Three-queue scheduler, time slice=10ms



73

MLFQ: BATCH AND INTERACTIVE JOBS

- $A_{arrival\_time}$ =0ms, A_{run_time}=200ms,
- B_{run_time} =20ms, $B_{arrival\_time}$ =100ms

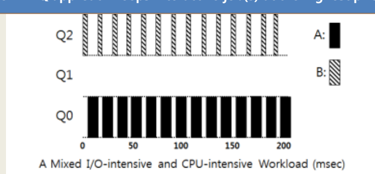


74

MLFQ: BATCH AND INTERACTIVE - 2

- Continuous interactive job (B) with long running batch job (A)
  - Low response time is good for B
  - A continues to make progress

The MLFQ approach keeps interactive job(s) at the highest priority



75

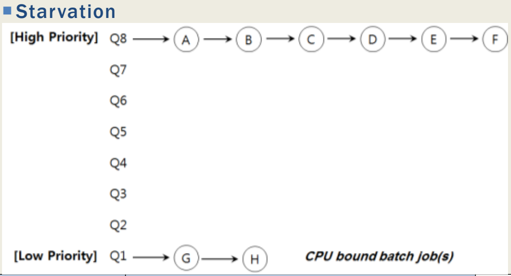
OBJECTIVES – 4/15

- Questions from 4/10
- Assignment 0
- C Tutorial - Pointers, Strings, Exec in C
- Chapter 6: Limited Direct Execution
- Chapter 7: Scheduling Introduction
  - Scheduling metrics
    - Turnaround time, Jain's Fairness Index, Response time
  - FIFO, SJF, STCF, RR schedulers
- Chapter 8: Multi-level Feedback Queue
  - MLFQ Scheduler
  - **Job Starvation**
  - Gaming the Scheduler
  - Examples

76

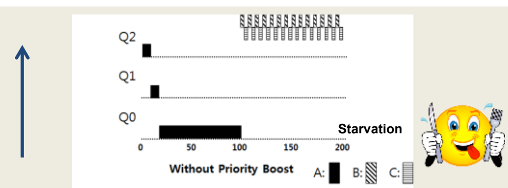
MLFQ: ISSUES

- Starvation



77

RESPONDING TO BEHAVIOR CHANGE

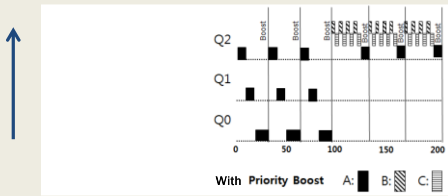


- Priority Boost
  - Reset all jobs to topmost queue after some time interval S

78

RESPONDING TO BEHAVIOR CHANGE - 2

- With priority boost
  - Prevents starvation



79

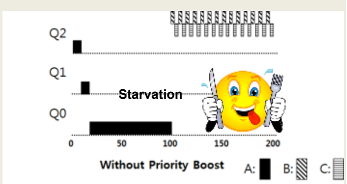
KEY TO UNDERSTANDING MLFQ – PB

- Without priority boost:
- **Rule 1:** If Priority(A) > Priority(B), A runs (B doesn't).
- **Rule 2:** If Priority(A) = Priority(B), A & B run in RR.
- **KEY**: If time quantum of a higher queue is filled, then we don't run any jobs in lower priority queues!!!

80

STARVATION EXAMPLE

- **Consider 3 queues:**
- Q2 – HIGH PRIORITY – Time Quantum 10ms
- Q1 – MEDIUM PRIORITY – Time Quantum 20 ms
- Q0 – LOW PRIORITY – Time Quantum 40 ms
- Job A: 200ms no I/O
- Job B: 5ms then I/O
- Job C: 5ms then I/O
- Q2 fills up, starves Q1 & Q0
- A makes no progress



81

OBJECTIVES – 4/15

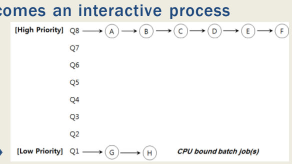
- Questions from 4/10
- Assignment 0
- C Tutorial - Pointers, Strings, Exec in C
- Chapter 6: Limited Direct Execution
- Chapter 7: Scheduling Introduction
  - Scheduling metrics
    - Turnaround time, Jain's Fairness Index, Response time
  - FIFO, SJF, STCF, **RR** schedulers
- Chapter 8: Multi-level Feedback Queue
  - MLFQ Scheduler
  - Job Starvation
  - **Gaming the Scheduler**
  - Examples

82

MLFQ: ISSUES - 2

- Gaming the scheduler
  - Issue I/O operation at 99% completion of the time slice
  - Keeps job priority fixed – never lowered
- Job behavioral change
  - CPU/batch process becomes an interactive process

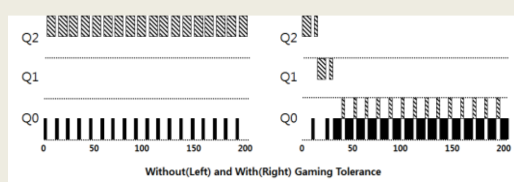
Priority becomes stuck



83

PREVENTING GAMING

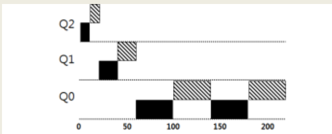
- Improved time accounting:
  - Track total job execution time in the queue
  - Each job receives a fixed time allotment
  - When allotment is exhausted, job priority is lowered



84

MLFQ: TUNING

- Consider the tradeoffs:
  - How many queues?
  - What is a good time slice?
  - How often should we “Boost” priority of jobs?
  - What about different time slices to different queues?



Example) 10ms for the highest queue, 20ms for the middle, 40ms for the lowest

April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.85

85

OBJECTIVES – 4/15

- Questions from 4/10
- Assignment 0
- C Tutorial - Pointers, Strings, Exec in C
- Chapter 6: Limited Direct Execution
- Chapter 7: Scheduling Introduction
  - Scheduling metrics
    - Turnaround time, Jain’s Fairness Index, Response time
  - FIFO, SJF, STCF, **RR** schedulers
- Chapter 8: Multi-level Feedback Queue
  - MLFQ Scheduler
  - Job Starvation
  - Gaming the Scheduler
  - **Examples**

April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.86

86

PRACTICAL EXAMPLE

- Oracle Solaris MLFQ implementation
  - 60 Queues →
    w/ slowly increasing time slice (high to low priority)
  - Provides sys admins with set of editable table(s)
  - Supports adjusting time slices, boost intervals, priority changes, etc.
- Advice
  - Provide OS with hints about the process
  - Nice command → Linux

April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.87

87

MLFQ RULE SUMMARY

- The refined set of MLFQ rules:
- **Rule 1:** If Priority(A) > Priority(B), A runs (B doesn’t).
- **Rule 2:** If Priority(A) = Priority(B), A & B run in RR.
- **Rule 3:** When a job enters the system, it is placed at the highest priority.
- **Rule 4:** Once a job uses up its time allotment at a given level (regardless of how many times it has given up the CPU), its priority is reduced(i.e., it moves down on queue).
- **Rule 5:** After some time period S, move all the jobs in the system to the topmost queue.

April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.88

88

Jackson deploys a 3-level MLFQ scheduler. The time slice is 1 for high priority jobs, 2 for medium priority, and 4 for low priority. This MLFQ scheduler performs a Priority Boost every 6 timer units. When the priority boost fires, the current job is preempted, and the next scheduled job is run in round-robin order.

Job	Arrival Time	Job Length
A	T=0	4
B	T=0	16
C	T=0	8

(11 points) Show a scheduling graph for the MLFQ scheduler for the jobs above. Draw vertical lines for key events and be sure to label the X-axis times as in the example. Please draw clearly. An unreadable graph will loose points.

89

EXAMPLE

- Question:
- Given a system with a quantum length of 10 ms in its highest queue, how often would you have to boost jobs back to the highest priority level to guarantee that a single long-running (and potentially starving) job gets at least 5% of the CPU?
- Some combination of n short jobs runs for a total of 10 ms per cycle without relinquishing the CPU
  - E.g. 2 jobs = 5 ms ea; 3 jobs = 3.33 ms ea, 10 jobs = 1 ms ea
  - n jobs always uses full time quantum (10 ms)
  - Batch jobs starts, runs for full quantum of 10ms
  - All other jobs run and context switch totaling the quantum per cycle
  - If 10ms is 5% of the CPU, when must the priority boost be ???
  - <u>**ANSWER**</u> → ***Priority boost should occur every 200ms***

April 15, 2025 | TCSS422: Operating Systems [Spring 2025] School of Engineering and Technology, University of Washington - Tacoma | L5.90

90

QUESTIONS

91