TCSS 422: OPERATING SYSTEMS

Introduction to OS Schedulers

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

April 9, 2024 | TCSS422: Operating Systems [Spring 2024]
School of Engineering and Technology, University of Washington - Tacoma

1

TEXT BOOK COUPON

- 10% off textbook code: **MYLIBRARY10** (*through Friday Apr 12*)
- 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 9, 2024 | TCSS422: Operating Systems [Spring 2024]
School of Engineering and Technology, University of Washington - Tacoma | L5.2

2

TCSS 422 – OFFICE HRS – SPRING 2024

- **Tuesdays after class until 7:00pm**
  Hybrid (In-person/Zoom)
  - This session will be in person in CP 229.
  - Zoom will be monitored when no student is in CP 229.
- Thursdays after class until 7:00pm – Hybrid (In-person/Zoom)
  - Additional office time will be held on Thursdays after class when there is high demand indicated by a busy Tuesday office hour
  - When Thursday Office Hours are planned, Zoom links will be shared via Canvas
  - Questions after class on Thursdays are always entertained even when the formal office hour is not scheduled

April 9, 2024 | TCSS422: Operating Systems [Spring 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.3

3

OBJECTIVES – 4/9

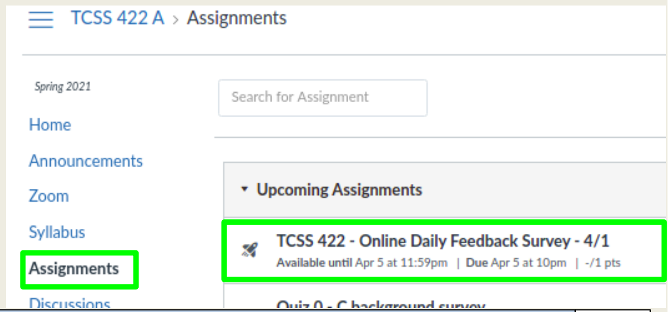
- Questions from 4/4
- Assignment 0
- C Tutorial - Pointers, Strings, Exec in C
- Chapter 6: Limited Direct Execution
- Chapter 7: Scheduling Introduction
  - Scheduling metrics
    - Turnaround time, Jain's Fairness Index, Response time
  - FIFO, SJF, STCF, RR schedulers
- Chapter 8: Multi-level Feedback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

April 9, 2024 | TCSS422: Operating Systems [Spring 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.4

4

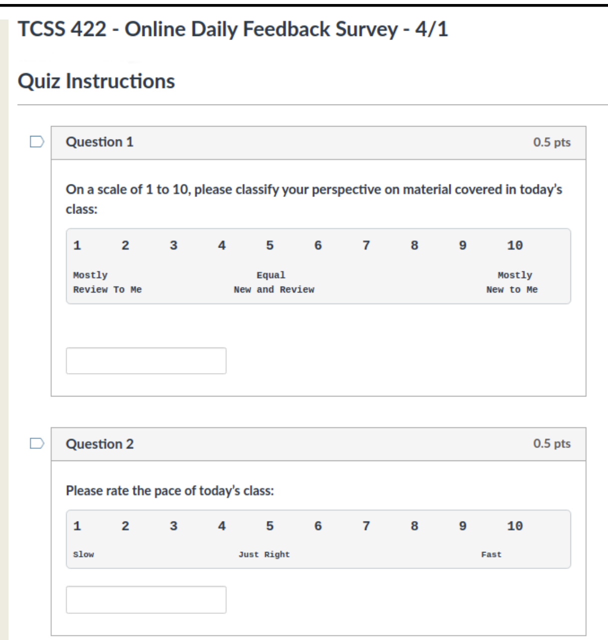
ONLINE DAILY FEEDBACK SURVEY

- 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 9, 2024	TCSS422: Computer Operating Systems [Spring 2024] School of Engineering and Technology, University of Washington - Tacoma	L5.5

5



April 9, 2024 — TCSS422: Computer Operating Systems [Spring 2024]
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 (33 respondents):
- 1-mostly review, 5-equal new/review, 10-mostly new
- **Average – 6.44 (↓ - previous 6.56)**

- Please rate the pace of today's class:
- 1-slow, 5-just right, 10-fast
- **Average – 5.21 (↓ - previous 5.38)**

7

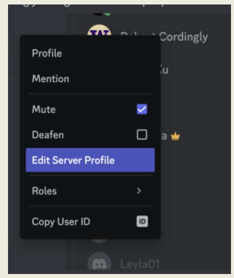
FEEDBACK FROM 4/4

- ***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

8

TCSS 422 DISCORD SERVER

- Please join the TCSS 422 A – Spring 2024 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



9

OBJECTIVES – 4/9

- Questions from 4/4
- **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

10

TCSS 422 – SET VMS

- School of Engineering and Technology hosted Ubuntu 22.04 VMs for TCSS 422 – Spring 2024 are created
- Connection information on how to access SET VMs has been emailed to students who requested BMs

11

OBJECTIVES – 4/9

- Questions from 4/4
- 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

12

OBJECTIVES – 4/9

- Questions from 4/4
- 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 9, 2024	TCSS422: Operating Systems [Spring 2024] School of Engineering and Technology, University of Washington - Tacoma	L5.13

13

CH. 6: LIMITED DIRECT EXECUTION

April 9, 2024 TCSS422: Operating Systems [Spring 2024]
School of Engineering and Technology, University of Washington - Tacoma L5.14

14

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

15

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?

16

MULTITASKING

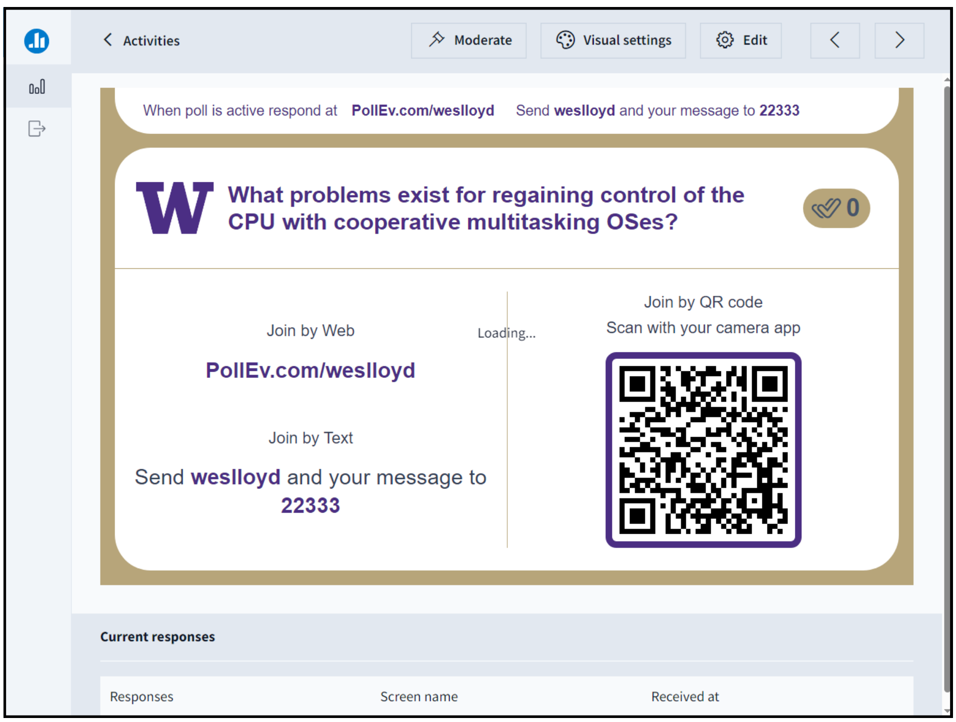
- How/when should the OS regain control of the CPU to switch between processes?

- Cooperative multitasking (mostly pre 32-bit)
  - < \
  - Op
    - Illegal operations
    - 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

April 9, 2024	TCSS422: Operating Systems [Spring 2024] School of Engineering and Technology, University of Washington - Tacoma	L5.17

17



18

QUESTION: MULTITASKING

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

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?

MULTITASKING - 2

- Preemptive multitasking (32 & 64 bit OSes)
- >= Mac OSX, Windows 95+

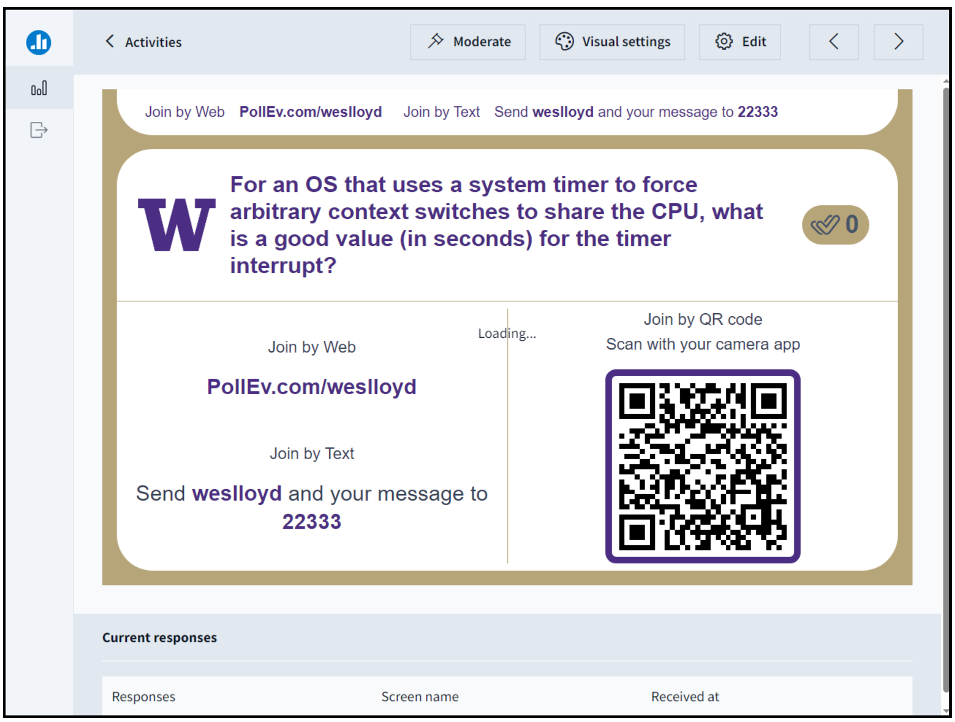
- Timer
  - Rais
  - Inter
    1. Current program is halted
    2. Program states are saved
    3. OS Interrupt handler is run (kernel mode)

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

- (PollEV) What is a good interval for the timer interrupt?

April 9, 2024	TCSS422: Operating Systems [Spring 2024] School of Engineering and Technology, University of Washington - Tacoma	L5.21

21



22

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?

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)

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**

25

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.

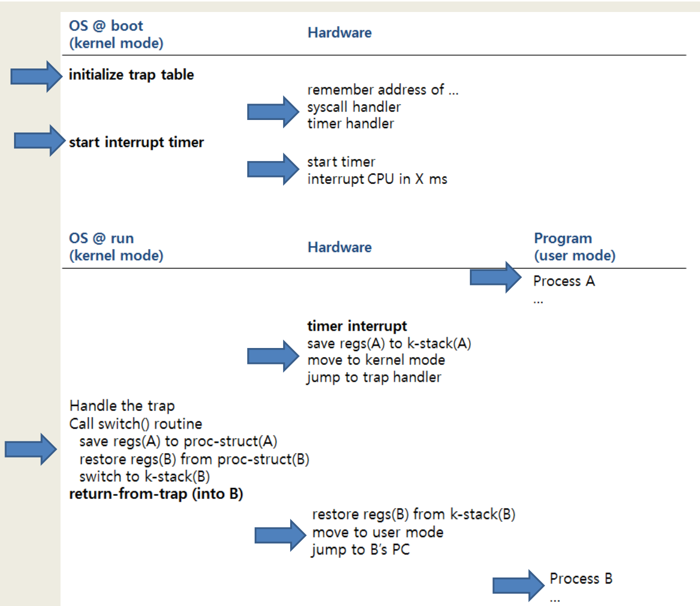
26

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

27



28

OS @ boot (kernel mode)	Hardware	
initialize trap table		
	remember address of … syscall handler timer handler	
start interrupt timer		
	start timer interrupt CPU in X ms	
OS @ run	**Hardware**	**Program**

Context Switch

Call switch() routine save regs(A) to proc-struct(A) restore regs(B) from proc-struct(B) switch to k-stack(B) **return-from-trap (into B)**		
	restore regs(B) from k-stack(B) move to user mode jump to B's PC	
		Process B …

29

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

30

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

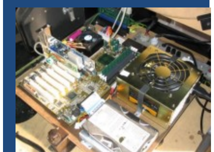
31

OBJECTIVES – 4/9

- Questions from 4/4
- 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

32

CHAPTER 7- SCHEDULING: INTRODUCTION



April 9, 2024 TCSS422: Operating Systems [Spring 2024]
School of Engineering and Technology, University of Washington - Tacoma L5.33

33

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 9, 2024	TCSS422: Operating Systems [Spring 2024] School of Engineering and Technology, University of Washington - Tacoma	L5.34

34

SCHEDULING METRICS - 2

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

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

- 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

35

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

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

36

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

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

$$\frac{(.2 + .7 + .1) = 1^2 = 1}{n \cdot (.2^2 + .7^2 + .1^2)}$$

$n \cdot (.04 + .49 + .01) = 3 \cdot (.54) = 1.62$

$\frac{1}{1.62} = .62$

37

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

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

38

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

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

$$\frac{.3\bar{3} + .3\bar{3} + .3\bar{3} = (.99)^2 \sim .9999 \sim 1}{n \cdot (.33^2 + .33^2 + .33^2)}$$

$$n \cdot (.1089 + .1089 + .1089)$$

$$3 \cdot (.3267) \rightarrow .9801$$

$$\downarrow$$

$$\frac{.99}{.9801} = 1.01 \qquad \frac{.9999}{.9801}$$

39

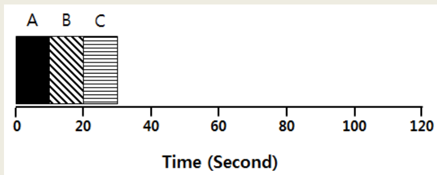
OBJECTIVES – 4/9

- Questions from 4/4
- 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

40

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 9, 2024 | TCSS422: Operating Systems [Spring 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.41

41

OBJECTIVES – 4/9

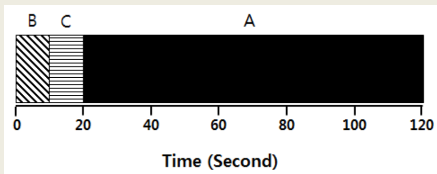
- Questions from 4/4
- Assignment 0
- C Tutorial - Pointers, Strings, Exec in C
- Chapter 6: Limited Direct Execution
- Chapter 7: Scheduling Introduction
  - Scheduling metrics
    - Turnaround time, Jain's Fairness Index, Response time
  - FIFO, SJF, STCF, RR schedulers
- Chapter 8: Multi-level Feedback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

April 9, 2024 | TCSS422: Operating Systems [Spring 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.42

42

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



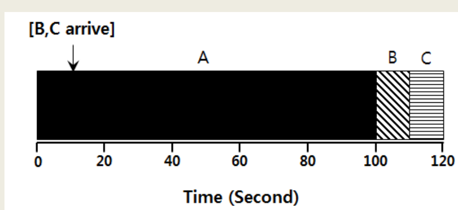
$$Average\ turnaround\ time = \frac{10 + 20 + 120}{3} = 50\ sec$$

April 9, 2024 | TCSS422: Operating Systems [Spring 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.43

43

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



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

April 9, 2024 | TCSS422: Operating Systems [Spring 2024] School of Engineering and Technology, University of Washington - Tacoma | L5.44

44

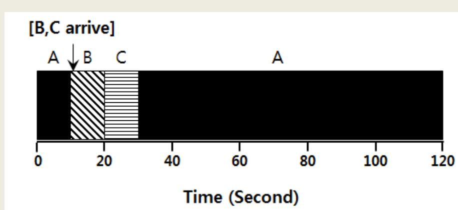
OBJECTIVES – 4/9

- Questions from 4/4
- 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

45

STCF: SHORTEST TIME TO COMPLETION FIRST

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



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

46

WE WILL RETURN AT ~4:55PM

April 9, 2024 TCSS422: Operating Systems [Spring 2024]
School of Engineering and Technology, University of Washington - coma L5.47

47

OBJECTIVES – 4/9

- Questions from 4/4
- 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 9, 2024	TCSS422: Operating Systems [Spring 2024] School of Engineering and Technology, University of Washington - Tacoma	L5.48

48

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?

OBJECTIVES – 4/9

- Questions from 4/4
- 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

RR: ROUND ROBIN

- Run each job awhile, then switch to another distributing the CPU evenly (fairly)
- Scheduling Quantum is called a time slice
- Time a mu time period.

Process	Burst Time
P1	12

RR is fair, but performs poorly on metrics such as turnaround time

Round Robin scheduling algorithm Gantt chart

Scheduling Quantum = 5 seconds

P1	P2	P3	P4	P5	P1	P2	P4	P1

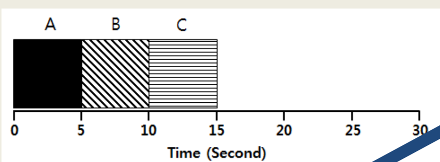
0 5 10 14 19 24 29 32 37 39

51

RR EXAMPLE

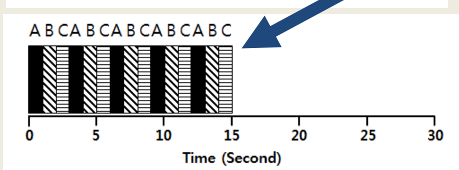
- ABC arrive at time=0, each run for 5 seconds

OVERHEAD not considered



SJF (Bad for Response Time)

$$T_{average\ response} = \frac{0+5+10}{3} = 5sec$$



RR with a time-slice of 1sec (Good for Response Time)

$$T_{average\ response} = \frac{0+1+2}{3} = 1sec$$

52

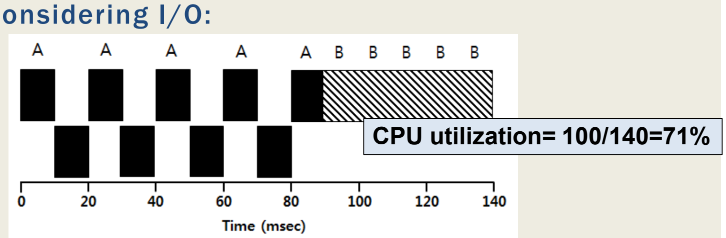
ROUND ROBIN: TRADEOFFS

Short Time Slice		Long Time Slice
Fast Response Time	⟷	Slow Response Time
High overhead from context switching		Low overhead from context switching

- Time slice impact:
  - Turnaround time (for earlier example): ts(1,2,3,4,5)=14,14,13,14,10
  - Fairness: round robin is always fair, J=1

SCHEDULING WITH I/O

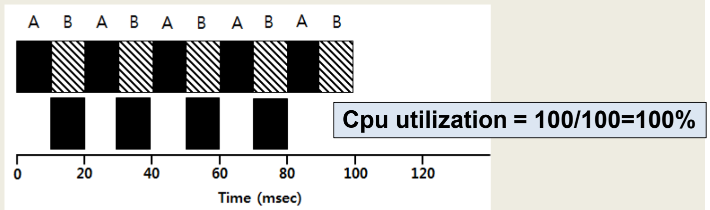
- STCF scheduler
  - A: CPU=50ms, I/O=40ms, 10ms intervals
  - B: CPU=50ms, I/O=0ms
  - Consider A as 10ms subjobs (CPU, then I/O)
- Without considering I/O:



Poor Use of Resources

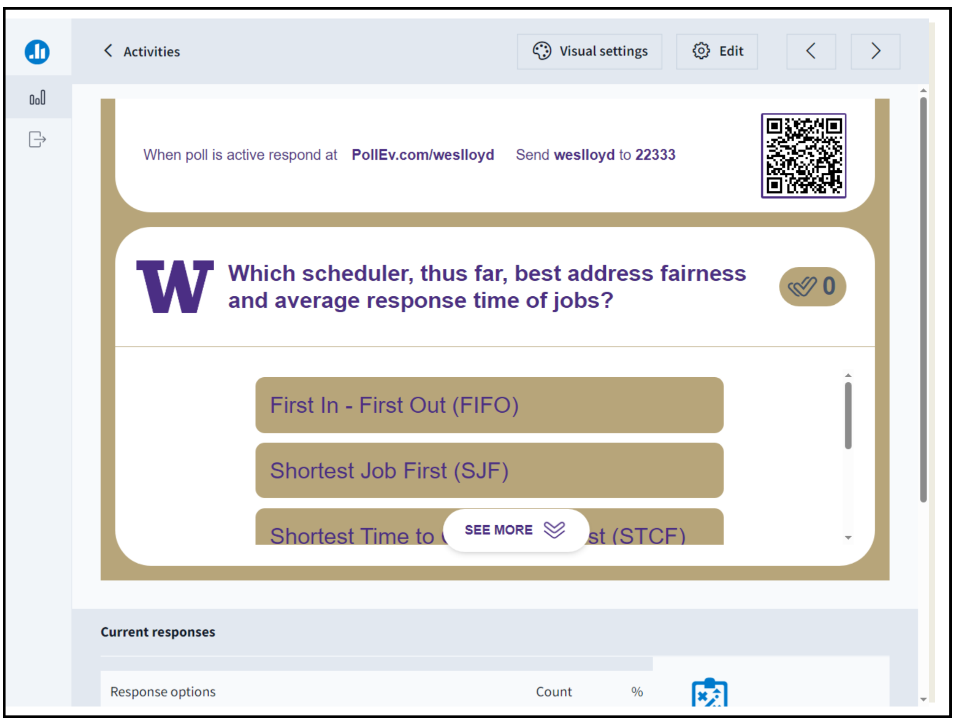
SCHEDULING WITH I/O - 2

- When a job initiates an I/O request
  - A is blocked, waits for I/O to compute, frees CPU
  - STCF scheduler assigns B to CPU
- When I/O completes → raise interrupt
  - Unblock A, STCF goes back to executing A: (10ms sub-job)



Overlap Allows Better Use of Resources

55



56

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

57

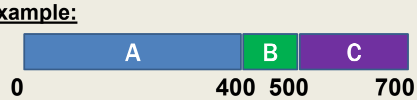
SCHEDULING METRICS

- Consider Three jobs (A, B, C) that require: $time_A$=400ms, $time_B$=100ms, and $time_C$=200ms

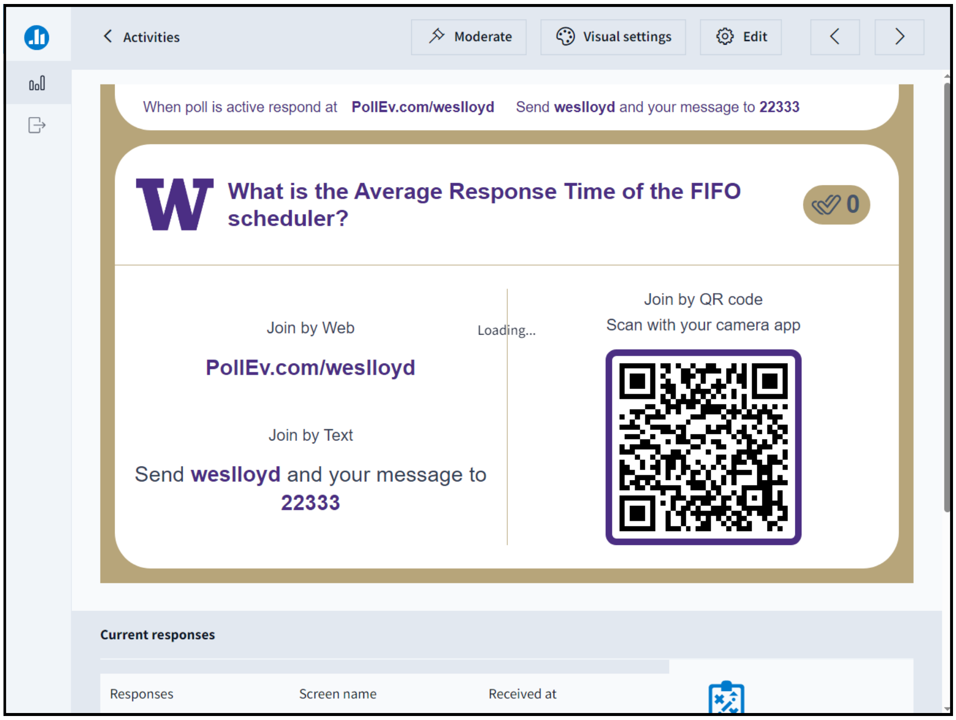
- All jobs arrive at time=0 in the sequence of A B C.

- Draw a scheduling graph to help compute the **average response time (ART)** and **average turnaround time (ATT)** scheduling metrics for the FIFO scheduler.

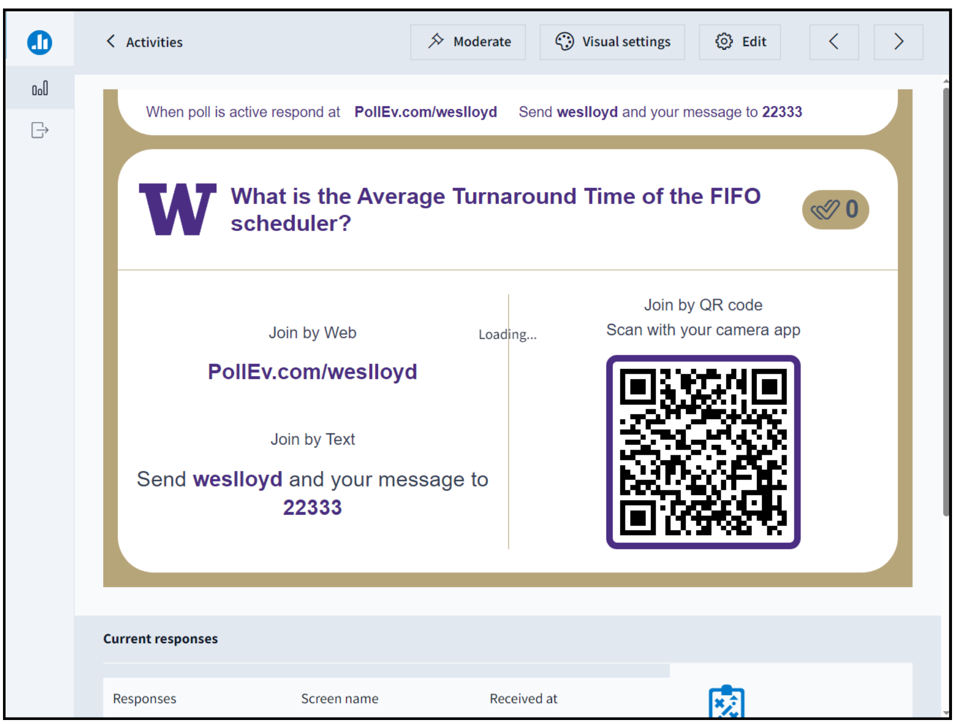
Example:



58


Activities
Moderate
Visual settings
Edit
When poll is active respond at PollEv.com/weslloyd Send weslloyd and your message to 22333
What is the Average Response Time of the FIFO scheduler?
0
Join by Web
PollEv.com/weslloyd
Join by Text
Send weslloyd and your message to 22333
Loading...
Join by QR code
Scan with your camera app
Current responses
Responses
Screen name
Received at

59


Activities
Moderate
Visual settings
Edit
When poll is active respond at PollEv.com/weslloyd Send weslloyd and your message to 22333
What is the Average Turnaround Time of the FIFO scheduler?
0
Join by Web
PollEv.com/weslloyd
Join by Text
Send weslloyd and your message to 22333
Loading...
Join by QR code
Scan with your camera app
Current responses
Responses
Screen name
Received at

60

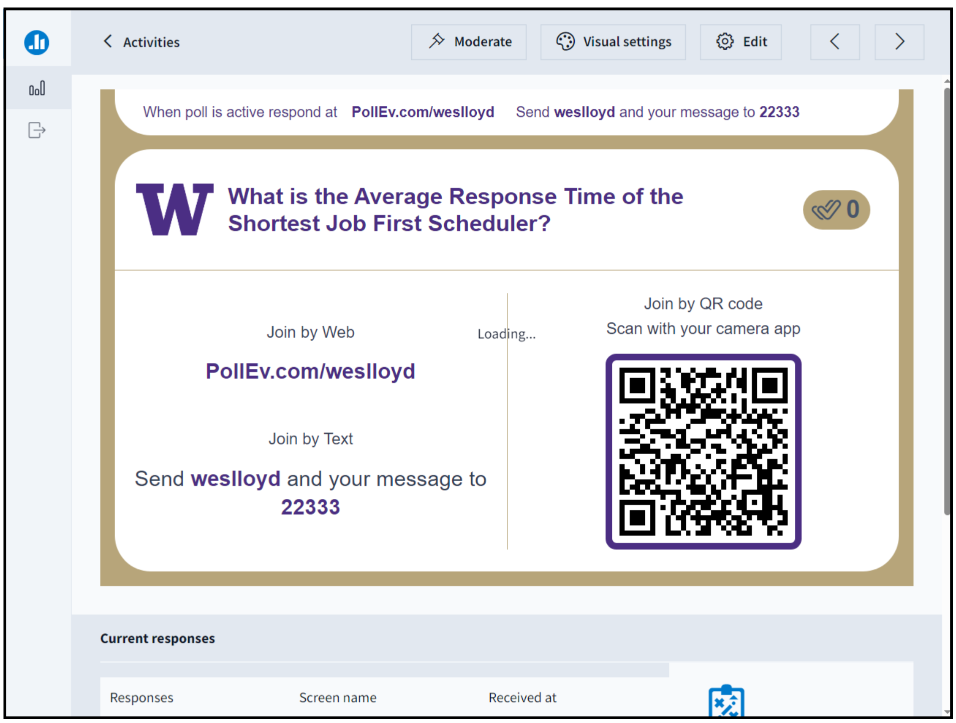
SCHEDULING METRICS

- 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.

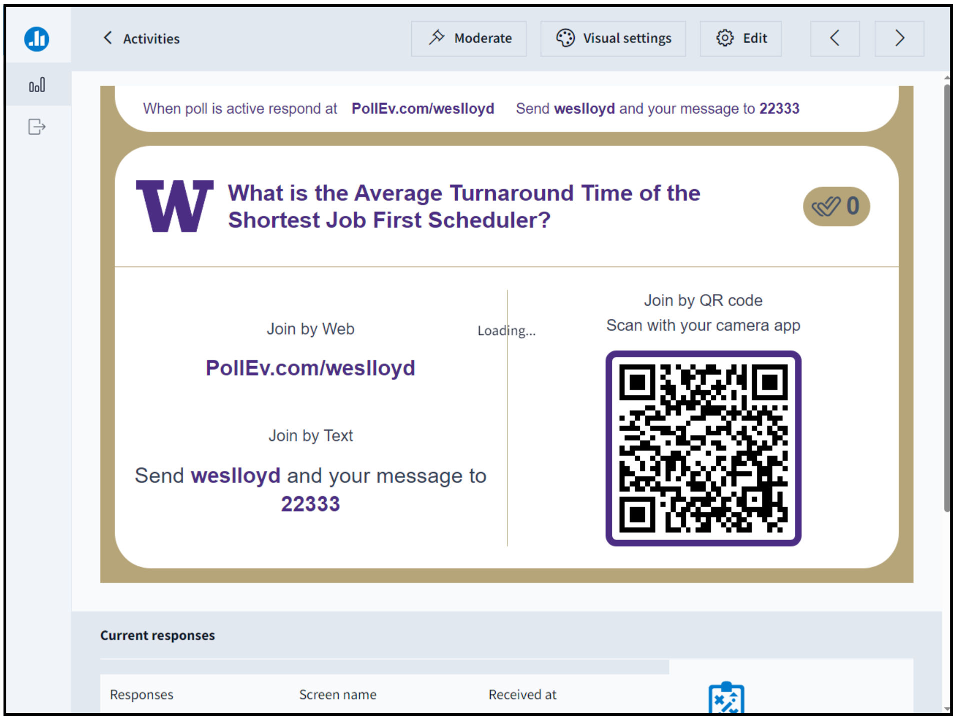
Example:

April 9, 2024	TCSS422: Operating Systems [Spring 2024] School of Engineering and Technology, University of Washington - Tacoma	L5.61

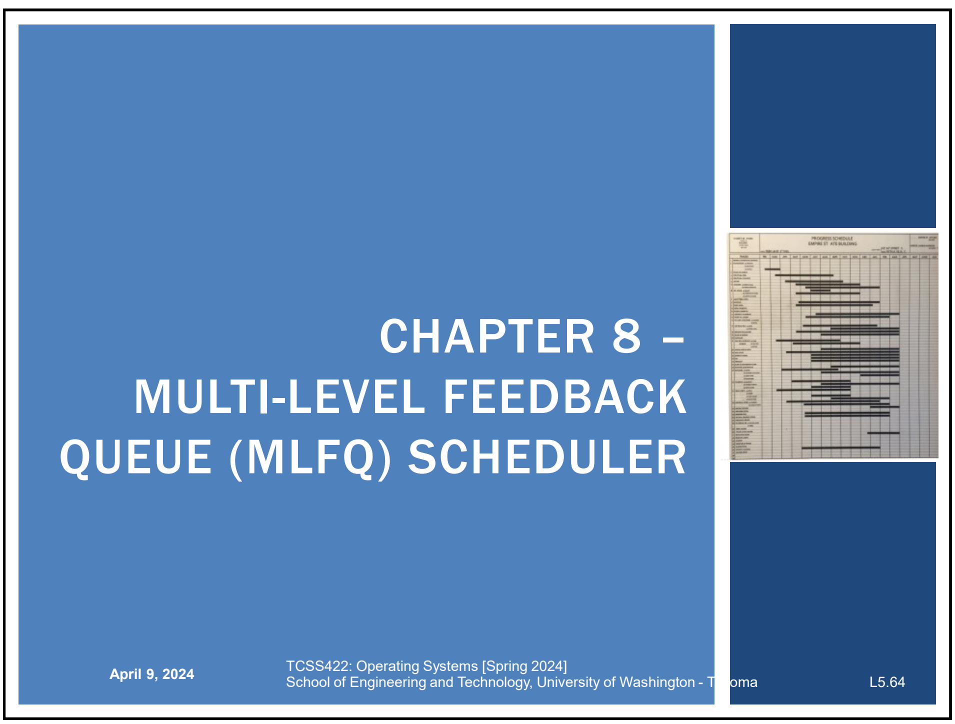
61



62


Activities
Moderate
Visual settings
Edit
When poll is active respond at PollEv.com/weslloyd Send weslloyd and your message to 22333
What is the Average Turnaround Time of the Shortest Job First Scheduler?
0
Join by Web
PollEv.com/weslloyd
Loading...
Join by QR code
Scan with your camera app
Join by Text
Send weslloyd and your message to 22333
Current responses
Responses
Screen name
Received at

63


CHAPTER 8 – MULTI-LEVEL FEEDBACK QUEUE (MLFQ) SCHEDULER
April 9, 2024
TCSS422: Operating Systems [Spring 2024]
School of Engineering and Technology, University of Washington - Tacoma
L5.64

64

OBJECTIVES – 4/9

- Questions from 4/4
- 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

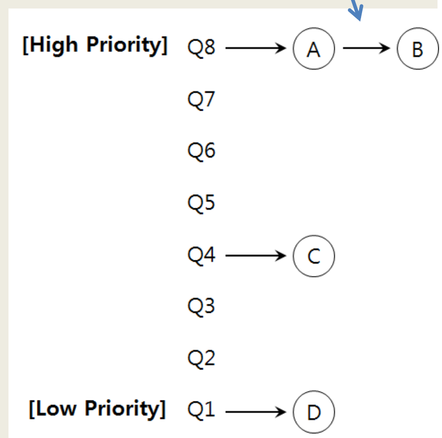
MULTI-LEVEL FEEDBACK QUEUE

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

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



67

MLFQ: DETERMINING JOB PRIORITY

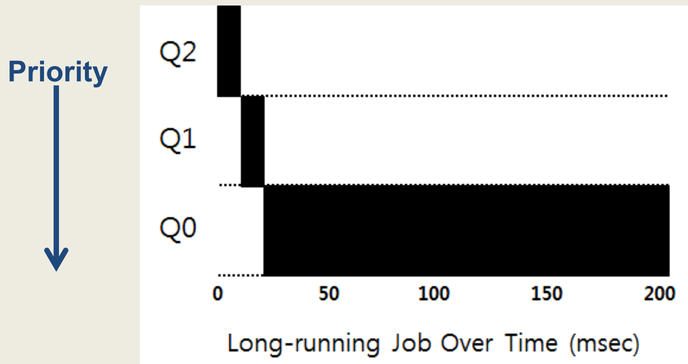
- New arriving jobs are placed into highest priority queue
- If a job uses its entire time slice, priority is reduced (↓)
  - Jobs appears CPU-bound ("batch" job), not interactive (GUI/UI)
- If a job relinquishes the CPU for I/O priority stays the same

MLFQ approximates SJF

68

MLFQ: LONG RUNNING JOB

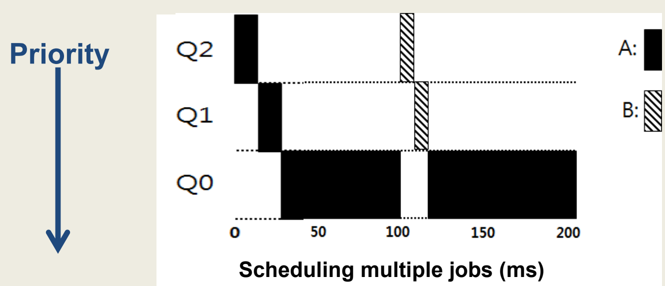
- Three-queue scheduler, time slice=10ms



69

MLFQ: BATCH AND INTERACTIVE JOBS

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

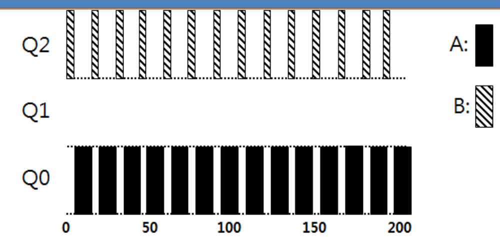


70

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



A Mixed I/O-intensive and CPU-intensive Workload (msec)

April 9, 2024	TCSS422: Operating Systems [Spring 2024] School of Engineering and Technology, University of Washington - Tacoma	L5.71

71

OBJECTIVES – 4/9

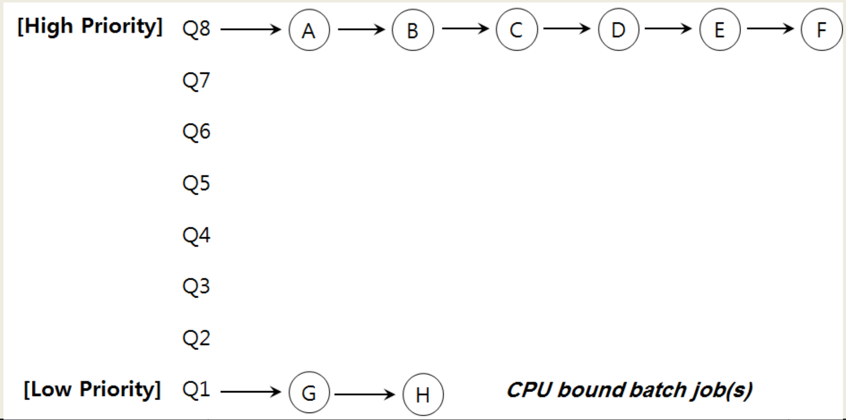
- Questions from 4/4
- Assignment 0
- C Tutorial - Pointers, Strings, Exec in C
- Chapter 6: Limited Direct Execution
- Chapter 7: Scheduling Introduction
  - Scheduling metrics
    - Turnaround time, Jain's Fairness Index, Response time
  - FIFO, SJF, STCF, RR schedulers
- Chapter 8: Multi-level Feedback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

April 9, 2024	TCSS422: Operating Systems [Spring 2024] School of Engineering and Technology, University of Washington - Tacoma	L5.72

72

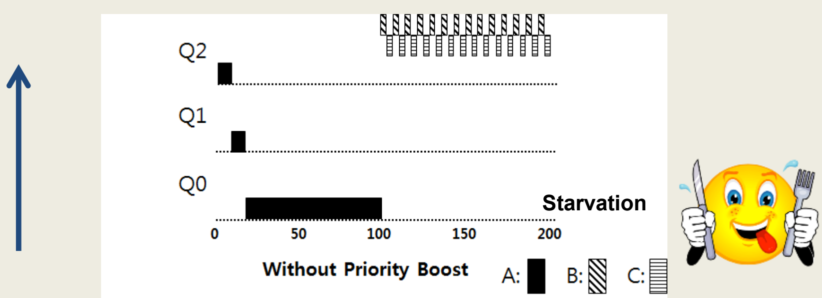
MLFQ: ISSUES

- Starvation



73

RESPONDING TO BEHAVIOR CHANGE

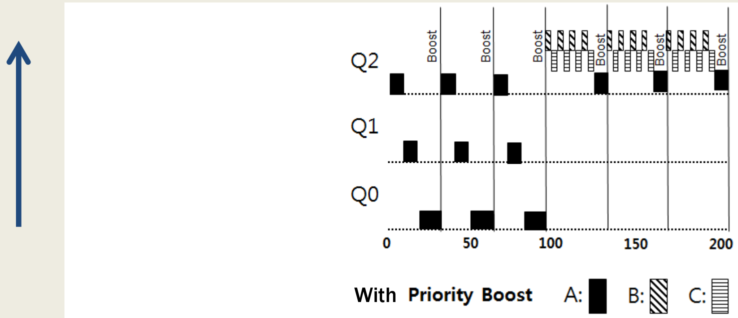


- Priority Boost
  - Reset all jobs to topmost queue after some time interval S

74

RESPONDING TO BEHAVIOR CHANGE - 2

- With priority boost
  - Prevents starvation



75

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!!!

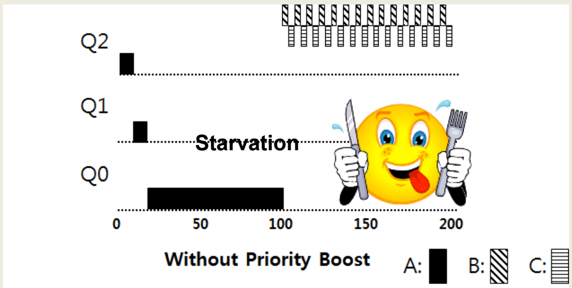
76

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

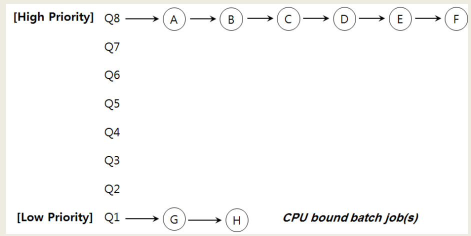


OBJECTIVES – 4/9

- Questions from 4/4
- 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

MLFQ: ISSUES - 2

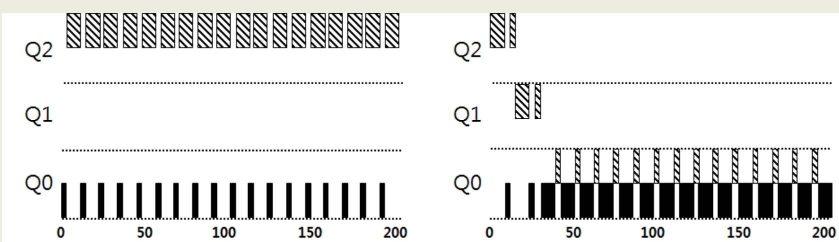
- Gaming the scheduler
  - Issue I/O operation at 99% completion of the time slice
  - Keeps job priority fixed – never lowered
- Job behavioral change
  - CPU/batch process becomes an interactive process



Priority becomes stuck

PREVENTING GAMING

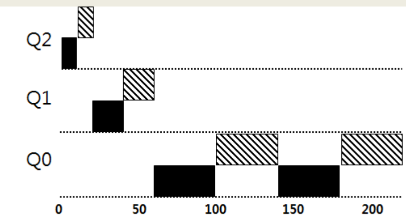
- Improved time accounting:
  - Track total job execution time in the queue
  - Each job receives a fixed time allotment
  - When allotment is exhausted, job priority is lowered



Without(Left) and With(Right) Gaming Tolerance

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

OBJECTIVES – 4/9

- Questions from 4/4
- 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**

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

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.

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.

HIGH |
|
|
MED |
|
|
LOW |________________________________

0

85

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 9, 2024	TCSS422: Operating Systems [Spring 2024] School of Engineering and Technology, University of Washington - Tacoma	L5.86

86

QUESTIONS

87