

TCSS 422: OPERATING SYSTEMS

Common Scheduling Algorithms, Multi-level Feedback Queue (MLFQ) Scheduler, Proportional Share Schedulers



Wes J. Lloyd

School of Engineering and Technology

University of Washington - Tacoma

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.1

1

TEXT BOOK COUPON

■ 10% off textbook code: **LIBRARY10** (through Friday Apr 14)

■ <https://www.lulu.com/shop/andrea-arpaci-dusseau-and-remzi-arpaci-dusseau/operating-systems-three-easy-pieces-softcover-version-100/paperback/product-14mjrrgk.html>

■ With coupon textbook is only \$19.80 + tax & shipping

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.2

2

OFFICE HOURS – SPRING 2023

■ **Tuesdays:**

■ 2:30 to 3:30 pm - CP 229 / Zoom

■ **Fridays**

■ \*1:30 to 2:30 pm – Zoom / (CP 229-on some days)

■ Also available after class

■ Or email for appointment

> Office Hours set based on Student Demographics survey feedback

\* time may be occasionally rescheduled due to faculty meeting conflicts

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.3

3

PANEL AND Q&A ON SOCIAL JUSTICE  
APRIL 18

■ It is fine to view TCSS 422 lecture recording to attend this event:

SSW&E&I Committee Presents

**SOCIAL JUSTICE VISIONS FOR TACOMA**

Join our panel of local practitioners and activists

All Students Welcome!

Scan to pre-register and help plan for food!

**Tuesday, April 18th**  
**4:00pm - 6:30pm**  
**William Philip Hall (WPH)**

UNIVERSITY OF WASHINGTON - TACOMA

For Accommodations: contact Disability Services at 253.941.4388

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.4

4

OBJECTIVES – 4/13

■ **Questions from 4/11**

■ Assignment 0

■ C Tutorial - Pointers, Strings, Exec in C

■ Quiz 1 – Active Reading Chapter 9

■ Chapter 7: Scheduling Introduction

■ Chapter 8: Multi-level Feedback Queue

- MLFQ Scheduler
- Job Starvation
- Gaming the Scheduler
- Examples

■ Chapter 9: Proportional Share Schedulers

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.5

5

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

TCSS 422 A > Assignments

Spring 2023

Home

Announcements

Zoom

Syllabus

**Assignments**

Search for Assignment

Upcoming Assignments

TCSS 422 - Online Daily Feedback Survey - 4/1

Available until Apr 5 at 11:59pm | Due Apr 5 at 10pm | -13 pts

Discussions

Quizzes

Checklist

Grades

April 13, 2023

TCSS422: Computer Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.6

6

Slides by Wes J. Lloyd

L6.1

TCSS 422 - Online Daily Feedback Survey - 4/1

Quiz Instructions

Question 10.5 pts

On a scale of 1 to 10, please classify your perspective on material covered in today's class:

12345678910

Mostly Review to MeEqual New and ReviewMostly New to Me

Question 20.5 pts

Please rate the pace of today's class:

12345678910

SlowJust RightFast

April 13, 2023

TCSS422: Computer Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.7

7

MATERIAL / PACE

Please classify your perspective on material covered in today's class (39 respondents):

1-mostly review, 5-equal new/review, 10-mostly new

Average – 7.64 (↑ - previous 7.58)

Please rate the pace of today's class:

1-slow, 5-just right, 10-fast

Average – 6.34 (↑ - previous 6.08)

April 13, 2023

TCSS422: Computer Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.8

8

FEEDBACK FROM 4/11

I'm having trouble wrapping my head around the scheduling metrics concepts, can you take some time to explain it again?

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.9

9

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 11, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L5.10

10

SCHEDULING METRICS - 2

Scheduling Metric #2: Fairness

Jain's fairness index

Quantifies if jobs receive a fair share of system resources

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

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

April 11, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L5.11

11

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

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

$$J(.2, .7, .1) = \frac{1^2}{3 \cdot .62} = .62$$

April 11, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L5.12

12

Compute average turnaround time for Shortest Job First Scheduler

Job A arrives at t=0, runtime=100  
Job B arrives at t=10, runtime=10  
Job C arrives at t=20, runtime=10

[B,C arrive]

Time (Second)

April 11, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.13

13

FEEDBACK - 2

- *The most that is not clear to me is the context switch.*
- *Since context switching allows multiple processes to make progress but also has some overhead that increases the overall runtime, how do we determine exactly how often to cause a context switch/determine the size of a time slice?*
- It is not necessary for users to determine how often to context switch processes
- The Linux operating system scheduler does this for us
- This is discussed in Ch.9 - Linux Completely Fair Scheduler (CFS)
- In this course we seek to understand "the big picture", but not fine grained detail on how CFS works
- Command to view context switches: `pidstat 1 -w`

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.14

14

OBJECTIVES – 4/13

- Questions from 4/11
- **Assignment 0**
- C Tutorial - Pointers, Strings, Exec in C
- Quiz 1 – Active Reading Chapter 9
- Chapter 7: Scheduling Introduction
- Chapter 8: Multi-level Feedback Queue
  - MLFQ Scheduler
  - Job Starvation
  - Gaming the Scheduler
  - Examples
- Chapter 9: Proportional Share Schedulers

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.15

15

OBJECTIVES – 4/13

- Questions from 4/11
- Assignment 0
- **C Tutorial - Pointers, Strings, Exec in C**
- Quiz 1 – Active Reading Chapter 9
- Chapter 7: Scheduling Introduction
- Chapter 8: Multi-level Feedback Queue
  - MLFQ Scheduler
  - Job Starvation
  - Gaming the Scheduler
  - Examples
- Chapter 9: Proportional Share Schedulers

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.16

16

OBJECTIVES – 4/13

- Questions from 4/11
- Assignment 0
- C Tutorial - Pointers, Strings, Exec in C
- **Quiz 1 – Active Reading Chapter 9**
- Chapter 7: Scheduling Introduction
- Chapter 8: Multi-level Feedback Queue
  - MLFQ Scheduler
  - Job Starvation
  - Gaming the Scheduler
  - Examples
- Chapter 9: Proportional Share Schedulers

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.17

17

OBJECTIVES – 4/13

- Questions from 4/11
- Assignment 0
- C Tutorial - Pointers, Strings, Exec in C
- Quiz 1 – Active Reading Chapter 9
- **Chapter 7: Scheduling Introduction**
- Chapter 8: Multi-level Feedback Queue
  - MLFQ Scheduler
  - Job Starvation
  - Gaming the Scheduler
  - Examples
- Chapter 9: Proportional Share Schedulers


April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.18

18

CHAPTER 7-  
SCHEDULING:  
INTRODUCTION



April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.19

19

CHAPTER 7

■ Chapter 7: Scheduling Introduction

■ Scheduling metrics

■ Turnaround time, Jain's Fairness Index, Response time

■ FIFO, SJF, **STCF**, RR schedulers

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.20

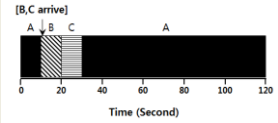
20

STCF:  
SHORTEST TIME TO COMPLETION FIRST

■ Consider: duration a=100sec, b/c=10sec

■ A<sub>len</sub>=100 A<sub>arrival</sub>=0

■ B<sub>len</sub>=10, B<sub>arrival</sub>=10, C<sub>len</sub>=10, C<sub>arrival</sub>=10



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

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.21

21

CHAPTER 7

■ Chapter 7: Scheduling Introduction

■ Scheduling metrics

■ Turnaround time, Jain's Fairness Index, **Response time**

■ FIFO, SJF, **STCF**, RR schedulers

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.22

22

SCHEDULING METRICS - 3

■ Scheduling Metric #3: **Response Time**

■ Time from when job arrives until it starts execution

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

■ STCF, SJF, FIFO

■ can perform poorly with respect to response time

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

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.23

23

CHAPTER 7

■ Chapter 7: Scheduling Introduction

■ Scheduling metrics

■ Turnaround time, Jain's Fairness Index, Response time

■ FIFO, SJF, STCF, **RR** schedulers

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.24

24

RR: ROUND ROBIN

Run each job awhile, then switch to another distributing the CPU evenly (fairly)

Scheduling Quantum is called a time slice

Time slice is a multiple of the 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

P1	P2	P3	P4	P5	P1	P2	P4	P1
0	5	10	14	19	24	29	32	37

Scheduling Quantum = 5 seconds

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.25

25

RR EXAMPLE

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

A

B

C

0

5

10

15

20

25

30

Time (Second)

SJF (Bad for Response Time)

A

B

C

A

B

C

A

B

C

0

5

10

15

20

25

30

Time (Second)

OVERHEAD not considered

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

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

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

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.26

26

ROUND ROBIN: TRADEOFFS

Short Time Slice

Fast Response Time

High overhead from context switching

Long Time Slice

Slow Response Time

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

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.27

27

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:

A

A

A

A

A

A

A

A

A

A

0

20

40

60

80

100

120

140

Time (msec)

CPU utilization= 100/140=71%

Poor Use of Resources

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.28

28

SCHEDULING WITH I/O - 2

When a job initiates an I/O request

A is blocked, waits for I/O to complete, 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)

A

B

A

B

A

B

A

B

A

B

0

20

40

60

80

100

120

Time (msec)

Cpu utilization = 100/100=100%

Overlap Allows Better Use of Resources

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.29

29

Respond at [pollen.com/westloyd](https://pollen.com/westloyd)

Text WESLOYD to 22333 once to join, then 1, 2, 3, 4, 5...

Which scheduler, thus far, best address fairness and average response time of jobs?

First In - First Out (FIFO)

Shortest Job First (SJF)

Shortest Time to Completion First (STCF)

Round Robin

None of the Above

All of the Above

1

2

3

4

5

6

Powered by Poll Everywhere

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollen.com/app](https://pollen.com/app)

Total Results: 0

30

Slides by Wes J. Lloyd

L6.5

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 13, 2023

TCCS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

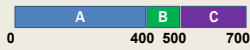
L6.31

31

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 13, 2023

TCCS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.32

32


What is the Average Response Time of the FIFO scheduler?

Powered by  Poll Everywhere

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [poller.com/app](https://poller.com/app)

33

What is the Average Turnaround Time of the FIFO scheduler?

Powered by  Poll Everywhere

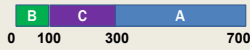
Start the presentation to see live content. For screen share software, share the entire screen. Get help at [poller.com/app](https://poller.com/app)

34

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 13, 2023

TCCS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.35

35

What is the Average Response Time of the Shortest Job First Scheduler?

Powered by  Poll Everywhere

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [poller.com/app](https://poller.com/app)

36

### What is the Average Turnaround Time of the Shortest Job First Scheduler?

“ 7.75 milli ”

“ 2ms ”

“ Too long :( ”

“ 1000 ”

Powered by Poll Everywhere

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [poll.com/help](https://poll.com/help)

37

WE WILL RETURN AT  
4:50PM

April 13, 2023

TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.38

38

OBJECTIVES – 4/13

- Questions from 4/11
- Assignment 0
- C Tutorial - Pointers, Strings, Exec in C
- Quiz 1 – Active Reading Chapter 9
- Chapter 7: Scheduling Introduction
- Chapter 8: Multi-level Feedback Queue
  - MLFQ Scheduler**
    - Job Starvation
    - Gaming the Scheduler
    - Examples
- Chapter 9: Proportional Share Schedulers

April 13, 2023

TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.39

39

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

April 13, 2023

TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.40

40

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 13, 2023

TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.41

41

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

[High Priority]

Q8 → A → B

Q7

Q6

Q5

Q4 → C

Q3

Q2

[Low Priority]

Q1 → D

April 13, 2023

TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.42

42

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 13, 2023

TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.43

43

MLFQ: LONG RUNNING JOB

- Three-queue scheduler, time slice=10ms

April 13, 2023

TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.44

44

MLFQ: BATCH AND INTERACTIVE JOBS

- A<sub>arrival\_time</sub> =0ms, A<sub>run\_time</sub> =200ms,
- B<sub>run\_time</sub> =20ms, B<sub>arrival\_time</sub> =100ms

April 13, 2023

TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.45

45

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

April 13, 2023

TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.46

46

OBJECTIVES – 4/13

- Questions from 4/11
- Assignment 0
- C Tutorial - Pointers, Strings, Exec in C
- Quiz 1 – Active Reading Chapter 9
- Chapter 7: Scheduling Introduction
- Chapter 8: Multi-level Feedback Queue
  - MLFQ Scheduler
  - Job Starvation
  - Gaming the Scheduler
  - Examples
- Chapter 9: Proportional Share Schedulers

April 13, 2023

TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.47

47

MLFQ: ISSUES

- Starvation

April 13, 2023

TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.48

48



## L6.49

L6.50

L6.51

L6.S2

L6.53

L6.54

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

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.55

55

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 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.56

56

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 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.57

57

MLFQ RULE SUMMARY

- The refined set of MLFQ rules:
  - Rule 1:** If  $\text{Priority}(A) > \text{Priority}(B)$ , A runs (B doesn't).
  - Rule 2:** If  $\text{Priority}(A) = \text{Priority}(B)$ , 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 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.58

58

OBJECTIVES – 4/13

- Questions from 4/11
- Assignment 0
- C Tutorial - Pointers, Strings, Exec in C
- Quiz 1 – Active Reading Chapter 9
- Chapter 7: Scheduling Introduction
- Chapter 8: Multi-level Feedback Queue
  - MLFQ Scheduler
  - Job Starvation
  - Gaming the Scheduler
  - Examples
- Chapter 9: Proportional Share Schedulers

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.59

59

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.

60

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	10
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 lose points.

time slice is job time

61

### EXAMPLE

■ Question:

■ Given a system with a quantum length of 10 ms **for all jobs** 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?

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.62

62

### EXAMPLE

■ Question:

■ Given a system with a quantum length of 10 ms **for all jobs** 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?

0.05 PB = 10  
PB = 10 / 0.05 = 200 ms

63

### EXAMPLE

■ Question:

■ Given a system with a quantum length of 10 ms **for all jobs** 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 in highest queue (10 ms)
- Batch jobs starts, runs for full quantum of 10ms, pushed to lower queue
- 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 ???

■ **ANSWER** → Priority boost should occur every 200ms

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.64

64

### OBJECTIVES – 4/13

- Questions from 4/11
- Assignment 0
- C Tutorial - Pointers, Strings, Exec in C
- Quiz 1 – Active Reading Chapter 9
- Chapter 7: Scheduling Introduction
- Chapter 8: Multi-level Feedback Queue
  - MLFQ Scheduler
  - Job Starvation
  - Gaming the Scheduler
  - Examples
- **Chapter 9: Proportional Share Schedulers**

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.65

65

### CHAPTER 9 - PROPORTIONAL SHARE SCHEDULER

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.66

66

OBJECTIVES – 4/13

Chapter 9: Proportional Share Schedulers

Lottery scheduler

Ticket mechanisms

Stride scheduler

Linux Completely Fair Scheduler

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.67

67

PROPORTIONAL SHARE SCHEDULER

Also called fair-share scheduler or lottery scheduler

Guarantees each job receives some percentage of CPU time based on share of "tickets"

Each job receives an allotment of tickets

% of tickets corresponds to potential share of a resource

Can conceptually schedule any resource this way

- CPU, disk I/O, memory

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.68

68

LOTTERY SCHEDULER

Simple implementation

Just need a random number generator

- Picks the winning ticket

Maintain a data structure of jobs and tickets (list)

Traverse list to find the owner of the ticket

Consider sorting the list for speed

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.69

69

LOTTERY SCHEDULER IMPLEMENTATION

head → Job A Tix100 → Job B Tix50 → Job C Tix250 → NULL

```
1 // counter: used to track if we've found the winner yet
2 int counter = 0;
3
4 // winner: use some call to a random number generator to
5 // get a value, between 0 and the total # of tickets
6 int winner = getRandom(0, totaltickets);
7
8 // current: use this to walk through the list of jobs
9 node_t *current = head;
10
11 // loop until the sum of ticket values is > the winner
12 while (current) {
13     counter = counter + current->tickets;
14     if (counter > winner)
15         break; // found the winner
16     current = current->next;
17 }
18 // 'current' is the winner: schedule it...
```

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.70

70

OBJECTIVES – 4/13

Chapter 9: Proportional Share Schedulers

Lottery scheduler

Ticket mechanisms

Stride scheduler

Linux Completely Fair Scheduler

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.71

71

TICKET MECHANISMS

Ticket currency / exchange

User allocates tickets in any desired way

OS converts user currency into global currency

Example:

- There are 200 global tickets assigned by the OS

User A → 500 (A's currency) to A1 → 50 (global currency)  
→ 500 (A's currency) to A2 → 50 (global currency)

User B → 10 (B's currency) to B1 → 100 (global currency)

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.72

72

TICKET MECHANISMS - 2

- Ticket transfer
  - Temporarily hand off tickets to another process
- Ticket inflation
  - Process can temporarily raise or lower the number of tickets it owns
  - If a process needs more CPU time, it can boost tickets.

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.73

73

LOTTERY SCHEDULING

- Scheduler picks a **winning** ticket
  - Load the job with the winning ticket and run it
- Example:
  - Given 100 tickets in the pool
    - Job A has 75 tickets: 0 - 74
    - Job B has 25 tickets: 75 - 99

Scheduler's winning tickets: 63 85 70 39 76 17 29 41 36 39 10 99 68 83 63  
Scheduled job: A B A A B A A A A A B A B A

- But what do we know about probability of a coin flip?

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.74

74

COIN FLIPPING

- Equality of distribution (fairness) requires a lot of flips!

Similarly, Lottery scheduling requires lots of "rounds" to achieve fairness.

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.75

75

LOTTERY FAIRNESS

- With two jobs
  - Each with the same number of tickets (t=100)

When the job length is not very long, average unfairness can be quite severe.

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.76

76

LOTTERY SCHEDULING CHALLENGES

- What is the best approach to assign tickets to jobs?
  - Typical approach is to assume users know best
  - Users are provided with tickets, which they allocate as desired
- How should the OS automatically distribute tickets upon job arrival?
  - What do we know about incoming jobs a priori ?
  - Ticket assignment is really an open problem...

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.77

77

OBJECTIVES – 4/13

- Chapter 9: Proportional Share Schedulers
  - Lottery scheduler
  - Ticket mechanisms
  - **Stride scheduler**
  - Linux Completely Fair Scheduler

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.78

78

STRIDE SCHEDULER

- Addresses statistical probability issues with lottery scheduling
- Instead of guessing a random number to select a job, simply count...

April 13, 2023

TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.79

79

STRIDE SCHEDULER - 2

- Jobs have a "stride" value
  - A stride value describes the counter pace when the job should give up the CPU
  - Stride value is **inverse in proportion** to the job's number of tickets (more tickets = smaller stride)
- Total system tickets = 10,000
  - Job A has 100 tickets →  $A_{stride} = 10000/100 = 100$  stride
  - Job B has 50 tickets →  $B_{stride} = 10000/50 = 200$  stride
  - Job C has 250 tickets →  $C_{stride} = 10000/250 = 40$  stride
- Stride scheduler tracks "pass" values for each job (A, B, C)

April 13, 2023

TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.80

80

STRIDE SCHEDULER - 3

- Basic algorithm:
  - Stride scheduler picks job with the lowest pass value
  - Scheduler increments job's pass value by its stride and starts running
  - Stride scheduler increments a counter
  - When counter exceeds pass value of current job, pick a new job (go to 1)
- KEY:** When the counter reaches a job's "PASS" value, the scheduler passes on to the next job...

April 13, 2023

TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.81

81

STRIDE SCHEDULER - EXAMPLE

- Stride values
  - Tickets = priority to select job
  - Stride is inverse to tickets
  - Lower stride = more chances to run (higher priority)

Priority

C stride = 40

A stride = 100

B stride = 200

April 13, 2023

TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.82

82

STRIDE SCHEDULER EXAMPLE - 2

- Three-way tie: randomly pick job A (all pass values=0)
- Set A's pass value to A's stride = 100
- Increment counter until > 100
- Pick a new job: two-way tie

Pass(A) (stride=100)	Pass(B) (stride=200)	Pass(C) (stride=40)	Who Runs?
0	0	0	A
100	0	0	B
100	200	0	C
100	200	40	C
100	200	80	C
100	200	120	A
200	200	120	C
200	200	160	C
200	200	200	...

Tickets

C = 250

A = 100

B = 50

Initial job selection is random. All @ 0

C has the most tickets and receives a lot of opportunities to run...

April 13, 2023

TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.83

83

STRIDE SCHEDULER EXAMPLE - 3

- We set A's counter (pass value) to A's stride = 100
- Next scheduling decision between B (pass=0) and C (pass=0)
  - Randomly choose B
- C has the lowest counter for next 3 rounds

Pass(A) (stride=100)	Pass(B) (stride=200)	Pass(C) (stride=40)	Who Runs?
0	0	0	A
100	0	0	B
100	200	0	C
100	200	40	C
100	200	80	C
100	200	120	A
200	200	120	C
200	200	160	C
200	200	200	...

Tickets

C = 250

A = 100

B = 50

C has the most tickets and is selected to run more often ...

April 13, 2023

TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.84

84

STRIDE SCHEDULER EXAMPLE - 4

- Job counters support determining which job to run next
- Over time jobs are scheduled to run based on their priority represented as their **share of tickets**...
- Tickets are analogous to job priority**

Pass(A) (stride=100)	Pass(B) (stride=200)	Pass(C) (stride=40)	Who Runs?
0	0	0	A
100	0	0	B
100	200	0	C
100	200	40	C
100	200	80	C
100	200	120	A
200	200	120	C
200	200	160	C
200	200	200	...

Tickets  
C = 250  
A = 100  
B = 50

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.85

85

OBJECTIVES – 4/13

- Chapter 9: Proportional Share Schedulers**
  - Lottery scheduler
  - Ticket mechanisms
  - Stride scheduler
  - Linux Completely Fair Scheduler**

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.86

86

LINUX: COMPLETELY FAIR SCHEDULER (CFS)

- Large Google datacenter study:  
"Profiling a Warehouse-scale Computer" (Kanev et al.)
- Monitored 20,000 servers over 3 years
- Found 20% of CPU time spent in the Linux kernel
- 5% of CPU time spent in the CPU scheduler!
- Study highlights importance for high performance OS kernels and CPU schedulers!

Figure 5: Kernel time, especially time spent in the scheduler, is a significant fraction of WSC cycles.

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.87

87

LINUX: COMPLETELY FAIR SCHEDULER (CFS)

- Loosely based on the stride scheduler
- CFS models system as a Perfect Multi-Tasking System
  - In perfect system every process of the same priority (class) receive exactly  $1/n^{\text{th}}$  of the CPU time
- Each scheduling class has a runqueue
  - Groups process of same class
  - In class, scheduler picks task w/ lowest **vruntime** to run
  - Time slice varies based on how many jobs in shared runqueue
  - Minimum time slice prevents too many context switches (e.g. 3 ms)

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.88

88

COMPLETELY FAIR SCHEDULER - 2

- Every thread/process has a scheduling class (policy):
- Normal classes:** SCHED\_OTHER (TS), SCHED\_IDLE, SCHED\_BATCH
  - TS = Time Sharing
- Real-time classes:** SCHED\_FIFO (FF), SCHED\_RR (RR)
- How to show scheduling class and priority:
- #class**  
`ps -elfc`
- #priority (nice value)**  
`ps ax -o pid,ni,cls,pri,cmd`

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.89

89

COMPLETELY FAIR SCHEDULER - 3

- Linux  $\geq$  2.6.23: Completely Fair Scheduler (CFS)
- Linux  $<$  2.6.23: O(1) scheduler
- Linux maintains simple counter (vruntime) to track how long each thread/process has run
- CFS picks process with lowest vruntime to run next
- CFS adjusts timeslice based on # of proc waiting for the CPU
- Kernel parameters that specify CFS behavior:

```
$ sudo sysctl kernel.sched_latency_ns
kernel.sched_latency_ns = 24000000
$ sudo sysctl kernel.sched_min_granularity_ns
kernel.sched_min_granularity_ns = 3000000
$ sudo sysctl kernel.sched_wakeup_granularity_ns
kernel.sched_wakeup_granularity_ns = 4000000
```

April 13, 2023

TCSS422: Operating Systems [Spring 2023]  
School of Engineering and Technology, University of Washington - Tacoma

L6.90

90

COMPLETELY FAIR SCHEDULER - 4

- `Sched_min_granularity_ns` (3ms)
  - Time slice for a process: busy system (w/ full runqueue)
  - If system has idle capacity, time slice exceed the min as long as difference in `vruntime` between running process and process with lowest `vruntime` is less than `sched_wakeup_granularity_ns` (4ms)
- Scheduling time period is: total cycle time for iterating through a set of processes where each is allowed to run (like round robin)
- Example:
  - `sched_latency_ns` (24ms)
  - if (proc in runqueue < `sched_latency_ns`/`sched_min_granularity`) or
  - `sched_min_granularity` \* number of processes in runqueue

Ref: [https://www.gyrfuturals.com/sched\\_min\\_granularity\\_ns-sched\\_latency\\_ns-efw-timeline-processes/](https://www.gyrfuturals.com/sched_min_granularity_ns-sched_latency_ns-efw-timeline-processes/)

April 13, 2023

TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.91

91

CFS TRADEOFF

- **HIGH**
  - `sched_min_granularity_ns` (timeslice)
  - `sched_latency_ns`
  - `sched_wakeup_granularity_ns`

reduced context switching → less overhead  
poor near-term fairness
- **LOW**
  - `sched_min_granularity_ns` (timeslice)
  - `sched_latency_ns`
  - `sched_wakeup_granularity_ns`

increased context switching → more overhead  
better near-term fairness

April 13, 2023

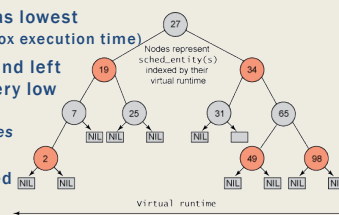
TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.92

92

COMPLETELY FAIR SCHEDULER - 5

- Runqueues are stored using a linux red-black tree
  - Self balancing binary tree - nodes indexed by `vruntime`
- Leftmost node has lowest `vruntime` (approx execution time)
- Walking tree to find left most node has very low big O complexity:  $\sim O(\log N)$  for  $N$  nodes
- Completed processes removed



April 13, 2023

TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.93

93

CFS: JOB PRIORITY

- Time slice: Linux **"Nice value"**
  - Nice predates the CFS scheduler
  - Top shows nice values
- Process command (nice & priority):  
`ps ax -o pid,ni,cmd,%cpu, pri`
- Nice Values: from -20 to 19
  - Lower is **higher** priority, default is 0
  - Vruntime is a weighted time measurement
  - Priority weights the calculation of vruntime within a runqueue to give high priority jobs a boost.
  - Influences job's position in rb-tree

```
static const int prio_to_weight[40] = {
    /* -20 */ 88761, 71755, 56883, 46273, 36291,
    /* -15 */ 29154, 23254, 18705, 14949, 11916,
    /* -10 */ 9549, 7639, 6180, 4984, 3986,
    /* -5 */ 3121, 2501, 1991, 1586, 1277,
    /* 0 */ 1054, 805, 635, 526, 423,
    /* 5 */ 335, 272, 215, 172, 137,
    /* 10 */ 110, 87, 70, 56, 45,
    /* 15 */ 36, 29, 23, 18, 15,
    /* 20 */ 12
};
```

April 13, 2023

TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.94

94

COMPLETELY FAIR SCHEDULER - 6

- CFS tracks cumulative job run time in `vruntime` variable
- The task on a given runqueue with the lowest `vruntime` is scheduled next
- `struct sched_entity` contains `vruntime` parameter
  - Describes process execution time in nanoseconds
  - Value is not pure runtime, is weighted based on job priority
  - Perfect scheduler → achieve equal `vruntime` for all processes of same priority
- Sleeping jobs: upon return reset `vruntime` to lowest value in system
  - Jobs with frequent short sleep **SUFFER !!**
- Key takeaway:  
**Identifying the next job to schedule is really fast!**

April 13, 2023

TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.95

95

COMPLETELY FAIR SCHEDULER - 7

- More information:
- Man page: "man sched" : Describes Linux scheduling API
- <http://manpages.ubuntu.com/manpages/bionic/man7/sched.7.html>
- <https://www.kernel.org/doc/Documentation/scheduler/sched-design-CFS.txt>
- [https://en.wikipedia.org/wiki/Completely\\_Fair\\_Scheduler](https://en.wikipedia.org/wiki/Completely_Fair_Scheduler)
- See paper: The Linux Scheduler – a Decade of Wasted Cores
- <http://www.ece.ubc.ca/~sasha/papers/eurosys16-final29.pdf>

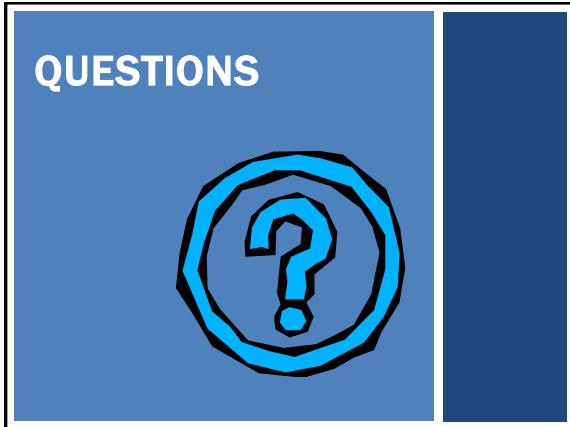
April 13, 2023

TCSS422: Operating Systems (Spring 2023)  
School of Engineering and Technology, University of Washington - Tacoma

L6.96

96





97