CPU Scheduling

CSC 213 – October 3, 2022
Agenda for Today

1. Bug of the Day
2. Week 6 Assignment
3. Q&A
4. CPU Scheduling
5. Wrap Up
Bug of the Day
What's wrong here?

```c
void make_int_array(int* ptr, size_t count) {
    ptr = malloc(sizeof(int) * count);
}

int main() {
    int* arr;
    make_int_array(arr, 4);
    arr[0] = 0;
    arr[1] = 1;
    ...
}
```

**Explanation**
The `make_int_array` function calls `malloc`, saves the pointer to a local, and then the pointer disappears.

Memory leak! (ptr never freed)
Segfault! (arr[0] = 0)
Assignment Questions

What happened to the chdir tests you were going to write?

I didn't get these working. If your assignment passes the gradescope tests and does not use the chdir system call, you may assume it is correct (it probably is).
Lab Questions

Do we need to support freeing large objects?
   No. You can just ignore attempts to free large objects.

Should we check the magic number in both free and usable size?
   Yes.
CPU Scheduling
Scheduler Basics

Why do we need a CPU scheduler?

The OS is running multiple tasks, but the CPU can't do everything at the same time. Just like memory, time on the CPU is a limited resource that the OS needs to allocate. Making those allocations is the scheduler's job. The OS is choosing what to run, and the scheduler tries to make that choice efficiently.
Scheduling Terminology

Batch Scheduler
A scheduler that starts jobs and runs them to completion.

Preemption
When a scheduler interrupts a running job to switch to another job. This can happen in response to a variety of events: timers, new jobs, etc.

Quantum
A short unit of time that jobs can run before being preempted. Not all preemptive schedulers use a quantum, but some do.

Context Switch
When a scheduler switches from one job to another.
Scheduling Algorithms

First In, First Out (FIFO)
Schedule jobs in the order they arrive.

Shortest Job First (SJF)*
Schedule the shortest job first.

Shortest Time to Completion First (STCF)*
Schedule the job that will finish first.

Round Robin
Cycle through jobs executing each for one quantum.
Evaluating Schedulers

How can we measure a scheduler's performance? When is each measurement important?

Fairness: one long job shouldn't prevent others from making progress.

Efficiency: how much of the available CPU time is actually given to running tasks, versus overhead spent scheduling?

Turnaround Time: the time between when a job arrives, and when it is completed.

Response Time: the time between when a job arrives and when it is started.
Preemptive Scheduling

Most systems use preemptive schedulers. Why do you think this is the case?
A preemptive scheduler will have a lower response time than a batch scheduler. It isn't guaranteed, but lower turnaround time is likely.
Users expect computer to be responsive, even while long tasks run in the b.g.

How do you decide between a long or short scheduler quantum?
If the quantum is too short, we will waste a lot of time context switching.
If the quantum is too long, response time will be bad.
Scheduler Examples
Illustrating Schedules

Show schedules for FIFO, SJF, STCF, and RR (quantum = 2).

A arrives T=0, length=4; B arrives T=1, length=2; C arrives T=3, length=1
Scheduling Consequences
Advanced Scheduling *(if there's time)*

What kinds of workloads are particularly bad for these simple schedulers? Why?

**FIFO**

One very long task starts first, followed by lots of short tasks that will have to wait until the long task finishes.

**Round Robin**

Lots of short tasks that are slightly longer than the quantum would make turnaround time longer, and incur more context switches.

One long-running task will take a very long time to finish if it is interrupted by new short tasks.
Wrap Up
Reminders

Assignment
The listing directories assignment is due today

Lab
The memory allocator lab is due on Wednesday

Reading
Read the instructions for Wednesday's lab before class

Mentor Office Hours
Sunday 1–4pm
Tuesday 8–9pm
Wednesday 8–9pm