

Outline

•Last time

- Finish off processes/threads (chaps 4/5)
- Assignment #2 due

•This time

- CPU Scheduling (chapt 6)
- Assignment #3 out (due Thurs Oct 11)

•Next time

- Finish CPU Scheduling (chapt 6)
- Intro to synchronization (chapt 7)

•Midterm: Tues Oct 16 (closed books, closed notes)

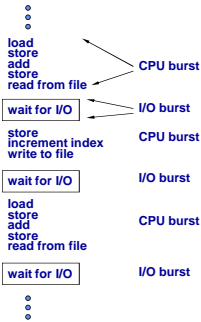
CS414: Operating Systems

Before we start

•Blake's office hours are on Wed for the next two weeks

CS414: Operating Systems

Motivation for Scheduling: CPU and I/O Bursts



- I/O-bound process: many short CPU bursts
- CPU-bound process: very long CPU bursts

- What should happen to the CPU when a process performs I/O?
- What should happen if a process never/rarely performs I/O?

CS414: Operating Systems

Scheduling

•Multiprogramming:

- running more than one process at a time enables the OS to increase system utilization and throughput by overlapping I/O and CPU activities

•[Policy vs. Mechanism]

- what to do vs. how to do it
- separation is crucial! WHY?

•[Process Execution State] remember from a few classes ago

- new, ready, running, waiting, terminated

•[Long-Term Scheduling]

- How does the OS determine the degree of multiprogramming (the number of jobs executing at once in primary memory)?
- Invoked very infrequently (seconds, minutes → may be slow)

•[Short-Term Scheduling]

- How does the OS select a process from the ready Q to execute?
- Invoked very frequently (milliseconds → must be fast)

CS414: Operating Systems

Context Switches: Voluntary vs. Involuntary

•Voluntary

- context switcher is invoked by the process that intends to relinquish the CPU (**nonpreemptive** scheduling); "yield" is for cooperative model
- I/O requests and termination can also invoke context switcher

•Involuntary

- Interrupt handler executes context switcher (preemptive scheduling)
- Interval timer is used

CS414: Operating Systems

Scheduling Criteria

•user-oriented, performance-related

- [Response Time] (can be referred to as "wait time") time from submission of job to start of execution of job
- [Turnaround Time] time from submission of job to completion of job
- [Deadlines] time at which computation must complete

•user-oriented, other

- [Predictability] job should run about the same amount of time regardless of load

•system-oriented, performance-related

- [Throughput] jobs completed per unit time
- [Processor Utilization] percentage of time that processor is busy

•system-oriented, other

- [Fairness] jobs should be treated the same
- [Balancing Resources] keep all resources busy

CS414: Operating Systems

Algorithm Evaluation

•[deterministic modeling]

- take a particular predefined workload and evaluation algorithm(s) (like gantt charts); gives exact numbers -- easy to compare; limited applicability (too specific); (this is what we'll focus on)

•[queuing models]

- use *distribution* of characteristics of arrivals (arrival times and execution times); math can sometimes be difficult; too many independent assumptions

•[simulation]

- programming a model of the computer system; random-number generators; can be expensive to develop

•[implementation]

- actual implement it and evaluate it in "real operation"; difficulties: cost, what if environment changes?

CS414: Operating Systems

Nonpreemptive Scheduling Policies

Important: Assume jobs are independent unless otherwise stated

- First-in First-Out (FIFO)**: execute jobs to completion in the order of their arrival

•**Example**: assume context switch time of 0 seconds; if 2 jobs arrive at the same time, then job with "lower number" arrived first

Job	Arrival	Duration	Response	Turnaround
1	0	50		
2	0	40		
3	10	30		
4	10	20		
5	20	10		

Advantage: simple

Disadvantages: short jobs may wait behind long jobs; may lead to poor overlap of I/O and CPU; not appropriate for timesharing situations

What does the Gantt chart look like?

CS414: Operating Systems

"Random" as a Policy?

- Would you schedule the previous job set via a "Random" policy?

•Forget about deterministic modelling

•Pros: fast scheduling, good as a baseline (?)

•Cons: not predictable, often mismatched with metrics (maybe?)

CS414: Operating Systems

Shortest Job Next (SJN or SJF)

- Schedule the job that has the least (expected) amount of work (CPU) left to do

- Provably optimal with respect to minimizing the average waiting time

•**Problem**: impossible to predict the amount of time left a job has (although we can use past performance to predict the future)

- Preemptive SJN is sometimes called SRTN (shortest remaining time next)

- Problem**: gets jobs out of the system, but may starve jobs

•**Example**: Assume context-switch time of 0

Job	Arrival	Duration	Turnaround (FIFO)	Turnaround (SJN)
1	0	50		
2	0	40		
3	10	30		
4	10	20		
5	20	10		

CS414: Operating Systems

Nonpreemptive Deadline Scheduling

- In *hard real-time systems*, jobs must complete execution before their deadlines; failure to do so can result in the potential loss of human life

- Example** hard real-time systems:

•**Example**: Assume context-switch time of 0

Job	Arrival	Duration	Deadline	Start Time
1	0	350	575	
2	0	125	550	
3	0	475	1050	
4	0	250	(none)	
5	0	75	200	

CS414: Operating Systems