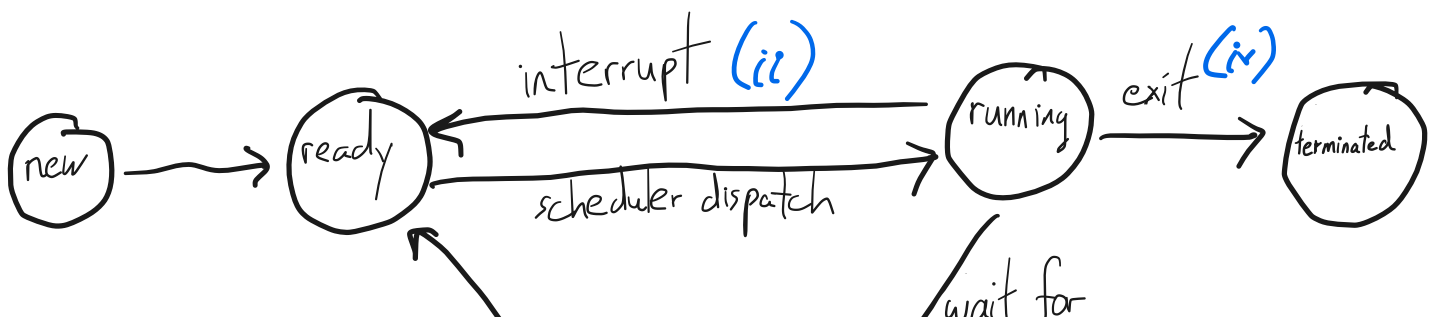


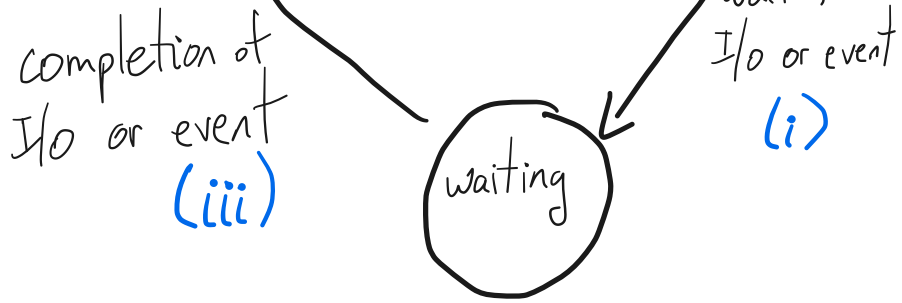
- 1. Last time
 - 2. Scheduling intro
 - 3. Scheduling disciplines
 - FIFO
 - SJF
 - RR
 - Incorporating I/O
 - Priority
 - MLFQ
 - Lottery/Stride
 - 4. Lessons and conclusions
-

2. Scheduling intro

High-level problem: OS has to decide which process (or thread) to run.

A. When scheduling decisions happen
process state/transitions:





scheduling decisions happen when a process/thread:

- (i) Switches from running to waiting
- (ii) Switches from running to ready
- (iii) Switches from waiting to ready
- (iv) Exits

preemptive scheduling vs. non-preemptive scheduling

B. Metrics and criteria

turnaround time: time for each process/thread to ^{complete}

waiting / response / output time: time spent waiting for something to happen

system throughput: # completed processes / unit of time

fairness: often conflicts w/ efficiency

C. Context switching has a cost
 CPU time in kernel (save and restore registers, switch address spaces)
 indirect costs (TLB shutdowns, processor cache, OS caches)

3. Scheduling disciplines

Assume first that process/threads do no I/O. (unrealistic; relax it later)

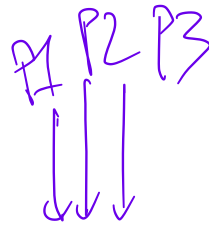
A. FCFS/FIFO

- run job until done

ex: P1 needs 24s

P2 needs 3s

P3 needs 3s



$$\frac{3 \text{ processes}}{(24 + 3 + 3)} = \frac{3 \text{ processes}}{30s} = 0.1 \text{ proc/sec.}$$

total?

avg tt?

0.1 processes/sec.

$$\frac{24 (P1) + 27 (P2) + 30 (P3)}{3} = 27s.$$

$$P3 = 3 \text{ sec}, P2 = 6 \text{ sec}, P1 = 30s$$

$$39/3 = 13$$

B. SJF + STCF

SJF: choose job w/ shortest upcoming CPU burst

STCF: preemptive version of SJF

example:

process	arrival time	burst time
P1	0	7 5
P2	2	4 2
P3	4	1
P4	5	4

time: 0 1 2 3 4 5 6 7 8 9 10 11 ...

P1 P1 P2 P2 P3 P2 P2 P4 P4 P4 P4 P1 ...

C. Round robin (RR)

- add a timer

- quantum

all processes the same length?

what if jobs are ...
 ex: 2 jobs of 50 time units each, quantum is 1.
 - avg tt? 99.5 quanta
 - if we did FCFS? 75 quanta.

D. Incorporating I/O
 motivating example:

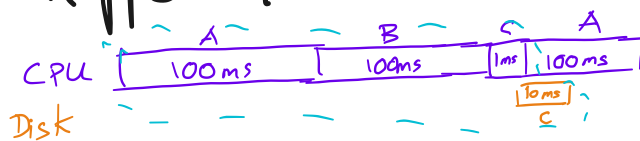
3 jobs

A, B: CPU-bound, run for a week

C: I/O-bound, loop: 1ms of CPU,
 10ms of disk I/O

what happens if we use FIFO?

what happens if we use RR w/ 100ms quantum?



disk utilization $\approx 5\%$

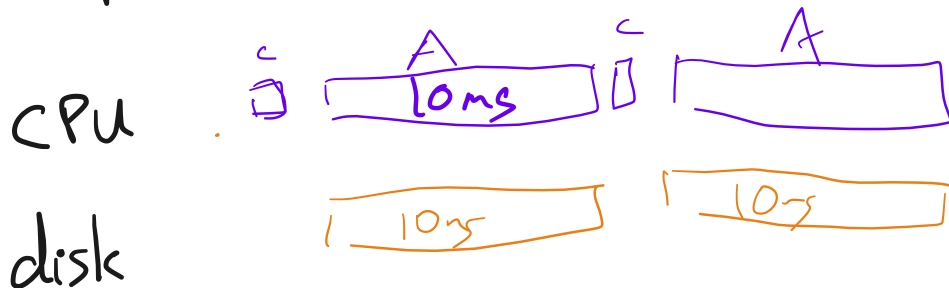
$$\frac{1}{C} \frac{10 \text{ ms}}{201 \text{ ms}} \approx \frac{10 \text{ ms}}{200 \text{ ms}} = 5\%$$

what happens if we use RR ≈ 1 ms quantum?



(exercise)

what happens if we use STCF?



disk utilization? $\approx 90\%$

context switches?

EWMA

t_n : length of process's n^{th} CPU burst
 τ_{n+1} : estimate for the $n+1^{\text{st}}$ burst

$$0 < \alpha \leq 1$$

$$\tau_{n+1} \leftarrow \alpha \cdot t_n + (1-\alpha) \tau_n$$

$$= \alpha t_n + (1-\alpha)\alpha t_{n-1} + (1-\alpha)^2 \alpha t_{n-2} + \dots + (1-\alpha)^n \cdot \alpha t_0$$

Each older term given exponentially less weight.

E. Priority scheme



F. MLFQ

Three ideas:

- multiple queues, with different priority
- RR w/in each queue
- feedback: change prio based on how much/
how little process has used the CPU.

G. lottery and stride scheduling

P_i gets t_i tickets

$$T = \sum t_i$$

prob. of "winning" next quantum is t_i/T .

H. Linux: CFS

~ stride scheduling

4. Lessons

- (i) Know your goals!
- (ii) Compare against optimal
- (iii) There are different schedulers that interact