CSc 360
Operating Systems
Processes

Jianping Pan

Summer 2015

P1 is already out!

- Due Monday, May 25, 5pm on connex
  - site remains open until midnight
  - but if it crashes after 5pm, we cannot fix it
  - so please submit a copy before 5pm
- Tutorial starting this Friday, ECS 104
- A realistic shell interpreter (RSI), able to
  - execute external programs
  - change directories (i.e., internal commands)
  - execute programs in background

Too challenging?

- We are here to help you
  - follow the suggested approach
    - discussed in the first lecture---it's effective!
  - attend lectures and tutorials: both!
  - get started earlier!
  - Connex forum (discussion group)
    - get help and help others
  - CSC consultant clinic/office
  - office hours (tutorial and lecture instructors)

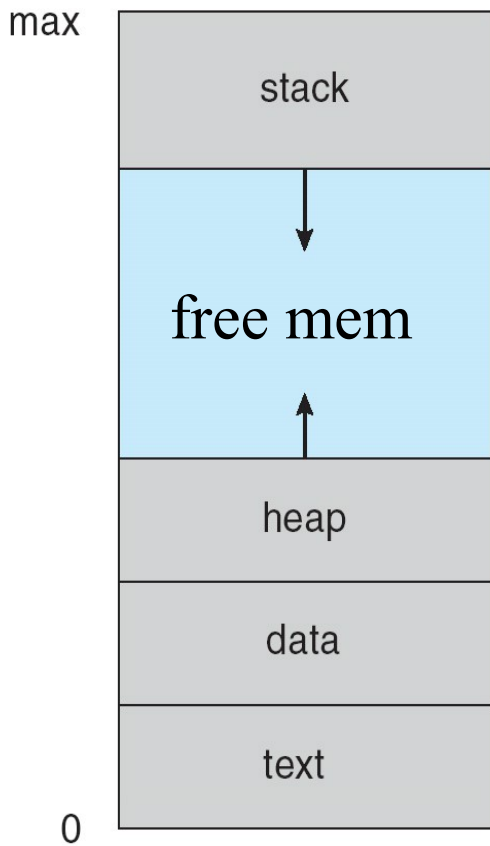
Interested in problem solving?

- UVic Programming Club
  - also selection for ACM ICPC competition
    - http://www.csc.uvic.ca/icpc
  - mailing list
    - http://groups.google.com/group/uvicicpc
  - recruitment for
    - undergrad and first-year grad: potential contestants
    - all students: student coaches
  - previous achievements
    - http://panlab.cs.uvic.ca/webb/viewtopic.php?t=3414
- Training sessions planned
  - first meeting May 20, 5:30pm in ECS660

Responsible use of computers

- Through this course, we will know better about operating systems, how they work, and tricks and tips
- You can practice these things and skills on your own computers
- Do not attempt to trick or compromise computers also used by others
- See UVic policies: IT Policy 6030

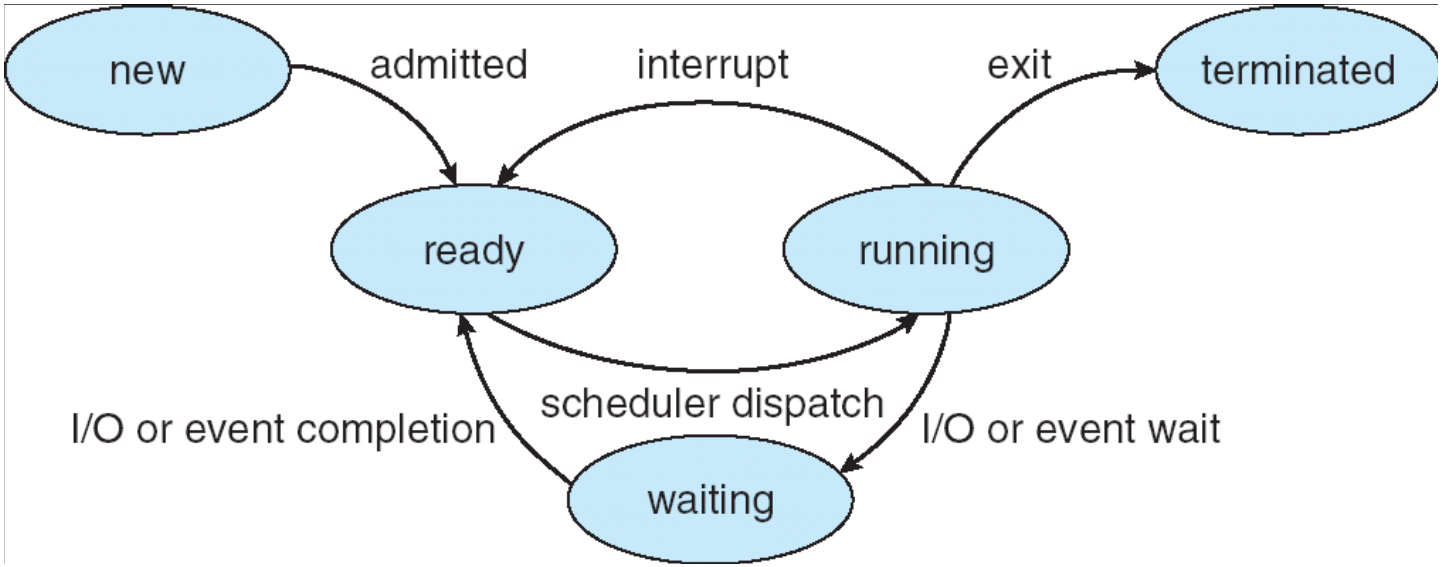
Processes



- Process: a program in execution
- Program: passive entity
  - static binary file on storage
    - e.g., gcc -o hello hello.c; ls -l hello
    - -rwxrwxr-x 1 *user group size date/time* hello
- Process: active entity; resource allocated!
    - ./hello
    - text (code); data (static), stack, heap
    - process control block

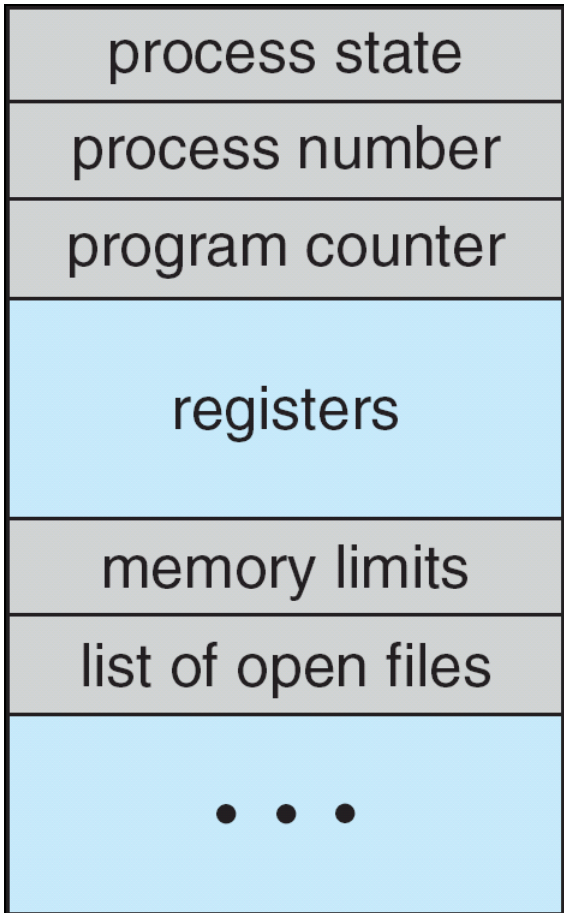
Process states

- E.g., one CPU (core)
  - one running process at any time
  - maybe many ready/waiting processes



Process control blocks

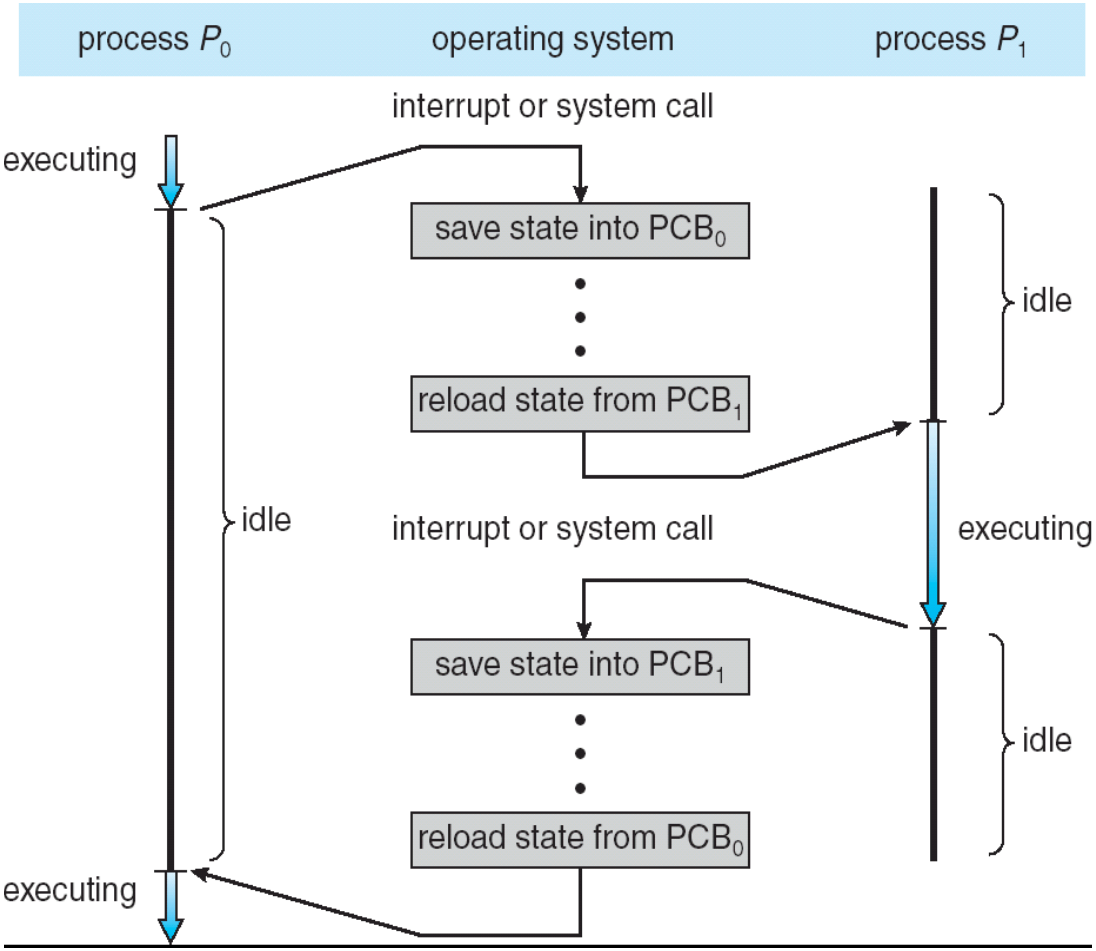
- PCB: keep track processes
  - state: ready/running, etc
  - CPU
    - PC, registers, priority, etc
  - memory
    - memory control information
  - I/O
    - e.g., list of opened files
  - accounting



"PCB exhausted!"

Context switching

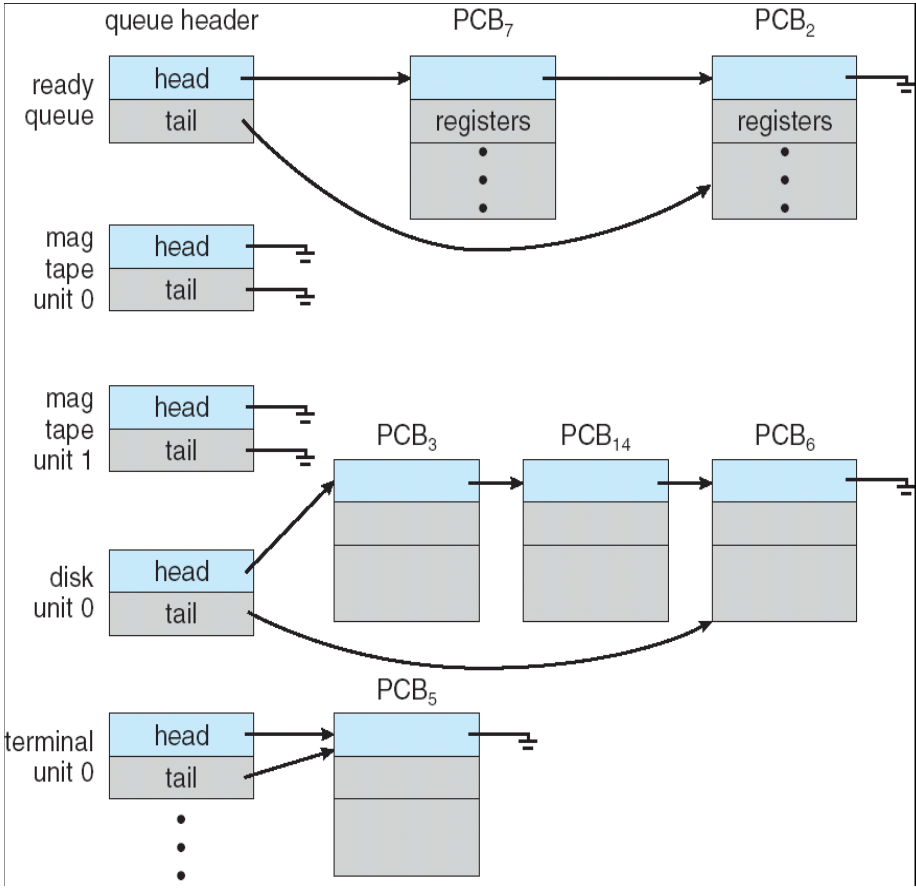
- Context switch
  - save states
  - restore states
- When
  - timer
  - I/O, memory
  - trap
  - system call



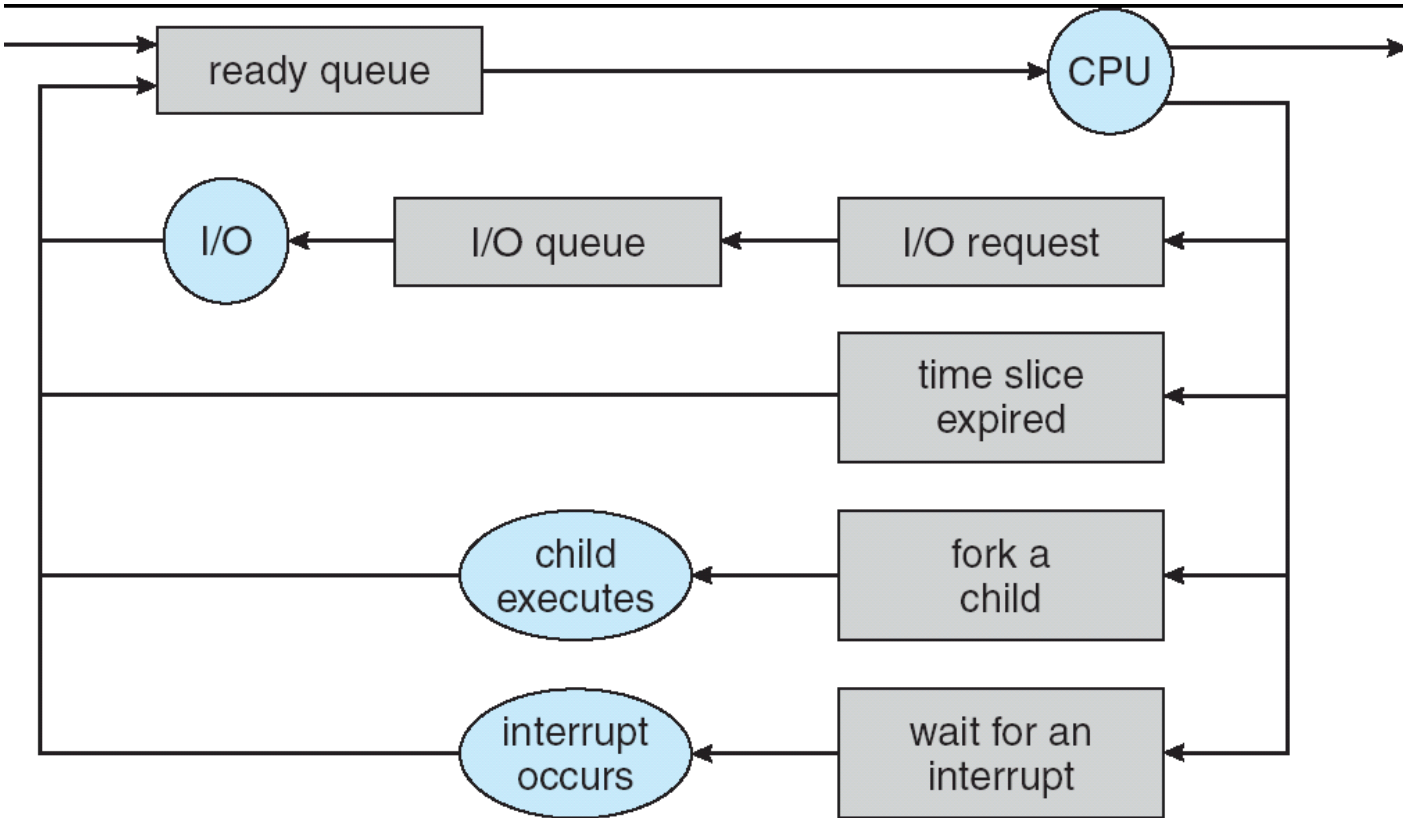
when more than two processes

Process scheduling

- Multiprogramming
  - utilization
- Timesharing
  - interactive
- Scheduling queues
  - linked list
  - ready queue
  - I/O queue



Queuing system



Queuing scheduler

- Who's the next?
- Long-term scheduler
  - job scheduler (spooling)
  - get to ready queue
  - CPU-intensive vs I/O intensive
- Short-term scheduler
  - CPU scheduler
  - frequency vs overhead

where's the long-term scheduler/gatekeeper?

More on scheduling

- Medium-term scheduler
  - who is NOT the next
    - reduce the degree of multiprogramming
  - swap-in/out
- Scheduling algorithms
  - first-come-first-server, shortest-job-first, priority, round-robin, fair and weighted fair, …
  - more in Chapter 5

we do a lot of scheduling in our daily life too!

This lecture

- Process and process scheduling
  - process vs program
  - process control block
    - context switch: what to save/restore
  - process scheduling
- Explore further
  - process status: /bin/ps
  - top CPU processes: /usr/bin/top

Next lecture

- Process operations and IPC
  - read OSC7 Chapter 3 (or OSC6 Chapter 4)