Agenda
1. MapReduce (discussion on how to solve word count problem using process & pipes)
2. Assign & discussion
VM of a Linux process

Different

identified for each process

Process-specific data structures

Physical memory

Kernel code and data

Stack

Memory mapped shared lib.

Runtime heap

data

text

Kernel VM

page tables

Process VM

Permits sharing of code and data while preserving memory isolation between processes.
Multilevel Page Table

1. Single page tables can be very large.

2. Do not have to create page table entries for VM pages that VM address space
   can be brought into memory as needed.

3. Level 1 table in memory

Ref. Computer Systems A Prog. Perspective (CSAPP)
Unix Kernel & File System

- Assumes hardware supports user & kernel mode interrupt & exception mechanism.

Userland

User programs & Libs.

Sys call interface

file subsys.

buffer

block I/O

device driver

Process control

Sched.

Memory

Inter process comm

H/w Control

H/w.
Buffering / Caching

adv =
1. hides details of h/w such as alignment which makes program easier to develop & more portable.

2. reduce i/o ops.
   - delayed write
   - improves latency & throughput

disadv =:
1. program can never be sure that the write was propagated to the device
2. memory copy overhead
Think about how this restriction on what transitions are permissible in which states ensures kernel integrity.

Critical section disables interrupts c.s. enables interrupt
- Kernel d.c.s should always be in consistent state

Atomic

Diagram:
- Process state transitions
- Node 1: User running
- Node 2: Kernel running
- Node 3: Ready to run
- Node 4: Asleep
- Arrows indicate transitions: return, interrupt, schedule process, wakeup.
File system

File system layout

User file descriptor table

File table

inode

local
global
```c
fd1 = open("/etc/passwd", RDONLY)
fd2 = open("local", O_RDWR)
fd3 = open("/etc/passwd", O_WRONLY)
```

Think about implications of this design.