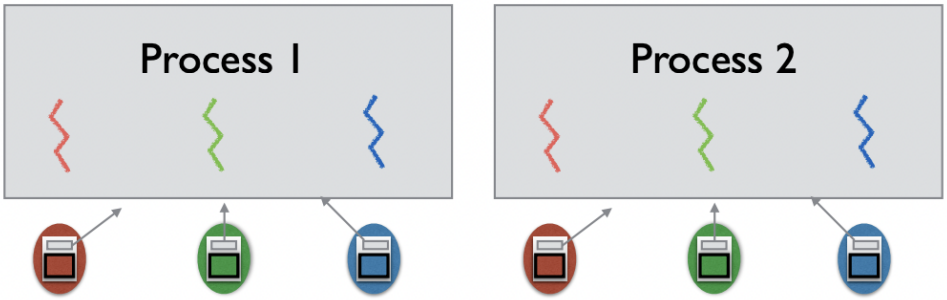
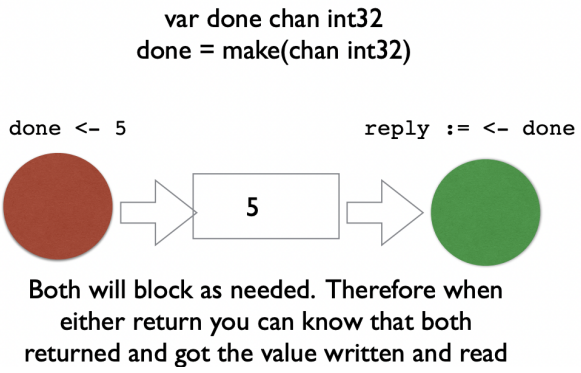
CAS CS 350
Lec 6

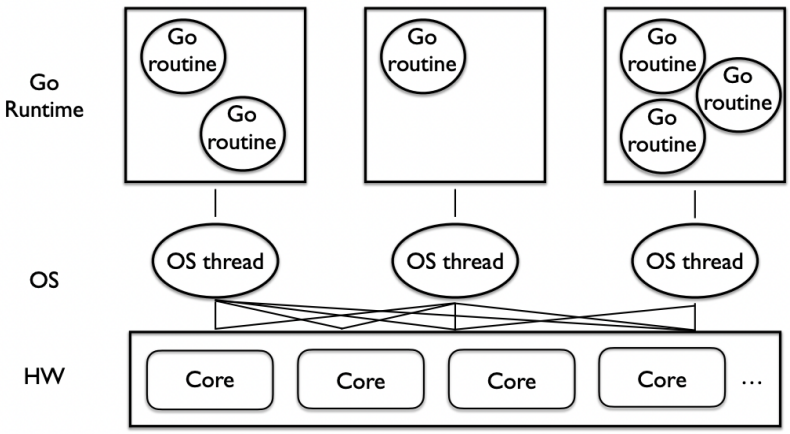
MapReduce (cont.) & Google File System (GFS)

1. Each process has its own virtual address space
   a. 

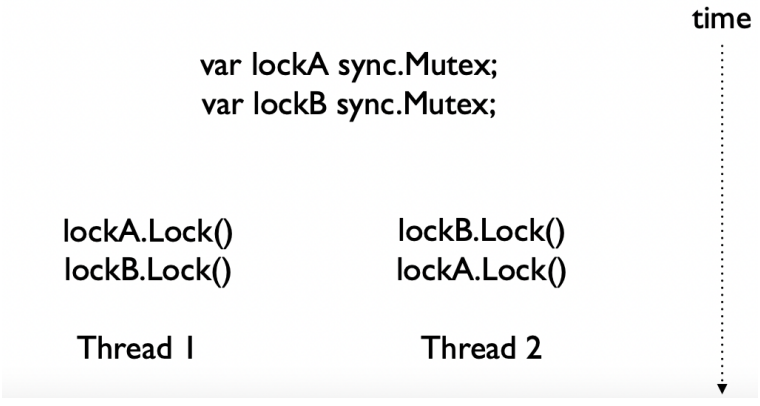
   b. Different threads → pass the pointer
2. Channels: Go Channels
   a. 

   b. In practice, you can think of channel as lock and buffer
   c. Channels are the medium through which goroutines can communicate with each other. Channel is a pipe that allows a goroutine to either put or read data
   d. Goroutine is a function that executes independently in a concurrent fashion. It is a lightweight thread that is managed by go runtime.

3. OS and HW threads vs Go routines
   a. Thread → can be regarded as series of tasks that can be executed concurrently with other instructions
   b. Goroutines are a lightweight abstraction over OS threads → single software thread can run many Goroutines (Goroutines are scheduled to execute by the Go runtime while HW threads are scheduled to operate by the kernel)
   c. A single OS thread can run many Goroutines
   d. Assigning Go routines to OS threads for execution is handled by the Go runtime
   e. Assigning OS threads to HW threads is done but the OS kernel

f. 

4. OS threads vs Go routines
    a. Scheduling OS threads is expensive because it requires context switching (operation system needs to move one operation system to another → when it does so, you need to save the thread, restore the thread you need to schedule → especially expensive when the state of the program is large (when using big memory)
    b. Creating and scheduling Goroutines is cheap → we can easily have 100k Goroutines in a program
5. Dead Lock
    a. Dead Lock → threads are waiting for each other → have to kill one of them
    b. In other words, two processes known as goroutines run concurrently (at the same time) and block each other's pathways
    c. Can also happen in channels since channels use locks

d. 

    e. Here, thread 1 locks lockA and thread2 locks lockB → don't want this to happen