CS 219 Lecture 30

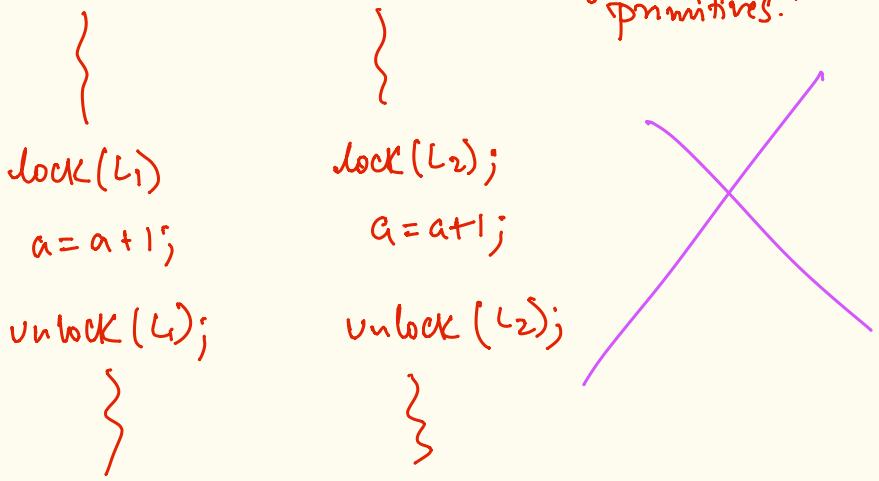
Lab 9 – pthreads (API & usage)

Lab 10 – futex w/ xv6 + semaphores in user space.

* last comments on synchronization.

① common accesses need common locks

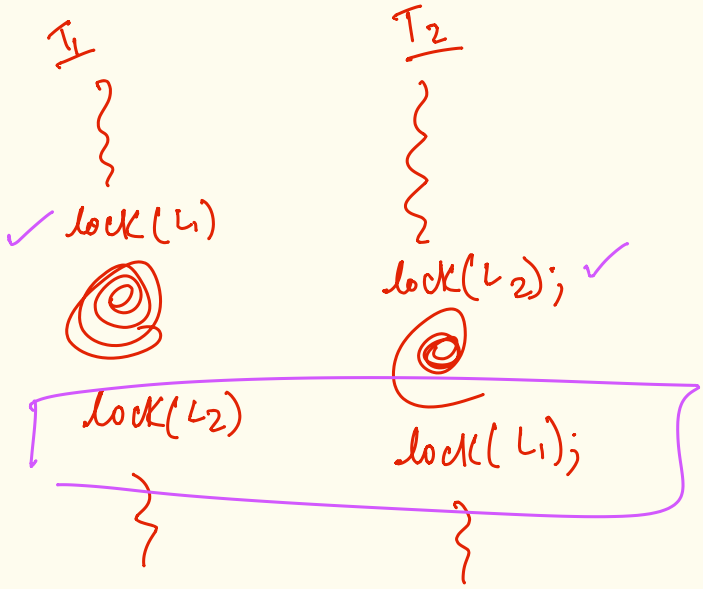
synch. primitives.



② deadlocks!

└ circular / inter-dependency of locks which can stall progress.

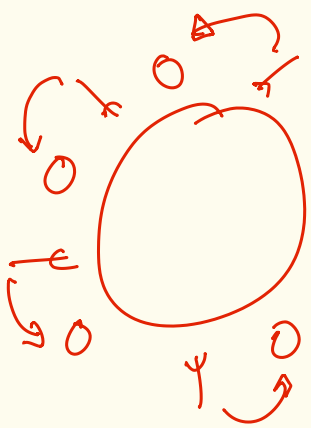
(i)



deadlock!

(ii) circular dependency

dining philosophers problem



* each philosophee does the following

- pick left spoon
- pick right spoon
- eat
- think
- repeat

(3) what is the critical section?

~ associate a lock w/ a PCB entry

~ associate a lock w/ the PCB array / list

~ associate lock with allocproc

~ or w/ findPCB in allocproc

~ or fork()

(*) the file abstraction

- persistent data / information / state
  - reuse all artefacts
  - across a power cycle:

~ file: is an end point/ abstraction of the OS for this purpose!

(*) requirements of abstraction to engage/ make useful a persistent store/ storage system

~ persistent

~ operations set: read, write, truncate, findsize, erase, ...

~ identification / naming ~ organization

~ access control ~ reliability ~ efficiency!

abstractions to work w/ persistent stores -

(i) file ~ sequence of linearly ordered bytes.

(ii) rdbms ~ tables.

(iii) Key-value store.

built on top of hardware!

HDD

SSD

Tapedrive