Threads and Synchronization

Yannis Smaragdakis, U. Athens
Threads

• A thread is an independent flow of control
  – i.e., an execution of a program
    • with its own instruction pointer
    • and stack, since that's determined by executed instructions

• Multiple threads may be in the same process
  – or in the kernel
  – they share everything else: heap, static area

• In a modern OS, threads (not processes) are the fundamental unit of scheduling/execution
User and Kernel-Level Threads
Synchronization

- Shared memory: parts of the address space are visible to two threads
- Can also be done with processes
  - just map the same physical pages in two address spaces
- Threads and processes are a continuum
  - shared memory can make processes be like threads
- Shared memory raises the need for synchronization
  - otherwise: race conditions
Mutex Locks

• Most common synchronization structure: mutual-exclusion lock (mutex)

• In two states: locked or unlocked

• Operations:
  – init: create, in unlocked state
  – lock/acquire:
    • if unlocked, atomically make it locked
    • else, no progress until it is unlocked and successfully acquired
  – unlock/release:
    • set mutex to unlocked

• See xv6 spinlock and sleeplock implementations
Condition Variables

- Used in combination with mutexes
  - *monitor-style programming*
- General-purpose waiting for long periods of time
- Operations:
  - init(): create, empty queue
  - wait(m):
    - *atomically* unlock mutex, put thread in “waiting” queue
    - when thread exits waiting queue, lock/acquire mutex
  - signal/notify:
    - remove one thread from waiting queue
- Typically in user space - kernel has modest needs
  - most general synchronization primitives, can implement all policies
Semaphores (Dijkstra ~1965)

- Like enhanced mutexes
  - mutexes = \textit{binary semaphores}
- Instead of locked/unlocked, a counter
- Operations:
  - \texttt{init(n)}: create, with counter n
  - \texttt{down/P}:
    - if counter positive, \textit{atomically} decrement
    - else, no progress until it is positive
  - \texttt{up/V}:
    - increment counter
- Variations where increment/decrement are by >1
Applications

• Lots of textbook synchronization problems
  - bounded buffer
  - readers/writers
  - dining philosophers