CS3103 Operating System

Content

- Ch1 Overview:
  - Basic concepts
- Ch2 Process:
  - Basic process concepts
    - Process State
    - Process Scheduling
    - Context Switch
    - Process Creation
    - Process Termination
    - Inter-process Communication
      - Message Passing Model & Shared Memory Model
      - Direct & Indirect
      - Synchronization: Blocking & Non-blocking
- Ch3 Thread:
  - Basic Thread concepts:
    - Comparison between thread and process
    - User thread & Kernel thread
      - One-to-one threading
      - Many-to-one threading
      - Many-to-many threading
      - Combined Model
    - Comparison between User thread and Kernel thread
    - Thread Cancellation
    - Thread Pools
    - Context Switch
- Ch4 Scheduling:
  - Criteria: CPU utilization, throughput, waiting time
    - Turnaround time: time takes from a request was submitted till it is finished
    - Response time: time takes from when a request was submittted until the start of execution
  - Policy: FCFS, SJF, RR
  - Estimation of next CPU burst
  - Multiple feedback queues
- Ch5: Synchronization
  - Background
    - race condition
    - critical section

        - critical section
            - Mutual Exclusion
            - Progress
            - Bounded Waiting
    - Mutex Lock
        - Peterson's Solution
    - Atomic Hardware Instructions
        - TestAndSet
        - Swap
    - Semaphore
        - Wake() operation
        - Signal() operation
    - Classical Synchronization problems
        - Producer-Consumer Problem
        - Readers and Writers Problem
        - Dining-Philosophers Problem
- Ch6: DeadLock
    - Conditions:
        - Mutual exclusion
        - Hold and wait
        - No resource preemption
        - Circular wait
    - Resource allocation graph
    - Deadlock prevention
        - Circular Wait: Careful design of lock acquisition
        - Hold-and-Wait: Try to grab all the locks at first
        - No Preemption: trylock() *Disadvantage: might cause livelock*
        - Mutual Exclusion: Using atomic instruction (No mutex at all)
    - Deadlock Avoidance (Always in the safe state, more risky)
        - Banker's Algorithm (Multiple instance per resource)
        - Resource allocation graph (single instance per resource)
    - Deadlock Detection
        - Single Instance: wait-for-graph
        - Multi-instance Resources: similar to Banker's algorithm
- Ch7: Memory Management
    - Basic Concepts:
        - MMU (Hardware device that maps virtual to physical address)
        - Swapping
        - Dynamic Storage Allocation

        - First-fit
        - Best-fit
        - Worst-fit
    - Fragmentation
        - External & Internal
    - Paging Techniques
- Ch8: Page Replacement
    - Performance:
        - EAT = (1-p) * memory_access_time + p * page_fault_service_time
    - Algorithm
        - FIFO
        - LRU
        - second-chance (clock) replacement
    - Thrashing
        - A process is busy swapping pages in and out, spending a lot of time in paging rather than executing (if a process does not have enough pages, then page-fault rate is high)
- Ch9: File
    - Basic concepts:
        - File attributes
        - File operations
        - Access models
        - Directory Structure
            - single level (hard to group, naming problems)
            - two level (difficult to share files among the users)
            - tree structured
        - Acyclic-graph
        - general graph
    - File system mounting
    - File sharing (remote file sharing)
    - Access control
        - Access mode: read, write, execute
        - classes of users: owner, group, others
        - e.g. chmod 664 (owner 110, group 110, others 100)
    - File System Layers
    - Allocation Methods
        - Contiguous Allocation
        - Linked Allocation
        - Index Allocation

    - Free space management
        - Bitmap free-space management
        - linked free space
        - grouping
        - counting
- Ch10: Storage
    - Basic concepts
    - Disk Scheduling
        - FCFS
        - SSTF (shortesst seek time first)
        - SCAN (elevator algorithm, starts from one end and moves towards the other end, like an elevator)
        - C-SCAN (moves to one end and handle requests, when reaches the end, immediately go back to the other end and handle request)
        - C-LOOK (similiar to C-SCAN, but does not move the the end of the disk, only goes as far as the last request in each direction)
    - Flash Translation Layer (FTL)
        - Firmware inside the flash device to make the NAND flash memory appear as a block device
        - includes: Address translator, Garbage collector, Wear leveler
        - Address Translator:
            - Page level, block level and hybrid
        - Garbage Collector
            - Because block is the minimum level for us to do the erase operation
        - Wear Leveling
            - Dynamic vs Static (note that Dynamic is more static while static is more dynamic)
            - Dynamic: wear leveling only for hot blocks
            - Static: proactively moving cold data to young blocks

Ch1: Overview of OS

- Interrupt
- Monolithic kernel: the entire OS runs in kernel mode (less overhead, more efficient)
- Microkernel: part of the OS runs in the user mode, and part runs in kernel mode (more secured)
- Application Program Interface (API) a high-level abstraction that encapsulates system calls, while a system call is a request directed at the operating system's kernel

Ch2: Process

inter-process communication (IPC)

- message passing and shared memory
- comparison (shared memory more time efficient, message passing could be used in a distributed environment, shared memory more complex to implement)
- direct communications and indirect communications (with mailboxes)

Ch3: Thread

- Key difference from process: Threads of the same process share the same memory space Note that thread does not share registers
- Comparison with process (thread more light-weight, thread less overhead, thread share memory space)
- Comparison between user thread and kernel thread (user thread implemented by user oriented to user do not worry about implementation, kernel thread has full functionalities, other kernel threads could still execute if one is blocked, kernel thread has larger context switch overhead)
- Context Switch: Before interrupt service routine, save the registers (program counter, stack pointer) to the stack, and use one stack per task

Ch4: Scheduling

Ch5: Synchronization

Ch6: DeadLock

Ch7: Memory Management

- logical and Physical Address space
    - logical: generated by the CPU
    - physical: address seen by the memory unit
    - same in compile-time, differ in execution-time
- Dynamic Storage-Allocation
    - First-fit: allocate the first hole
    - Best-fit: allocate the smallest hole that is big enough
    - worst-fit: allocate the largest hole
- Fragmentation
    - External: total memory space enough to satisfy a request but not contiguous

- Internal: allocated memory may be bigger than requested memory

Ch8: Page Replacement

algorithms: FIFO, LRU, Clock Replacement

Ch9: File system

Attributes: Name, Identifier, Type, Location, Size
Operations: create, open, read/write, seek, delete, truncate
Open files: file handler, file descriptor (file table, lock table, file cache, I/O buffer needed)
Directory Organization:

- Single level:
  - name problems and grouping problems
- Two level:
  - different users can have the same name for different files
  - efficient to search
  - cannot group files // solves the naming problem
- Tree Structured
  - Efficient in searching, can group files, convenient naming
  - Does not support sharing
- Acyclic-graph
  - Dangling pointer problem: solve the problem with backpointers/reference counter
- general graph
  - allow cycles, but use garbage collection
  - does not allow links that create cycles

File mounting: a file system must be mounted before it can be accessed
Access Control: read-write-execute
Remote File Sharing: network file system, common internet file system
File control block: inode number, and many other details
File System Layers:

- Devices -> I/O -> Basic file system -> file-organization module -> logical file system -> application programs

Contiguous allocation:

- simple, fast, random access
- External segamentation, increasing file size is difficult
  Linked allocation
- flexible in terms of file size, does not suffer external segmentation
- slow, no random access, pointer overhead
  index allocation:
- fast, random access, no external fragmentation
- large pointer overhead

Free space:

- linked: difficult to allocate contiguous free blocks
- grouping: uses indexes to group free blocks
- counting: starting block + # of contiguous blocks

Ch10: Storage

Positioning time = seek time(move to the cylinder) + rotation latency(rotate under the disk)
Disk scheduling:

- FCFS
- SSTF
- SCAN
- C-SCAN
- C-LOOK

Flash Translation Layer (FTL)

- Address Translator
  - page level, block level, hybrid
    - Hybrid: data blocks, and log/update blocks
- Garbage Collector
  - select victim
  - live-page copying
  - erase victim
- Wear Leveling
  - dynamic: wear leveling only for the hot blocks
  - static: wear leveling for all blocks by proactively moving cold data to young blocks
- Update Propagation: Though the log to determine what should be done finally