Module I

OS

Interface b/w hardware & software

Objective

* Convenience
* efficiency
* evolve

Operations & Functions

* Security
* Control over system performance
* Job accounting
* Error detection aids
* Coordination b/w other software and users
* Memory Management
* Processor management
* Device management
* File management

Some System calls:
fork(), exit(),
read(), write(),
getpid(),
close()

Services

* Handling I/O
* Program Execution
* Manipulation of file system
* Error detection & Handling
* Resource allocation
* Accounting
* Protection

System Call

Manipulation are controlled through

Interface b/w process and OS

- Process Control
- File management
- Communication
- Device management
- Information management

Boot process

BIOS
↓
Boot Leader
↓
Kernal
↓
OS Log

Types

Structure

Simple structure

* No well defined structure
* Small
* Not well separ.

Layered Structure

*OS broken into pieces.
* Each layer can use function of lower layer only.
*Layer 0- Hardware.
* Layer N- UI.

Micro kernal

* Removing non-essential components from kernal
* Results in smaller kernal

Modular structure

* best Structure
*Implementing non-essential programs as loadable
* kernal only contain essental Modular programs

Monolithic system

* Whole OS works under kernal.
* Large kernal size due to large no. of modules.

Batch Operating System

Jobs → OS → To batches → CPU

Time Sharing OS

Each task is given some time to operate, Multitasking system.

Multiprocessing System

* Symmetric
* Assymmetric

Real time OS

* Response time
* Used in strict environment.
* Timed system

Network OS

* Run on a server
* Shared Access is enabled

Distributed OS

* Independent CPU & Memory and processor Using Interconnection n/w

Job Queue

- Set of all process residing in memory

Ready Queue

- Set of all process waiting in main memory for read, write, execute.

Device Queue

- Waiting for I/O.

Schedulers

* Long term
* Short term
* Medium term.

Scheduling Algo.

* FCFS.
* SJF
* Priority → Each process have some priority
* Round Robin. Quantum time.

Scheduling Criteria

- CPU utilization: Max.
- Throughput: Max.
- Turn-around time: Min

- Bounder buffer
- Unbounded buffer

Shared Memory System

* Producer producing to buffer consumer consuming from buffer.

* Used in distributed system

* Synchronize working of two process without sharing the same addr. space.

Message Passing System

* Send (message) receive(message).
* Direct or indirect communication.
* Synchronize & Asynchronize
* Automatic & explicit buffering

Synchronization

* Blocking Send
* Non-blocking send.
* Blocking receive
* Non-blocking receive.

Module II

- Cooperating process
- Independent process

Interprocess Communication

Cooperating process (requires IPC)

* Information sharing
* Computation speedup.
* Modularity
* Convenience.

(Producer-Consumer problem)

* Shared Memory
* Message Passing

Process

* Process state: new, ready running, waiting & termination
* PCB: State, number, counter, Register
* Program in execution
* Context switch

Threads

* A path in process.
* Single & Multi-thread system

Application → Web Server, Docs, Games

Benefits

* Resource sharing
* Responsiveness
* Economy
* Scalability

Operations on Process

- Process creation
- Process termination

CPU Scheduler

* Preemptive
* Non-preemptive

Process Creation

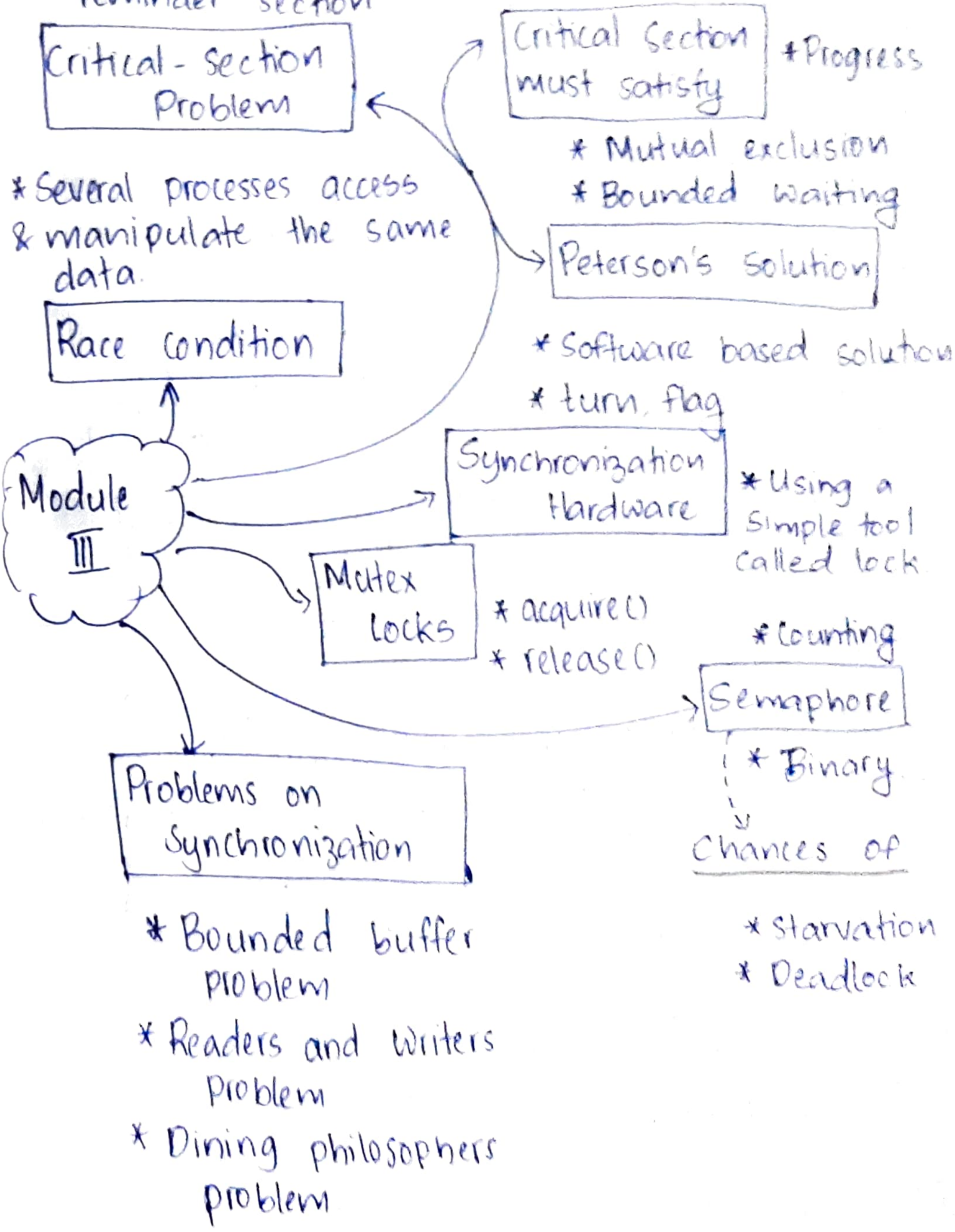
* In UNIX: fork().
* New process contain addr. of parent process for communication.

Associated system calls

* exec()
* exit() → All of it's child will die.
* wait()

Process Termination return an integer value

- critical section: shared data is accessing
- entry section, exit section, reminder section



Module III

Critical-Section Problem

Critical Section must satisfy

*Progress

* Mutual exclusion

* Bounded waiting

Peterson's Solution

* Software based solution

* turn, flag

* Several processes access & manipulate the same data.

Race condition

Synchronization Hardware

* Using a simple tool called lock

Mutex Locks

* acquire()

* release()

* Counting

Semaphore

* Binary

Chances of

* Starvation

* Deadlock

Problems on Synchronization

* Bounded buffer problem

* Readers and Writers Problem

* Dining philosophers problem

Logical address space: set of all logical addr. generated by a program

Module IV

Virtual Memory

* Secondary memory is treated as Main memory

Demand Paging

- Memory Miss
- Memory Hit

Page Replacement

* A page replacement scheme whenever there is a page fault.

* FCFS Page Replacement
* LRU Page Replacement
* Optimal Page Replace
* Second chance Algo.

Reference bit 0 : Remove the page
Reference bit 1 : give one more chance.

Dynamic Linking → shared libraries → Dynamic Loading

Memory

* Cache Memory
* RAM

- consists of large array of bits/bytes, each with its own address.

Address Binding

Binding is a mapping from one address space to another

Same logical & physical addr

- Compile time
- Load time
- Execution time

*Done by MMU

Logical address: addr. generated by CPU.
Physical address: addr. Seen by memory unit

Protection

- Base (low addr) and Limit (Range) Registers

Linking

- Dynamic
- Static

Dynamic loading

*Loading a program into the main-memory on demand

Swapping

* process swaped temp out of memory to a backing store and brought into memory for continued execution

Contiguous Memory Allocation

Fixed Size Partitioning

* Internal fragmentation

Variable sized Partitioning

* External fragmentation.

Hole (large size partiti

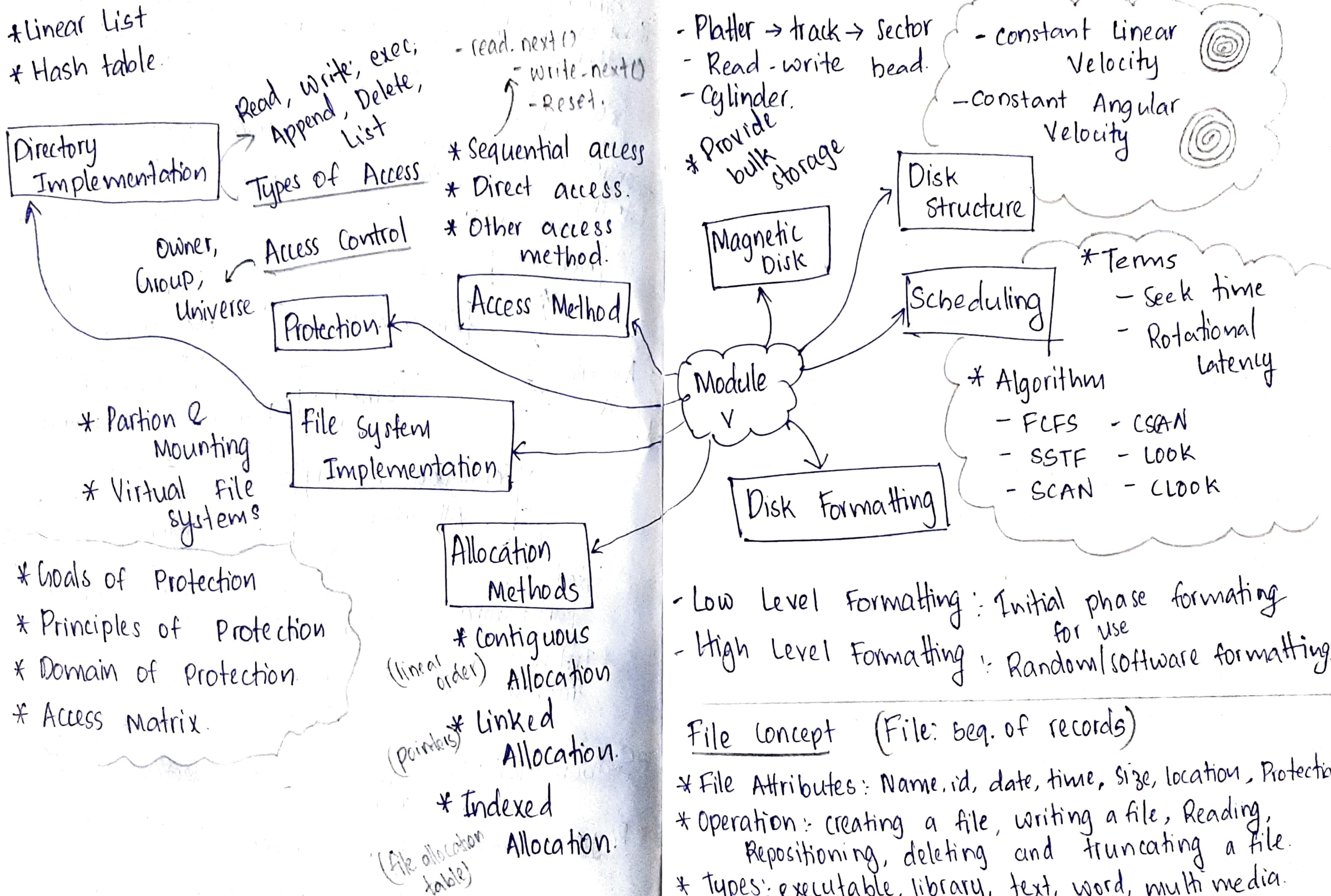
*Beft fit, Worst fit, first fit

* Segmentation : Views memory as a collection of variable sized segments.

* Paging : Pages & Frames method (+ Compaction. (re-shuffle))

Translation Look Ahead Buffer TLB

It's a fast cache memory



Module V

Directory Implementation

* Linear List
* Hash table

Types of Access

Read, write, exec, Append, Delete, List

Access Control

Owner, Group, Universe

Protection

Access Method

* Sequential access
  - read.next()
  - write.next()
  - Reset.
* Direct access.
* Other access method.

File System Implementation

* Partion & Mounting
* Virtual File systems

* Goals of Protection
* Principles of Protection
* Domain of Protection
* Access Matrix.

Allocation Methods

* Contiguous Allocation (linear order)
* Linked Allocation. (pointers)
* Indexed Allocation. (File allocation table)

Magnetic Disk

- Platter → track → Sector
- Read-write head.
- Cylinder.
* Provide bulk storage

Disk Structure

- constant Linear Velocity
- constant Angular Velocity

Scheduling

* Terms
  - Seek time
  - Rotational latency
* Algorithm
  - FCFS
  - SSTF
  - SCAN
  - CSCAN
  - LOOK
  - CLOOK

Disk Formatting

- Low Level Formatting: Initial phase formating for use
- High Level Formatting: Random/software formatting

File Concept (File: seq. of records)

* File Attributes: Name, id, date, time, size, location, Protectio
* Operation: creating a file, writing a file, Reading, Repositioning, deleting and truncating a file.
* Types: executable, library, text, word, multi media.
* File structure: Fixed-Variable length.