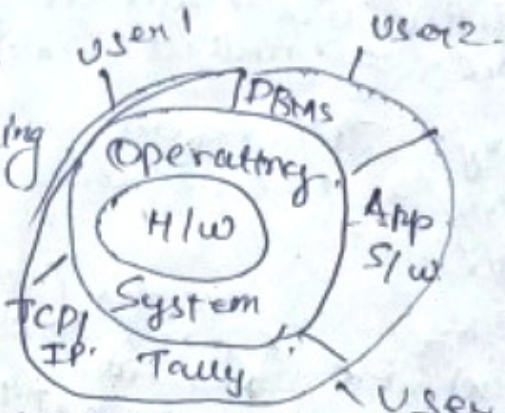
11.11.2024. O.S

This is a type of a system s/w.
It create interface between user and h/w.

OS-designed Approach.
layer based
Kernel based

layer based.
Bottom layer: H/w.
machine level programming is used for direct H/w access. It is hard and it is dependtent on h/w.




It acts as: resource manager and resource allocator.
2 types of resources:
(i) H/w resource (memory, processor, I/O device)
(ii) S/w resource.
system is divided into 3 modules
1. memory
2. processor
3. I/O module.

S/w resources are file and application s/w.
OS just work like government.

## OS Design Goals

(i) Primary Goal: User convinient / user friendly.
(ii) Secondary Goal: Efficiency: Throughput must be maximum (max. no. of jobs finished per unit time)
(iii) Turnaround time / latency (latency must be less)

$$TAT = CT - AT$$

BT = Burst Time

Q. Assume we have 4 process $P_1, P_2, P_3, P_4$.

| Process No. (Pno) | AT | BT (ms) |
|---|---|---|
| $P_1$ | 0 | 4 |
| $P_2$ | 1 | 3 |
| $P_3$ | 2 | 5 |
| $P_4$ | 3 | 6 |

Process is two types: CPU Bound Process, IO Bound process.
CPU Bound wants more time of CPU.

Process Vs Program:
Program (set of Inst).
Program: lazy
Program: pasive

Program under execution: process.
Process: Operating body
Process: Active and Dynamic

Process of Running (States of Process)
New → Ready → Running → wait / Block. → Terminate

Process will each time change its state.

Program will not need resources. Process will need Process.

CPU Bound will be more in Run state.
I/O Bound will be more in the wait Block

new state is born / creation of process.
Then process will go to the ready state.
Ready state is in memory.

New → Ready: Long Term Scheduler.
Ready → Running: Short Term scheduler

In Ready state; new process wait to run on CPU

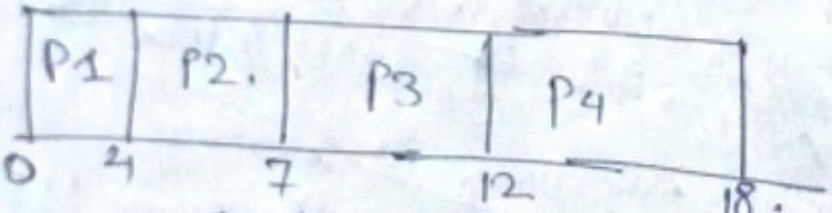
Short Term scheduler runs on the algorithm.

FCFS, Criteria: AT; Non Primitive.

| PnO | AT | BT | CT | CT-AT TAT | WT = TAT - BT WT |
|---|---|---|---|---|---|
| $P_1$ | 0 | 4 | 4 | 4 | 0 |
| $P_2$ | 1 | 3 | 7 | 6 | 3 |
| $P_3$ | 2 | 5 | 12 | 10 | 5 |
| $P_4$ | 3 | 6 | 18 | 15 | 9 |

*Non Primitive: One process from ready to run will not go again to ready que.

Ready
$P_4$ ~~B~~ ~~$P_3$~~ ~~$P_2$~~ $P_1$




WT must be less; Throughput is High, Avg TAT and RT (In-preemptive)

RT = First response - AT

Throughput: no. of jobs (process) completed per unit of time
Time
is called Throughput.

next class: Types of OS

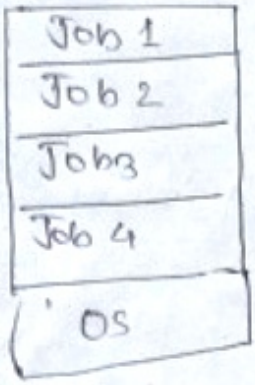
15.11.24. OS

Types of OS

1. Batch OS.
2. multi-programming OS.
3. multi-tasking OS. (Timesharing OS).
4. real time OS. (RTOS).
5. multi-processor OS.
6. Network OS.
7. Distributed OS.

## Batch-OS

similar type of jobs, batch was made.
language: JCL: job control language.
used during: 1st Before 1st Gen.
It is pure non-preemptive.
(no new job start until current job is finished)




Process

CPU Bound (B.T=Burst Time)

IO Bound (I.OBT: Input Output Burst time).

*Disadvantage was: most of time. CPU was idle.

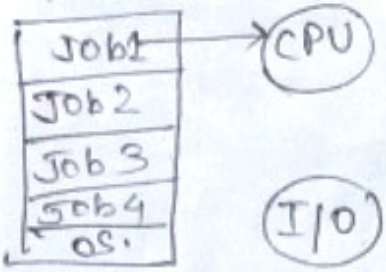
Better sol.

for maximum utilisation of CPU.

CPU, busy throughput is high.

batch: jobs of similar nature.

← Suppose Job 1 was selected for CPU execution. That time Job 1 went I/O operation. Job 1 goto I/O device; CPU become Idle.




* user cannot directly interact with system. The system has an operator. in 1st Gen / 2nd Gen computer. To operator, we gave task, the operator made batches and execution was batch by batch.

* user cannot set priority: It is pure non preemptive. Job 4 has priority high than Job 1. Until and unless Job 1 is not finished, Job 4 not start.

* user cannot directly interact with the machine.
  user cannot set priority.

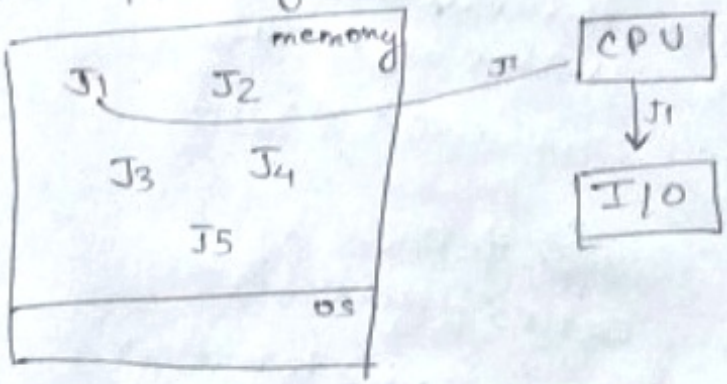
* Decreased Throughput of the system.
  - Throughput: no. of jobs completed per unit time.
  Throughput will decrease in batch system due to cpu is idle.

* if the job is completed completely then only another job will be scheduled onto the CPU.

* fcfs: (first come first serve) scheduling algorithm is used in batch operating system.

* IBM developed an OS - OS/2 is an example of batch operating system (outdated).

2nd Operating System was multi-programming OS:




J1 was scheduled to CPU. (from ready to run)
{J1, J2 ... J5} are in ready state in memory.
J1 required I/O devices; so J1 goes to I/O device
Then automatically acc. to algorithm other jobs {J2 .... J5} are scheduled to the CPU.
while J1 is doing I/O operation.
By this maximum utilisation of CPU.
Another job will be brought from ready to run.

OS If the job is leaving the CPU to perform I/O operation then another job which is ready for execution will be scheduled on the CPU.

CPU - Scheduling Algorithms.
In different or different Algorithms are used.
Main utilization of CPU is done in multiprogramming.
for Input/output: Job is in wait/Block, then it send to ready queue.
Increased throughput of the system.
eg.
windows, unix, linux are multiprogramming OS.
Major drawback (Batch OS): CPU was Idle, Throughput Low.

Process has different states:
① new state: process creation happens.
(Program: set of instructions) under execution is process. In new state, there is creation of process OR process was already created.
② ready state: process will wait to run on the CPU.
③ run state: process will go to CPU
OS has a program called scheduler.
Scheduler is of 3 types
LTS - long Term scheduler (new to ready)
STS - short term scheduler (ready to run).
MTS - mid term scheduler

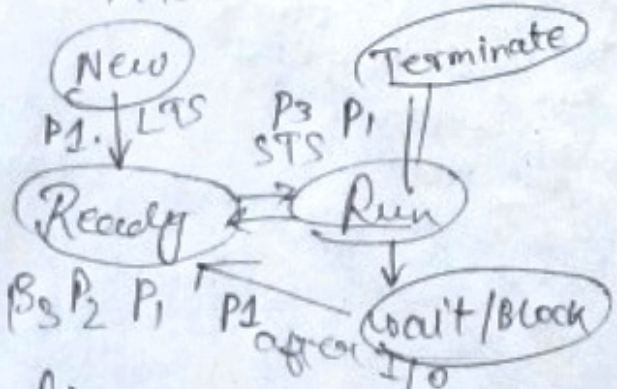


fig.

④ wait/block: process come from run state to this block due to I/O requirement. Then send to ready state after I/O completion.

④ Short-term scheduler will use algorithms to schedule the process to running state.

⑤ Terminate state: After Run state when process gets killed / terminated.
eg download 100% complete (Then download process is Terminated).
Normal Termination: Successful execution of program
close / Terminate: 80% downloaded; canceld.

fig. is described.

P1 go to CPU to run.
P1 need I/O operation; go to wait / block state.
P3 has more priority, P3 sent to CPU to run.
and P1 sent back to Ready state.