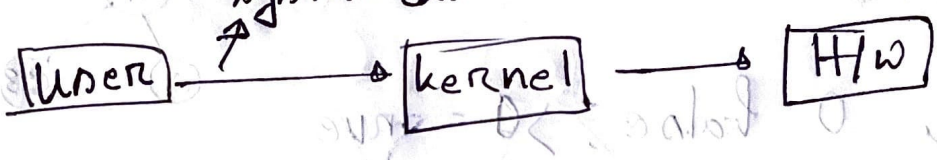
User system call এর মাধ্যমে kernel কে request করবে এবং kernel hardware manupulate করে।



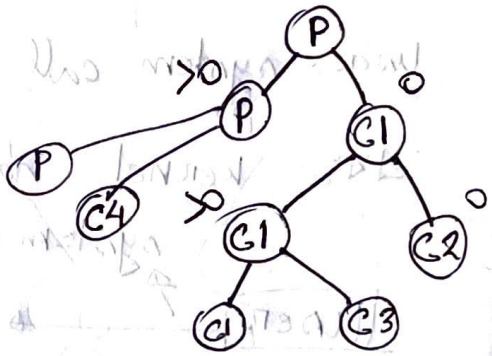
Bios stored in EEPROM.

kernel: OS এর এর মধ্যে program, device কীভাবে কাজ করবে, central module of operating system.

bootstrap program: initial program executed when running.

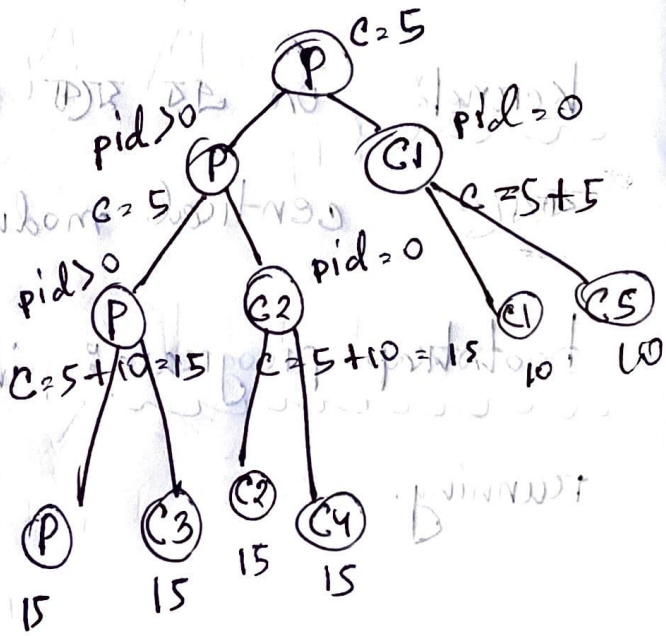
CSE321

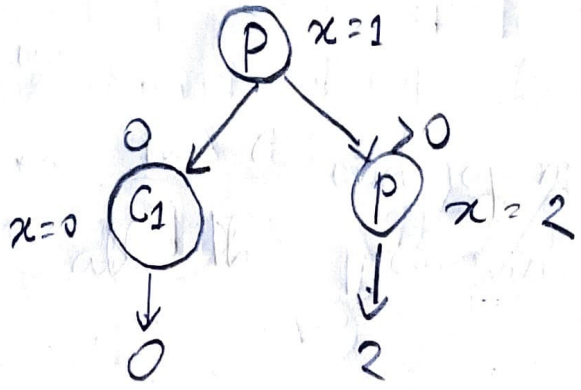
```
if (fork() || fork())
    fork()
```



OR → ||, 0 = false, >0 = true

#
```
c = 5
pid = fork()
if (pid == 0) {
    c += 5
}
else { pid = fork()
    c += 10
}
fork()
print(c)
```





Process

Process / user program / tasks

যে task processor এর কাছে complete হয়ে আছে সেগুলো process.

process is a program that in in execution

program counter : stores current activity of the process.

Stack : temporary data (local variable).

data : Global variable.

heap : dynamically allocated memory.

text : code

Process

এপিএস

same program এর অনেকগুলো process থাকতে পারে।

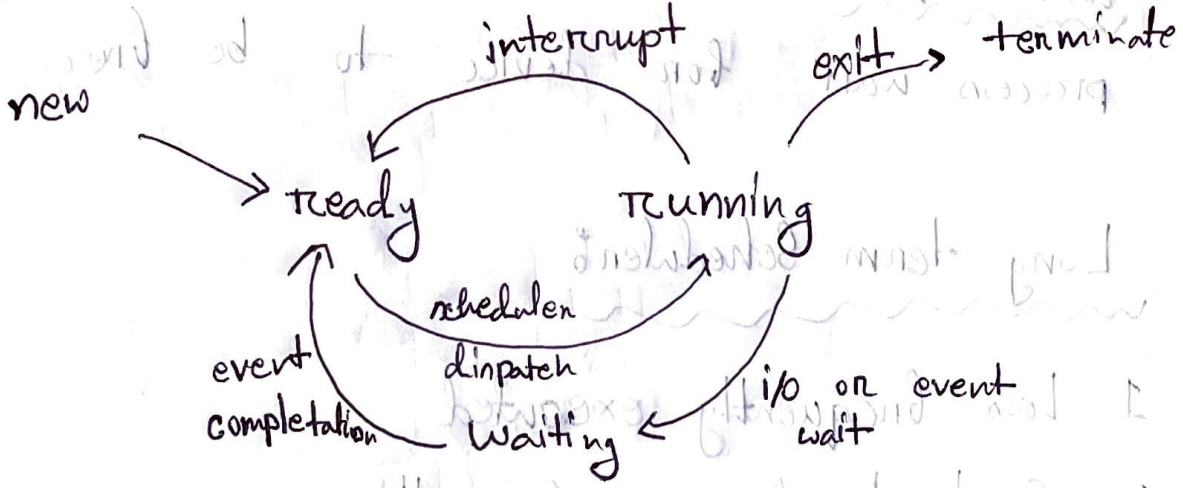
state of process ঃ memory পাওয়া অপেক্ষা
cpu busy থাকলে waiting state

HDD → RAM Memory → CPU

New | Running

Ready state ঃ Waiting to be assigned to a processor.

Process state diagram



Job queues

1. Resides in Secondary memory.
2. Keeps all the processes of the system

Ready queues

1. Resides in RAM / main memory.
2. Keeps all the processes that are waiting to be executed.

device queue:

process wait for device to be free.

Long term Schedulers

1. Less frequently executed
2. Controls degree of multiprogramming.

Short term Schedulers

1. More frequently executed.

(cpu idle ~~bad~~ বসে থাকলে overhead হয়)

Medium term Scheduler

1. time-sharing system may use this scheduler.
2. Swapping reduce the degree of multiprogramming.

partially executed
swapped-out process

Context switch: Storing currently executed process and restoring the next process.

forks

total output = 2^n [n = fork()]

∴ no. of child process = $2^n - 1$

Threads

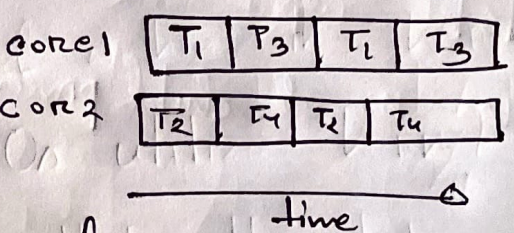
A thread contains - ① Thread ID ④ Stack
② Registers set
③ Program counter

↓
unit of cpu utilization

Threads share code section, data section and os resources belonging to the same process.

Parallelism implies a system can perform more than one task at a time simultaneously

Concurrency: supports more than one task implies a system can making progress



Advantages of multi-threading:

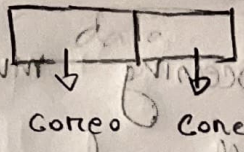
1. Responsiveness: One thread can provide rapid responsiveness while others are blocked or slowed down for intensive calculation.
2. Resource sharing: Threads share code, data, and other os resources which allows

multiple tasks to perform at same time simultaniously,

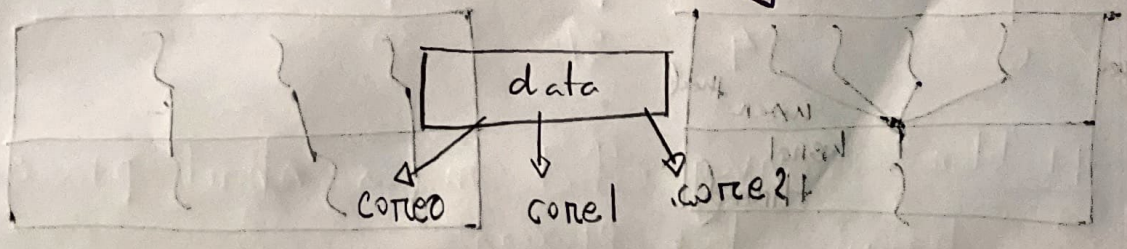
Economys Creating and managing thread are more faster than performing same task for processes

Scalability: It means program can take advantage of multiple processors if available. Unlike single threaded program it can use only one processor regardless of how many available. while multi-threaded program ~~share~~ spread work among ~~prog~~ processor making them faster and efficient.

Data parallelism: distribute subset of same data across cores.



Task parallelism distribute threads across cores, each thread performing unique operation.



Amdal's law: Identifies performance gain by adding additional cores to an application that has both parallel and serial component.

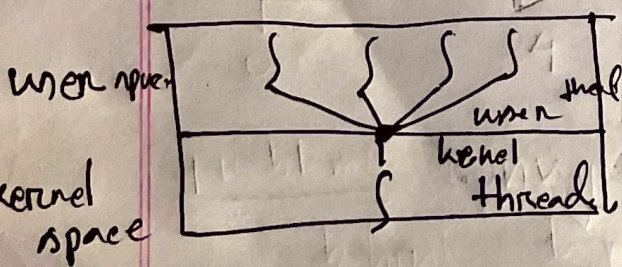
$$\text{speed up} \leq \frac{1}{S + \left(\frac{1-S}{N}\right)}$$

S = serial
N = no. of core

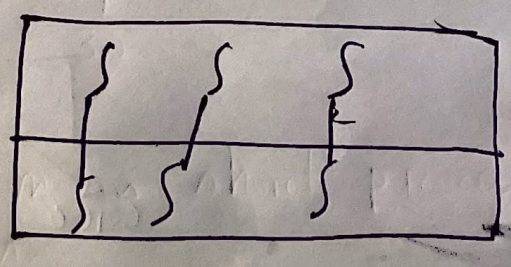
User threads supported above the kernel without kernel support. Programmer put this in their application

Kernel thread supported within the kernel of the os itself. Os utilize kernel thread by processing multiple task on system call at a time.

Many to ones



one to ones



Pthread:

1. Pthread stands for POSIX thread. They are a set of standard ~~that~~ for creating and managing thread in a program

2. Working: Global variable are shared among all the thread. So all thread within program can access and modify these variable.

3. Synchronization: One thread can wait for other thread to finish his execution before it can start its own.

4. Pthread begain executing in specified function. So function need to be defined, and it will start running.

Java thread:

1. Java thread are like pthread but for Java programing language.
2. Java thread are managed by JVM
3. They are created by ① extending thread class
   ② implementing runnable interface.

Cpu Scheduling

Criteria of scheduling:

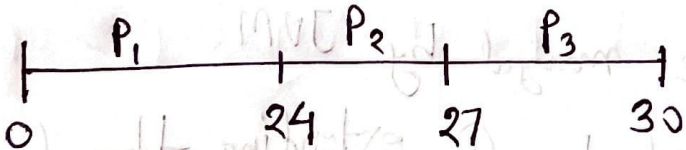
1. Turnaround time: amount of time to execute a particular process. (exit time - arrival time)
2. Waiting time: total time process wait in ready queue (turn around time - burst time).
3. Response time: total time of first response from cpu. (the first time process get cpu - arrival time)

SJF / SRTF / Round Robin

4. Cpu utilization 5. Throughput

FCFS

Process	burst time	completation time	T.A.T	W.T
P_1	24	24	24	0
P_2	3	27	27	24
P_3	3	30	30	27

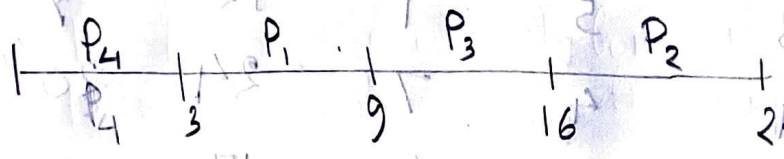


Problems: convoy effect means short process behind long process.

Shortest Job first (SJF)

Preemptive: If a new process comes with CPU burst less than remaining time of current of current process, preempt.

P	a	burst	Comp	t.a.t	w.t
P_1	0	6	9	9	3
P_2	0	8	24	24	16
P_3	0	7	16	16	9
P_4	0	3	3	3	0



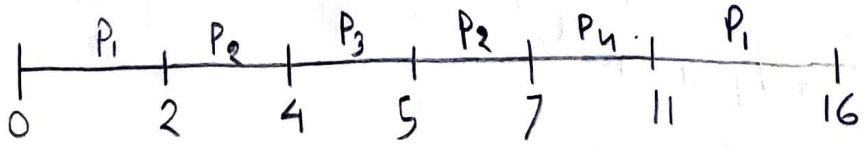
Avg w.t = $\frac{3+16+9}{4} = 7$

Avg t.a.t = $\frac{9+24+16+3}{4}$

= 13

SRJF

P	a	burst	completation	t.a.t	wait
P_1	0	~~7~~5	16	16	9
P_2	2	~~4~~2	~~7~~	5	1
P_3	4	1	5	1	0
P_4	5	4	11	6	2



Avg w.t = 3

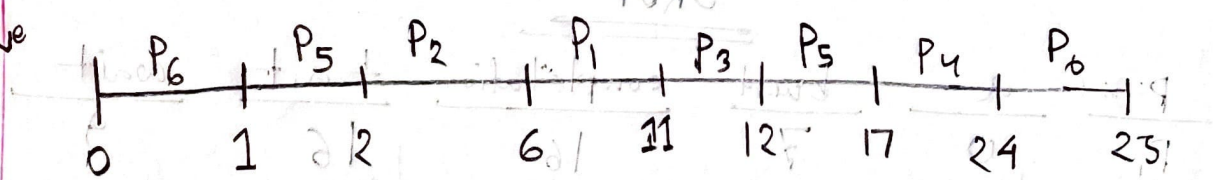
Avg t.at = 7

Priority Scheduling

problems Starvation - means low process may never execute

Solutions Aging is as time progresses increase the priority of process.

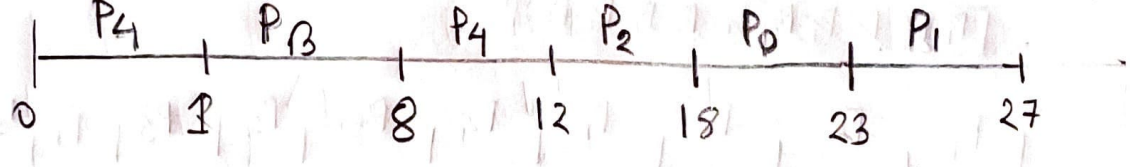
P	Priority	arrival	burst	comp	t.a.t	w.t
P_1	2	3	5	11	8	3
P_2	1	2	4	6	4	0
P_3	3	5	1	12	7	6
P_4	4	4	7	24	20	13
P_5	3	1	5	17	16	10
P_6	5	0	1	25	25	23



Avg wait = 9.167

higher number = high priority

P	a	Priority	b	comp	t.at	w.t
P_0	3	2	5	23	20	15
P_1	2	1	4	27	25	21
P_2	5	3	6	18	13	7
P_3	1	4	7	8	7	0
P_4	0	3	54	12	12	7



Round Robin

#Each process get small unit of CPU time which is time quantum. No process wait more than $(n-1)q$ time unit

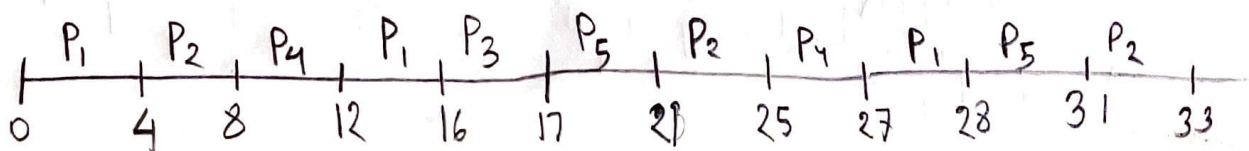
quantum time = 4

queue $P_1\ P_2\ P_4\ P_1\ P_3\ P_5\ P_2\ P_4\ P_1\ P_5\ P_2$

P	a	b	comp	t.at	w.t
P_1	0	5,	28	28	19
P_2	1	10, 6	33	32	22
P_3	5	1	17	12	11
P_4	3	6 2	27	24	18
P_5	8	7 3	31	23	16

Avg wt = 17.2

Avg. tat = 24

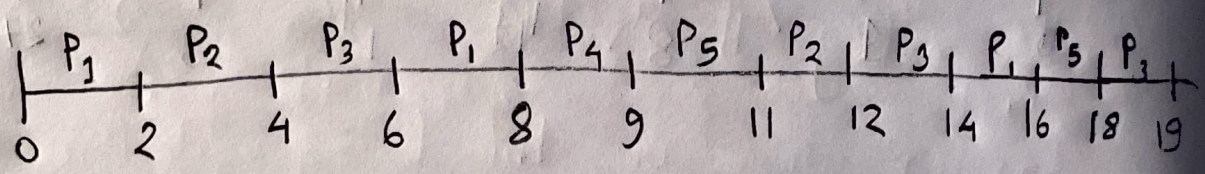


round robin

response	P	arrival	burst	comp	t.a.t	w.t
0	P_1	0	6 4 2	16	16	10
1	P_2	1	3 1	12	11	8
2	P_3	2	5 3 1	19	17	12
5	P_4	3	1	9	6	5
5	P_5	4	4 2	18	14	10

$q = 2$

$P_1\ P_2\ P_3\ P_1\ P_4\ P_5\ P_2\ P_3\ P_1\ P_5\ P_3$



$$\text{throughput} = \frac{5}{19} = 0.263$$

$$\frac{\text{total process}}{\text{total unit of time}}$$