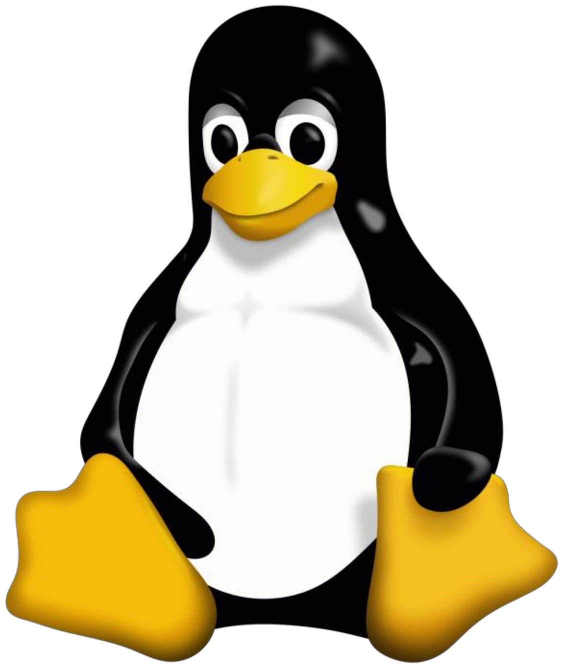
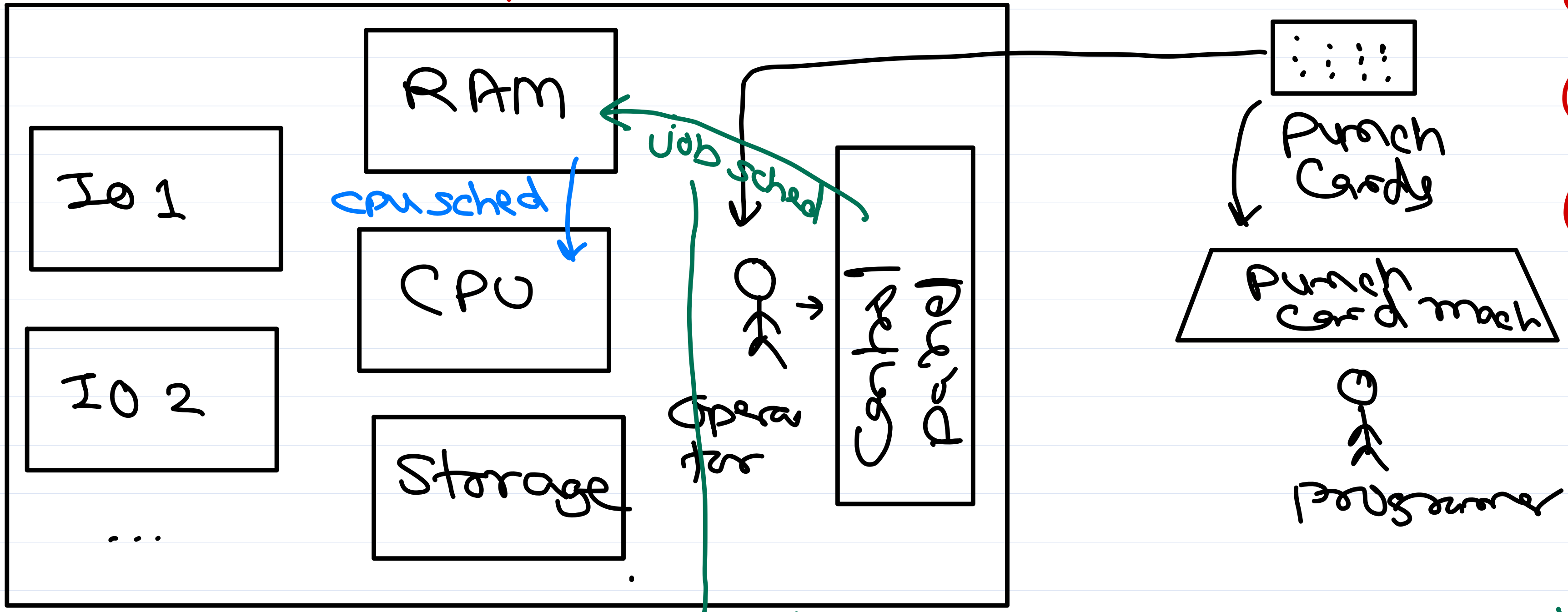


Operating System Concepts

Sunbeam Infotech

Mainframe systems

mainframe computer



1. Resident monitor
2. Batch system
3. multi-programming
   - Loading multiple programs in main memory
   - Aim: Better CPU utilization
   - Degree of multi-prog: num of prog that can be loaded in mem

→ decides programs to be loaded in RAM.
Combn of IO bound & CPU bound jobs is for better CPU & IO utilization.
Modern OS don't have job sched.

4. multi-tasking / time sharing.

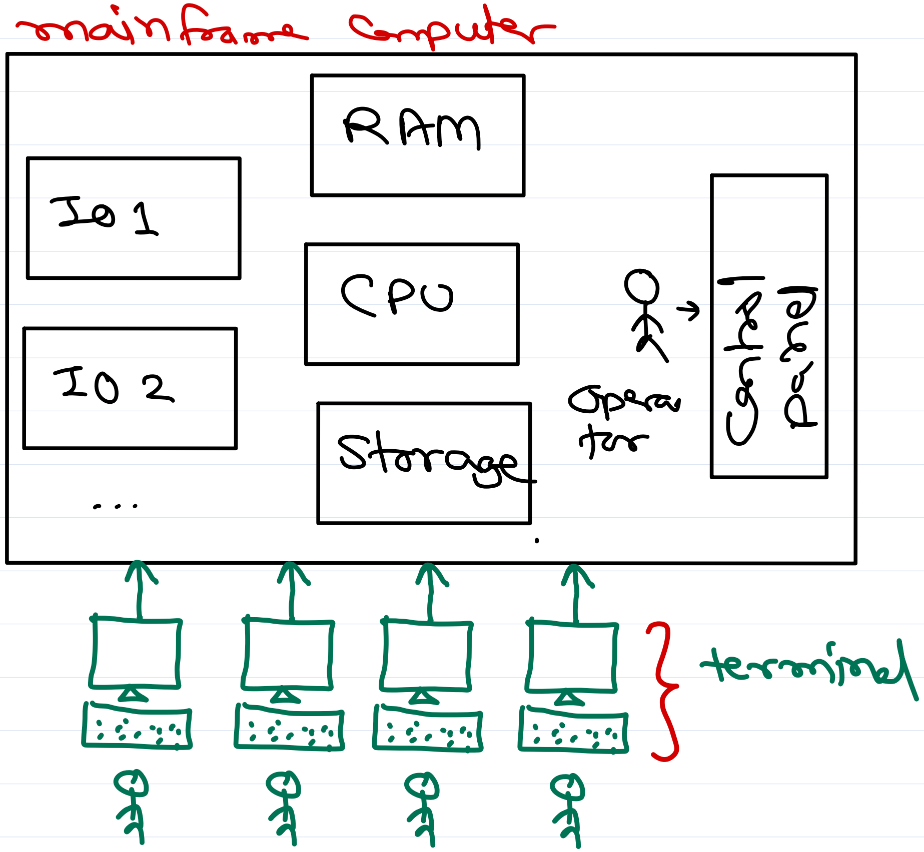
program exec time
→ CPU burst time
→ IO burst time

IO bound process: IO time > CPU time
CPU bound process: CPU time > IO time

Mainframe systems

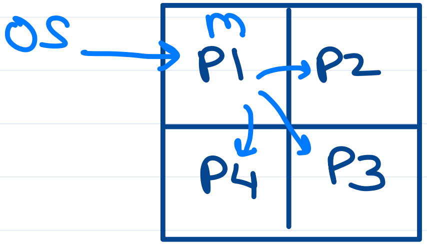
④ Multi-tasking (Time-sharing)
- sharing CPU time among multiple programs present in main memory & ready for execution.
  * Process based multi-tasking
    - share CPU time in independent processes.
  * Thread based multi-tasking
    - share CPU time in multiple threads of same/diff processes.
    - Threads are created to do multiple tasks concurrently within a single process.
  - a.k.a. multi-threading.

⑤ multi-user:
- multiple users execute multiple programs concurrently on same computer.

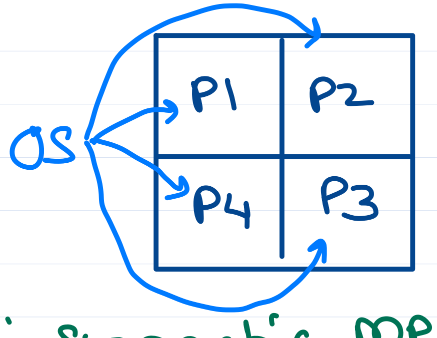


OS types

1. mainframe systems
   e.g. UNIX, IBM360, ...
2. desktop systems
   e.g. Linux, Mac, Windows
3. parallel Systems
   - Using multiple CPUs in same computer to perform tasks.

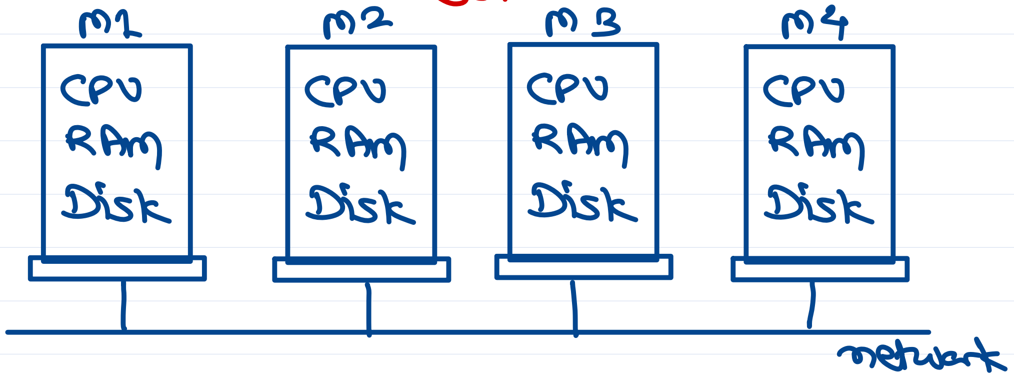


asymmetric MP.



symmetric MP.
e.g. Linux, Windows Vista+.

4. Distributed Systems



cluster: set of Computers connected in a network for a dedicated purpose

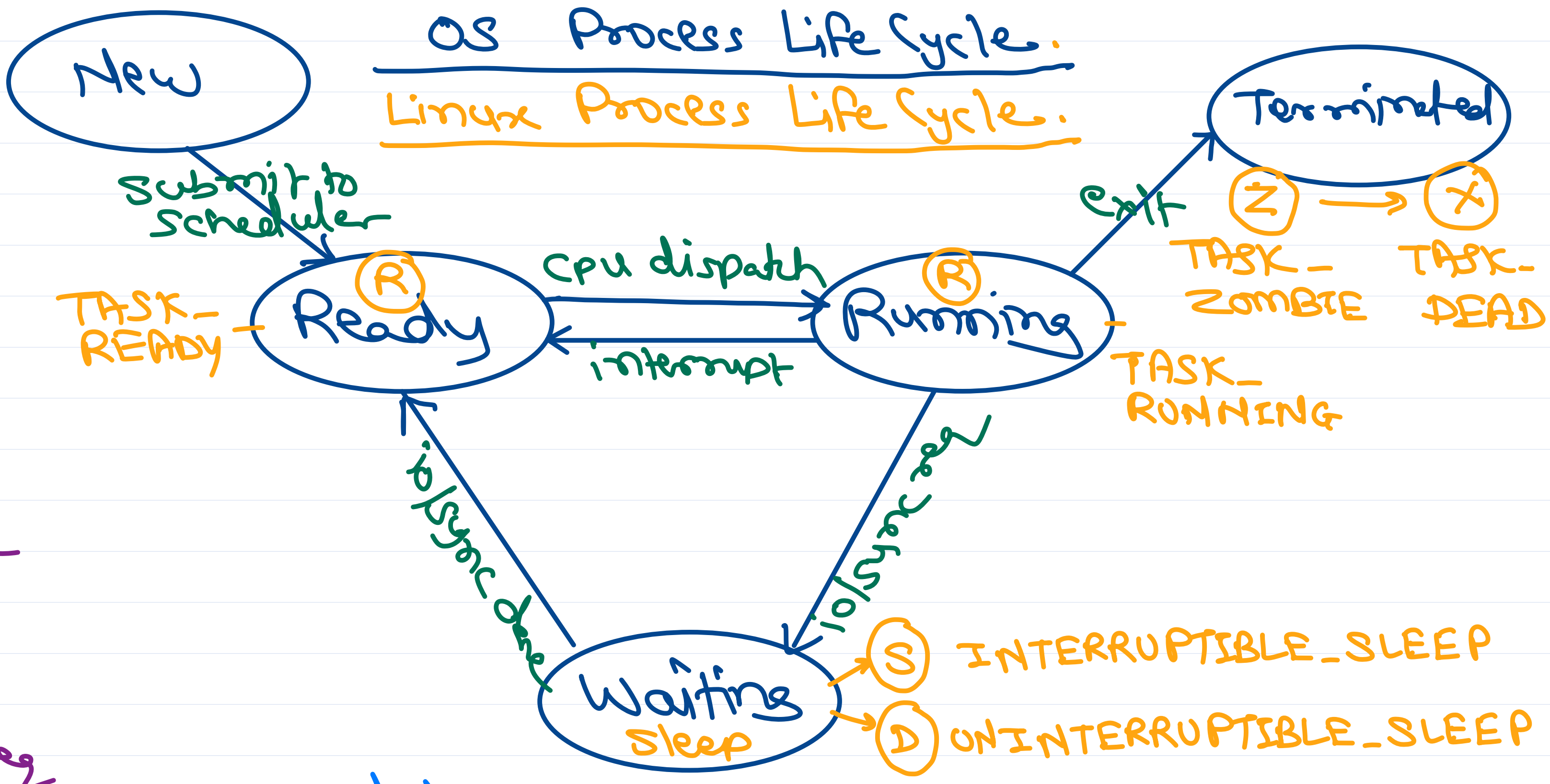
5. hand held systems
   e.g. Symbian, Android, iOS, ...
6. real time system
   accuracy of result not only depends on correctness of calculation but also depend on time in which result is produced.

   e.g. Win CE, FreeRTOS, µCOS, Xenomai, RTAI, µITron, pSos, VxWorks, ...

Process Life Cycle

OS data structures

1. job queue / process list
   - list of all processes (PCB)
2. ready queue / run queue
   - list of processes ready for execution on CPU.
   - CPU scheduler pick up process from ready queue and then dispatcher load it into CPU.
3. waiting queues
   - processes doing IO/sync eg are added into wait queue of respective IO devices / sync objs.



CPU scheduler called

1. Running → Terminated
2. Running → Waiting

} non-preemptive scheduling aka. cooperative scheduling

3. Running → Ready
4. Waiting → Ready

} pre emptive scheduling

CPU scheduling

Scheduling criterias

1. CPU utilization ↑
2. waiting time ↓
3. turn around time ↓
   min = CPU time + IO time
4. response time ↓
5. throughput ↑

Scheduling algorithms

1. FCFS
2. SJF
3. Priority
4. RR
5. Fair share

FCFS (non-preemptive)

Process	time	wait	TAT
P1	3	0	3
P2	3	3	6
P3	24	6	30

avg wait

$= \frac{0+3+6}{3}$

$= 3$

avg TAT

$= \frac{3+6+30}{3}$

$= 13$

P1	P2	P3
0 – 3	3 – 6	6 – 30

P1 P2 P3

Process	time	wait	TAT
P1	24	0	
P2	3	24	
P3	3	27	

avg wait

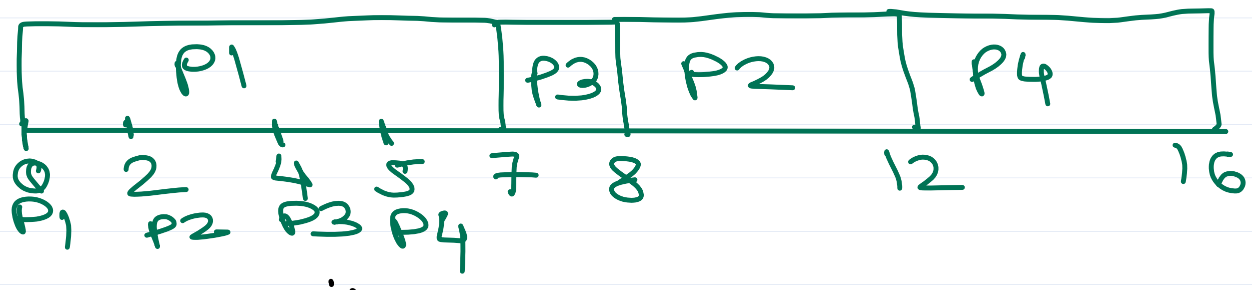
$= 17.$

Convoy effect

CPU Scheduling

SJF (non-preemptive)

process	arrival	time	wait
P1	0	7	0
P2	2	4	6
P3	4	1	3
P4	5	4	7

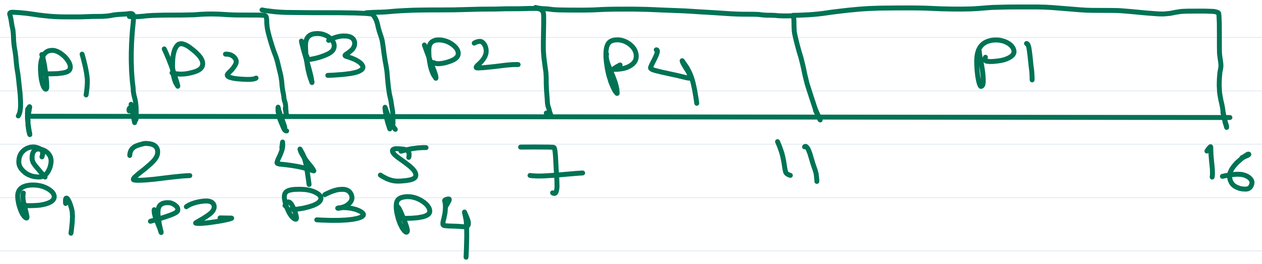


avg wait = 4 ms.

* gives min avg wait time.

SJF (preemptive) = SRTF

process	arrival	time	wait
P1	0	7 - 5x	9
P2	2	4 - 2x	1
P3	4	1 - 1x	0
P4	5	4 - 4x	2



avg wait = 3 ms

CPU scheduling

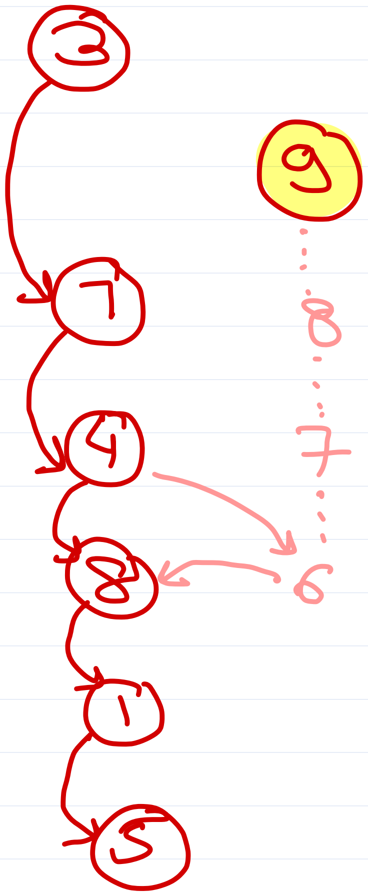
Priority Sched

- non-preemptive
- preemptive

each process is asociated with a number called as "priority".
Usually, lower number indicate higher priority.

due to high priority processes, a low priority process may not get enough CPU time for execution
⇒ Starvation.

increase priority of starved processes periodically so that it will get cpu time for execution sooner ⇒ aging.



RR (preemptive)

assign fixed time slice/ time quantum to each process repeatedly.

process	time	rem	wait	resp
P1	53	~~33~~ ~~13~~ x	81	0
P2	17	x	20	20
P3	68	~~48~~ ~~28~~ ~~8~~ x	94	37
P4	24	~~4~~ x	97	57

avg wait = 73

min resp time.

Time quantum = 20

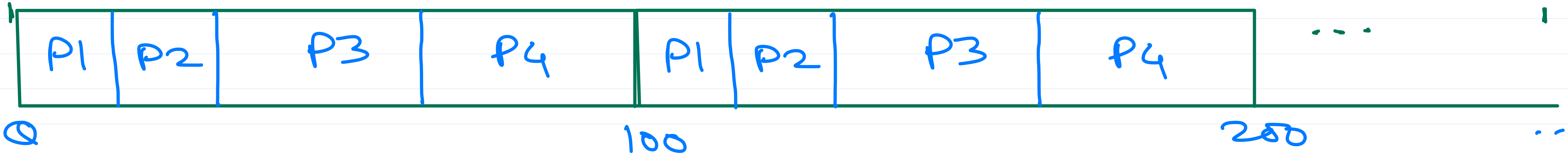
P1	P2	P3	P4	P1	P3	P4	P1	P3	P3

0 20 37 57 77 97 117 121 134 154 162

CPU scheduling

Fair share algo

epoch time = 100 ms



process	priority
P1	10 (L)
P2	10 (L)
P3	5 (H)
P4	5 (H)

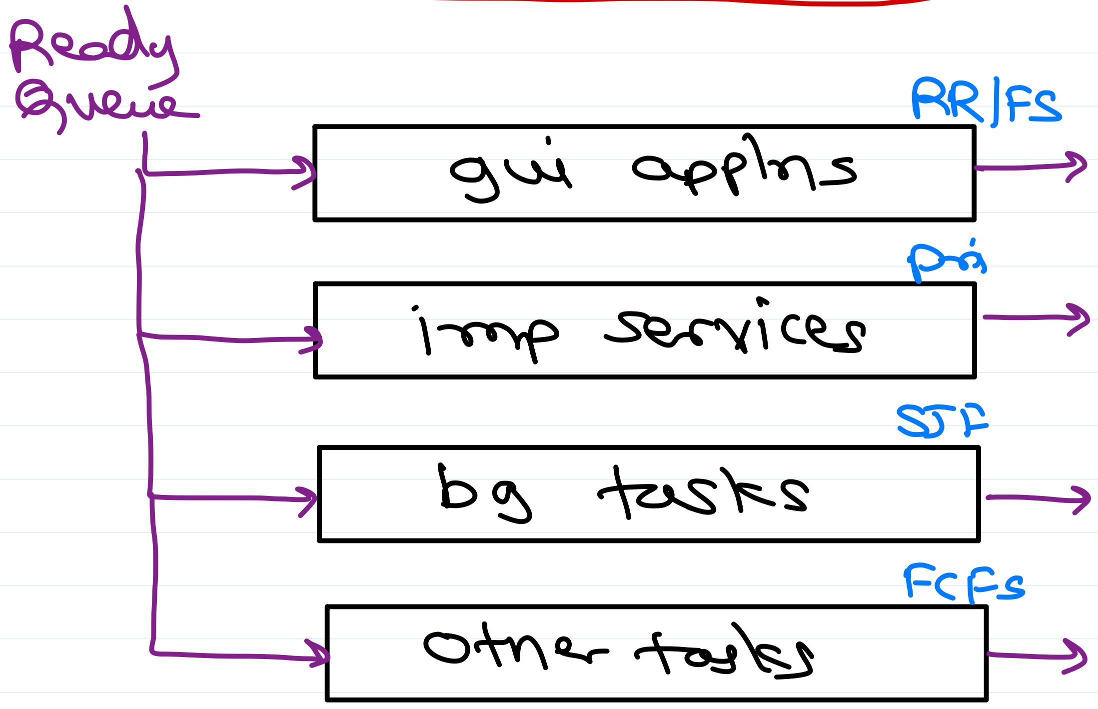
Called as "nice" value in Linux.

CPU time divided into epoch times and in each epoch cpu time share is given to each ready process based its priority.

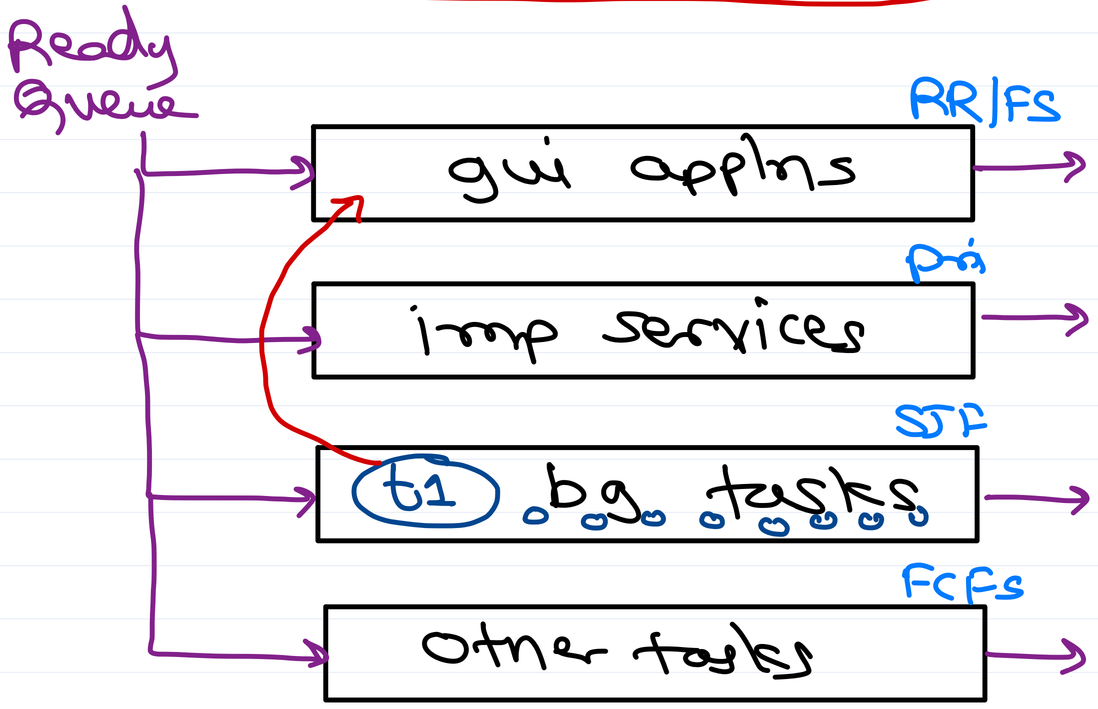
High priority process gets more CPU time w.r.t. low priority process.

CPU scheduling

multi-level sched queue



multi-level feedback queue



Linux sched classes

* Realtime sched
  - (A) SCHED_FIFO
  - (B) SCHED_RR
* Non-realtime sched
  - (C) SCHED_OTHER/NORMAL (fair share)
  - (D) SCHED_BATCH
  - (E) SCHED_IDLE

VI editor

- text editor in CLI
- developed by BSD UNIX
  - UCB - Bill Joy
- VI improved - vim

cmd> vim filepath

- VI editor modes
  - Command mode (default).
    - press esc.
  - insert/edit mode.
    - press "i"

Commands

1. save/write → :w
2. quit → :q
3. save & quit → :wq
4. copy cur line → yy
   copy n lines → n yy
   copy line x to line z → :x,zy
   copy after cursor → y$
   copy before cursor → y^
   copy cur word → yw
5. cut (same as copy) → y → d
6. paste → p
7. undo → u
8. redo → ctrl + R
9. run linux cmd → :! command

SUNBEAM

Thank you!

Nilesh Ghule <nilesh@sunbeaminfo.com>