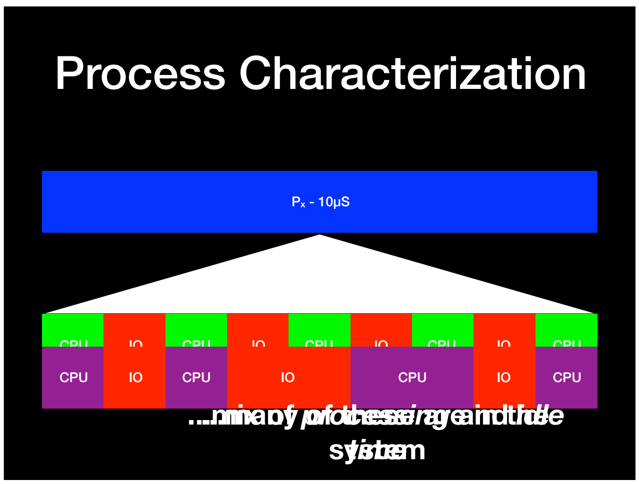
1

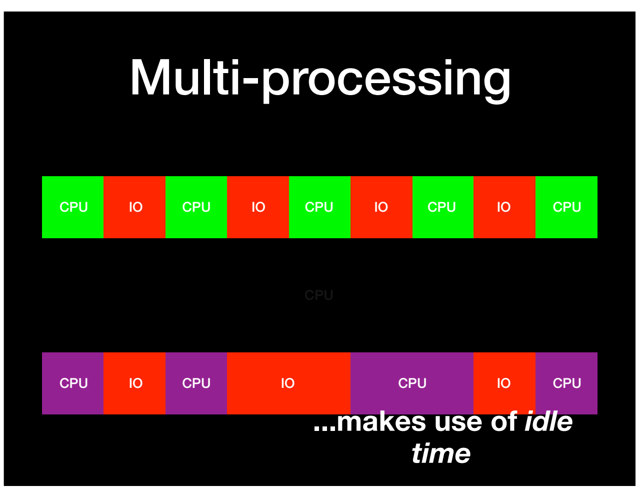
Scheduling
Process Management
COS 450 - Fall 2018

2

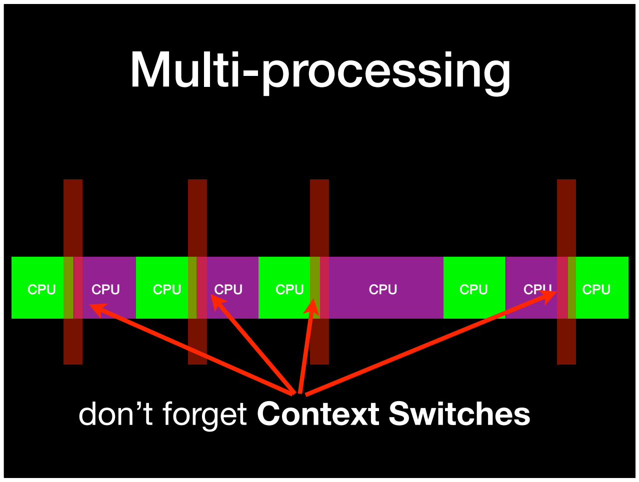


Process Characterization
Pₓ - 10μS
CPU
IO
CPU
IO
CPU
IO
CPU
IO
CPU

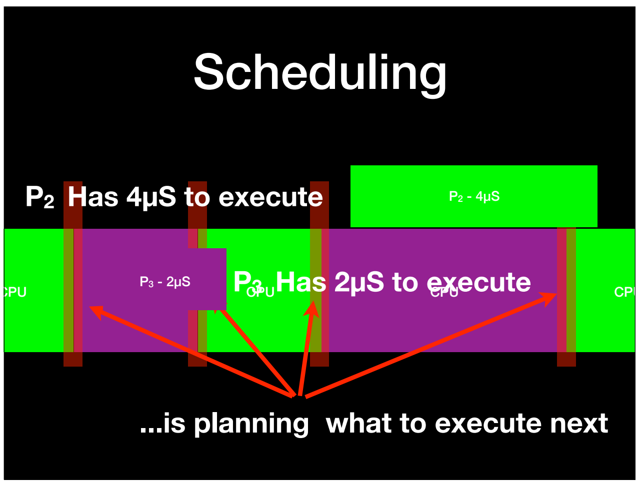
3



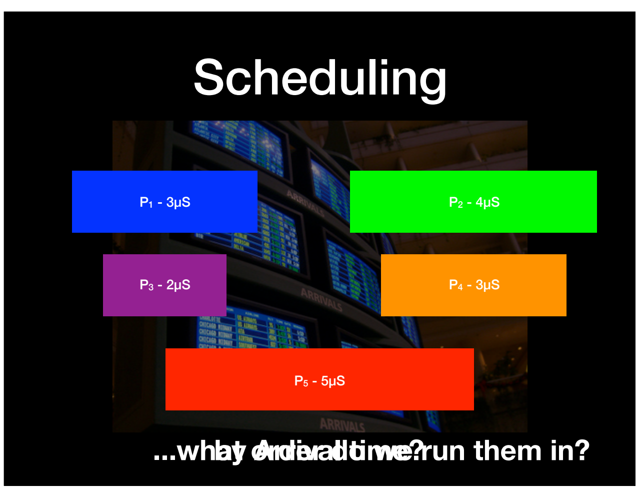
Multi-processing
CPU
IO
CPU
IO
CPU
IO
CPU
IO
CPU
CPU
IO
CPU
IO
CPU
IO
CPU
...makes use of *idle time*


Multi-processing
CPU
CPU
CPU
CPU
CPU
CPU
CPU
CPU
CPU
don't forget **Context Switches**

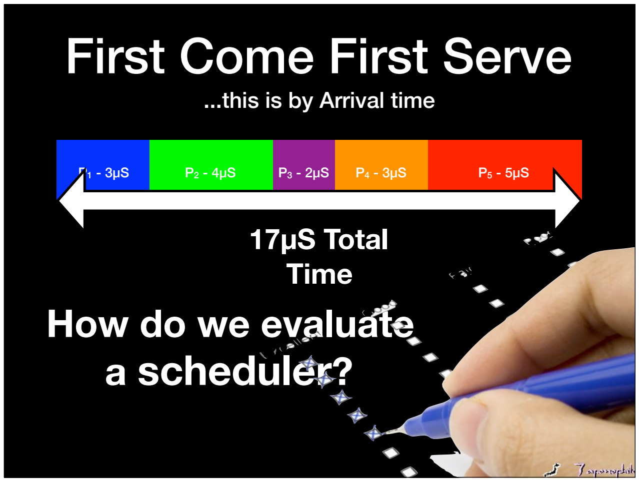
4


Scheduling
P2 Has 4μS to execute
P2 - 4μS
P3 - 2μS
P3 Has 2μS to execute
...is planning what to execute next

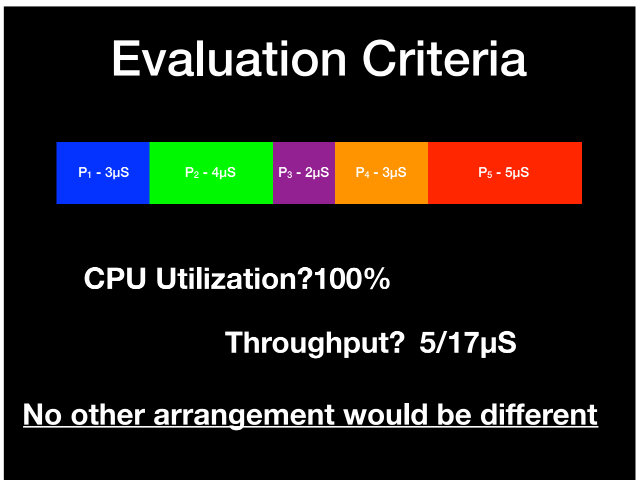
5


Scheduling
P1 - 3μS
P2 - 4μS
P3 - 2μS
P4 - 3μS
P5 - 5μS
run them in?

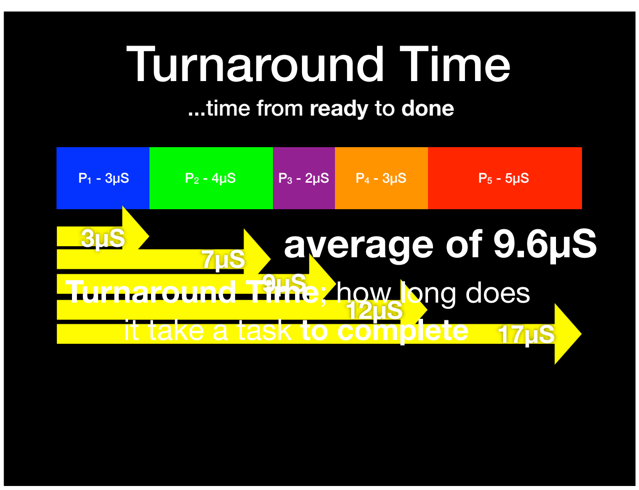
6


First Come First Serve
...this is by Arrival time
P_1 - 3µS
P_2 - 4µS
P_3 - 2µS
P_4 - 3µS
P_5 - 5µS
17µS Total Time
How do we evaluate a scheduler?

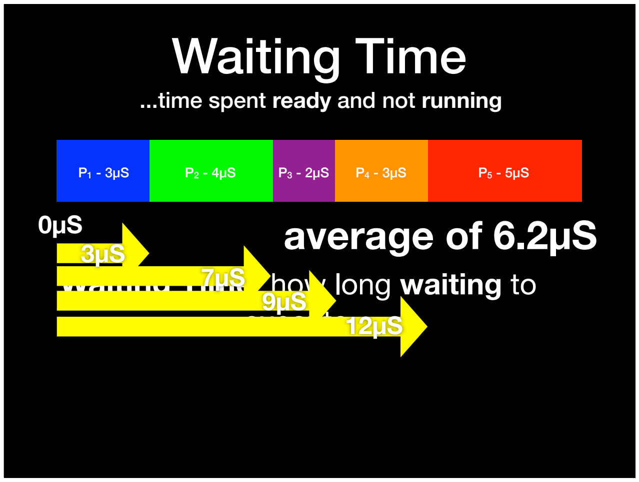
7


Evaluation Criteria
P_1 - 3µS
P_2 - 4µS
P_3 - 2µS
P_4 - 3µS
P_5 - 5µS
CPU Utilization?100%
Throughput? 5/17µS
No other arrangement would be different

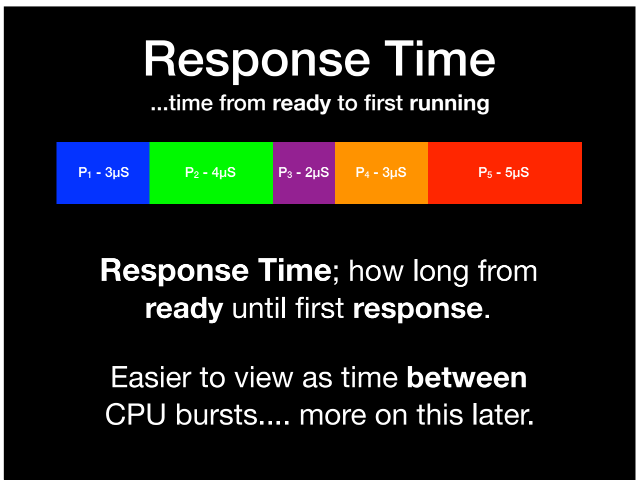
8


Turnaround Time
...time from ready to done
P_1 - 3µS
P_2 - 4µS
P_3 - 2µS
P_4 - 3µS
P_5 - 5µS
3µS
7µS
9µS
12µS
17µS
average of 9.6µS
Turnaround Time: how long does it take a task to complete

9


Waiting Time
...time spent ready and not running
P1 - 3µS
P2 - 4µS
P3 - 2µS
P4 - 3µS
P5 - 5µS
0µS
3µS
7µS
9µS
12µS
average of 6.2µS
long waiting to

10


Response Time
...time from ready to first running
P1 - 3µS
P2 - 4µS
P3 - 2µS
P4 - 3µS
P5 - 5µS
Response Time; how long from ready until first response.
Easier to view as time between CPU bursts.... more on this later.

11

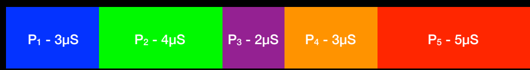

First Come First Serve
...evaluated
P1 - 3µS
P2 - 4µS
P3 - 2µS
P4 - 3µS
P5 - 5µS
Criteria
FCFS
Can we do better?
Response Time
n/a

12

13

Shortest Job First

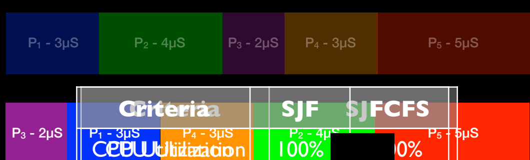
...run the job with the least to do



14

Shortest Job First

...run the job with the least to do



Criteria	SJF	FCFS
CPU Utilization	100%	0%
Throughput	5/17	/17
Turnaround Time	8.8	.6
Waiting Time	5.4	.2
Response Time	n/a	/a

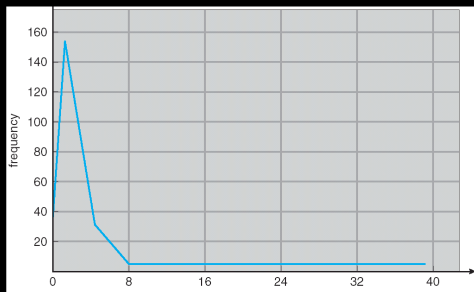
15

Optimal Solution

Shortest Job First is the **Optimal** scheduling algorithm.

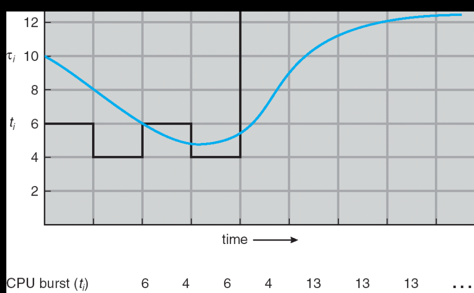
...the problem is we never know how long a burst is.

16



CPU bursts are typically **short**

17



we can **estimate** from history

18

Starvation

SJF also can cause

starvation

of large jobs

No image suitable for class

19

What about priority?

P_1 - 3µS, p=3

P_2 - 4µS, p=4

P_3 - 2µS, p=2

P_4 - 3µS, p=3

P_5 - 5µS, p=5

...SJF is a special case of priority scheduling

20

priority = length & age

P_1 - 3µS, p=3

P_2 - 4µS, p=4

P_3 - 2µS, p=2

P_4 - 3µS, p=1

P_5 - 5µS, p=2

...adjust priority to reflect a burst's age

21

Evaluation Criteria

P_4 - 3µS, p=1 | P_3 - 2µS, p=2 | P_5 - 5µS, p=2 | P_1 - 3µS, p=3 | P_2 - 4µS, p=4

Criteria	SJF	FCFS	Priorit
CPU Utilization	100%	100%	
Throughput	5/17	5/17	
Turnaround Time	8.8	9.6	
Waiting Time	5.4	6.2	
Response Time	n/a	n/a	

22

When **must** we make a scheduling decision?

When a process goes from **running** to **waiting**

When a process **terminates**

23

When **can** we make a scheduling decision? (preemptive)

When a process goes from **waiting** to **ready**

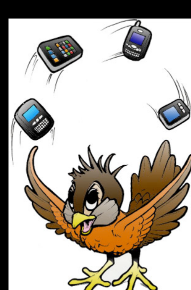
When a process goes from **running** to **ready**

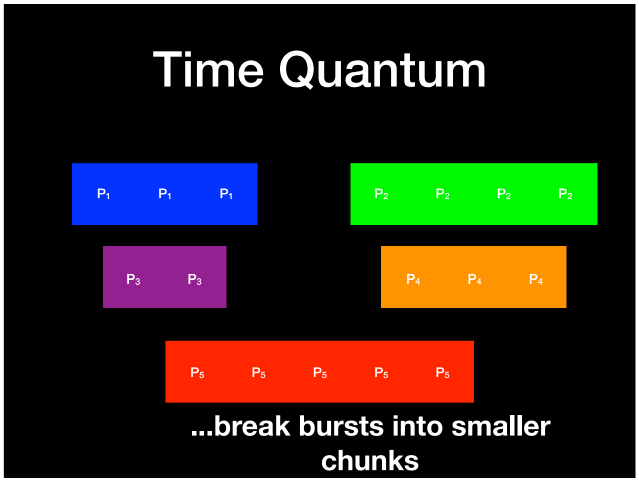
24

Round Robin Scheduling

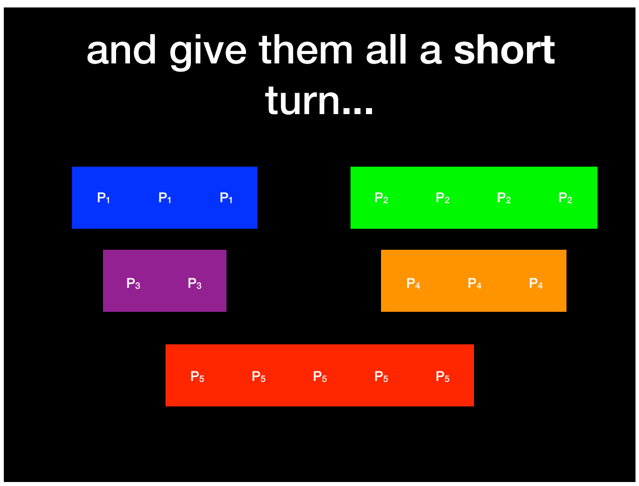
Take advantage of
preemptive
capabilities

the timer is our friend

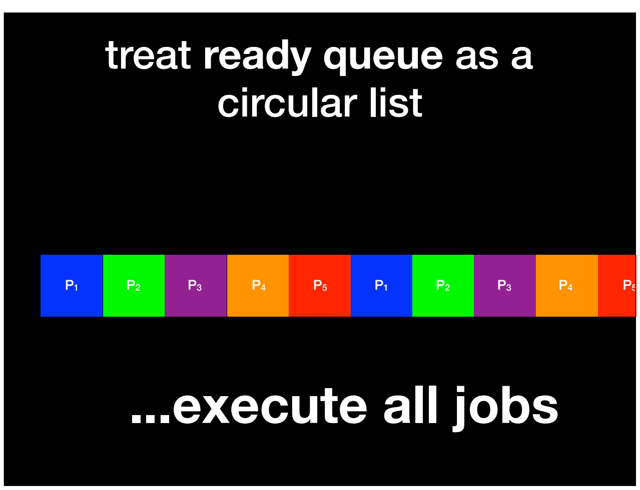



Time Quantum
P_1 P_1 P_1
P_2 P_2 P_2 P_2
P_3 P_3
P_4 P_4 P_4
P_5 P_5 P_5 P_5 P_5
...break bursts into smaller chunks

25


and give them all a short turn...
P_1 P_1 P_1
P_2 P_2 P_2 P_2
P_3 P_3
P_4 P_4 P_4
P_5 P_5 P_5 P_5 P_5

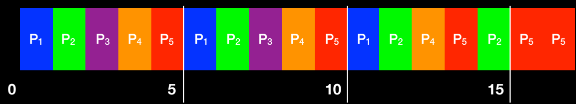
26


treat ready queue as a circular list
P_1 P_2 P_3 P_4 P_5 P_1 P_2 P_3 P_4
...execute all jobs

27

28

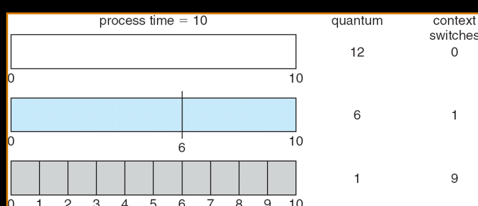
Evaluation Criteria



Criteria	SJF	FCFS	Priority	RR q=1
CPU Utilization	100%	100%	100%	100%
Throughput	5/17	5/17	5/17	5/17
Turnaround Time	8.8	9.6	9.6	
Waiting Time	5.4	6.2	6.2	
Response Time		~13.6		

29

Quantum Size

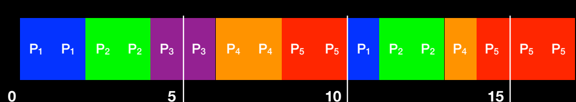


smaller quantum...

...**more** context switches

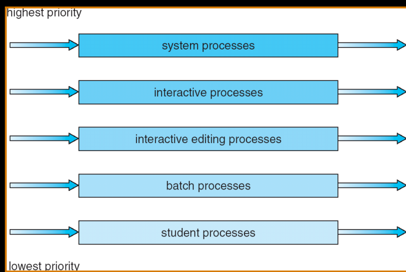
30

Evaluation Criteria



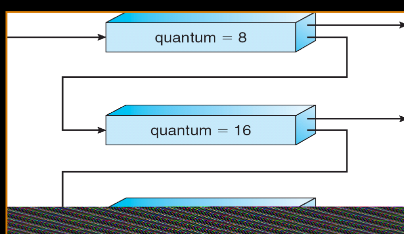
Criteria	SJF	FCFS	Priority	RR q=1	RR q=2
CPU Utilization	100%	100%	100%	100%	100%
Throughput	5/17	5/17	5/17	5/17	5/17
Turnaround Time	8.8	9.6	9.6	12.8	
Waiting Time	5.4	6.2	6.2	9	
Response Time		~13.6		2.51	

Multi-level Scheduling



exhaust **all** high priority jobs before lower priority

31



Multilevel w/Feedback

32

Multiple Processors

asymmetric

symmetric

processor affinity

scheduling is more involved

33

Asymmetric

34

Symmetric

35

Processor Affinity

36

37

Real-Time Systems

hard real-time required to complete critical task - guaranteed

soft real-time critical tasks get priority

38

Thread Scheduling

39

Evaluation Methods

Deterministic - predetermined workload

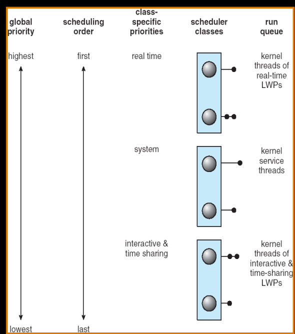
Queuing Model - purely mathematical

Simulation - write code to simulate

Implementation - try it!

Solaris

40



Three classes of schedulers

- real-time
- kernel
- interactive

Windows XP

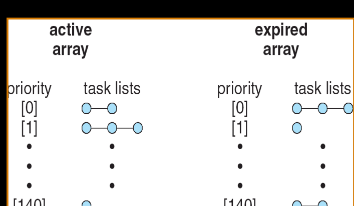
41

Several classes based on priority

	real-time	high	above normal	normal	below normal	idle priority
time-critical	31	15	15	15	15	15
highest	26	15	12	10	8	6
above normal	25	14	11	9	7	5
normal	24	13	10	8	6	4
below normal	23	12	9	7	5	3
lowest	22	11	8	6	4	2
idle	16	1	1	1	1	1

Linux

42



time-sharing prioritized credit based

real-time (soft) priority FCFS and RR

Questions (5.1)

43

Why is it important to distinguish **I/O-bound** processes from **CPU-bound** processes?

Questions (5.2)

44

How and when do these conflict?

- CPU Utilization & Response time
- I/O Utilization & CPU Utilization

Questions (5.3)

45

Which of these could result in **starvation**?

- FCFS
- SJF
- RR
- Priority

Questions (5.6)

Consider RR-variant where ready queue contains **pointers** to PCBs.

- What would be the effect of two pointers to **the same process**?
- Could we alter basic RR to do this w/o pointers?

46

Scheduling

End of Section

47