CPU Scheduling

44-550: Operating Systems

CPU Scheduling

- Determining what CPU resources are available to running processes is one of the responsibilities of the OS
- Must be efficient and effective
- More than one *policy*
  - When does a process move from ready to run?
- The OS must decide when it should switch *context*

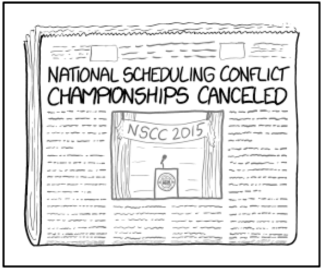


Figure: From `https://xkcd.com/1542/`

Types of Scheduling

- Long term
  - Decides which processes to load into memory
  - Decides which process to start based on order and priority
- Medium Term
  - Schedule processes based on resources required
  - Suspend processes that cannot run (maximum claim on resource exceeds that available)
- Short Term (CPU Scheduling)
  - Allocates CPU time among runnable processes
  - Very fast execution vital
    - A quick decision is more important than the optimal decision
  - Uses a *ready list* to determine which processes are ready to run

Some Definitions

CPU Burst

The amount of time the process uses the processor before it is no longer ready

Time Slice

A discrete, finite unit of time. When talking CPU scheduling, equal time slices are called *quanta* (which is the plural of *quantum*)

Context Switch

A process in which the context of the current process is saved, the CPU is deallocated from that process, and allocated to a new process (and the new context is loaded). A significantly expensive operation.

Scheduler Functions

- Selects the next process to get CPU time
    - Obtains process from ready queue, loads the context
- De-allocates the CPU from the currently running process
- Allocates the CPU to the newly selected process

Context Switches

- Can happen:
  - At the end of the CPU burst
  - Process is interrupted by the OS
  - Process has completed the time slice
- OS may have different classes of processes, or it may be fair
  - All processes are treated the same, or...
  - Processes are given a priority set by the OS or the user

Scheduling Policies

- Non-preemptive
  - Process executes until CPU burst is complete
- Preemptive
  - Process can get interrupted while executing
    - Time slice expires
    - Higher priority may be in ready queue

Considerations

- CPU Utilization
- Throughput
    - Number of processes executed and completed in a certain time period
- Process average wait time
- Average turnaround time
    - Average time from start to finish
- Average response time
    - Time from when a process sends a command to the OS until the response is received
- Fairness
    - How processes are treated

A Sampling of Policies

- First Come First Served (FCFS)
- Shortest Job First (SJF)
- Round Robin (RR)
- Shortest Remaining Time (SRT)

No one policy is superior to all others; it becomes a balancing act and determining what characteristics of each policy are important

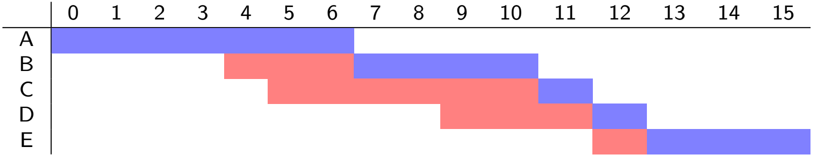
First Come First Served

- Implemented with a queue (FIFO)
- Arrival order determines the selection of next process to run
- Non-preemptive

Process	Burst Time(t)	Arrival Time
A	7	0
B	4	4
C	1	5
D	1	9
E	3	12

🟥 Waiting 🟦 Running

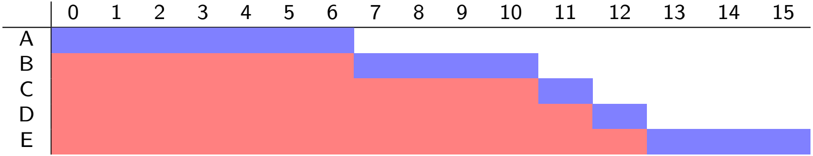
FCFS Example



Process	Required Time (T)	Wait Time	Total Time
A	7	0	7
B	4	3	7
C	1	6	7
D	1	3	4
E	3	1	4
Average		2.6	5.8
		Wait Time	Turnaround

Throughput:$5/16 = 0.3125$

What if all jobs get there simultaneously?



Process	Required Time (T)	Wait Time	Total Time
A	7	0	7
B	4	7	11
C	1	11	12
D	1	12	13
E	3	13	16
Average		8.6	11.8
		Wait Time	Turnaround

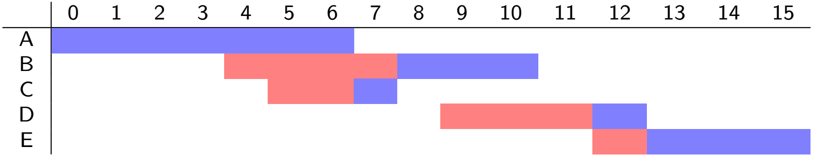
Throughput:$5/16 = 0.3125$

Ouch! We can do better than that! We should try some other scheduling policies. This is dead simple to implement, though.

FCFS: the Convoy Effect

- CPU heavy jobs will hold CPU until exit or I/O
  - I/O is rare in CPU burst intensive processes
- Have to intelligently deal with I/O bursts and CPU bursts
- Example:
  - CPU bound runs (I/O bound idle)
  - CPU bound blocks
  - I/O bound jobs run, quickly block on I/O
  - CPU bound runs again
  - I/O Completes
  - CPU bound still runs while I/O devices idle

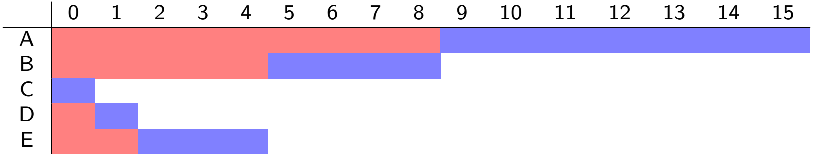
Shortest Job First (SJF) (Original Problem)



Process	Required Time (T)	Wait Time	Total Time
A	7	0	7
B	4	4	8
C	1	2	3
D	1	3	4
E	3	1	4
Average		2	5.2
		Wait Time	Turnaround

Throughput:$5/16 = 0.3125$

Shortest Job First (SJF) (All Arrive at Time 0)



Process	Required Time (T)	Wait Time	Total Time
A	7	9	16
B	4	5	9
C	1	0	1
D	1	1	2
E	3	2	5
Average		3.4	6.6
		Wait Time	Turnaround

Throughput:$5/16 = 0.3125$

SJF

- SJF doesn't always minimize Turnaround Time (though it will miminize Wait Time)
- Requires a "psychic" CPU
  - Not completely sure how long the CPU bursts for a process are
  - Can estimate based on past behavior, though
- Lots of short jobs could push out a long running job