All algorithms that require more explanation will use this example below:

P1 0 CPU 10 IO 3 CPU 10 IO 4 CPU 2

P2 3 CPU 100 IO 2 CPU 50 IO 2 CPU 4

P3 7 CPU 3 IO 2 CPU 2 IO 1 CPU 3 IO 1 CPU 2 IO 5 CPU 1

P4 11 CPU 10 IO 20 CPU 20

P5 90 CPU 3 IO 2 CPU 2 IO 1 CPU 3 IO 1 CPU 2 IO 5 CPU 1

FCFS: No preempt

Uses no parameters

Take the next process in the ready queue, process for cpu burst time, once process burst time is complete, if the process is finished, it will end, otherwise it will go to the end of the queue and the scheduler will give the next process the cpu.

All algorithms that require more explanation will use this example below:

P1 0 CPU 10 IO 3 CPU 10 IO 4 CPU 2

P2 3 CPU 100 IO 2 CPU 50 IO 2 CPU 4

P3 7 CPU 3 IO 2 CPU 2 IO 1 CPU 3 IO 1 CPU 2 IO 5 CPU 1

P4 11 CPU 10 IO 20 CPU 20

P5 90 CPU 3 IO 2 CPU 2 IO 1 CPU 3 IO 1 CPU 2 IO 5 CPU 1

Virtual Round robin: Preempt

Uses 1 parameter: Quantum

Like FCFS but preemption is added to switch between processes

For instance, if there are 5 processes and the time quantum = 5,  
It will run through process 1 for 5 seconds (if the process burst is shorter than time quantum, will voluntarily end and scheduler will move on, else run for full quantum time), then if the process has not finished, it will be added to the end of the queue and process 2 will be given the cpu for 1 time quantum. In the case that the process blocks before or at quantum time, the process will be pushed into a aux queue which will have priority over the other queue.

Using example:

Time 0: P1 arrives, starts  
Time 3: P2 arrives, P1 Timeout, burst times: P2 = 100, P1 = 7, P2 starts  
Time 6: P2 timeout, burst times: P1 = 7, P2 = 97, P1 starts  
Time 7: P3 arrives  
Time 9: P1 timeout, burst times: P2 = 97, P3 = 3, P1 = 4, P2 starts  
Time 11: P4 arrives  
Time 12: P2 timeout, burst times: P3 = 3, P1 = 4, P4 = 10, P2 = 94, P3 starts  
Time 15: P3 blocks for 2 IO, burst times: P1 = 4, P4 = 10, P2 = 94, P1 starts   
Time 17: P3 unblocks, priority queue  
Time 18: P1 timeout, burst times: P3 = 2, P4 = 10, P2 = 94, P1 = 1, P3 starts  
Time 20: P3 blocks for 1 IO, burst times: P4 = 10, P2 = 94, P1 = 1, P4 starts  
Time 21: P3 unblocks, priority queue  
Time 23: P4 timeout, burst times: P3 = 3, P2 = 94, P1 = 1, P4 = 7, P3 starts  
Time 26: P3 blocks for 1 IO, burst times: P2 = 94, P1 = 1, P4 = 7, P2 starts  
Time 27: P3 unblocks, priority queue  
Time 29: P2 timeout, burst times: P3 = 2, P1 = 1, P4 = 7, P2 = 91, P3 starts  
Time 31: P3 blocks for 5 IO, burst times: P1 = 1, P4 = 7, P2 = 91, P1 starts  
Time 32: P1 blocks for 3 IO, burst times: P4 = 7, P2 = 91, P4 starts  
Time 35: P4 timeout, P1 unblocks, priority queue, burst times: P1 = 10, P2 = 91, P4 = 4, P1 starts  
Time 36: P3 unblocks, priority queue  
Time 38: P1 timeout, burst times: P3 = 1, P2 = 91, P4 = 4, P1 = 7, P3 starts  
Time 39: P3 exits, burst times: P2 = 91, P4 = 4, P1 = 7, P2 starts  
Time 42: P2 timeout, burst times: P4 = 4, P1 = 7, P2 = 88, P4 starts  
Time 45: P4 timeout, burst times: P1 = 7, P2 = 88, P4 = 1, P1 starts  
Time 48: P1 timeout, burst times: P2 = 88, P4 = 1, P1 = 4, P2 starts  
Time 51: P2 timeout, burst times: P4 = 1, P1 = 4, P2 = 85, P4 starts  
Time 52: P4 blocks for 20 IO, burst times: P1 = 4, P2 = 85, P1 starts  
Time 55: P1 timeout, burst times: P2 = 85, P1 = 1, P2 starts  
Time 58: P2 timeout, burst times: P1 = 1, P2 = 82, P1 starts  
Time 59: P1 blocks for 4 IO, burst times: P2 = 82, P2 starts  
Time 62: P2 timeout, burst times: P2 = 79, P2 starts  
Time 63: P1 unblocks, priority queue  
Time 65: P2 timeout, burst times: P1 = 2, P2 = 76, P1 starts  
Time 67: P1 exits, burst times: P2 = 76, P2 starts  
Time 70: P2 timeout, burst times: P2 = 73, P2 starts  
Time 72: P4 unblocks, priority queue  
Time 73: P2 timeout, burst times: P4 = 20, P2 = 70, P4 starts  
…

All algorithms that require more explanation will use this example below:

P1 0 CPU 10 IO 3 CPU 10 IO 4 CPU 2

P2 3 CPU 100 IO 2 CPU 50 IO 2 CPU 4

P3 7 CPU 3 IO 2 CPU 2 IO 1 CPU 3 IO 1 CPU 2 IO 5 CPU 1

P4 11 CPU 10 IO 20 CPU 20

P5 90 CPU 3 IO 2 CPU 2 IO 1 CPU 3 IO 1 CPU 2 IO 5 CPU 1

SRT (Shortest Time Remaining): Preempt

Uses 2 parameters: ServiceGiven and Alpha

Runs the shortest burst remaining, will stop a process if another process arrives or readies with a shorter remaining time. If serviceGiven is true, the comparison will be based on the cpu service times that would run. If serviceGiven is false, the comparison will be based on the exponential service prediction.

Using example:

Time 0: P1 arrives, starts  
Time 3: P2 arrives, burst times are: P1=7, P2 = 100  
Time 7: P3 arrives, Burst times are: P1 = 3, P2 = 100, P3 = 3, P3 runs  
Time 10: P3 blocks for IO 2, Burst times are: P1 = 3, P2 = 100, P1 runs  
Time 11: P4 arrives, Burst times are: P1 = 2, P2 = 100, P4 = 10  
Time 12: P3 readies, Burst times are: P1= 1, P2 = 100, P3 = 2, P4 = 10  
Time 13: P1 blocks for IO 3, Burst times are: P2 = 100, P3 = 2, P4 = 10, P3 runs  
Time 15: P3 blocks for IO 1, Burst times are: P2 = 100, P4 = 10, P4 runs  
Time 16: P1 readies, P3 readies, Burst times are: P1 = 10, P2 = 100, P3 = 3, P4 = 9, P3 runs  
Time 19: P3 blocks for IO 1, Burst times are: P1 = 10, P2 = 100, P4 = 9, P4 runs  
Time 20: P3 readies, Burst times are: P1 = 10, P2 = 100, P3 = 2, P4 = 8, P3 runs  
Time 22: P3 blocks for IO 5, Burst times are: P1 = 10, P2 = 100, P4 = 8, P4 runs  
Time 27: P3 readies, Burst times are: P1 = 10, P2 = 100, P3 = 1, P4 = 8, P3 runs  
Time 28: P3 exits, Burst times are: P1 = 10, P2 = 100, P4 = 8, P4 runs

… will not finish this

All algorithms that require more explanation will use this example below:

P1 0 CPU 10 IO 3 CPU 10 IO 4 CPU 2

P2 3 CPU 100 IO 2 CPU 50 IO 2 CPU 4

P3 7 CPU 3 IO 2 CPU 2 IO 1 CPU 3 IO 1 CPU 2 IO 5 CPU 1

P4 11 CPU 10 IO 20 CPU 20

P5 90 CPU 3 IO 2 CPU 2 IO 1 CPU 3 IO 1 CPU 2 IO 5 CPU 1

HRRN (Highest Response Ratio Next): No preempt

Will take the process that has the highest response ratio and put into the CPU

All algorithms that require more explanation will use this example below:

P1 0 CPU 10 IO 3 CPU 10 IO 4 CPU 2

P2 3 CPU 100 IO 2 CPU 50 IO 2 CPU 4

P3 7 CPU 3 IO 2 CPU 2 IO 1 CPU 3 IO 1 CPU 2 IO 5 CPU 1

P4 11 CPU 10 IO 20 CPU 20

P5 90 CPU 3 IO 2 CPU 2 IO 1 CPU 3 IO 1 CPU 2 IO 5 CPU 1

Feedback: Preempt

Uses parameters: Quantum and Priority

Like virtual round robin, but every time a process times out due to the quantum time, the process is set 1 degree lower in priority, and gets added to that level’s queue.

Using the example:

Time 0: P1 arrives, P1 starts

IMPORTANT

Calculating exponential service times:

Sn+1 (next predicted service time) = (alpha \* service time of past cpu) + (1-alpha)(current predicted service time)