Why an OS has to schedule processes

Processes or jobs competing for CPU are not of the same type or have the same level of importance. For example some processes are CPU bound while other processes are IO bound. In the later case when a process blocks for IO the OS should let other processes use the CPU otherwise the computation resource is not going to be utilized as expected. Here is an overall list of scheduling goals sorted by system type:

All systems

1. Fairness: give each process a fair share of the CPU
2. Enforcement: ensure that the stated policy is carried out
3. Balance: keep all parts of the system busy

Batch Systems

1. Throughput: maximize jobs per unit time
2. Turnaround: minimize the time users wait for jobs
3. CPU utilization: keep the CPU as busy as possible

Interactive Systems

1. Response time: respond quickly to user requests
2. Proportionality: meet user expectations

Real-time Systems

1. Meet deadlines: missing deadlines is a system failure
2. Predictability: same type of behavior for each time slice

So we now know why scheduling is important. Let us explore some of the well known scheduling algorithms:

First Come First Serve (FCFS)

1. Serve the jobs in the order they arrive
2. Non-Premptive scheduling algorithm
3. Simple and easy to implement
4. Longer jobs delay shorter jobs

Shortest Job First (SJF)

1. Short jobs complete first
2. Long jobs may starve
3. Rarely used because time is hard to estimate

Shortest Remaining Time First (SRTF)

1. This is the preemptive form of SJF
2. Reevaluate when a new job is submitted
3. Rarely used because time is hard to estimate

Three Level Scheduling

1. Scheduling jobs for admission to memory
2. Scheduling jobs to use the CPU
3. Memory scheduler to move jobs back to disk

Round Robin Scheduling

1. Give each ready process a time slot called quantum
2. Too short quantum hearts efficiency
3. Too long quantum hearts responsiveness

Priority Scheduling

1. Assign a priority to each running process
2. Decrease priority as process runs
3. Increase priority when a process waits for IO
4. Group processes with the same priority then use round robin

Shortest Process Next Scheduling

1. Serve the process that will finish the soonest
2. Guess completion time based on previous runs

Lottery Scheduling

1. Give processes tickets for CPU time
2. More tickets a process gets more CPU time it gets
3. Tickets can be transferred between cooperating processes

Multi-Level Feedback Queue Scheduling

1. A process can move between queues and have a different priority with each queue
2. If a process uses too much CPU time it will be moved to a lower priority queue
3. If a process waits too long in the lower priority queue it may be moved to a higher priority queue
4. Aging prevents starvation