Processes states: a cpu (central processing unit)

Processesin a system cycle between two states: a CPU (Central Processing Unit) burst, inwhich calculations are carried out, and an I/O (input/output) burst, in whichdata is either sent to or received by the system. The Central Processing Unit (CPU)attempts to maximise the efficiency and ‘ fairness’ of a CPU, by allowing aprocess to utilise the CPU, whilst another waits for an I/O. If this isn’taddressed, CPU cycles are lost whilst the CPU waits for a single process’s I/O. The role of the CPU scheduler is to find the subsequent process for an inactiveCPU, from a queue of immediately available processes. The order in which the processesare handled depends on the algorithm implemented by the CPU scheduler.

Thereare two types of algorithms applied: non-preemptive and preemptive. Nonpreemptivealgorithms must be applied applied when a new process is required, i. e. wheneither a process is terminated, or when a process transitions from a running towaiting state. However, when a process goes from either a running state toready state, or from waiting state to ready state, preemptive algorithms areoptimum. In order to ensure that preemptive algorithms can be implemented, itmust be ensured that the hardware can support time interrupts. Once the schedulingalgorithm has selected a process, the dispatcher hands control of the CPU toit.

This involves a context switch, a user mode switch and then a jump to thecorrect location in the newly loaded program. The time taken for this to complete is called the dispatchlatency. Whenattempting to choosing the optimum algorithm to implement in CPU scheduling, there is criteria to adhere by in order to make the best decision: CPUutilisation should be maximised, in order not waste a single CPU cycle whereverpossible, and the ‘

throughput’ (number of processes executed per unit time) shouldalso be maximised. However, the turnaround time, waiting time and response timeshould all be minimised as much as possible. http://www.

studytonight. com/operating-system/cpu-schedulingForthe following evaluation of algorithms, a single CPU burst is assumed perprocess. First-Come First Served (FCFS) is the most primitive form ofscheduling, involving a first-in, first-out basis for the queue order, and is anon-preemptive algorithm. The benefits of using FCFS is that it is simple inapplication and allows the process being executed to complete its CPU burst. Therefore, there is no need to context switch and take CPU away from process preemptively. However, as well as having long waiting times, FCFS can also slow the systemdown in a process known as the convoy effect; preventing the CPU from being optimallyused as after the large process finished, the CPU goes through periods of high intensityprocessing (with a minimal I/O) and vice versa 3.

RoundRobin (RR) shares similarities with FCFS, however, the CPU bursts are givenlimits called time quantum. If the process completes before the time quantumlimit runs out, the schedule continues in a first in, first out manner. However, if the quantum timer limit expires, the time interrupt occurs and the processis placed at the back of the queue (the algorithm is preempetive). This isideal for small tasks, as even if a large process is placed higher in the queuethan a short one, each get a ‘ fair’ attempt.

However, this proves problematicwith processes of equal time, where FCFS could have a lower throughput and theoverhead produced through ‘ context switching’ reduces the CPU utilisation 3. Shortest-Job-FirstScheduling (SJF) arranges the queue in ascending order of length of CPU bursttime.