Uniprocessor Scheduling Simulator

Operating System Project (CS 307)

Group No. 06 (Ashish Kumar Bedi - B13204 and Mukarram Tailor - B13318)

What is scheduling?

In computing, scheduling is the method by which work specified by some means is assigned to resources that complete the work.

The resources may be virtual computation elements such as threads, processes or data flows, which are in turn scheduled onto hardware resources such as processors, network links or expansion cards.

Scheduler : Importance in OS

A scheduler is what carries out the scheduling activity.

Scheduler is fundamental to computation itself, and an intrinsic part of the execution model of a computer system.

Schedulers are often implemented in an OS so they keep all compute resources busy (as in load balancing), allow multiple processes to share system resources effectively.

The concept of scheduling makes it possible to have computer multitasking with a single central processing unit (CPU).

Scheduler : Features

A scheduler may aim at one of many goals, for example:

i) maximizing throughput (the total amount of work completed per time unit)

ii) minimizing response time (time from work becoming enabled until the first point it begins execution on resources)

iii) minimizing latency (the time between work becoming enabled and its subsequent completion)

Scheduler : Functioning

Our scheduler keeps track of the processes as well as their states, viz. blocked, ready, or running.

All the Scheduling Criteria are based on the following methods:

I) User-oriented Criteria

ii) System-oriented Criteria

User oriented Criteria

- User-oriented criteria is the behavior of the system as perceived by a single user.
- **Quantitative:**
- Turnaround Time – The time from when a process is first added to the group of ready executable processes to the time when it finishes executing and exits. Discussed in much greater detail below.
- Response Time – The time from when the request is first made to the time when the response starts to be received. From a user’s point of view, response time is a better measure than turnaround time because a process can produce feedback to the user while the process is still executing.
- **Qualitative:**
- Predictability – For any given job, it should run in approximately the same amount of time regardless of the load on the system.

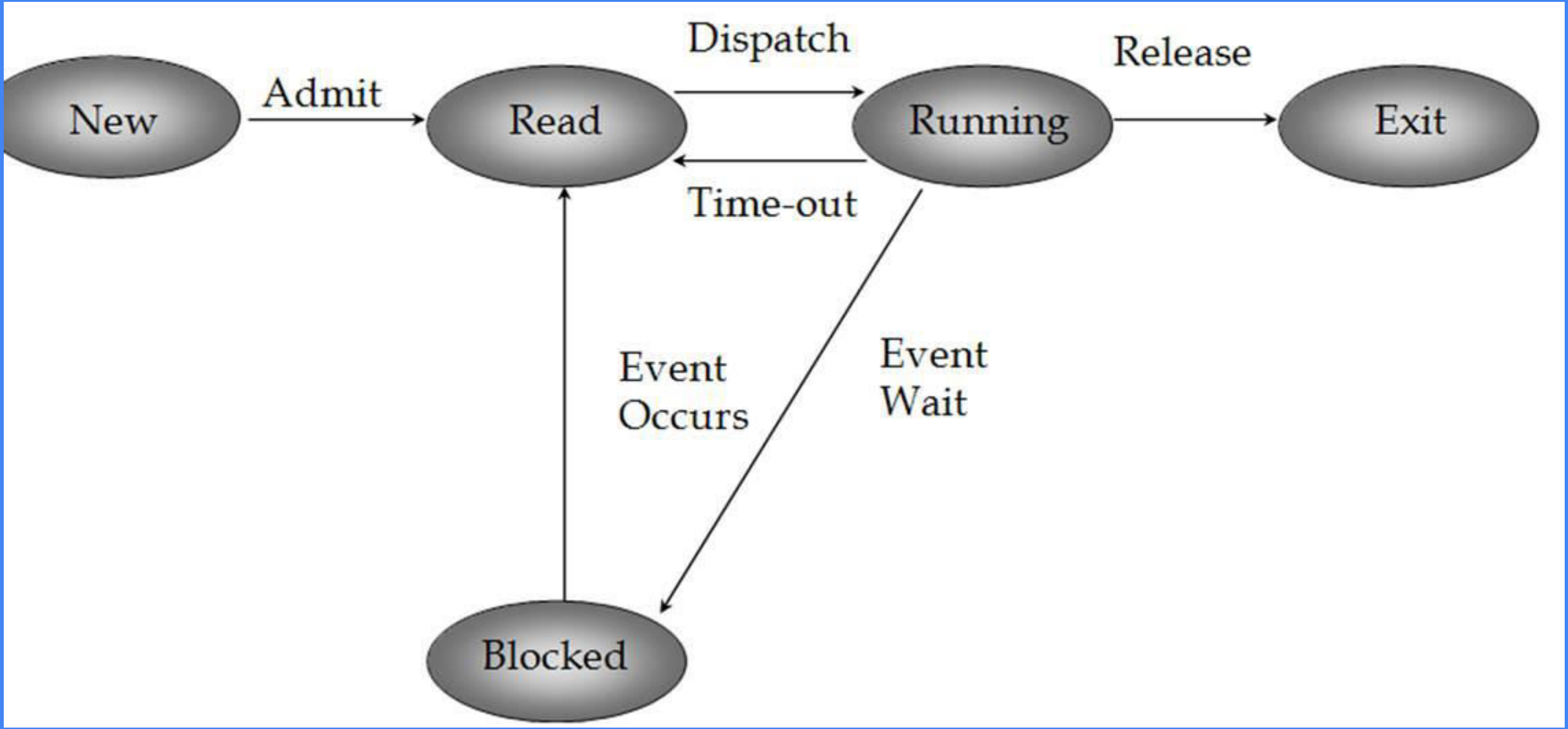
System oriented Criteria

- System-oriented criteria focus on effective and efficient utilization of the processor.

- **Quantitative:**
- Throughput – The number of processes completed per unit of time. Depends a lot on the average length of the processes, but is still affected by the scheduling policy.
- Processor Utilization – The percentage of time that the processor is busy. More important in multi-user systems than single user systems.

- **Qualitative:**
- Fairness – Processes should be treated the same, and no process should suffer starvation.

Process State Diagram



Assumption and Rules

1) Process status diagram followed by the program : -

$$\begin{tikzcd}
\text{NEW} \arrow[r, dashed] & \text{READY} \arrow[rr, shift left] & & \text{RUNNING} \arrow[ll, shift left] \arrow[dl] \\
& & \text{BLOCKED} \arrow[ul] &
\end{tikzcd}$$

Assumption and Rules(contd.)

2) All the processes in input file which have not encountered their arrival cycle yet, will be marked as status "NEW"

3) Simulator CPU will only pick process from ready queue for running, therefore, blocked process cannot become running immediately, instead, it should become ready first.

4) All processes' ids will be consecutive positive decimal integers, from m to n ($0 <= m < n <$ infinite).

Major Scheduling Algorithms used

First Come First Serve (FCFS) Scheduling

Shortest-Remaining-Time-First (SRTF) Scheduling

Round Robin(RR) Scheduling

Round Robin

- Round Robin is a policy that uses preemption based on a clock interrupt at periodic intervals.

- **Are all types of processes treated fairly?**
- Yes, all are treated fairly In order to schedule processes fairly, a round-robin scheduler generally employs time-sharing, giving each job a time slot or quantum (its allowance of CPU time), and interrupting the job if it is not completed by then. The job is resumed next time a time slot is assigned to that process. If the process terminates or changes its state to waiting during its attributed time quantum, the scheduler selects the first process in the ready queue to execute. In the absence of time-sharing, or if the quanta were large relative to the sizes of the jobs, a process that produced large jobs would be favored over other processes.

Round Robin (continued)

- Round Robin algorithm is a pre-emptive algorithm as the scheduler forces the process out of the CPU once the time quota expires.

- In the implementation of our Round Robin Algorithm, we have kept the CPU time utilization for any process as 2 seconds, i.e., any process, no matter what its total CPU time is, will remain in the Running State of a maximum of 2 seconds at one go.

First-Come-First-Served (FCFS)

- FCFS is a non-preemptive policy that selects and runs a process until completion based on FIFO.

Are all types of processes treated fairly?

- No, they are not. Longer processes are going to fare better than shorter processes, especially in the case when a short process arrives just after a long process.

- Processor-bound processes are favored over I/O-bound processes. Suppose we have a collection of processor-bound and I/O-bound processes. When a processor-bound process is running, all I/O-bound processes will have to wait. Some of these I/O-bound processes will be in a blocked state, but could move to the ready state even while the processor-bound process is running once they receive their required input or finish their output operation. When the processor-bound process finally finishes executing, the I/O-bound process will execute only to become quickly blocked by I/O operations again.

Shortest Remaining Time First (SRTF)

- Shortest remaining time, also known as shortest remaining time first (SRTF), is a scheduling method that is a preemptive version of shortest job next scheduling. In this scheduling algorithm, the process with the smallest amount of time remaining until completion is selected to execute.

- Since the currently executing process is the one with the shortest amount of time remaining by definition, and since that time should only reduce as execution progresses, processes will always run until they complete or a new process is added that requires a smaller amount of time.

- **Are all types of processes treated fairly?**
- Penalizes long processes. Short processes are able to cut in line ahead of longer processes.

Basic Implementation Introduction

- Multiple queues are maintained.

- If processes come after CPU running for a few cycles (all processes are NEW at a certain cycle), program will indicate CPU is idle and output CPU IDLE.

Basic Logic for each cycle :

Check new coming processes, put them into ready queue.

Scan blocked queue's processes, consume their i/o time, if i/o is finished, move them to ready queue

Running arbitration: if there is one process running, check preemptive, compare with ready queue, if fulfill preemptive, replace current running process; check current running process's CPU time, if consumed 1/2, move to blocked queue, if consumed all, delete else, running cycle++

Then, if no process is running (running one may be moved after former steps) check ready queue, pick the one fulfilled running condition as running and running cycle ++.

Thank You.

Group No. 06 (Ashish Kumar Bedi - B13204 and Mukarram Tailor - B13318)