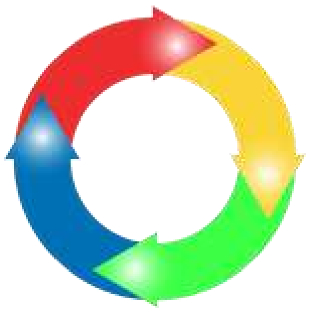
Introduction

Modern operating systems support multitasking via CPU time sharing. The operating system manages CPU time by implementing a scheduling algorithm. In this article, we are going to briefly discuss round robin scheduling algorithm. To better understand the topic, you may check the following posts…



- Multitasking terms
- Scheduling algorithms
- Evaluate a scheduling algorithm

Let us get started…

What is round robin algorithm?

Round robin is one of the most popular process scheduling algorithms due to its simplicity and fairness. Each process gets a chance to execute by the CPU for a fixed amount time slice called the quanta. The operating system alternate (i.e. context switching) the CPU between processes that are ready for execution in a circular order without priority (it can be priority based as well). It is a preemptive algorithm as the scheduler forces the process out once its time slice is expired. It is as simple as that. Let us now outline how the algorithm works…

How does it work?

Below is a short summary of how the algorithm works…

- Processes are served in the order they arrive to the system. An equal CPU time slice called quanta is allocated for each process to execute
- If the time slice expires before a process completes, the operating system forces that process out
- If a process completes or blocks for IO during its time slice, the scheduler picks another ready process from the queue
- The preempted process is placed at the back of the queue and has to wait all other waiting processes to get another chance as the CPU goes through processes in a circular way

Performance

Round robin performance mainly depends on the length of the quanta and number of processes. If the time slice is very long this makes the algorithm behaves like a first come first serve algorithm. On the other hand, if the time slice is very short, this enables the CPU to cycle through all processes quickly which is a good thing for interactive applications. The downside for interactivity is the overhead of context switching. Context switching requires saving process state so that they can be resumed later. Context switching wastes CPU time instead of doing real work.

Advantages

There are various advantages for using round robin algorithm to schedule processes. Here is a short list…

- Simple and easy to implement algorithm
- Starvation free as every process gets a chance to execute
- Better response time
- Fair algorithm as every process gets an equal amount of CPU time
- Fast decision making (i.e. low overhead to decide what to run next)

Disadvantages

There is no such a thing called free lunch. Round robin is not the best choice all the time. Here are two…

- The overhead of context switching is expensive as it requires saving state specially when the time slice is short
- May not be the best choice for interactive applications

Evaluation

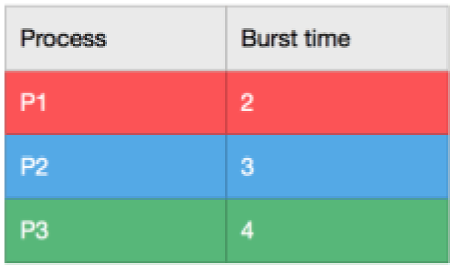
In order to evaluate the performance of round robin scheduling algorithm we need to compute turnaround time and waiting time for each process where…

- Turnaround time is the difference between completion time and arrival time
- Waiting time is the difference between turnaround time and burst time (i.e. running time)

Let us take an example…

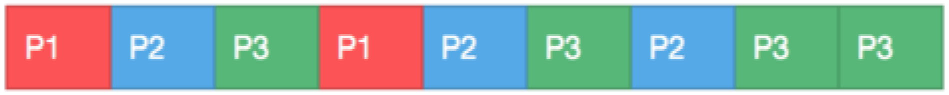
- We have 3 processes arriving in the order P1, P2, P3
- Burst time is the amount of execution time a process needs to finish
- Assume the quantum time = 1

We need to compute the waiting time and turnaround time for each process



Process	Burst time
P1	2
P2	3
P3	4

This is the Gantt chart…



Just looking at the colors, it is easy to note that P1 has 2 bursts (2 red blocks), P2 has 3 bursts and P3 has 4. It is also easy to calculate the waiting time…

1 P1 waiting time = 2 (blocks 2 & 3)
2 P2 waiting time = 4 (blocks 1, 3, 4, 6)
3 P3 waiting time = 5 (blocks 1, 2, 4, 5, 7)
4 Average wait time = (2 + 4 + 5)/3 = 3.67

Let us also calculate the turnaround time assuming all processes arrived to the system at the same time (time = 0)

1 P1 turnaround time = 4 (position of second red block)
2 P2 turnaround time = 7 (position of third blue block)
3 P3 turnaround time = 9 (position of 4th green block)
4 Average turnaround time = (4 + 7 + 9)/3 = 6.67

Summary

- One of the most popular process scheduling algorithms due to its simplicity and fairness
- Each process gets a chance to execute by the CPU for a fixed amount time slice called the quanta
- Performance mainly depends on the length of the quanta and number of processes
- Main advantages are starvation free, fairness and simplicity
- Main disadvantage is the overhead due to context switching specially when the time is lice is very short