International Journal of Progressive Sciences and Technologies (IJPSAT)
ISSN: 2509-0119.

http://ijpsat.ijsht-journals.org
Vol. 27 No. 1 June 2021, pp.458-463

Scheduling Algorithms for CPU

Basma Mohammed algubbi, Raheel mahdi baka, Zainab Ahmad Kerwad

Abstract – The process scheduling, is one of the most important tasks of the operating system. We present in this paper some types of scheduling algorithms of CPU. One of the most common scheduling algorithms used by the most operating systems is the Round Robin method in which, the ready processes waiting in ready queue, seize the processor for a short period of time known as the quantum (or time slice) circularly. But the most important issue in RR algorithm, which highly affects its efficiency. It determines the amount of the time quantum. Assigning very small and very large values for this parameter, will lead to performance decrease due to increasing context switching overhead, and degradation of the RR method to FCFS algorithm consequently. We will suggest a solution of allocated time quantum.

Keywords – CPU, Scheduling, Algorithm, Operating System.

I. INTRODUCTION

Central processing unit (CPU) scheduling is the mechanism by which operating systems (OS) allocate the CPU resources to processes. Scheduling is required for multiprogramming, and it is one of the fundamental of OS activities (Aas, 2005). [1]

The operating system (OS) is the most important program needed for starting up. Managing the tasks each of which are performed by one of the management units. The process management, is one of these management units, allocates the processor to the processes using several allocating algorithms. The scheduling criterion is to give fairness to each process and to provide efficiency in CPU usage. One of the most common algorithms in processor allocation is the Round Robin (RR) algorithm in which, the ready processes waiting in ready queue, are dispatched sequentially and allocate the processor for certain period of time known as time quantum (q) or time slice. If a process is finished during its time quantum, releases the processor, otherwise the processor is pre-empted by the OS and is allocated to the next ready process waiting in front of the ready queue and the current process will be moved to the end of the this queue . The value of the quantum parameter is selected in 10-100 milliseconds range [2]. Each value will lead to a specific performance and will affect the algorithm's efficiency by affecting the processes' waiting time. In this paper we proposed new method to solve problem by allocating time quantum in Round Robin process algorithm to minimize the average waiting time of the processes. The content of the paper is organized as the follows: In section 2, previous works. In section 3 basic concepts of scheduling. Section 4 presents the scheduling algorithms which is being the main section of this paper. In section 5, we described the proposed model, and section 6, covers summary and conclusions.

II. PREVIOUS WORKS

The importance of the problem has already raised the attention of the researchers and study in this field still continues. Some more important works are listed below:-

Albielmona [4] did an overall review over many scheduling algorithms. Proportional share scheduling algorithm proposed is combining the small overhead of the RR method by protecting the short processes. [5] The capability of re-adjusting the weights enables the algorithm to have a more fair behavior. Nieh et al. [6] proposed a new scheduling algorithm known as Virtual Time

Corresponding Author: Zainab Ahmad Kerwad

Round-Robin (VTRR) with complexity of O(1). By combining fair queuing algorithms with the RR scheduling algorithm. Nayak et al. [7] developed a new scheduling algorithm called Improved Round Robin (IRR) by dynamically calculating the time quantum value in order to decrease the average waiting time of the processes and total number of context switches. Siregar [8] used Genetic Algorithm to determine optimum time quantum value in Round Robin CPU scheduling algorithm to minimize the average waiting time of the processes.

III. BASIC CONCEPTS

The following steps describe these concepts:-

Utilization: keep the CPU as busy as possible (from 0% to 100%).

Throughput: number of processes that complete their execution per time unit.

Waiting time: sum of the periods spent waiting in the ready queue. Turnaround time: amount of time to execute a particular process; which is the interval from time of submission of a process to the time of completion.

Response time: amount of time it takes from when a request was submitted until the first response is produced.

Time quantum: also known as time slice, the amount of time that a process captures the processor.

Context Switching: Switching the processor from the current process to next ready, includes storing and retrieving the values of flags and some important registers.

Starvation: occurs if a low priority process never runs.

IV. SCHEDULING ALGORITHMS

The scheduling algorithms can be divided into two types, preemptive a non- preemptive.

The first type, allocate the processor to the processes exclusively until completion. While preemptive type, the processor from the process at a specific time slice before completion and switch to the next processes.

1.1 First-Come, First Served (FCFS)

FCFS is one the non-preemptive; it is the simplest CPU scheduling algorithm. The process that requests the CPU first is allocated the CPU first. The average waiting time under FCFS is long.

Example (1) shows (FCFS) method.

Table 1. Example for FCFS method

Proce	Burst Time *(milliseconds)*
P1	**24**
P2	**3**
P3	**3**

Suppose that the processes arrive in the order: *P1*, *P2*, *P3*. The Gantt chart for the schedule is:

Waiting time for *P* = 0; *P2* = 24; *P3* = 27 Average waiting time:

(0 + 24 + 27) / 3 = 17 *(milliseconds)*

1.2 Shortest-Job-First Scheduling (SJF)

Associate with each process the length of its next CPU burst. We use these lengths to schedule the process with the shortest time.

Below are two schemes of (SJF):

Non-preemptive: once CPU given to the process it cannot be preempted until completes its CPU burst.

Preemptive: if a new process arrives with CPU burst length less than remaining time of current executing process, preempt. This scheme is known as the Shortest-Remaining-Time-First (SRTF). If two processes have the same length CPU burst, FCFS scheduling is used. SJF is optimal gives minimum average waiting time for a given set of processes. Table (2) shows the example of 4 processes with the arrival time and them burst time:

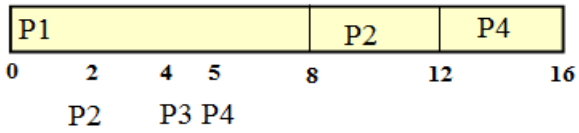
burst time:

Table 2. Example for preemptive SJF method

Process	Arrival time	Burst time
P1	0	7
P2	2	4
P3	5	1
P4	4	5

Non-preemptive SJF schedule:

Table 3. shown non-preemptive method



As shown in table (3), the average waiting time (0+6+3+7)/4=4.

Preemptive SJF schedule:

Table 4. shown preemptive method

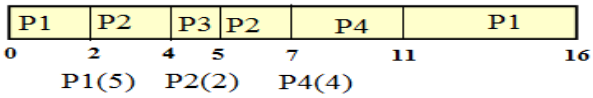


Table (4) shows the average waiting time as (9+1+0+2)/4=3.

1.3 Round Robin (RR)

The Round Robin is one the pre-emptive, it is similar to FCFS, but preemption is added to switch between processes. Each process gets a small unit of CPU time (*time quantum or time slice*), usually 10-100 milliseconds. After this time has elapsed, the process is preempted and added to the end of the ready queue. After the time slice is expired an interrupt will occur and a context switch.

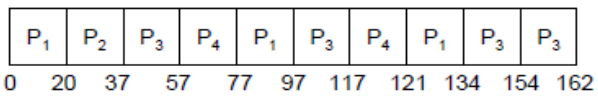
The following example explain the RR method.

RR with Time Quantum = 20

Table 5. Example for RR method

Proce	Burst Time	Waiting Time of each Process
P1	53	0+(77-20)+(121-97)=81
P2	17	20
P3	68	37+(97-57)+(134-117)=94
P4	24	57+(117-77)=97

Table 6. RR method



In table (6) the average Waiting Time calculates as (81+20+94+97) / 4 = 73.

1.4 Priority Scheduling

A priority number (integer) is associated with each process. The CPU is allocated to the process with the highest priority (smallest integer ≡ highest priority). SJF is a special case of general priority scheduling, where priority is the predicted next CPU burst time. Priorities can be defined either internally or externally:

• **Internally** use of some measurable quantity or quantities to compute the priority of a process. For example, memory requirements, number of open files, ratio of average I/O burst to average CPU burst have been used in computing priorities.

• **Externally priority** is set by criteria that are external to the O.S, such as importance of the process.

1.5 Multilevel Queue Scheduling

Ready queue is partitioned into separate queues such as:

foreground (interactive)

background (batch)

Each queue has its own scheduling algorithm:

foreground – RR

background – FCFS

Scheduling must be done between the queues, as listed below:

Fixed priority scheduling; i.e., serve all from foreground then from background. Possibility of starvation.

Time slice – each queue gets a certain amount of CPU time which it can schedule amongst its processes 80% to foreground in RR; 20% to background in FCFS.

1.6 Multilevel Feedback Queue Scheduling

Like multilevel scheduling, a process can move between various queues; aging can be implemented this way. If a process waits too long in a lower-priority queue may be moved to a higher-priority queue (this form of aging to prevent starvation). If a process uses too much CPU time, it will be moved to lower-priority queues. This leaves I/O bound and interactive processes in the higher-priority queues.

The example below has shown this method, as the three queues listed below:

Q0 time quantum 8 milliseconds

Q1 time quantum 16 milliseconds

Q2 FCFS

A new job enters queue Q0 which is served FCFS. When it gains CPU, job receives 8 milliseconds. If it does not finish in 8 milliseconds, job is moved to queue Q1. At Q1 job is again served FCFS and receives 16 additional milliseconds. If it still does not complete, it is preempted and moved to queue Q2.

V. The Proposed Model

In this section, we will present and propose new model for optimizing the time quantum value in RR scheduling algorithm.

In the proposed model, it is assumed that:

There is n number of ready processes waiting in the ready queue.

Ready queue is divided into two queues.

Queue number one is for short processes and allocated time quantum (Q1).

Queue number two is for long processes and allocated double Time Quantum (Q2).

Q2= 2*Q1.

The following example shows the proposed model:

Considering that the Proposed Model algorithm for four compute-bound processes. If we have four processes P1 (takes 53 seconds), P2 (takes 17 seconds), P3 (takes 68 seconds), and P4 (takes 24).

Table 7. Example for proposed model

Process	Burst Time
P1	53
P2	17
P3	68
P4	24

Assume Time Quantum =Sum (Burst Time) / Number of process.

Time Quantum= (53+17+68+2440≃ 4/)

TQ1=40, TQ2= 2*40.

Table 8. the proposed method

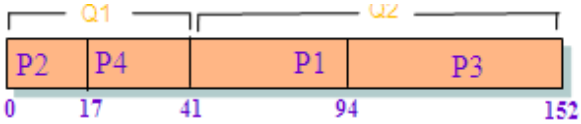


Table (8) shows the average Waiting Time for the proposed model as (0+17+41+94) / 4 = 152 / 4=38.

By combining between RR method and the proposed method we get that the average waiting time diminish in the last method, and this decline returns to dividing the queue of ready processes into two queues Q1 for short processes and Q2 for long processes, and duplicated time quantum of Q2.

VI. Conclusions

The main goal of the CPU scheduling is distribution of the CPU time among the ready processes, and to give fairness to each process. Also, to provide efficiency in CPU usage. In this paper, we proposed new model developed to solve the problem. By allocated time quantum in Round Robin process to minimize the average waiting time of the processes.

References

[1]. W. Stallings, Operating Systems: Internals and Design Principles. 6th Edition, Prentice Hall, 2008.

[2]. M. Fahimi, Operating Systems, vol. 1, 1st Edition, Tehran: Jelve Publishing, 1992.

[3]. Aas, J. (2005). Understanding the Linux 2.6.8.1 CPU Scheduler, Silicon Graphics Inc, Mountain View, CA.

[4]. R. Abielmona, "Scheduling Algorithmic Research", Department of Electrical and Computer Engineering, Ottawa-Carleton Institute, 2000.

[5]. T. Helmy, and A. Dekdouk, "Burst Round Robin: As a Proportional-Share Scheduling Algorithm", IEEE, Proceedings of the fourth IEEE-GCC Conference on towards Techno-Industrial Innovations, pp. 424-428, 2007.

[6]. J. Nieh, Ch. Vaill, and H. Zhong, "Virtual-Time Round-Robin: An O(1) Proportional Share Scheduler." Proceeding of the 2001 USENIX Annual Technical Conference, USA, 2001.

[7]. D. Nayak, S. K. Malla, and D. Debadarshini, "Improved Round Robin Scheduling using Dynamic Time Quantum", International Journal of Computer Applications, Vol. 38, No. 5, 2012.

[8]. M. U. Siregar, "A New Approach to CPU Scheduling Algorithm: Genetic Round Robin", International Journal of Computer Applications, Vol. 47, No. 19, 2012.

[9]. Collins. B (2001) "An Approach to Scheduling Task Graph with Contention inCommunication";ACM, http//www.ait.nrl.navy.mil/pgmt-for web/pdf/an-approach- toscheduling.pdf.

[10]. Oliver Sinnen ,(2007)," Task Scheduling For Parallel Systems", John Wiley , Canada.

[11]. http://www.cse.sc.edu/

[12]. http://prr.hec.gov.pk/Chapters/487S-4.pdf

[13]. http://www.arabteam2000forum.c

[14]. http://www.cs.nyu.edu/courses/spring02/ V22.0202-001/lectures/lect9.pdf

[15]. http://dspace.mak.ac.ug/bitstream/123456789/527/3/lakuma-joel-cit-masters-report.pdf