IMPROVED DYNAMIC ROUND ROBIN PROCESS SCHEDULING

Sumit Dalal[1], Vikram Singh[2]

[1]M.Tech Scholar, [2]Professor, Department of Computer Science,
Ch. Devi Lal University, Sirsa,(India)

ABSTRACT

Process Scheduling is one of the fundamental concepts of operating system. Round robin process scheduling algorithm is optimal process scheduling algorithm in time sharing systems. The efficiency of the processor depends on the choice of time slice in time sharing systems. The time quantum taken in round robin algorithm is static due to which throughput of the processor decreases. In this paper, selection of time quantum is discussed and a new process scheduling algorithm for time sharing systems named as improved dynamic round robin algorithm is proposed. Proposed algorithm incorporates advantages of both "round robin" and "shortest job first" process scheduling algorithms which tends to reduce chances of starvation and increases throughput of the system. Round robin process scheduling algorithm has high process switch rates and accordingly average waiting time, average turnaround time and system throughput are adversely affected. This situation can be improved by using proposed process scheduling algorithm. In this research, analysis of number of process switches, the average waiting time and the average turnaround time of processes in round robin process scheduling algorithm, "efficient dynamic round robin" and proposed "improved dynamic round robin" process scheduling algorithm has been done.

Keyword: Process Switch, Process Scheduling, Round Robin, Shortest Job First, Efficient Dynamic Round Robin and Improved Dynamic Round Robin.

I. NTRODUCTION

The processor being the most precious resource of the computer system, should be utilised efficiently. Apart from this, processes must get justified treatment in getting their turn at processor. Here, the role of process scheduling comes in to picture. Sharing of computer resources between multiple processes along the time coordinate is called process scheduling. Ajit (2010) describes criteria for selection of a process:

1. **Process switch:** A process switch is earmarked by transfer of control from one process to another. This involves storing the status of outgoing process and preparing and loading the context of incoming process. So, a process switch incurs processing overhead and goal of a process scheduling policy remains to minimize the count of process switches.
2. **Throughput:** Throughput of a computer system is described as number of processes completed in a unit time. Throughput of a system, among other factors, depends upon number of process switches and varies inversely with it.

3. **Processor Utilisation:** It is defined as the fraction of time processor is in use. Usually, maximization of processor utilisation is a major goal of the process scheduling.
4. **Waiting Time:** Waiting time is the total time a process has to wait in ready queue. The waiting time is not the measurement of time when a process executes or does I/O completion.
5. **Turnaround Time:** Turnaround time is the total time which a process takes to complete – waiting and executing.
6. **Response Time:** Response time of a process is the time elapsed between entering of the last character of the input and appearance of first character of the output. It is applicable to interactive processes.

II. RELIMINARIES

The various process scheduling algorithms are: -

1. First come first serve scheduling: It is the simplest process scheduling algorithm. Herein, the processor is allocated to the processes in the order of their arrival in the ready queue.
2. Shortest job first scheduling: In this scheduling the process with the shortest CPU burst time is allocate the processor first. This requires advance knowledge or estimations about processes' burst times.
3. Priority scheduling: Here processes are assigned numerical priority levels and at a given time the process with the highest priority is assigned the processor.
4. Round robin scheduling: Used in timesharing systems, this scheduling policy is characterized by division of processor time in small equal time slices and allocating these processor-time slices to ready state processes on equitable and round robin basis.

III. ELATED WORK

Efforts have been made to modify RR in order to give better average turnaround time, average waiting time and minimize context switches. The main disadvantages of RR are static time quantum, larger waiting time and response time, large number of context switches, low throughput. Many research works has been done to improve the performance of the RR scheduling algorithm. Rami J. (2009) has proposed a new approach - SAAR algorithm – which uses dynamic time quantum which is repeatedly adjust by the burst time of running processes. SRBRR algorithm proposed by Mohanty & Behera (2010) uses median for dynamic time quantum. Varma (2012) proposed a new approach – called ISRBRR- for time quantum i.e. *ceil(sqrt(median * highest_burst_time))*. Mittal & Raman (2014) proposed EDRR, which uses a new formula for calculating time quantum. In this paper, a comparison of RR, EDRR and IDRR – the newly proposed algorithm has been done.

IV. ROPOSED IDRR ALGORITHM

For the proposed algorithm, input process load is arranged in a queue in ascending order of processes' burst times (*CPUBurstTime*). The algorithm starts with a non-null queue of ready processes (*ReadyQueue*) and picks up the process at the queue-front. The time quantum (*TimeQuantum*) of 1^{st} batch of processes is set equal to burst time of shortest process. After completion of one round of processes *TimeQuantum* is updated as follows:

$$Mean = \frac{\sum_{i=0}^{n} CPU\ Burst\ Time}{n-1} \qquad \text{(i)}$$

If $\frac{no\ of\ processes}{2} \neq 0$ **Then**

Median = *CPUBurstTime* of middle process

Else $Median = \frac{\sum_{i=n/2}^{\frac{n}{2}+1} CPU\ Burst\ Time}{2} \qquad \text{(ii)}$

Time Quantum = Ceiling (Sqrt ((*Mean* * Max(*CPUBurstTime*) + (*Median* * Min(*CPUBurstTime*))) (iii)

4.1 Proposed scheduling algorithm

1. Arrange processes in order of increasing *CPUBurstTime*.

2. Set *TimeQuantum* = *CPUBurstTime* of 1st process.

3. While (*ReadyQueue* != NULL)

a) Allocate processor to each process in *ReadyQueue* on round robin basis.

b) Update *ReadyQueue* - If a new process arrives, arrange *ReadyQueue* in ascending order of *CPUBurstTime* including processes that did not complete in last round.

c) Calculate *Mean* using equation **(i)**

d) Set the *TimeQuantum* using equation **(iii)**

4. Stop

4.2 Flowchart

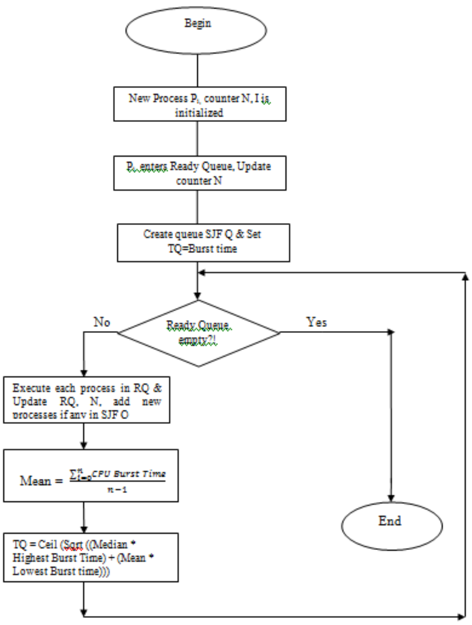

Figure A. Flowchart of IDRR scheduling

V. ILLUSTRATIONS & RESULTS

For the purpose of illustration of the algorithm, let us consider following seven processes:

Process Name	Burst Time (ms)	Arrival Time
A	20	0
B	70	0
C	10	0
D	45	10
E	20	20
F	15	30
G	42	30

Figure B. Input Table

At T=0, three processes have arrived in the ready queue. Now as per algorithm *TimeQuantum* for first round of execution is equal to the burst time of shortest job i.e. 10ms. After running all three processes (C, A, B in that order) for 10ms each, ready queue is updated. By this time (i.e. at 30ms) four new processes (D, E, F, G) have arrived in the ready queue. Processes are arranged in order of their increasing *CPUBurstTime*. Now, *Time Quantum* for next round is calculated using (iii) above.

A. According to simple Round Robin (RR) algorithm (*TimeQuantum*=30ms), there are 10 no. of process switches.

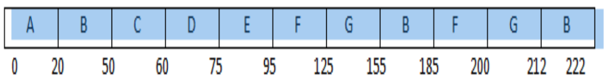

Figure C. Simple round robin Gantt chart

B. According to Efficient Dynamic Round Robin (EDRR) algorithm, there are 9 no. of process switches.

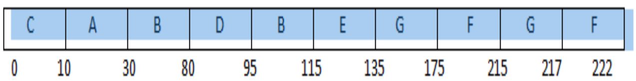

Figure D. EDRR Gant chart

C. According to proposed algorithm Improved Dynamic Round Robin (IDRR), there are 8 no. of process switches.

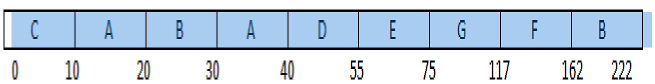

Figure E. IDRR Gantt chart.

	No. of Context Switches	Avg. Waiting Time	Avg. Turnaround Time
RR	10	94.471	126.2857
EDRR	9	73	117.714
IDRR	8	52.714	94.4

Figure F. Result table (no. of switches, avg. waiting time and avg. turnaround time)

VI. CONCLUSION

The paper introduces and proposes a new process scheduling algorithm. Comparison of various algorithms i.e. round robin (RR), efficient dynamic round robin (EDRR) and the proposed algorithm improved dynamic round robin (IDRR) has been done. It is concluded that the proposed algorithm is more efficient than RR and EDRR as it has reported shorter average waiting time, shorter average turnaround time and lesser number of process switches as compared to round robin, so it decreases the operating system' overhead. The proposed algorithm is the combination of the shortest job first CPU scheduling algorithm and the round robin CPU scheduling algorithm with efficient & dynamic time quantum.

REFERENCES

[1]. Mittal, P.K. & Raman (2014), "An Efficient Dynamic Round Robin CPU Scheduling Algorithm," International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4, Issue 5, May 2014, ISSN: 2277-128X.

[2]. Abdulrazaq, A., Salisu, A., Ahmad, M.M., and Saleh, E.A. (2014), "An Additional Improvement in Round Robin (AAIRR) CPU Scheduling Algorithm," International Journal of Advanced Research in Computer Science and Software Engineering , 4 (2), 601-610.

[3]. Abdullahi, I and Junaidu, S. B. (2013), "Empirical Framework to Migrate Problems in Longer Job First Scheduling Algorithm," International Journal of Computer Applications, 75 (14), 9-14

[4]. Arora, H. (2013), "An Improved CPU Scheduling Algorithm," International Journal of Applied Information Systems, Volume 6, No. 6, December 2013 – ISSN: 2249-0868.

[5]. Rajput, I.S. (2012), "A Priority based Round Robin CPU Scheduling Algorithm for Real Time Systems", International Journal of Innovations in Engineering and Technology, Vol. 1 Issue 3 Oct 2012.

[6]. Abbas, N., Ali, K., and Seifedine, K. (2011), "A New Round Robin Based Scheduling Algorithm for Operating Systems: Dynamic Quantum Using the Mean Average," International Journal of Computer Science Issues, 8 (4), 224-229.

[7]. Mohanty, R., Behera, H.S., Patwari, K. (2011), "Priority Based Dynamic Round Robin (PBDRR) Algorithm with Intelligent Time Slice for Soft Real Time Systems, International Journal of Advanced Computer Science and Applications, Vol. 2, No.2, February 2011.

[8]. Yadav, R.K., Mishra, A.K., Prakash, N. and Sharma, H. (2010), "An Improved Round Robin Scheduling Algorithm for CPU Scheduling", (IJCSE) International Journal on Computer Science and Engineering, Vol. 02, No. 04, 1064-1066, 2010.

[9]. Ajit, S, Priyanka, G and Sahil, B. (2010), "An Optimized Round Robin Scheduling Algorithm for CPU Scheduling," International Journal on Computer Science and Engineering , 2 (7), 2382-2385.

[10]. Rami J.M. (2009), "Self-Adjustment Time Quantum in Round Robin Algorithm Depending on Burst Time of Now Running Processes", American J. of Applied Sciences 6(10):1831-1837, 2009.