Volume 2, Issue 6, June 2012 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

A New Proposed Precedence based Round Robin with Dynamic Time Quantum (PRRDTQ) Scheduling Algorithm For Soft Real Time Systems

H. S. Behera
Computer Science and Engineering
VSSUT, Burla, Odisha,India.
hsbehera_india@yahoo.com

Brajendra Kumar Swain
Computer Science and Engineering
VSSUT, Burla, Odisha,India.
brajendra.swain@gmail.com

Abstract— **Scheduling is a fundamental function of operating system. Almost all computer resources are scheduled before use. Round Robin (RR) performs optimally in timeshared system but it is not suitable for real time system because it gives more number of context switches, larger waiting and turnaround time. In this paper, we have proposed a new variant of Round Robin scheduling algorithm, known as Precedence based Round Robin with Dynamic Time Quantum (PRRDTQ) is suitable for soft real time system. This algorithm gives precedence to all processes according to their priority and burst time, then applies the Round Robin algorithm on it. Proposed algorithm is developed by taking dynamic mean time quantum in to account. Our experimental result shows that our proposed algorithm performs better than algorithm in MRR [1] and PBDRR [2] in terms of reducing the number of context switches, average waiting time and average turnaround time.**

Keywords— **Context Switch, Real Time Operating System, Round Robin Algorithm, Scheduling, Turnaround Time, Waiting Time.**

I. INTRODUCTION

Modern Operating Systems are moving towards multitasking environments which mainly depends on the CPU scheduling algorithm since the CPU is the most effective or essential part of the computer. Scheduling refers to the way processes are assigned to run on the available CPUs, since there are typically many more processes running than available CPUs. Round Robin is considered the most widely used CPU scheduling algorithm [7, 8], also used for flow passing scheduling through a network device [9].

Real Time Operating Systems (RTOS) are the ones that are designed to provide results within a specific time-frame. It must have well defined fixed and response time constraints and the processing must be done within the defined constraints or the system will fail. RTOS are basically divided into three types: hard, firm and soft. In hard real time systems, failure to meet

Deadline or response time constraints leads to system failure. In firm real time systems, failure to meet deadline can be tolerated. In soft real time systems, failure to meet deadline doesn't lead to system failure, but only performance is degraded [4]. There are many different scheduling algorithms which vary in efficiency according to the environment. The Criteria for a good scheduling algorithm depends, on the following measures:

- Fairness: all processes get fair share of the CPU,
- Efficiency: keep CPU busy 100% of time,
- Response time: minimize response time,
- Turnaround: minimize the time batch users must wait for output,
- Throughput: maximize number of jobs per hour.

Round Robin (RR) which is the main concern of this research is one of the oldest, simplest and fairest and most widely used scheduling algorithms, designed especially for time-sharing systems. RR is similar to FCFS except that preemption is added to processes [7, 8].

First-Come-First-Served (FCFS) is the simplest scheduling algorithm, it simply queues processes in the order that they arrive in the ready queue. Processes are dispatched according to their arrival time on the ready queue. Being a non-preemptive discipline, once a process has a CPU, it runs to completion. The FCFS scheduling is fair in the formal sense or human sense of fairness but it is unfair in the sense that long jobs make short jobs wait and unimportant jobs make important jobs wait [7, 8] .

Shortest Job First (SJF) is the strategy of arranging processes with the least estimated processing time remaining to be next in the queue. It works under the two schemes (preemptive and non-preemptive). It's provably optimal since it minimizes the average turnaround time and the average waiting time. The main problem with this discipline is the necessity of the previous knowledge about the time required for a process to complete. Also, it undergoes a starvation issue especially in a busy system with many small processes being run [7, 8].

II. RELATED WORK

C.Yashuwaanth proposed a Modified RR(MRR) algorithm which overcomes the limitations of simple RR and is suitable for the soft real time systems [1].

In real time systems, the rate monotonic algorithm is the optimal fixed priority scheduling algorithm whereas the earliest-deadline-first and minimum-laxity-first algorithms

are the optimal dynamic priorities scheduling algorithms as presented by Rakesh Mohanty, H. S. Behera in PBDRR [2] and Liu and Laylandin their paper [3]. S. Baskiyarand N. Meghanathan have presented a survey on contemporary Real Time Operating System (RTOS) which includes parameters necessary for designing a RTOS, its desirable features and basic requirements[4]. A dynamically reconfigurable system can change in time without the need to halt the system. David B. Stewart and PradeepK.Khosla proposed the maximum-urgency-first algorithm, which can be used to predictably schedule dynamically changing systems [5]. The scheduling mechanism of the maximum-urgency-first may cause a critical task to fail. The modified maximum urgency first scheduling algorithm by Vahid Salmani, SamanTaghaviZargar, and Mahmoud Naghibzadeh resolves the above mentioned problem [6].

III. PROPOSED APPROACH

. Our proposed algorithm calculates a new factor, called Precedence Factor (PFi) by combining the basic factors of the process i.e. priority and burst time.

$$PFi = (priority \times priority\_ratio) + (burst\ time \times burst\_ratio)$$

Where

Priority Ratio= Priority of the Process(Pi) / Total no of Processes

And

Burst Ratio= Burst time of the Process(BTi) / Σ Burst Times of the Processes

Then it assigns precedence to processes according to their factors. The process having lesser factor gets higher precedence. The processes are shorted according to the new precedence and round robin algorithm is applied with dynamic time quantum. The dynamic time quantum is computed by taking the mean of the remaining burst times. Processes with shorter burst time and higher priority is executed first resulting in better turnaround time and better waiting time.

IV. PROPOSED ALGORITHM (PSEUDO CODE)

1. I/P : Process(Pn), Burst Time(BTi), priority(Pi), Ready Queue(RQ) ;

   O/P: Context Switch (CS), Average Waiting Time(Awt), Average Turnaround Time(Atat).

2. Initialize Ready Queue=0,CS=0,Awt =0, Atat=0, Quantum Time (QT=0), n=number of processes;

3. /*initialization*/

4. For(i=1to n)
   {

   Priority_ratio(PRi)= 0;

   Burst_ratio(BRi)= 0;

   Remaning Burst Time(RBTi)= 0;

   Precedence Factor(PFi)=0;
   }

5. For(i=1 to n)
   {

   $PRi = \frac{\text{Priority of the Process(Pi)}}{Total\ no\ of\ Processes}$;

   $BRi = \frac{\text{Burst time of the Process(Pi)}}{\sum Burst\ Times\ of\ the\ Processes}$;

   RBTi= BTi;
   }

6. For(i=1 to n)
   {

   $PFi = (Pi \times PRi) + (BTi \times BRi)$;

   }

7. Processes are shorted and ready queue is filled according to the precedence factor(PFi);

8. While (Ready Queue!= NULL)

   $QT = \frac{\sum_1^n RBTi}{n}$;

9. Assign the CPU to the process on front of the Ready Queue ;

10. If ($RBTi \leq QT$)
    {

    Process(Pi) is completed and removed from the Ready Queue;

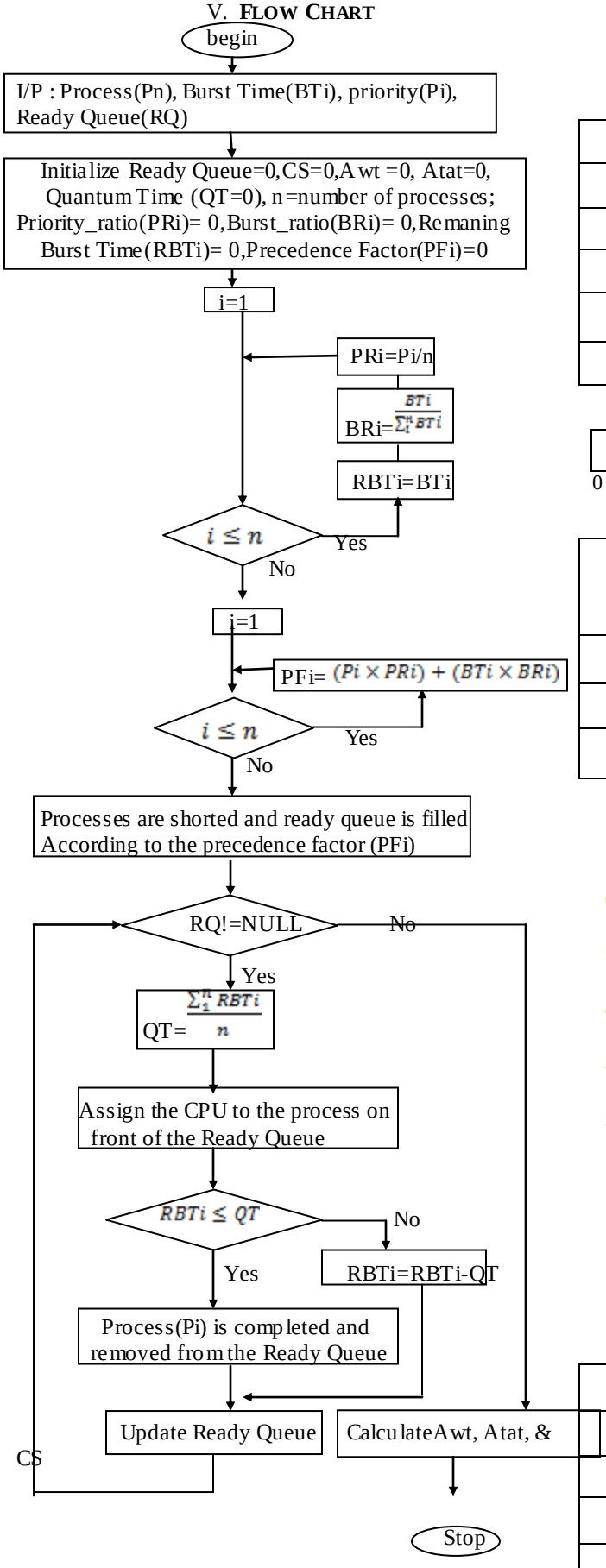
    }
    else
    {
    RBTi=BTi-QT;

    process is added to the tail of the RQ;
    }

    end while;

11. Calculate Awt, Atat, & CS;

V. FLOW CHART



VI. EXPERIMENTAL ANALYSIS

Case-1(Increasing Burst Time)

TABLE I
Data for processes in increasing order

Pi	BT	Priority	PR×P	BT×BR	PFi
1	5	2	0.8	2.3	3.1
2	12	3	1.8	1.8	3.6
3	16	1	0.2	3.2	3.4
4	21	4	3.2	5.67	8.87
5	23	5	5.0	6.9	11.9

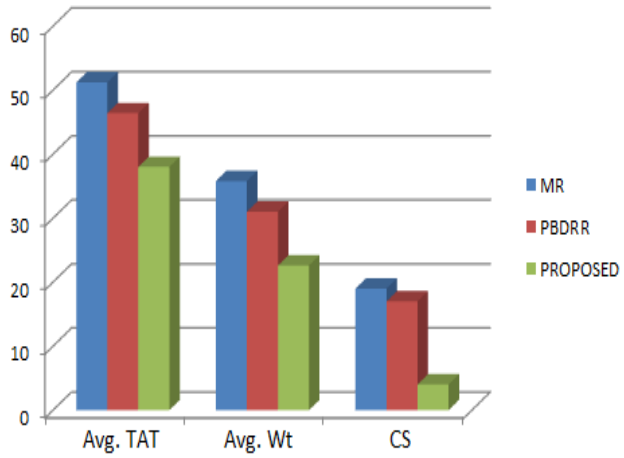
TABLE III
Gantt chart

P1	P3	P2	P4	P5
0	5	21	33	54	77

TABLE III
Comparison Table

ALGORITHM	PARAMETERS			
	Avg. TAT	Avg. WT	CS	Overhead
MRR	51.2	35.8	19	High
PBDRR	46.4	31	17	Average
PRRDTQ	38	22.6	4	Low

Graph-I:
Comparison of Performance of MRR, PBDRR and PROPOSED



CASE-2(Decreasing Burst Time)

TABLE IIIV
Data for Processes in Decreasing Order

Pi	BT	Priority	PR×P	BT×BR	PFi
1	31	2	0.8	12.4	13.2
2	23	1	0.2	6.44	6.64
3	16	4	3.2	3.2	6.4
4	9	5	5	0.54	5.54

5	1	3	1.8	0.03	1.83

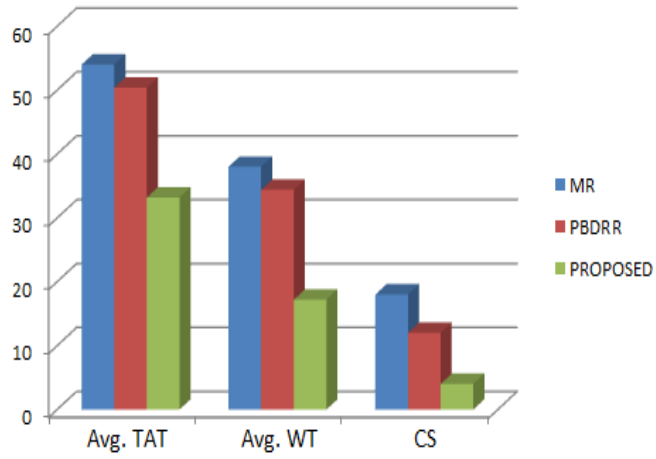
TABLE V
Gantt Chart

P5	P4	P3	P2	P1
0	1	10	26	49 80

TABLE VI
Comparison Table

ALGORITHM	PARAMETERS			
	Avg. TAT	Avg. WT	CS	Overhead
MR	54	38	18	High
PBDRR	50.4	34.4	12	Average
PRRDTQ	33.2	17.2	4	Low

Graph-II
Comparison of Performance of MRR, PBDRR And PROPOSED



CASE-3 (Random Burst Time)

TABLE VII
Data for Processes in Random Order

Pi	BT	Priority	PR×P	BT×BR	PFi
1	11	2	0.8	0.88	1.68
2	53	1	0.2	21.2	21.4
3	08	4	3.2	0.48	3.68
4	41	5	5	12.3	17.3
5	20	3	1.8	3	2.1

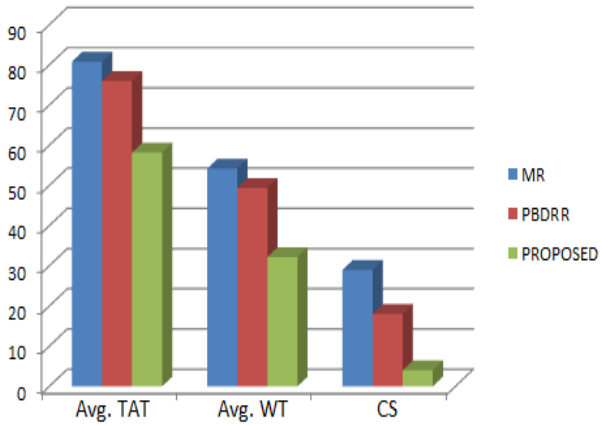
TABLE VIII
Gantt Chart

P1	P3	P2	P4	P5
0	11	31	39	80 133

TABLE IVX
Comparison Table

ALGORITHM	PARAMETERS			
	Avg. TAT	Avg. WT	CS	Overhead
MRR	80.8	54.2	29	High
PBDRR	76	49.4	18	Average
PRRDTQ	58.2	32.2	4	Low

Graph-III
Comparison of Performance of MRR, PBDRR And PROPOSED



VII. CONCLUSION

From the above comparisons, we observed that our new proposed PRRDTQ is performing better than the algorithms proposed in paper MRR[1] and PBDRR[2] in terms of average waiting time, average turnaround time and number of context switches thereby reducing the overhead and saving of memory spaces.

REFERENCES

[1] C. Yaashuwanth and R. Ramesh, : A New Scheduling Algorithm for Real Time System, International Journal of Computer and Electrical Engineering (IJCEE), Vol. 2, No. 6,pp 1104-1106, December, 2010.

[2] Rakesh Mohanty, H. S. Behera, Khusbu Patwari, Monisha Dash, M. Lakshmi Prasanna "Priority Based Dynamic Round Robin (PBDRR) Algorithm with Intelligent Time Slice for Soft Real Time Systems", (IJACSA)International Journal of Advanced Computer Science and Applications, Vol. 2, No.2, February 2011.

[3] C. L. Liu and James W. Layland : Scheduling Algorithmsfor Multiprogramming in a Hard-Real-Time Environment, Journal of the ACM(JACM), Vol. 20, Issue 1, January, 1973.

[4] S. Baskiyar and N. Meghanathan: A Survey On Contemporary Real Time Operating Systems, Informatica, 29,pp 233-240, 2005.

[5] David B. Stewart and Pradeep K. Khosla: Real-Time Scheduling of Dynamically Reconfigurable Systems,

Proceedings of the IEEE International Conference on Systems Engineering, pp 139-142, August, 1991.

[6] Vahid Salmani, Saman TaghaviZargar, and Mahmoud Naghibzadeh: A Modified Maximum Urgency First Scheduling Algorithm for Real-Time Tasks, World Academy of Science, Engineering and Technology. Vol. 9, Issue 4, pp 19-23, 2005.

[7] Silberschatz, Galvin and Gagne, Operating systems concepts, 8th edition, Wiley, 2009.

[8] Lingyun Yang, Jennifer M. Schopf and Ian Foster, "Conservative Scheduling: Using predictive variance to improve scheduling decisions in Dynamic Environments", Super Computing 2003, November 15-21, Phoenix, AZ, USA.

[9] Weiming Tong, Jing Zhao, "Quantum Varying Deficit Round Robin Scheduling Over Priority Queues", International Conference on Computational Intelligence and Security. pp. 252- 256, China, 2007.

