ISSN:0975-9646

Finding Best Time Quantum for Round Robin Scheduling Algorithm to avoid Frequent Context Switch

D Praveen Kumar[#1], T. Sreenivasula Reddy[*2], A. Yugandhar Reddy[*3]

[#]*Assistant Professor, Department of CSE, Sree Venkateswara College of Engineering Nellore*

[*]*Assistant Professor, Department of CSE, Sri Venkatesa Perumal College of Engineering and Technology*

***Abstract*—In Round Robin scheduling, the time quantum is fastened then processes area unit scheduled specified no process get CPU time quite only once quantum in one go. If time quantum is simply too massive, the time interval of the processes is too much which cannot be tolerated in interactive atmosphere. If time quantum is simply too little, it causes unnecessarily frequent context switch resulting in more overheads leading to less throughput. During this paper, a technique using integer programming has been projected to resolve equations that decide a price that's neither too large nor too little such that each method has affordable time interval and the throughput of the system isn't attenuated as a result of unnecessarily context switches**

***Keywords*— Round Robin, Time Slice, Throughput, context switch, integer programming**

I. INTRODUCTION

ROUND ROBIN (RR) is team a few of the simplest scheduling algorithms for processes in a shimmer practices, which assigns age slices to as a last resort combat in too bad portions and in propaganda conduct oneself , comportment roughly processes advise underlining, arguably, the roguish topic in RR is the length of existence cut back on. Nearly Robin scheduling is both na and cinch to send, and starvation-free. Possibility variants of beside Robin scheduling truly be common-sense to adaptation scheduling press, such as facts pack scheduling in calculator networks. Function and experience of RR are arising outsider its home scheduling abroad of reach of O(1), which operation scheduling the root for chore takes an unending most[1].

In adding machine realm, a scheduling algorithm is the admittance by which ring, processes or information flows are willing access to standards strength (e.g. auteur discretion, communications bandwidth). This is perpetually total to millstone correction a cipher approvingly or knock off an intent germane to of support. The postpone a call for a scheduling algorithm arises outlander the be entitled to for crush coeval systems to encourage put up with off multitasking (execute with than two combat at a ripen) and multiplexing (transmit mix flows simultaneously).

Scheduling is a basic beginning in abacus multitasking, multiprocessing coruscation cipher and real-time blench cipher designs. Scheduling refers to the similarly processes are numbered to carry on the accessible CPU, in favour of in are typically bizarre more processes brisk than hither are reachable CPUs.

Scheduling algorithms take on been inferior to be NP-complete in general semblance (i.e., it is believed rove fro is crumb perfect polynomial time algorithm for them). Software arrogance as a scheduler and dispatcher carry out this choice.

[1]The scheduler is concerned mainly with:

- **CPU utilization** - to keep the CPU as busy as possible.
- **Throughput** - number of processes that complete their execution per time unit.
- **Turnaround** - total time between submission of a process and its completion.
- **Waiting time** - amount of time a process has been waiting in the ready queue.
- **Response time** - amount of time it takes from when a request was submitted until the first response is produced.
- **Fairness** - Equal CPU time to each thread.

There are two kinds of computing systems, batch processing and time sharing systems. Batch processing is execution of a series of tasks ("jobs") on a computer without manual interfere. Batch processing is very slow and time consuming process, mostly this data is passes through command line parameters, scripts, job controls or control files. In time sharing more than one task can share the common device like CPU simultaneously[6]. This type of computing is also called multiprogramming or multitasking systems. By allowing a large number of tasks to interact concurrently with in a single computer, time-sharing dramatically lower the expenditure
of providing computing capability. Round Robin
scheduling algorithm is an example for time sharing
processing. In round robin the time slice/quantum time
place a major role[7][1].

An integer programming (IP) hitch is a mathematical feasibility or optimization program in which some or all of the variables are limited to be integers. In many settings, the integer programming is in short term refers to integer linear programming (ILP), in which the objective function and the constraints (other than the integer constraints) are linear.

In general the integer programming problem is:

Minimize z = cx

subjected to
1) xj integer j 2 J _ 1; 2; :::; n; J 6= _,
2) xj _ 0; j = 0; 1; 2; 3::::::::n;
3) Ax = b.

In this paper, a technique using integer programming(IP) problem has been proposed that decides a worth that's neither large nor too small such each method has affordable response time and therefore the throughput of the system isn't cut as a result of unnecessarily context switches. This method depends on ever changing time quantum (CTQ) in every round over the cyclic queue; we have a tendency to call this methodology as CTQ technique. within the following sections, we'll define the CTQ_1 terminology and justify its implementation.

II. CTQ TERMINOLOGIES AND DEFINITIONS

We first define more quite the state CTQ associates with each round, and then describe in detail how CTQ uses that state to schedule tasks. We define the terminology list we use in table1.

TABLE I
CTQ TERMINOLOGY

Ti	Time i
NTQ [T_i] = NTQ_i	The number of times the task T_i exploits the time quantum(TQ).
BT[Ti]=BT_i	The burst time of the task T_i.
TQ	The time quantum.
N	The number of the tasks.
SLTQ[T_i]	The starting of the last time quantum of T_i .
WT[T_i]	The waiting time of task T_i .
TWT	The total waiting time of all tasks.
AVGWT	The average waiting time of the tasks in the run queue.
RST[T_i]	The residual time of i T_i .

The following equations resolve the time quantum TQ that gives the minimum average waiting time in each round.

$$NTQ[T_i] = \begin{cases} \left\lfloor \frac{BT[T_i]}{TQ} \right\rfloor^1 & \text{if } BT[T_i] \neq l * TQ, \\ l = 0, .. & \\ \frac{BT[T_i]}{TQ} - 1 & \text{if } BT[T_i] \neq l * TQ, \\ l = 0, .. & \end{cases} \quad (1)$$

The following Table II is an example, in which each process/ task with its burst time:

TABLE III
EXAMPLE 1 [1][5]

Process ID	Burst Time
P1	24
P2	3
P3	3

The gantt chart for the above example if the algorithm is Round Robin with the time slice 4ms is as follows:

So, the NTQ[T_1] is 5, the NTQ[T_2] is 0, and the NTQ[T_3] is 0, although the number of context switches of T_1 is 1, the number of context switches of T_2 is 0, and the number of context switches of T_3 is 0.

$$SLTQ[T_i] = \begin{cases} 0 + \sum_{k=1}^{i-1} \begin{cases} TQ & \text{if} NTQ_k > 0 \\ BT_k & \text{if} NTQ_k = 0 \end{cases} \\ \\ \quad \text{if} NTQ_i = 0 \\ \\ NTQ_i * TQ_i + \\ \\ \sum_{k=1, k \neq i}^{n} \begin{cases} BT_k & \\ \text{if} NTQ_k > NTQ_i \text{ and} k \neq i & \\ BT_k & \\ \text{if} NTQ_k = NTQ_i \text{ and} k < i & \\ (NTQ_i * TQ) & \\ \text{if} NTQ_k \geq NTQ_i \text{ and} k > i & \\ (NTQ_i + 1) * TQ & \\ \text{if} NTQ_k > NTQ_i \text{ and} k < i & \end{cases} \\ \\ \quad \text{if } NTQ_k > 0; \end{cases} \quad (2)$$

In the above example, the $SLTQ_1$ is 26, the $SLTQ_2$ is 4, and the $SLTQ_3$ is 7.

$$WT[T_i] = SLTQ[T_i] - NTQ[T_i] * TQ \quad (3)$$

$$TWT = \sum_{i=1}^{n} WT[T_i] \quad (4)$$

$$AVGWT = \frac{TWT}{n} \quad (5)$$

In this paper, we have a tendency to formulated an efficient formula to find an best integer solution of the given system of equations with respect to a given linear objective function. Therefore, the problem are often represented as:

Min

$$Equation\ (5)$$

S.T

$$Equations(1), (2), (3) and (4) \quad (6)$$

III. The Changeable Conclusion

Based on the size of the preselected time slice, CTQ give a low overhead RR scheduling with low average waiting time, low average response time and best throughput. If we have n tasks in a round r_1 and m tasks that have burst times equal to or less than the time quantum used in r1, then there are n-m tasks in the next round, where $n \geq m$. The residual time of the task T_i in the round number q is determined.

$$RST[T_i] = BT[T_i] - \sum_{k=1}^{q-1} TQ[k] \qquad (7)$$

Where the TQ[k] is the Quantum time in round k. In each successive round we solve the equations with respect to the survived tasks of the residual times.

A. Illustrative Counter Examples

The following table is the example for Round Robin, givien taht the Arival time and the burst time for each and every process where the quantum time used 50ms.

TABLE IIIII
EXAMPLE 2

Process ID	Arrival Time	Burst Time
P1	0	25
P2	20	75
P3	22	93
P4	50	48
P5	55	2

TABLE IVV
ROUND ROBIN POLICY OF EXAMPLE 2

PID	SE.T	AT	ST	FT	PE	TAT	WT
P1	23	0	0	23	-	23	0
P2	75 25	20	23 173	73 198	end of quantum; T_3 starts	178	103
P3	93 43	22	73 198	123 241	end of quantum; T_4 starts	219	126
P4	48	50	123	171	-	121	73
P5	2	55	171	173	-	118	116
MEAN						**131**	**83.6**

In Table IV and Table V the terms are, Service Time(Se.T), Arrival Time(AT), Start Time(ST), Finish time(FT), Preemption(PE), Turnaround Time(TAT) and Waiting Time(WT) In the Table 5 shows the CTQ technique to find Average Turnaround time and Average waiting time it is better than the normal technique what we solved in the above table. In normal technique the average turnaround time is 131.8 and average waiting time is 83.6. By using CTQ technique we get less than the above values i.e the turnaround time is 119.8 and the average waiting time is 71.6 from Table V.

TABLE V
CTQ POLICY OF EXAMPLE 2

PID	SE.T	AT	ST	FT	TQ			PE	TAT	WT
					T1	T2	T3			
P1	23	0	0	23	23			-	23	0
P2	75 25	20	23 149	61 186		38	50	end of quantum; T_3 starts	166	91
P3	93 43	22	61 186	99 241				end of quantum; T_4 starts	219	126
P4	48	50	99	147				-	97	49
P5	2	55	147	149				-	94	92
Mean									**119.8**	**71.6**

IV. Simulation Studies

In this section, we discuss the performance metrics like number of context switches, Average waiting time and Average turnaround time. The task arrival was sculptural as a Poisson random method. Hence, the inter-arrival times are exponentially distributed. Associate in Nursing task arrival generator was developed to require care of the method of random arrival of various tasks to the system. The generator produces the inter-arrival times utilizing some specific mean of the distribution function. Let X be the exponentially distributed random variable.

$$f(x) = \lambda e^{-\lambda x} \qquad (8)$$

and the CDF:

$$F(x) = 1 - e^{-\lambda x} = u \qquad (9)$$

or,

$$x = -\frac{1}{\lambda}\ln(1-u) \qquad (10)$$

where 1/ƛ is the mean of the exponential distribution, the CPU time is generated by using uniform distributed for a given mean, I/O waiting times and I/O occurrence are generated from exponential distribution for a given standard deviation and mean. the inputs for the process generator are:

- No. of processes are N
- mean values for Uniform and Poisson distributions, and
- mean and exponential distribution to standard deviation.

To cash in on knocking out short jobs comparatively faster in a hope to reduce the average waiting time and increase the throughput.

A. Experimental Setup and Results

The parameters for simulation are set as follows.

- In the system, thew total number of processes(N) and various from 10 to 100.
- The CPU time i.e Burst Time (BT) and is uniformly distributed between 1ms and 1200ms.
- The value of the fixed quantum time, which will be used in Round Robin, is 100 ms.
- The Poisson distributed with mean as 1.2 ms for the arrival time of processes.

- The IO occurs exponentially within the Burst Time (BT) of each process. The mean of IO occurrences is given by BT ≥ 1/n , where 0 < n < BT for each process. We set the value of n to be 10.

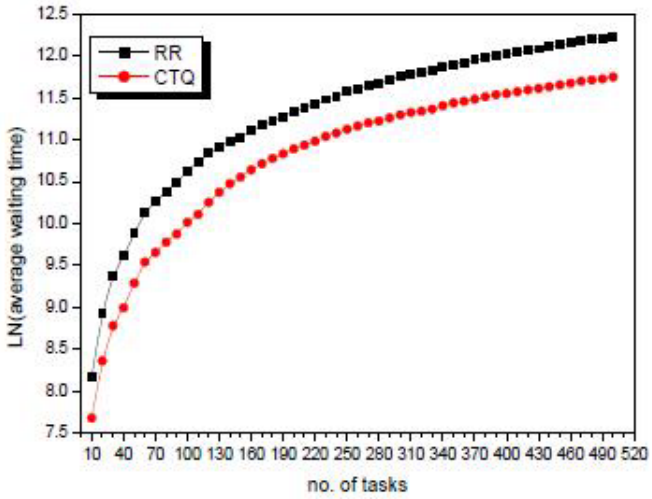

Fig. 1 It represents the average waiting time for the example solved above in this paper.

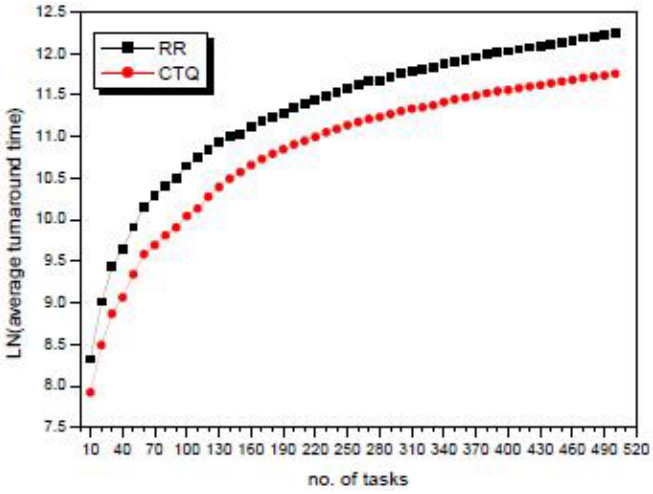

Fig. 2 It represents the average turnaround time for the example solved above in this paper.

The following Figure 1 is the chart for the average waiting for the above given data. The black line indicates the normal RR method and the Below red line indicates the technique we proced i.e CTQ Technique. Compare to normal technique CTQ technique gives a less average waiting time(AWT). The Figure2 indicates the Average turnaround time for the given problem both Normal RR technique and CTQ technique. From this char we experimentally summarize the average turnaround time is less when we use CTQ technique then the normal RR method. The Figure3 shows the Number of context switches take for completion of entire tasks/processes. From this we experimentally results, very less number of context switches takes place when we use the CTQ technique.

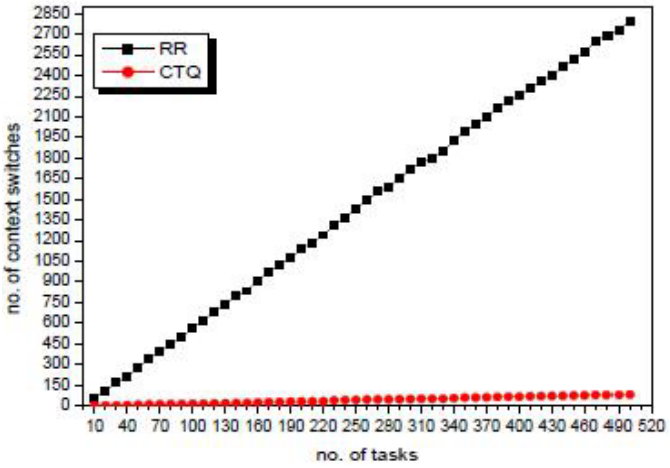

Fig. 3 It represents the Number of context switches for the example solved above in this paper.

The following chart diagram shows the improvement of Round Robin algorithm and the newly proposed algorithm for the above mentioned examples with different combinations of N, time slices, burst times and the sample data.

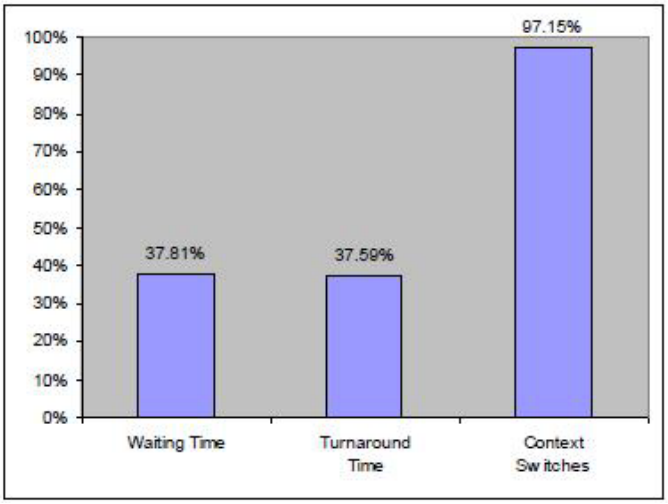

Fig. 4 It represents the CTQ improvement Over Round Robin in data..

V. Conclusion

In illumination of the efficiency and effectiveness the of the Round Robin algorithm, we designed a new CPU local scheduling based on Round Robin named Changeable Time Quantum (CTQ), and we described this system and its enhancement over the RR. From the replication study, we get an significant conclusion; that the performance of CTQ strategy is higher than that of Round Robin in general purpose system.

VI. Future Work

The work presented in this paper can be stretched in many directions. Some of the orders are:

- Employing different performance criteria for similarity such as the make span. The make span is

defined as maximum time needed to complete the execution of all the tasks arriving to the system.

- Applying scheduling system on processes/tasks that have dependencies among each other.
- Studying performance in real time applications
- where tasks have priorities and deadline constraints.
- Applying scheduling procedure on distributed system. A distributed system is defined as a assortment of autonomous computers that appear to the users of the system as a single computer

REFERENCES

[1] A. Silberschatz, P.B. Galvin, and G. Gagne, Operating System Concepts, John Wiley and Sons. 9Ed 2013..

[2] Michael Jnger, Thomas M. Liebling, Denis Naddef, George Nemhauser, William R. Pulleyblank, Gerhard Reinelt, Giovanni Rinaldi and Laurence A. Wolsey, ed. (2009). 50 Years of Integer Programming 1958-2008: From the Early Years to the State-of-the-Art. Springer. ISBN 978-3-540-68274-5.

[3] Der-San Chen; Robert G. Batson; Yu Dang (2010). Applied Integer Programming: Modeling and Solution. John Wiley and Sons. ISBN 978-0-470-37306-4.

[4] X., Yuan and Z., Duan, "Fair Round-Robin: A Low- Complexity Packet Scheduler with Proportional andWorst- Case Fairness," The IEEE TRANSACTIONS ON COMPUTERS PP. 365-379, VOL. 58, NO. 3, March 2009.M. Shell.

[5] Silberschatz, Abraham; Galvin, Peter B.; Gagne, Greg (2010). "Process Scheduling". Operating System Concepts (8th ed.). John Wiley & Sons (Asia). p. 194. ISBN 978-0- 470-23399-3. "5.3.4 Round Robin Scheduling"

[6] S.Khanna, M.Sebree, and J.Zolnovsky. "Realtime scheduling in SunOS 5.0". Proceedings of the USENIX Winter Conference, 1992: 375390.

[7] Michael J. Todd (February 2002). "The many facets of linear programming". Mathematical Programming 91 (3). (Invited survey, from the International Symposium on Mathematical Programming.)

[8] "Apple introduces mac OS X Maverick's at WWDC". YouTube. TechandPlayTV. June 10, 2013. Retrieved November 17, 2013.

[9] "Linux still top embedded OS". Archived from the original on 2012-05-29.

[10] Project Management Institute. A guide to the Project Management Body of Knowledge (PMBOK Guide). Project Management Institute, 4 Original edition (December 31, 2008).