Journal of Theoretical and Applied Information Technology
20th March 2015. Vol.73 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

A NEW PROPOSED DYNAMIC DUAL PROCESSOR BASED CPU SCHEDULING ALGORITHM

[1]G.SIVA NAGESWARA RAO, [2]DR.S.V.N. SRINIVASU, [3]DR. N SRINIVASU, [4]DR. O NAGA RAJU

[1]Assoc. Professor, Department of Computer Science and Engineering, K L University,sivanags@kluniversity.in

[2]Professor, Department of Computer Science, HOD, PNC&KR PG COLLEGE NARASARAO PET.

[3]Assoc. Professor, Department of Computer Science and Engineering, K L University,srinivasu28@kluniversity.in

[4]Assoc. Professor, Department of Computer Science , Govt. Degree College ,Macherla, onrajunrt1@gmail.com

ABSTRACT

Selection of appropriate CPU scheduling algorithm is important, because the OS scheduler has to context-switch the CPU among the various tasks that an operating system has to execute. Real time systems involve high number of context switches. Hence, RR scheduling is not suitable, as this can result in a few particular processes waiting in the pipeline for a long time, to get the CPU time. This paper proposes a dual-processor based scheduling technique, called as Dynamic Time Quantum algorithm. Here, one processor handles only computation-intensive tasks, whereas the other processor handles only I/O tasks. In this approach, the scheduler intelligently dispatches the process to the appropriate processor (CPU). After the processes are dispatched to the respective processors, the time quantum is calculated and the processes are executed in increasing order of their burst time. Experimental Analysis had proved that the proposed algorithm viz. Dynamic Time Quantum algorithm had outperformed the traditional algorithm which used a single CPU.

Keywords: *Time Quantum, Scheduling, Round Robin , Context Switches, Waiting Time, Turnaround Time.*

1. INTRODUCTION

When there is more than one process to be executed, a ready queue is maintained. Here in a two processor system, ready queue is maintained for each processor. The operating system follows a predefined procedure for selecting process from a number of processes waiting in the ready queue and assigns the CPU to the process. Careful attention is required to assure fairness and avoid starvation during allocation of CPU to the processes.

It is a good practice to schedule CPU-intensive processes separately from I/O–intensive processes, which means one CPU is exclusively dedicated for CPU-intensive processes and another CPU is exclusively dedicated for I/O-intensive processes.

In our proposed algorithm we have used percentage values to classify the processes into two groups of CPU-intensive and I/O-intensive processes.

A. Scheduling Algorithms:

Many CPU scheduling algorithms are used such as First Come First served scheduling (FCFS), shortest job First scheduling (SJF), Priority Scheduling etc. All the above algorithms are non-preemptive in nature and also not suitable for time sharing systems. In the First-Come-First-

Served (FCFS), the process that arrives first in the ready queue is allocated the CPU first. In SJF, when the CPU is available, it is assigned to the process that has the smallest next CPU burst. If two processes have same next CPU burst time, FCFS scheduling is used to break the tie. In priority scheduling algorithm a priority is given to each process and the process having highest priority is executed first and so on. Round Robin scheduling is similar to FCFS scheduling, but preemption is added to switch between processes. A small unit of time, called a time quantum or time slice is defined and the CPU scheduler goes around the ready queue,

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

allocating the CPU to each process for a time interval of up to 1 time quantum. The Round Robin (RR) Scheduling is one of the most popular scheduling algorithms found in computer systems today. In addition it is designed especially for time sharing systems and found in multiple processor systems.

B. Related Work:

In the recent past, a number of CPU scheduling mechanisms have been developed for predictable allocation of processor. Self-Adjustment Time Quantum in Round Robin Algorithm [2] is based on a new approach called dynamic time quantum in which, time quantum is repeatedly adjusted according to the burst time of the running processes. Dynamic Quantum with Readjusted Round Robin Scheduling Algorithm [1] uses the job mix order for the algorithm in [2]. According to [1], from a list of N processes, the process which needs minimum CPU time is assigned the time quantum first and then highest from the list and so on till the Nth process. Again in the 2nd round, the time quantum is calculated from the remaining CPU burst time of the processes and is assigned to the processes and so on. Both [2] and [1] are better than RR scheduling and overcomes the limitations of RR scheduling regarding the average waiting time, average turnaround time and context switch. Algorithm in [3] uses an approximation of K-means clustering algorithm to group processes of same kind together and dispatches them to appropriate processor. A new fair-share scheduling with weighted time slice [4] assigns a weight to each process and the process having the least burst time is assigned the largest weight. The time quantum is calculated dynamically, using weighted time slice method and then the processes are executed. [5] calculates the original time slice suited to the burst time of each processes and then dynamic ITS (Intelligent Time Slice) is found out in conjunction with the SRTN algorithm[7]. Algorithm in [6] is improved by using dynamic time quantum and multi cyclic time quantum.

C. Our Contribution:

We have proposed a new scheduling algorithm for two processor systems, that first separates the CPU-intensive and I/O-intensive processes into two groups and dispatch them to two different processors. Then the processes are executed in each processor. Our execution approach gives better result than Round-Robin (RR) and Dynamic Quantum with Readjusted Round Robin Scheduling Algorithm (DQRRR) in [1]. Instead of taking job mix order we have taken the processes in ascending order in the two ready queues of two processors and the time quantum is calculated using our proposed method which changes with the every round of execution and to verify if the reaming burst time of the current execution process is less than one time quantum then execute the same process otherwise go for the next process in the queue.

2. BACKGROUND PRELIMINARIES

A. Terminologies:

A *process* is a program in execution. *Ready queue* holds the processes waiting to be executed or to be assigned to the processer. *Burst time* (b_t) is the time, for which a process requires the CPU for execution. The time at which the process arrives is called the *arrival time* (a_t).*Time quantum* (t_q) *or time slice* is the period of time given to each process to have CPU. *Average waiting time* (a_{wt}) is the time gap between the arrival of a process and its response by the CPU. *Average Turnaround time* (a_{tat}) is the time gap between the instant of process arrival and the instant of its completion. Average Response time (a_{rt}) is the time taken to start responding a process. The number of times the CPU switches from one process to another is called the *context switches (cs).* PC_{CPU} and $PC_{I/O}$ are the percentage requirement of a process for CPU and I/O respectively.

B. Dynamic Quantum with Re-adjusted Round Robin [1] Scheduling Algorithm

The DQRRR scheduling [1] has improved the RR scheduling by improving the turnaround time, waiting time and number of context switches. Processes are arranged in
job mix order in the ready queue and time quantum is found using median method. The CPU scheduler goes around the ready queue, allocating the CPU to each process for a time interval of up to 1 time quantum. Again the time quantum is calculated from the remaining burst time of the processes and so on. New processes are added to the tail of the ready queue. The CPU scheduler picks the first process from the ready queue and allocates the CPU to the process for 1 time quantum.

3. PROPOSED APPROACH

The proposed algorithm DTPBCS finds the time quantum in an intelligent way which gives better result in a two-processor environment than Dynamic Quantum with Readjusted Round Robin Scheduling Algorithm [1](DQRRR) and RR scheduling. Out of two processors one is solely dedicated to execute CPU-intensive

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

processes (CPU_1) and the other CPU is solely dedicated to execute I/O-intensive processes (CPU_2). The algorithm is divided into part I and part II. Part I algorithm classifies a process and dispatches it into an appropriate ready queue. Part II algorithm calculates the time quantum for both CPUs in a dynamic manner in each cycle of execution. The time quantum is repeatedly adjusted in every round, according to the remaining burst time of the currently running processes. We have taken an approach to get the optimal time quantum,

where a percentage value <$PC_{CPU,}$ $PC_{I/O}$> is assigned to each processes. For instance, <75, 25> represents a process

whose time comprises 75% of CPU activities and 25% of I/O activities. For instance in database systems, a process can easily spend 70% of its time accessing the hard disk and 30% on computation. Here the shorter processes are executed first, to give better turnaround time and waiting time.

The Time Quantum (t_q) is calculated as below. t_{q1} (for CPU_1) =

$$\frac{\sum_{i=1}^{i=n_1}(PC_{CPU}[i] \cdot b_t[i])}{\sum_{i=1}^{i=n_1} PC_{CPU}[i]}$$

t_{q2} (for CPU_2) =

$$\frac{\sum_{i=1}^{i=n_2}(PC_{I/O}[i] \cdot b_t[i])}{\sum_{i=1}^{i=n_2} PC_{I/O}[i]}$$

Algorithm for both CPU_1 and CPU_2

a_{wt}=0,a_{tat}=0.
while (CPU_1 ready queue
queue!= NULL) find the
time quantum t_q
for i=1 to n_1
Put the processes with ascending
order of burst
time in ready queue end for
process for
i=1 to
n_1
if(b[i] < t_q)
p[i]=b[i]=
t_q and
rb[i]=0 else
if (b[i] = t)
p[i]=
t_q and
rb[i]=0
else
p[i]=t_q
and
rb[i]=b[i]-t_q **if**
rb[i] <=1 tq
continue the
same process
end of for
if rb[i]=0,remove the process from the ready queue
if rb[i] > 0, insert the process in the ready queue with rb[i]
end of while
a_{wt} , a_{tat} are calculated. stop & exit

Algorithm for CPU_2 is same as for CPU_1

Where, n1= number of processes in the ready queue for CPU1

n2= number of processes in the ready queue for CPU2

$PC_{CPU}[i]$ = Percentage requirement of process i for CPU

$PC_{I/O}[i]$ = Percentage requirement of process i for

I/O. Here p[i] is process i, b[i] is the burst time of process i, rb[i] is the remaining burst time of process i.

Illustration:

We have considered an example to demonstrate the above algorithm .The burst time sequence are:22, 31, 53, 69, 79 assigned to processes P1,P2,P3,P4,P5 along with Membership values <80, 20>, <79, 21>, <25, 75>, <11, 89>, <9, 91> respectively. The processes (P1,P2) having

PC_{CPU} more than 70 are fed into CPU_1 ready queue and the processes (P3 P4, P5) having

Journal of Theoretical and Applied Information Technology
20th March 2015. Vol.73 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

$PC_{I/O}$ more than 70 are fed into CPU_2 ready queue. Inside the ready queues processes are arranged in ascending order of burst time(b_t). Then the time quantum is calculated in each CPU using the proposed formula. Using the above algorithm the t_q calculated for CPU_1 are 26 and 5 and t_q for CPU_2 are 68, 6 and 5 respectively.

4. EXPERIMENTAL ANALYSIS

Experiments Performed

To evaluate the performance of our proposed algorithm, we have taken a set of processes in three different cases. This algorithm can work effectively with large number of data. In each case we have compared the experimental results of algorithm with the scheduling algorithm DQRRR [1] and with Round-Robin (RR) algorithm.

Increasing Order:

We consider six processes p1, p2, p3, p4, p5 and p6 arriving at time 0 with burst time 28, 52, 95, 110, 141, 153 respectively shown in Table I. Table II and Table III shows the comparing result of DQRRR algorithm, RR algorithm and our proposed TPCS algorithm for CPU_1 and CPU_2 respectively.

Table I: Data In Increasing Order

No. of process	b_t	PC_{CPU}	$PC_{I/O}$
P1	28	80	20
P2	52	70	30
P3	95	95	50
P4	110	25	75
P5	141	15	85
P6	153	10	90

Table II: Comparison Between DQRRR, RR And Our DDPCS Algorithm For CPU_1

Algorithms	TPCS	DQRRR	RR	DDPCS
t_q	61,34	52,43	35	61,34
a_{wt}	36	53.3	47.6	36
a_{tat}	94.3	111.6	106	94.3

Table III: Comparison Between DQRRR, RR And DDPCS For CPU_2

Algorithms	TPCS	DQRRR	RR	DDPCS
t_q	136,11,6	141,12	35	136
a_{wt}	165.7	167.3	248	**165.66**
a_{tat}	300.3	302	371.67	**255**

Table IV. Data In Decreasing Order

No. of process	b_t	PC_{CPU}	$PC_{I/O}$
P1	254	30	70
P2	209	82	18
P3	182	79	21
P4	99	12	88
P5	72	50	95
P6	58	91	90
P7	37	85	15

Journal of Theoretical and Applied Information Technology
20th March 2015. Vol.73 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table V: Comparison Between DQRRR, RR And DDPCS For CPU_1

Algorithms	TPCS	DQRRR	RR	DDPCS
t_q	118,78,13	120,75,14	35	118,78,13
a_{wt}	131.75	181	237	**102**
a_{tat}	253.25	302.5	358.5	**223**

Table VI: Comparison Between DQRRR, RR And TPCS For CPU_2

Algorithms	TPCS	DQRRR	RR	DDPCS
t_q	131,123	99,155	35	131
a_{wt}	81	114	183	81
a_{tat}	222.67	255.67	325	222

Random Order:

We consider six processes p1, p2, p3, p4, p5, p6 and p7 arriving at time 0 with burst time 94, 52, 39, 155, 113, 238 and 167respectively shown in Table VI. Table VII and Table IX shows the comparing result of DQRRR algorithm, RR algorithm and our proposed algorithm TPCS for CPU_1 and CPU_2 respectively.

Table VII. Data In Random Order

No. of process	b_t	PC_{CPU}	PC_{IO}
P1	94	12	88
P2	52	90	10
P3	39	85	15
P4	155	17	83
P5	113	77	23
P6	238	21	79
P7	167	90	91

Table VIII. Comparison Between DQRRR, RR And TPCS For CPU_1

Algorithms	TPCS	DQRRR	RR	DDPCS
t_q	66,47	52,61	35	66,47
a_{wt}	43.3	60.7	82.67	43.3
a_{tat}	111.3	128.7	452	111.3

Table IX. Comparison Between DQRRR, RR And TPCS For CPU_2

Algorithms	TPCS	DQRRR	RR	DDPCS
t_q	162,38,38	161,41,36	35	162,38
a_{wt}	230.25	280	356	**189.75**
a_{tat}	393.75	444	519.5	**353.25**

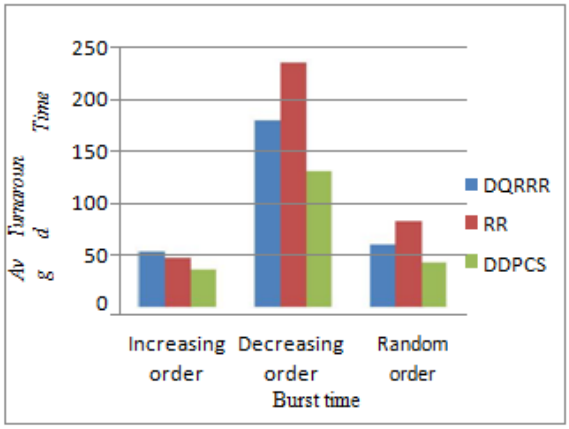

Fig. XIII: Comparison Of Average Turnaround Time Between DQRRR, RR And TPCS For CPU_1

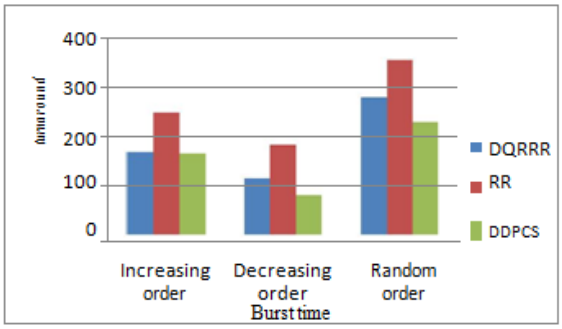

Fig XIV: Comparison Of Average Turnaround time between DQRRR, RR and TPCS for CPU_2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

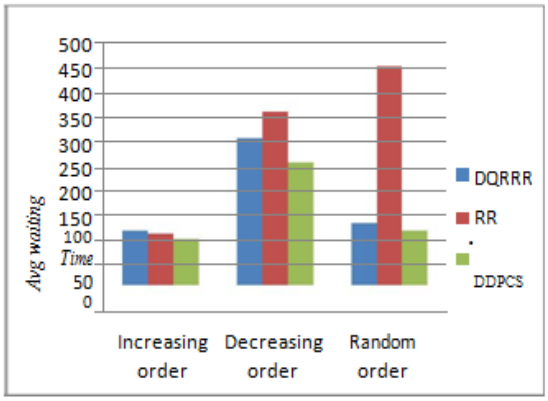

Fig XV: Comparison Of Average Waiting Time Between DQRRR, RR And TPCS For CPU_1

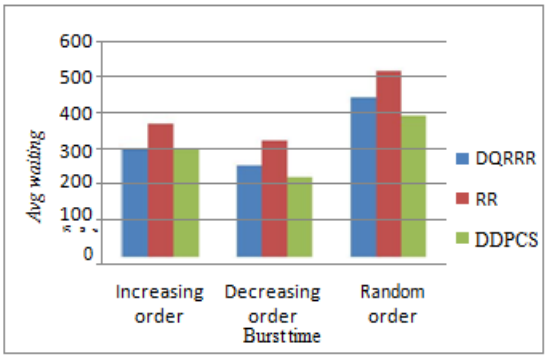

Fig XV: Comparison Of Average Waiting Time Between DQRRR, RR And TPCS For CPU_2

5. CONCLUSION

In this paper a new scheduling algorithm DTPCS, which is a modified Round Robin algorithm, used on a two processor system. One processor is designated exclusively for CPU-intensive processes and the other CPU is designated exclusively for I/O-intensive processes. The above comparisons show that the proposed DDPCS algorithm provides much better results than the algorithm proposed in [1] and Round-Robin algorithm[7] in terms of average waiting time, average turnaround time. This algorithm will implement by consider the arrival time of preemptive and non preemptive scheduling also.

REFERENCES

[1]H.S. Behera, R. Mohanty, Debashree Nayak "A New Proposed Dynamic Quantum with Readjusted Round Robin Scheduling Algorithm and its Performance Analysis", International Journal of Computer Applications(0975-8887) Volume 5- No.5, 10-15, August 2010.

[2] Rami J. Matarneh. "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.

[1] Nayak, Behera, R. Mohanty, Debashree

[3] Sanpawat Kantabutra, Parinya Kornpitak, Chengchai Naramittakapong "Dynamic Clustering-Based Round-Robin Scheduling Algorithm". Proceedings of the 3rd international symposium on communication and information technology (ISCIT2003) September 03-05,2003, Hatyai, Songkhla, Thailand

[4] H.S. Behera, Rakesh Mohanty, Jajnaseni Panda, Dipanwita Thakur, Subasini Sahoo "Experimental analysis of a new fair-share scheduling algorithm with waited time slice for real time systems". Journal of Global Research in Computer Science (ISSN-2229-371X), Volume 2, No. 2, 54-60, February 2011

[5] H.S. Behera, Simpi Patel, Bijaylaxmi Panda. "A new dynamic Round-robin and SRTN algorithm using variable original time slice and dynamic intelligent time slice for soft real time system". International Journal of Computer Applications (0975-8887), Volume 16, No.1, 54-60, February 2011

[6] .H.S. Behera, Rakesh Mohanty, Sabyasachi Sahu, Sourav Kumar Bhoi, "Design and performance evaluation of multi cyclic round robin(MCRR) algorithm using dynamic time quantum" Journal of global research in computer science(ISSN-2229-371X), volume 2, No.2, February 2011.

[7].Silberschatz,A.,P.B.GalvinandG.Gange,2004." Operating systems concepts". 7th Edn.,John wiley and Sons, USA. ,ISBN:13:978-0471694663,pp:944.