Two Queue based Round Robin Scheduling Algorithm for CPU Scheduling

Srishty Jindal
FET, MRIU
Faridabad

Priyanka Grover
FET,MRIU
Faridabad

ABSTRACT
Multitasking and multiuser operating system's performance depends on the efficiency of scheduling algorithm. Most commonly used Round Robin scheduling algorithm may not give optimal result if the burst time of processes is very high as compared to the time quantum of Round Robin algorithm. In this case, context switching and turnaround time of processes is very high. In this paper, a two queue based Round Robin Scheduling Algorithm is proposed. In the proposed approach two queues are used. One queue is exclusively used for CPU intensive processes and other queue is used for I/O intensive processes. This reduces the waiting time and turnaround time when there are less or equal numbers of I/O intensive processes. Performance Analysis depending upon CPU intensive and I/O intensive processes shows that it provides better results.

Keywords
Scheduling, Round Robin Algorithm, Context Switching, CPU Burst time, I/O Burst time

1. INTRODUCTION
Operating systems use scheduling algorithm to provide services to user on performing different tasks. These Scheduling algorithms are used when the multiple processes compete for the CPU at the same time & the scheduler then allocates the job to the CPU for execution according to the algorithm. There are several scheduling algorithms such as First Come First Serve(FCFS) Scheduling[1], Shortest Job First(SJF) Scheduling, Priority Scheduling, and Round Robin Scheduling[2]. Each Scheduling algorithm has some advantages and some disadvantages. Round Robin Scheduling algorithm is however widely used.

2. ROUND ROBIN SCHEDULING ALGORITHM
The round robin scheduling algorithm is designed especially for time-sharing systems. It is similar to FCFS scheduling, but preemption is added to switch between processes. A small unit, called a time quantum is defined. A time quantum is generally from 10 to 100 milliseconds. The ready queue is treated as circular queue. The CPU scheduler goes around the ready queue, allocating the CPU to each process for a time interval of up to 1 time quantum.

To implement RR scheduling, we keep the ready queue as a FIFO queue of processes. New processes are added to the tail of the ready queue. The CPU scheduler picks the first process from the ready queue, sets a timer to interrupt after 1 time quantum, and dispatches the process. One of the two things happens:

1) The process may have a CPU burst of less than 1 time quantum.

2) If it doesn't finish within the time slice, place it at the end of the queue, and choose the next item from the queue.

It Provides some sort of fairness (since all jobs are treated equally), but takes a long time to finish everything because each job gets very little time on the CPU (so the average turnaround time is high).RR allocates the CPU uniformly (fairly) across participants.

3. RELATED WORK
Sanpawat Kantabutra, Parinya Kornpitak, and Chengchai Naramittakapong[5] discussed dynamic clustering-based Round-Robin scheduling algorithm (DCBRRSA) for two processor system. In which one processor is exclusively for CPU-intensive processes and the other processor is exclusively for I/O-intensive processes. This scheduling algorithm uses an approximation of K-means clustering algorithm to group processes of the same kinds together and dispatches them to appropriate processors. Simulations have been done and comparisons are made between the uni-processor and two processor systems using the original Round-Robin algorithm and the two-processor system. Their results show that the average execution time of a process that uses the DCBRRSA is significantly less than that of the other two scheduling methods.

Richard Roehl and Brian Johnson [2] developed a new algorithm based on Round robin Scheduling and Priority Scheduling. This scheduling algorithm can be applied for to Operating system that implements a Round Robin Scheduler.

In this, tasks which have consumed or consume more CPU than their peers get equal scheduling priority on the CPU. By implementing a priority queue in the scheduler one can easily improve the performance of processes that require short bursts of CPU while continuing to service the processes with higher usage demand.

In this algorithm three queues are implemented. One queue is for kernel task, second queue for device driver task and third queue for user processes. Out of which highest priority is given to kernel tasks & lowest priority is given to user process. Each queue is processed until it is empty. Highest priority queue is executed first then execution continues according to their priorities.

H.S. Behera and Brajendra Kumar Swain[7] proposed a precedence based Round Robin with dynamic time quantum Scheduling Algorithm for soft real time system. In this algorithm, processes are given precedence according to their burst time and priority, then Round Robin algorithm is applied on it with dynamic mean time quantum. This algorithm performs better than other algorithms in terms of reduced context switching, waiting time and turnaround time.

Aashna Bisht, Mohd.Abdul Ahad and Sielvic Sharma[8] discussed a dynamic time quantum for round robin process scheduling algorithm which directly effect the turnaround time, waiting time and number of context switches. Time quantum is calculated dynamically on the basis of burst time of processes waiting in the ready queue.

Ms. Rashmi A. Dhumal, Ms. Tabassum A. Maktum and Ms. Lata Ragha[9] discussed a dynamic quantum based Genetic Round Robin algorithm in order to improve the performance. In this paper, time quantum is calculated dynamically by using Genetic approach as it provides the optimal solution for a problem. Initially, the time quantum is the median of all the processes burst time, then genetic algorithm is used to calculate fittest chromosome i.e. one with minimum average waiting time.

4. PROPOSED TWO QUEUE BASED ROUND ROBIN SCHEDULING ALGORITHM

In this proposed algorithm, Round-Robin scheduling algorithm is chosen because it is one of the most popular scheduling algorithms. In scheduling processes it is a good idea to schedule CPU-intensive processes separately from I/O-intensive processes [3,4,5]. It can reduce response time significantly for interactive I/O processes. Processes are executed in 1:1 ratio.

Likewise, CPU-processes also benefit from having less accumulated context switch time occurring with I/O processes and can generally run until they complete their time quantum. Thus, it is expected that more processes will be completed on average. In this scheduling algorithm we can use an approximation of the well-known K-means clustering algorithm [6] to classify the processes into two groups of CPU-intensive and I/O-intensive process.

In this algorithm two ready queues are maintained. One queue is solely dedicated to CPU-bound processes and another queue is used to store I/O bound processes. Processes may arrive at any time. These processes are classified in two classes. The two classes are CPU intensive processes & I/O intensive processes. One processor is responsible for execution of CPU intensive processes; second processor is responsible for I/O intensive processes. CPU intensive processes are those Processes in which there is more than 50% processing. Processes that spent more than 50% (of their burst time) in waiting queue are I/O intensive processes. Two queues are maintained one for each type of processes. Queue for CPU intensive processes is called Queue1 and the queue for I/O intensive processes is called Queue2. All the processes are sent to ready queues accordingly. On both of the ready queues Round Robin Algorithm is applied. These queues are updated as soon as any process arrive or completes its execution. T is assigned as the arrival time of first process. Each time the process enters, first its type is determined whether it is CPU bound or I/O bound. Whenever the burst time of a process is more than the time quantum, then the processes is inserted at the end of that queue with its remaining burst time. Processes which are arrived in-between are inserted first, and then the remaining part of previous process is inserted in the queue. This algorithm executes the processes in 1:1 ratio with CPU and I/O bound process. It means one process is executed from queue1 and one process from queue2.

4.1 Algorithm

Pseudo code: Enter the name nm, number no, type c, arrival time ar, burst time bt of n number of processes. I2 is the counter which will count the number of processes completed. M is the time quantum of algorithm and t is the total time.

two-queues(nm,no,c,ar,bt)

1) assign t=p[i].ar;
2) while(i<n)//while input queue is empty
3) {
4) if(p[i].c==1)
5) Put the values nm,no, ar,bt of that process in q1
6) else
7) Put the values nm,no, ar,bt of that process in q2
8) while((i2<n)&&((j<n1)||(k<n2))){/*j is the counter of array q1 and k is the counter of array q2*/
9) if(n1>j) {//If there are processes in queue q1
10) if(q1[j].bt<=m) //If burst-time is less than m the process is completed.
11) Compute the waiting time & burst time of the process.
12) else //If Burst-time is more than m it is to be put in queue q1
13) Check if more processes have come in between put them into queues.
14) Put the incomplete process in the end of the queue q1.
15) }
16) if(n2>k){//If there are processes in queue q2
17) if(q2[k].bt<=m)//If burst-time is less than m the process is completed
18) Compute the waiting time & burst time of the process.
19) else //If Burst-time is more than m it is to be put in queue q2
20) Check if more processes have come in between put them into queues.
21) Put the incomplete process in the end of the queue q2.}
22) }
23) }
24) Print the contents of queue1, queue2, and p.

5. RESULTS

Following are the three cases considered to analyze the actual efficiency of the algorithm. These three cases are taken according to the type of process we are taking our examples.

Case-1: Almost equal CPU bound and I/O bound Processes

In this case number of CPU bound processes is almost equal to the number of I/O bound processes. Results in this case are shown in table 1.

Table 1: Average waiting time results for case 1.

No. of Processes	Avg. Waiting time in Double Queue(in ms)	Avg. Waiting time in single Queue(in ms)
10	56	57.7
15	86	92.46
20	118.2	125.5
30	138.5	144.8

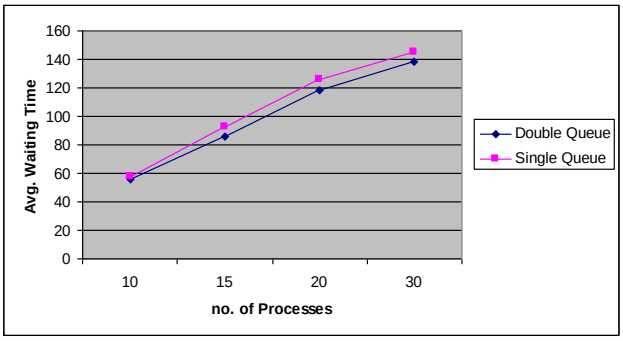

Fig 1: Comparison of average waiting time for case 1

Case-2: Less CPU /More I/O

In this case, the number of CPU bound processes is less than the number of I/O bound processes. Results in this case are shown in table 2.

Table 2: Average waiting time results for case 2.

No. of Processes	Avg. Waiting time in Double Queue(in ms)	Avg. Waiting time in single Queue(in ms)
10	45.2	46.2
20	103.85	114.75
30	158.03	167.16

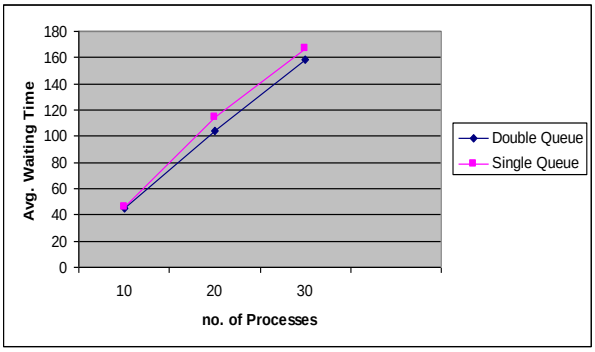

Fig 2: Comparison of average waiting time for case 2

Case-3: More CPU /Less I/O

In this case, number of CPU bound processes is more than the number of I/O bound processes. Results in this case are shown in table 3.

Table 3: Average waiting time results for case 3.

No.of Processes	Avg. waiting time in Double Queue (in ms)	Avg. waiting time in Single Queue (in ms)
10	34.3	33.3
20	81.1	79.3
30	123.5	120.7

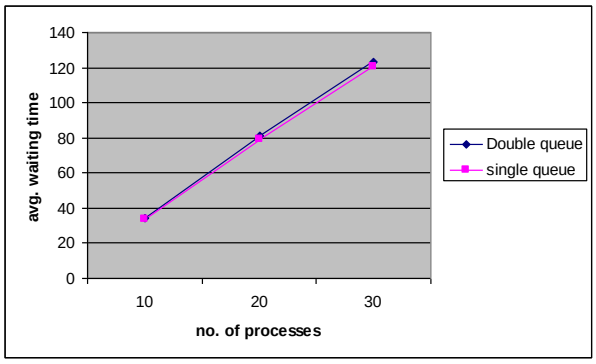

Fig 3: Comparison of average waiting time for case 3

6. CONCLUSION AND FUTURE SCOPE

Two-queue based Round Robin scheduling algorithm has been discussed. Three examples are taken to analyze its effectiveness. Results are compared with single queue. The following are the observations.

1. Two-queue algorithm is more efficient in terms of waiting time & turn around time in case when number of CPU bound processes are less than or equal to I/O bound processes.
2. As the number of processes increases, the average waiting time in single queue algorithm increases & the average waiting time in two-queue algorithm decreased as compared to single queue algorithm i.e. the gap between their waiting time increases. Hence, efficiency increases as number of processes increases.
3. As seen in the examples, when single queue algorithm is applied, some processes get access to the CPU after a long time while in case of two-queue algorithm more number of processes get access to the CPU in less time. Hence it gives better response time.

General observation is that two-queue algorithm results in less waiting time, turnaround time & response time. Multi-Queue Round Robin algorithm can be extended in the following directions.

1. Two-queue Round Robin algorithm can be implemented in several ratios with CPU & I/O bound process.
2. If the process is near completion, then extra CPU time quantum may be given to that process.
3. Process scheduling with multi-row concept can be implemented.
4. This can be extended to Real time Operating System environment to improve the efficiency.

7. REFERENCES

[1] Kantabutra Sanpawat, Kornpitak Parinya, and Naramittakapong Chengchai; "Dynamic Clustering based Round Robin Scheduling Algorithm":The Theory of Computation Group, Department of Computer Science Chiang Mai University Chiang Mai,2003.

[2] Bradley P.S.,Fayyad U.M., "Refining initial points for K-means clustering": Proc. Of 15th international conference on Machine Learning(ICML 98),pp 91-99,1998.

[3] Roehl Richard, Johnson Brian;"Designing a fairer Round robin Algorithm":31st Annual International Conference of The Computer Measurement Group, Inc., Orlando Florida USA, pp 1-8,2005.

[4] Silberschatz A. and Galvin P.B.; Operating System Concepts:Addison Wisley Publishing Company, New York, 1994.

[5] Andrew S. Tanenbaum. Modern Operating System. Prentice Hall, New Jersey,1992.

[6] E.G. Coffman and L. Kleinrock. Feedback queuing models for time-shared systems. Journals of the ACM, vol.15,549-576, October 1968.

[7] H.S. Beher and Brajendra Kumar Swain; "A New Prposed Precedence based Round Robin with Dynamic time Quantum Scheduling Algorithm for Soft Real Time Systems"; International Journal of Advanced Research in Computer Science and Software Engineering, Vol 2, Issue 6, June 2012.

[8] Aashna Bisht, Mohd. Abdul Ahad and Sielvic Sharma; "Calculating Dynamic Time Quantum for Round Robin process Scheduling Algorithm", International Journal of Computer Application, Vol 98, November 21, 2014.

[9] Ms. Rashmi A. Dhumal, Ms. Tabassum A. Maktum, and Ms. Lata Ragha; " Dynamic Quantum based Genetic Round Robin Algorithm", International Journal of Advanced Research in Computer and Commnication Engineering, Vol 3, Issue 3, March 2014.