Volume 6, Issue 1, January 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

A Comparative Study of Different Scheduling Algorithms for Real Time Operating System

[1]Rahul Karmakar, [2]Bikash Mondal, [3]Shehnaz Abdur Rauf, [4]Mallika Reddi, [5]Tuhin Kanti Maiti
[1]B.Tech, M.Tech, Assistant Professor, Dept. of CS, BU
[2]BCA, M. Sc (CS), [3, 4]B. Sc (CS), M. Sc CS)
[5]B. Sc (Pure Sc.), A-level (DOEACC), M. Sc (CS)

Abstract- Real-time systems play an important role in our modern society. The main objective of this paper is to compare the different scheduling algorithms in Real time system for scheduling the task, different scheduling algorithms were used. Most of the real-time systems are designed using priority based preemptive scheduling and worst case execution time estimates to guarantee the execution of high priority tasks. Workstation and personal computers are increasingly used for applications with real time characteristics such us speech understanding and synthesis, media computations and I/O and often concurrently executed with traditional non-real-time workstation. After that, we describe performance parameters we use to compare the performances of the various algorithms. We observe that the choice of a scheduling algorithm is important in designing a real time system. We conclude by discussing the results of the survey and suggest future research directions in the field of Real Time Operating System.

Keywords- Real-Time Operating Systems, Scheduling Algorithm, Deadline, Scheduler, Earliest Deadline First (EDF)

I. INTRODUCTION

The objective of a real-time task scheduler is to guarantee the deadline of tasks in the system as much as possible when we consider soft real-time system. Mostly all the real-time systems in existence use preemption and multitasking. There are two important properties of a task are the nature of the deadline and the arrival characteristics. The task arrival may be periodic with a constant interval, or the task may be stastically distributed and the inter-arrival time may be random in nature. The important requirement is that a time constrained task should be completed before the deadline set for that task expires. Depending on the real-time system, the nature of the deadline maybe different. Tasks with hard deadlines must be completed before the specified time. If the deadline is missed, the utility of execution for such a task in nil.

Real-time scheduling techniques can be largely divided into two categories: **Static** and **Dynamic**. **Static algorithms** assign priorities at design time. All assigned priorities remain constant for the lifetime of a task. **Dynamic algorithms** assign priorities at runtime, based on execution parameters of tasks.

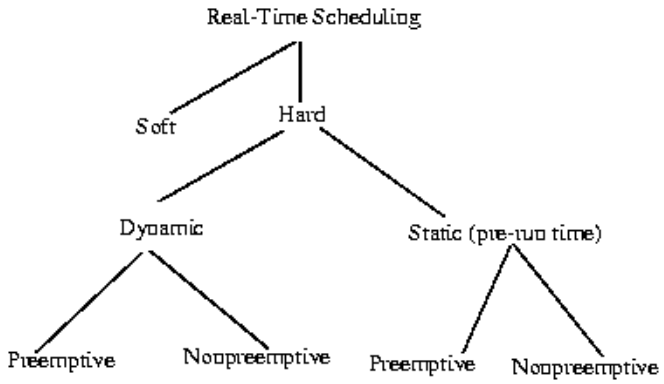
We observe that the choice of an operating system is important in designing a real time system. Designing real time systems involves choice for proper language, task portioning and merging and assigning priorities using a real time scheduler to manage response time. The depending upon scheduling objectives parallelism and communication may be balanced. The designer of scheduling policy must be determine critical tasks and assign them high priorities .however, care must be taken avoid starvation, which occurs when high priority tasks are always ready to run.

II. DIFFERENT SCHEDULING ALGORITHM

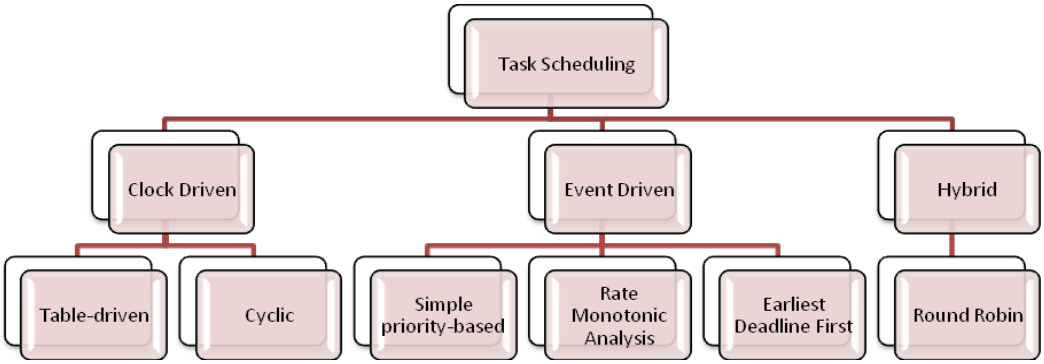
There are two main types of real-time systems:

Hard Real-Time System-System requires that fixed deadlines must be met otherwise disastrous situation may arise whereas in Soft Real-Time System, missing an occasional deadline is undesirable, but nevertheless tolerable.

Firm or Soft Real-Time System-System in which performance is degraded but not destroyed by failure to meet response time constraints is called soft real time systems



Different Task-Scheduling



A. Table-driven methods:

are schemes that allow you to look up information in a table rather than using logic statements (i.e. case, if). In simple cases, it's quicker and easier to use logic statements, but as the logic chain becomes more complex, table-driven code is simpler than complicated logic, easier to modify and more efficient.A table-driven method is quite simple. Use data structures instead of if-then statements to drive program logicThe concept is really best explained through an example:
Let's say you're running a restaurant and have a different number of seats for each table number.
Your logic to get the number of seats for a particular table might look something like

```
if table number == 1
   table has 4 seats
else if table number == 2
   table has 8 seats
```

so if you have 50 tables you would have 100 lines of code just to determine the number of seats.
Using table-driven methods, you could make an array with the index representing the table number and the value representing the number of seats, so your logic would instead look something like

```
tables [] = {4, 8, 2, 4, ...}
table seats = tables[table number]
```

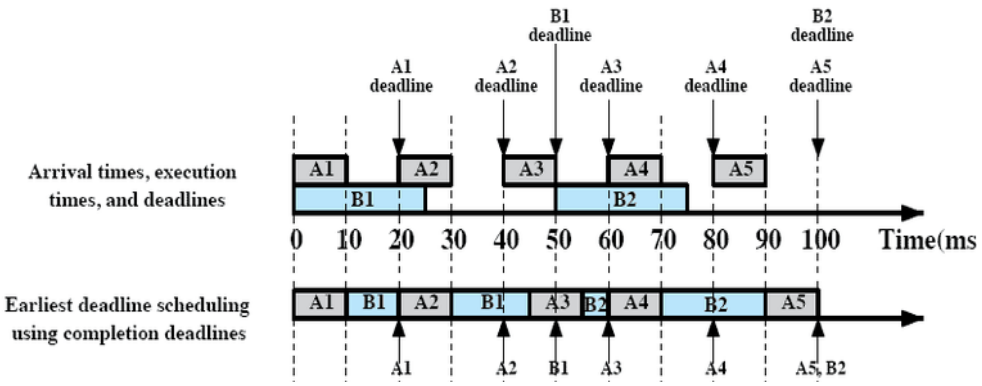
which is simpler, shorter, and easier to maintain.

B. The rate monotonic algorithm (RMA) :

In computer science, The rate monotonic algorithm (RMA) is a procedure for assigning fixed priorities to tasks to maximize their "schedulability." A task set is considered schedulable if all tasks meet all deadlines all the time. The algorithm is simple. Assign the priority of each task according to its period, so that the shorter the period the higher the priority. One major limitation of fixed-priority scheduling is that it is not always possible to fully utilize the CPU. Even though RMA is the optimal fixed-priority scheme, it has a worst-case schedule bound of:
$Wn = n(2(1/n) - 1)$
where n is the number of tasks in a system. As you would expect, the worst-case schedulable bound for one task is 100%. But, as the number of tasks increases, the schedulable bound decreases, eventually approaching its limit of about 69.3% .It is theoretically possible for a set of tasks to require just 70% CPU utilization in sum and still not meet all their deadlines.

C. Earliest deadline first (EDF):

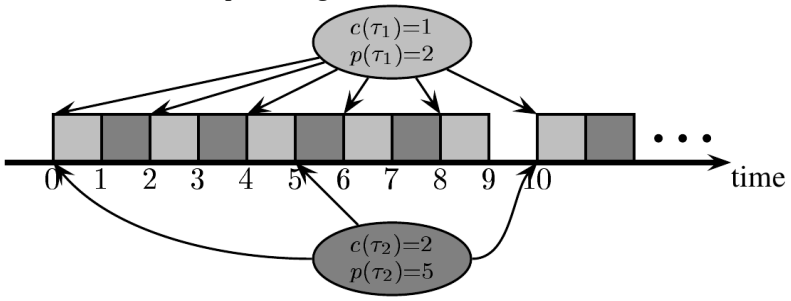
The **Earliest Deadline First (EDF)** algorithm is a dynamic priority-scheduling algorithm in which the priorities of individual jobs are based on their absolute deadlines. An EDF algorithm can generate a feasible schedule for a system of N independent, pre-emptable tasks as long as the total density of the system is less than 1. Hence EDF is an optimal scheduling algorithm. The priority of each task is decided based on the value of its deadline. The task with nearest deadline is given highest priority and it is selected for execution. This algorithm is simple and proved to be optimal when the system is preemptive, under loaded and there is only one processor.



D. Clock scheduling algorithm:

In scheduling the voltage at which a system operates and the frequency at which it runs, a scheduler faces two tasks: to predict what the future system load will be (given past behavior) and to scale the voltage and clock frequency accordingly. These two tasks are referred to as prediction and speed-setting .We consider one scheduler better than another if it meets the same deadlines (or has the same behavior) as another policy but reduces the clock speed for longer periods of time.

The schedulers we implemented are interval schedulers, so called because the prediction and scaling tasks are performed at fixed intervals as the system runs .At each interval, the processor utilization for the interval is predicted, using the utilization of the processor over one or more preceding intervals.



III. COMPARISON OF REAL TIME SCHEDULING ALGORITHMS

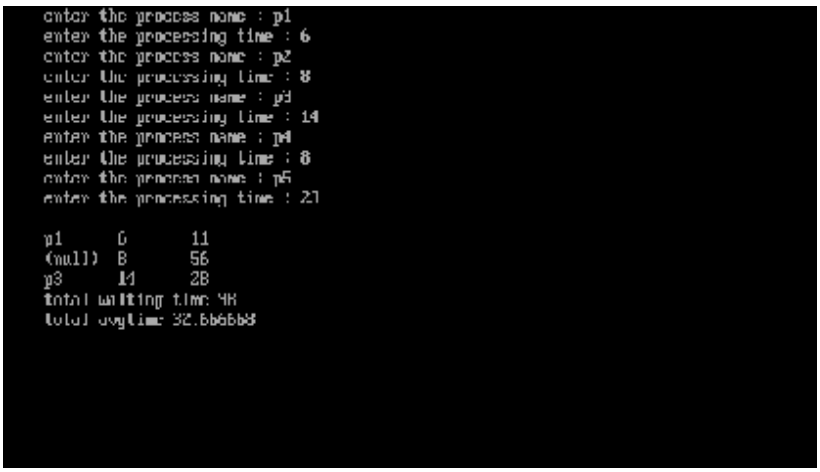
	Rate Monotonic Algorithm (RMA)	*Table-driven Algorithm (TDM)*	*Earliest Deadline First (EDF)*	*Clock Scheduling Algorithm (CSA)*
Implementation	***Simplest***	***Simple***	***Difficult***	***Difficult***
Processor Utilization	***Less***	***More Compared to RM***	***Full Utilization***	***Full Utilization***
Priority Assignment	***Static***	***Static***	***Dynamic***	***Dynamic***
Scheduling criterion	***Task Period***	***Relative Deadline***	***Deadline***	***Deadline***
Jitter Control	***Only for highest priority task***	***Only highest priority for task***	***Inefficient in overloaded systems***	***Inefficient in overloaded systems***

IV. SIMULATION RESULTS

A. Table-driven methods:

The major cycle of a set of tasks ST={T1,T2……Tn} is LCM ({P1,P2……Pn}) even when the tasks have arbitrary phasing.

OUTPUT:



B. The rate monotonic algorithm (RMA) :

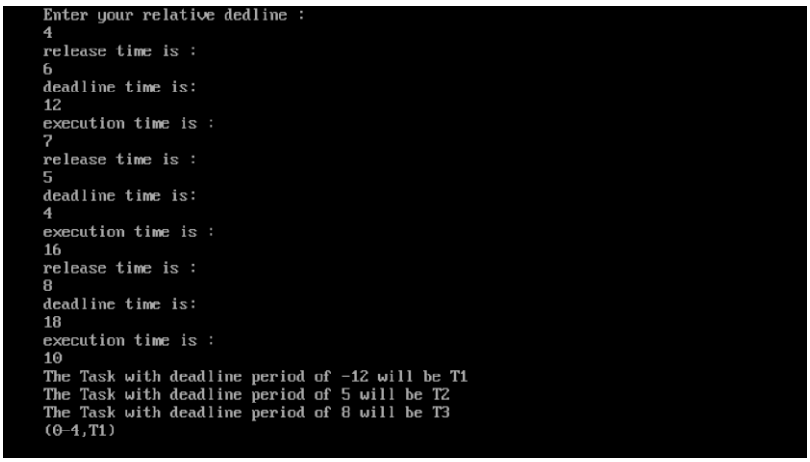
Enter execution time for p1: 5
Enter period for P1: 3
Enter execution time for p2: 8
Enter period for P2: 4
Enter execution time for p3: 12
Enter period for P3: 6

After compile the program it is easy and sure to schedule all process.

OUTPUT:

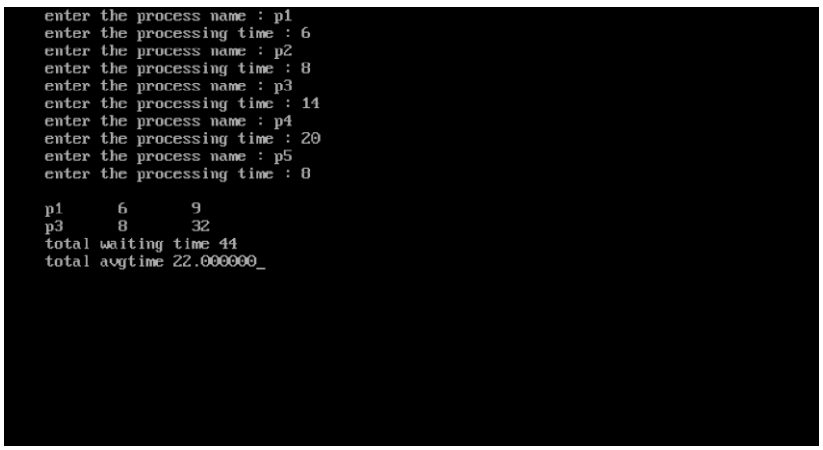
C. Earliest deadline first (EDF):

Output:



D. Clock scheduling algorithm:

Output:



V. CONCLUSION AND FUTURE SCOPE

From the comparative study it can be concluded that real time systems have became an inevitable part of our lives and their accurate performance has been a challenge. Scheduling a real time systems is done by static priority scheduling algorithms. Although dynamic priority scheduling algorithms can perform better than static priority in terms of CPU utilization, but it increases the overhead on the system. So, dynamic priority scheduling algorithms are not available in commercial real time systems. The various objectives of scheduling have been discussed. Study of scheduling algorithms have been done and it has been observed that pre-emptive scheduling with dynamic priorities works very well in case of scheduling tasks on real time systems. From the comparison of real time scheduling algorithms, it is clear that earliest deadline first is the efficient scheduling algorithm if the CPU utilization is not more than 100%. For hard real time systems, schedulability analysis can be done and calculations of probabilistic Worst Case Execution Time (WCET) analysis can also be done. Implementation of scheduling algorithm on FPGA can be done for scheduling tasks with dynamic priorities and schedulability of the task can be checked.

REFERENCES

[1] C.L. Liu and James W. Layland, "Scheduling Algorithms for Multiprogramming in a Hard Real Time Environment", Journal of the Association of Computing Machinery, Vol. 20, No. 1, January 1973, pp. 46-61.

[2] J. Liu, Real-Time Systems, Pearson Education, 2000.

[3] N. Fisher et al., "The Non-preemptive Scheduling of Periodic Tasks upon Multiprocessors", Journal of Real-Time Systems, vol. 32, n. 1-2, pp. 9-20, 2006.

[4] Marko Bertogna, Michele Cirinei, and Giuseppe Lipari, "Schedulability Analysis of Global Scheduling Algorithms on Multiprocessor Platforms", IEEE Transactions On Parallel And Distributed Systems, Vol. 20, No. 4, April 2009, pp. 553-566.

[5] L. Sha, R. Rajkumar, J.P. Lehoczky, "Priority Inheritance Protocols: An Approach to Real-Time Synchronization," IEEE Transactions on Computers, 1990.

[6] Clifford W. Mercer and Hideyuki Tokuda "Preemptibility in Real-Time Operating Systems"School of Computer Science, Carnegie Mellon University

[7] K. Ramamritham and J. A. Stankovic, "Scheduling algorithms and operating systems support *for real-time systems*," Proceedings of the IEEE, vol. 82, no. 1, pp. 55--67, January 1994.

[8] Pravanjan Choudhury, Rajeev Kumar and P.P. Chakrabarti, "Hybrid Scheduling of Dynamic Task Graphs with Selective Duplication for Multiprocessors under Memory and Time Constraints" IEEE Transactions On Parallel And Distributed Systems, Vol. 19, No. 7, July 2008, pp. 967-980.

ABOUT AUTHORS

[1]Mr. Rahul Karmakar
B.TECH (CSE), M.TECH (CSE)
Assistant Professor, Dept. of Comp. Sc, B.U, W.B
Email: rahulkarkamar6@gmail.com
Contact: +91 9434245749/+91 9933576628

[2]Mr. Bikash Mondal
BCA [WBUT], M. Sc (CS), [BU]
Email: bikash.abs89@gmail.com
bikashdgp19@gmail.com
Contact: +91 9002131286

[3]Miss. Shehnaz Abdur Rauf
B. Sc (CS)
M. Sc (CS)
Email: raufshehnaz@gmail.com
Contact: +91 9836639542

[4]Mrs. Mallika Reddi
B.Sc(CS)
M.Sc(CS)
Email: mallika.dasreddi@gmail.com
Contact: +91 892754717

[5]Mr. Tuhin Kanti Maiti
B. Sc (Pure Sc.)
A-level (DOEACC), M. Sc (CS)
Email: metuhin_123@rediffmail.com
Contact: +91 7602760403

