INTERNATIONAL JOURNAL OF RESEARCH SCIENCE & MANAGEMENT

PERFORMANCE ANALYSIS OF DYNAMIC PRIORITY SCHEDULING ALGORITHMS IN REAL TIME SYSTEMS

Amy Martin [1*],Pradeep Kumar T.S. [2],
[1*]School of Electronics Engineering (SENSE), VIT University, Tamil Nadu, India
[2]School of Computing Science and Engineering (SCSE), VIT University, Tamil Nadu, India
*Correspondence Author: amymartin1302@gmail.com

Keywords: EDF, RM, scheduling, real time systems, deadline, period

Abstract
A scheduling algorithm is the one which decides a schedule for a given set of tasks. There are a number of algorithms for task scheduling on a processor. Some of these algorithms are used for scheduling tasks on a multiprocessor system either under the partitioning scheme or under the global scheme. The optimal algorithm is the Earliest Deadline First (EDF) algorithm. If a set of task cannot be scheduled under EDF, no other scheduling algorithm can feasible schedule this task. Many improved versions of the existing conventional EDF algorithm have been proposed until now. In this paper, a study on the conventional EDF is done in detail. Finally, an evaluation of EDF algorithm using the processor values and static values is done.

Introduction
Real time systems are the ones in which the correctness of a system depends not only on the results of computations but also on the time at which the system produces its results [1]. In real time systems, there are fixed time constraints for each task to be performed. If the tasks are not completed in the specified time, the system is considered to be failed. Taking this into context, we can broadly classify real time systems into two: Soft Real Time Systems and Hard Real Time Systems [1].
Real time systems in which nothing catastrophic will happen if the deadlines are not met by each task are called Soft Real Time Systems [2]. This will only affect the performance of the system. For e.g. Multimedia. Whereas hard real time systems are those in which the result might be disastrous if deadlines are not me[4]t. In order to make sure the deadlines are met, a task scheduler is used.
Real time scheduling algorithms or techniques can be broadly categorized into two: dynamic and static. The priorities for tasks in a static algorithm are given at the design time. In a dynamic algorithm the priorities are assigned during the run time. The dynamic scheduling can again be subdivided into two categories, static priority and dynamic priority. The two prominent scheduling algorithms that fall into these two categories are Earliest Deadline First (EDF) and Rate Monotonic (RM) respectively [2]. The schedule may be preemptive, if a task can be interrupted by another task of higher priority or non- preemptive, where the task must be run to completion until it gets interrupted. Both EDF and RM algorithms are preemptive schedules.

Earliest deadline first
EDF is an optimal uniprocessor algorithm. Here tasks are preemptible and the priority is based on the deadlines. The task with the earliest deadline will have the highest priority. If a set of tasks cannot be scheduled on a single processor by EDF algorithm, then there is no other algorithm that can successfully schedule these tasks[3].

Rate monotonic
This algorithm can be overrated above all other algorithms used till today. It is a static priority algorithm which is used in uniprocessor systems. The task which has the shortest period will have the highest priority. Once the priority levels are fixed for each task, it remains the same until the end of scheduling.

Bin packing algorithm
This algorithm deals with task assignment of multiprocessor systems. It is based on the condition that the total processor utilization of each processor should not exceed a threshold [1,4]. The threshold value depends on the uniprocessor scheduling scheme used. In this paper, we use bin packing for EDF algorithm. Hence, in this case, uniprocessor utilization should not exceed 1.

Problem statement
This work is an analysis of Earliest Deadline First algorithm on a uniprocessor platform and extends the work to a multiprocessor. EDF is an optimal algorithm but when the scheduling load on the system is very high its performance decreases [3,6]. As mentioned earlier, when all tasks are independent the task that has the earliest deadline is taken at each scheduling point. Unlike the rate monotonic algorithm, this reason makes scheduling using EDF a tedious task. Even today, though EDF is an optimal algorithm, static priority algorithms are used in real time systems [5].
Certain terminology that must be understood before we get into the analysis is discussed below. For each event that occurs, it triggers a task which is to executed for that particular event [2]. Hence depending on the occurrence of the event the task will also

INTERNATIONAL JOURNAL OF RESEARCH SCIENCE & MANAGEMENT

be repeated. This can be either periodic or aperiodic. Each time a task occurs, it is called the instance of task. The time interval between time 0 and the instant where the actual deadline occurs is called absolute deadline, whereas Relative deadline is the time between the start of the task and the actual deadline [1].

Assumptions

In the examples given further, while scheduling tasks following assumptions are made:

1. There are no non-preemptible sections in any task taken and the cost of the pre-emption is insignificant.
2. Only processing necessities are noteworthy; memory, I/O and other source requirements are insignificant.
3. There are no precedence constraints; all tasks are independent.
4. The period of a task is equal to its relative deadline.
5. All tasks taken are released at the same time in the beginning

TABLE II
TERMS USED

Ei	Execution time in ms
Di	Deadline in ms
Pi	Period in ms
pi	Processors
U	Utilisation factor
Tw	Waiting time in ms
Taw	Average waiting time in ms

Mathematical underpinnings

We define the term Utilization, U

$$U = \sum_{i=1}^{n} ei/Pi$$

where ei= execution time

Pi= period

dmax= $max_{1 \le i \le n}\{d_i\}$

P= lcm(P1,…Pn) and $h_T(t)$ be the sum of the execution times ei of all tasks present in the given set T whose have absolute deadlines values less than t.

A given task set containing n tasks is not schedulable by EDF if

- u>1 or
- there exists

t < min { P + d_{max} , $\frac{u}{1-u}$ $max_{1 \le i \le n}$ {Pi - di}} such that $h_T(t) > t$.

For a set of tasks to be schedulable by RM, the total CPU utilization factor of the tasks should not be larger than $n(2^{\frac{1}{n}}) - 1)$

Where n represents the total number of tasks to be scheduled [1].

Let us a set of tasks and schedule it under RM and EDF for the analysis.

TABLE II
Set of Tasks by EDF and RM

Task	pi	ei
T1	4	1
T2	6	2
T3	8	3

INTERNATIONAL JOURNAL OF RESEARCH SCIENCE & MANAGEMENT

The given task set is schedulable under EDF if the necessary condition is satisfied.
i.e.

$$U = \sum_{i=1}^{n} ei/Pi$$

Should be less than or equal to one. Here,
U= 3/8 + 2/6 + 1/4 = 23/24
Here the given condition is satisfied.
We schedule with RM first:

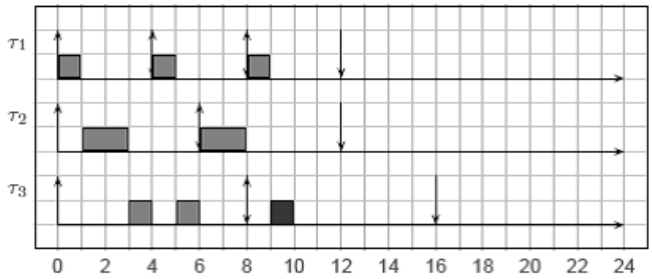

Fig. 1 Rate Monotonic Scheduling

It is seen that task T3 misses its deadline.
The same task when scheduled by EDF,

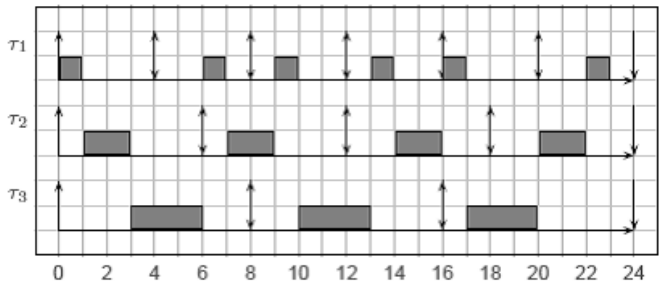

Fig. 2 EDF scheduling

All tasks meet its deadline.
Hence, we can say that EDF algorithm is an optimal scheduling scheme. If a given task set is not schedulable by EDF, then the task set cannot be scheduled by any other existing algorithm. We can also surmise that, there may be a task set whose CPU utilization value is greater than the necessary condition for RM but still schedulable by EDF [5].
Now let us a take a task set, which do not satisfy the conditions for EDF schedulability.

Table III
Set of tasks to be scheduled by bin packing

Task	ei	pi	Ui
T1	5	10	0.50
T2	7	21	0.33
T3	3	22	0.14
T4	1	24	0.04
T5	10	30	0.33
T6	16	40	0.40
T7	1	55	0.02

When scheduled by EDF, following results are shown:

INTERNATIONAL JOURNAL OF RESEARCH SCIENCE & MANAGEMENT

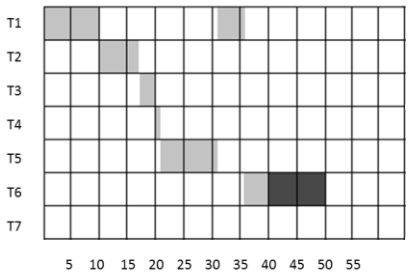

Fig. 3 EDF scheduling misses deadline

Task T6 misses its deadline at time 40 and so does the remaining task T7. This will be continued for the rest of the schedule too. Hence, if scheduled on a multiprocessor system, we can successfully schedule these tasks. Here we use more than one processor and assign task to each processor.

Extending our work to a multiprocessor system, Bin Packing algorithm is used. Tasks are scheduled under the condition that the total value of the utilizations which is assigned to each processor should always be less than or equal to one. Let us take the same set of tasks and schedule according to bin packing algorithm.

The first step is to reorder the tasks in the descending order of their utilization value. Then each task is assigned to a processor until it satisfies the condition that the utilization will be less than 1. If maximum value is reached, the next task is assigned to a new processor [1]. The scheduled task set is as shown below:

Table IV
Set of tasks to be scheduled by bin packing

Tasks	Ui	Processor Pi	Assignment vector
T1	0.50	P1	0.50
T6	0.40	P1	0.90
T2	0.33	P2	0.90,0.33
T5	0.33	P2	0.90, 0.66
T3	0.14	P2	0.90, 0.80
T4	0.04	P1	0.94, 0.80
T7	0.02	P1	0.96, 0.80

Hence the final schedule will be:
P1= T1, T6, T4, T7
P2= T2, T5, T3
Hence, the task set which was not schedulable by EDF on a uniprocessor system was successfully scheduled on a multiprocessor platform using Bin Packing algorithm.

Simulation and implementation

In this section we analyse the scheduling on a real time system. For the analysis of EDF, coding has been done using static values. The user can enter values including the number of tasks which is to be scheduled. According to the entered values tasks will be created and scheduled.

We take the following set of tasks and schedule it.

INTERNATIONAL JOURNAL OF RESEARCH SCIENCE & MANAGEMENT

Table V
Set of tasks to be scheduled by EDF

Task	pi	ei
T1	10	5
T2	22	3
T3	24	1
T4	55	1
T5	60	3
T6	90	9
T7	95	17

It was implemented using GCC compiler and on Linux platform. The output of the task being scheduled using the static values is as below:

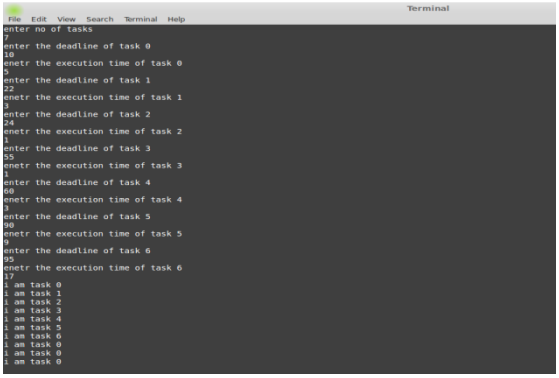

Fig. 4: Scheduling using static values

A comparison of both the implementation is done. It showed

1. The program which used static values used less CPU Resources.

Average waiting time (Taw):

Waiting time of each task when scheduled using EDF and bin packing can be calculated [7].

Calculating the waiting time of each task when scheduled using EDF, we get the following values:

T2=10
T3=13
T4=33
T5=39
T6=60
T7=69

The average waiting time, Taw=37.33

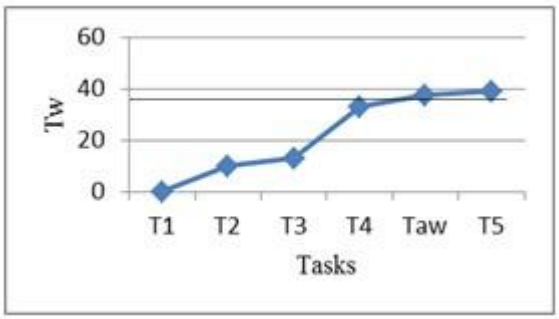

Fig. 5: Waiting time of tasks when scheduled by EDF

INTERNATIONAL JOURNAL OF RESEARCH SCIENCE & MANAGEMENT

The average waiting time when task set scheduled using bin packing algorithm is as shown below:

T2
3=7
T4=10
T5=10
T6=21
T7=43
T1 and T2 will have 0 waiting time. Hence average waiting time in this case,
Taw=18.2

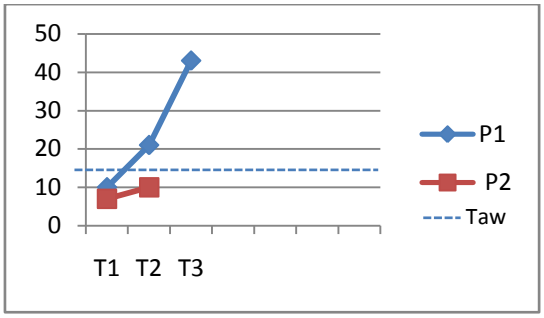

Fig. 6: Waiting time of tasks when scheduled by Bin Packing

Time complexity
The code implemented has the following complexities:
User time (in sec):0.00
System time (in sec): 0.00
Percentage of CPU this job got: 0%
Elapsed wall clock time: 1:06:64
Average shared text size (k bytes): 0
Maximum resident set size (k bytes): 512
Voluntary context switches: 16
Involuntary context switches: 3
Swaps: 0
File system inputs:0
File system outputs: 0
Page size (bytes): 4096
Exit status: 7

Conclusions

In this paper, a complete analysis of scheduling algorithms, EDF in particular was presented. Even though EDF is the most optimal algorithm present, it is still not used widely because of its overhead and schedulable complexities. Implementation of EDF on a uniprocessor system along with a comparative study of Bin Packing showed how a set of tasks is scheduled on a multiprocessor platform. With these understandings the work can be extended to reducing the complexity when implemented on a multiprocessor system. Since it is only EDF that uses the processor to the maximum utilization possible, if EDF can be used in both overloaded and under loaded conditions, it will give the maximum possible outputs.

References

1. Real – Time Systems, Kang G, C. M. Krishna.
2. A Comparative Study of Scheduling Algorithms for Real Time Task M.Kaladevi, M.C.A.,M.Phil.,1 and Dr.S.Sathiyabama, M.Sc.,M.Phil.,Ph.D,2 International Journal of Advances in Science and Technology, Vol. 1, No. 4, 2010.
3. A non-preemptive scheduling algorithm for soft real-time systems , Wenming Li, Krishna Kavi *, Robert Akl, Science Direct Computers and Electrical Engineering 33 (2007) 12–29
4. The improved EDF scheduling algorithm for embedded real time system in the uncertain environment, Xiaojie Li, Xianbo He, 3rd Internantional Conference on Advanced Computer Theory and Engineering(ICACTE)), 2010

INTERNATIONAL JOURNAL OF RESEARCH SCIENCE & MANAGEMENT

5. Real Time Systems, Jane W. S. Liu
6. Characteristics of EDF schedulability on Uniform Multiprocessor, Funk, S., Baruah, S., 15th Conference on Real- Time Systems, 2003. Proceedings.
7. Schedulability Analysis for Real-Time Systems with EDF Scheduling , Fengxiang Zhang, Student Member, IEEE, and Alan Burns, Senior Member, IEEE, IEEE transactions on computers, vol. 58, no. xx, xx 2009.