Overload Avoidance Algorithm for Real-Time Distributed System

A. F. M. Suaib Akhter [†], Mahmudur Rahman Khan[††], Md. Shariful Islam[†††]

Institute of Information Technology, University of Dhaka, Dhaka, Bangladesh [†,††]
Assistant Professor, Institute of Information Technology, University of Dhaka, Dhaka, Bangladesh [†††]

Abstract:
For real-time distributed systems dynamic scheduling has always been a challenging problem. Most of the systems handle overloaded condition after failure of some processes. Thus after recovery the system has to be involved in repairing the lost jobs. In the proposed algorithm the system will never enter in the overloaded state which will ensure that, none of the process will be failed which have already entered in the system. Proposed algorithm is a combination of a dynamic and a static scheduling algorithm. It uses EDF (Earliest Deadline First) which is the most widely used algorithm for dynamic scheduling and RM algorithm is the static algorithm for scheduling in such a condition when the probability of entering in overloaded condition is high. In regular load system will use EDF, but when the load reached in a certain level system will use RM algorithm until the system back to the safe state. A variable is assigned with each process to check whether the process will enter in the system and at the same time it decides when the system has to switch from one algorithm to another. The value of the variable depends on the current system load and remaining capacity of the system. By this way the algorithm will ensure resource utilization, efficiency and high performance of the system in any condition.

Key words:
Distributed system, EDF, Real-Time System, Scheduling algorithm.

1. Introduction

A real-time system is one that must process information and produce a response within a specified time, else risk severe consequences, including failure. If we consider the ATM network as an example of real time sys-tem, when the system become overloaded some transaction fails. There are many failed transaction where the system already have debited the amount from the customer account. Reversal i.e. credits the amount into customer account again will automatically start by the system just after get recovery from overloaded condition. On that time system again may fall into overloaded condition if number of reversal is high and by this way system have a high probability to enter into deadlock. Thus necessity of an overload avoidance algorithm is high, which will ensure that there will be no transaction failure. This will remove the hassle of Reversal which will make the system more efficient.

There are several scheduling algorithm to handle real time systems. Each of the algorithms provides best effort to make the system to increase the success ratio and effective CPU utilization as the performance of the algorithm measured using these two terms. EDF is the most widely used algorithm for handling real-time traffic. In EDF, jobs either all have deadlines until the beginning of service or deadlines until the end of service. In the former case, EDF is known to be optimal and, in the latter case, it is optimal if preemption is allowed [1].
In overload situations, EDF algorithm will result in rapidly decreasing of the system performance, even bringing in the domino effect [2]. For distributed real time system use of a static algorithm with EDF proved effective specially in overloaded condition. There are several methods proposed by the researchers to solve this problem. Static algorithm like RM algorithm performs well in overloaded condition. Thus these algorithms are used beside EDF to make the scheduling more efficient.
Previously proposed algorithms switched from EDF to another static algorithm after failure of more than two processes successively [3]. This will decrease the CPU utilization as well as the success ratio of the system. Recently some researcher proposed some algorithm by combining both the static and dynamic algorithm together to make the recovery process faster. Scheduling performance of these algorithms increases, but still these are not overload free i.e. the system has to use their resources to recover the partially done jobs.
Newly proposed algorithm will remove the possibility of failure by using a variable. A process will be served with probability *p*. A process with lower probability has less chance of complete. So whenever the probability will reach into a certain level the system will not allow any other process to enter in the system.
The scheduling will be EDF but after certain value of p the system will considered as overloaded and there is a chance of transaction failure. Then another static algorithm will come to schedule tasks in overloaded condition. Static algorithm will also calculate *p* and after crossing a certain value of *p* system will again perform scheduling by EDF.
In the rest of the paper the scheduling technique will be described as well as its advantages over other algorithms and performance analysis.

Manuscript received September 5, 2012
Manuscript revised September 20, 2012

2. System Model

The system considered here is a client/server distributed system with soft real-time constraints. It is basically a loosely coupled homogeneous system where each client can make scheduling decision independently. Each of the system has enough memory resources to support CPU in totally loaded condition.
All the tasks assigned here are preemptive and the system knows about the deadline and required computation time of the task when the task is released. System will check the current load whenever a new process is about to get chance in the system.
It will be ensured that the system will never enter in 100% loaded condition. Shifting from dynamic to static algorithm will be done when there is chance of overload. Thus served processes will not fail or wait for shifting rather than they will get chance according to their deadline.

3. Designing steps of the algorithm

In soft real-time systems, each task has a positive value. The goal of the system is to obtain as much value as possible. If a task succeeds, then the system acquires its value. If a task fails, then the system gains less value from the task ([4], [5]).
It is tried to increase the system performance with the help of a variable. When any process comes to get serve form a client, the system will calculate the average value of the system load for last minute.
The value of p will be:
p = 1 - Average Load of the CPU in last minute.
It will considered that the chance of meeting the deadline = p x 100 %.

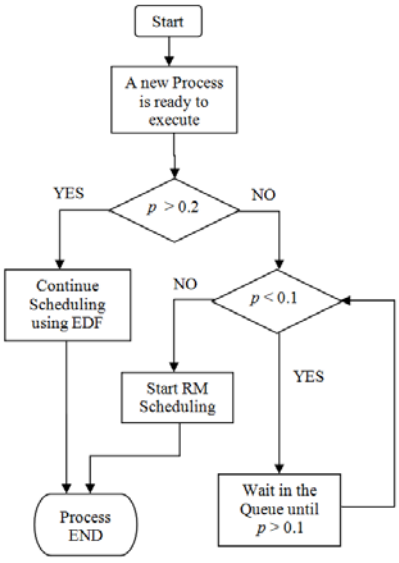


Figure 1: Flow Chart

In normal condition i.e. when the value of p remains above 0.2 or the chance of meeting the deadline is more than 20% the system will follow EDF. But when the value of p becomes less than 0.2, RM scheduling algorithm will come in front to serve the processes statically. RM is a static algorithm which assigns priorities to tasks based on their periods. Limitation of any static algorithm is that they are not as efficient as dynamic algorithm in under-loaded conditions [6]. But RM will perform scheduling until the value of p increases to 0.25 as p shows that the chance of entering overloaded state is removed.
The flow chart shows the details of the algorithm. Whenever a new process is waiting for the service of the system the value of p will be updated according to the system load and available resources. Then the algorithm will decide by which way it will be scheduled.

4. The Overload Avoidance Algorithm

In minimum loaded condition the system will follow EDF to schedule the tasks dynamically, but before overloaded condition the system will shift to RM algorithm which will serve the processes statically. The steps of the algorithms are:

- When a new process is waiting in the queue for service system will check its average load for the last minute by using $uptime system command. The value of p will calculate using this formula:
  p = 1- Average Load.

- If the value of $p > 0.2$ the system will follow EDF for scheduling.

- In EDF scheduling will be done by performing the following equations:
  Q = C – A
  Where Q: Queuing Time, C: Current-Time A: Arrival-Time, then
  R= Q - D
  Where, R: Remaining Time, D: Deadline

- Then the scheduler will check the packet with the smallest positive remaining time to serve [7]. By this way EDF performs its tasks dynamically.

- When the value of p become below 0.2, immediately the system shift to RM algorithm.

RM is a static algorithm which assigns priorities to tasks based on their periods. And RM algorithm continues scheduling until the value of p cross 0.25 or 0.3 depending on the capacity of the system.

4. Performance analysis

Proposed algorithm will proved a very essential one for those real time systems which have to maintain the atomicity criteria hardly. ATM network is one of the examples of this kind of real time system. This algorithm will help the ATM networks in two ways. Firstly, if the system is busy customers have to wait a little, but they will free from money loss due to half done transaction. Secondly, it is not required for the system to refund the amount to the customer's account as no transaction can be remain half-done. It will save system resource and remove approximately 90% reversals i.e. refund transaction.

We have used EDF algorithm which will dynamically handle the workload and its performance is quite good in under-loaded condition. When the performance of the algorithm i.e. success rate (SR) and Effective CPU Utilization (ECU) graph become lower we start RM algorithm so that this flow can't reached to zero state.

The system is said to be overloaded when even a clairvoyant scheduler cannot feasibly schedule the tasks offered to the scheduler [8]. As the system has very less chance to move to the overloaded state the performance will increase obviously.

We have considered the following performance criteria:

1. In real-time systems, deadline meeting is most important and we are interested in finding whether the task is meeting the deadline. Therefore the most appropriate performance metric is the Success Ratio and defined as [9].

$$SR = \frac{Number\ of\ tasks\ successfully\ scheduled}{Total\ number\ of\ tasks\ \ arrived}$$

2. Effective CPU Utilization (ECU) which can be defined as,

$$ECU = \sum_{i \in R} \frac{V_i}{T}$$

where,

• V is value of a task and,

o Value of a task = Computation time of a task, if the task completes within its deadline.

o Value of a task = 0, if the task fails to meet the deadline.

• R is the set of tasks, which are scheduled successfully i.e. completed within their deadline.

• T is total time of scheduling.[3]

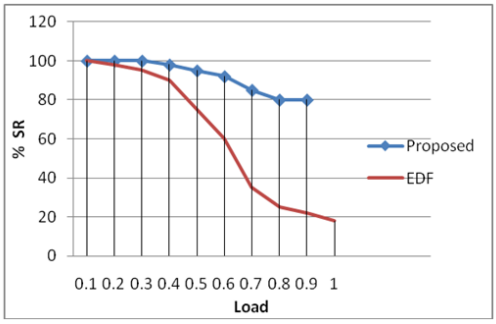
To compare the proposed algorithm with others we have consider different number of processors and by providing different amount of load getting the value of SR and ECU. By plotting the result into the graph the increment of performance can be shown clearly.

4. Results

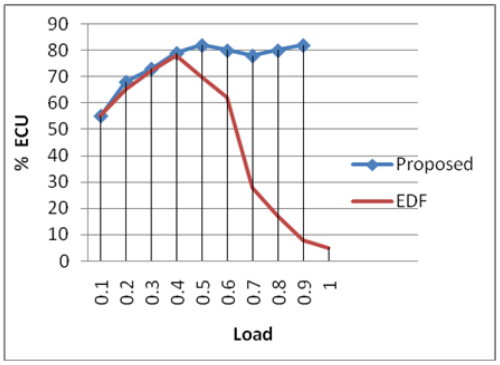
The proposed algorithm performs competitively better than EDF. Using the above SR and ECU calculation the generated graphs clearly shows the improvement. Two different computers having 2 and 5 processors are used to analyze the performance of EDF and the proposed algorithm. It shows that both Success Rate as the well as the CPU Utilization lines are consistent for the proposed algorithm where the lines of EDF goes down after a certain value. Graphs of SR are shown in figure 2 and 4 and ECU is shown in figure 3 and 4. These graphs show the improvement of our scheduling algorithm.

Conclusion:

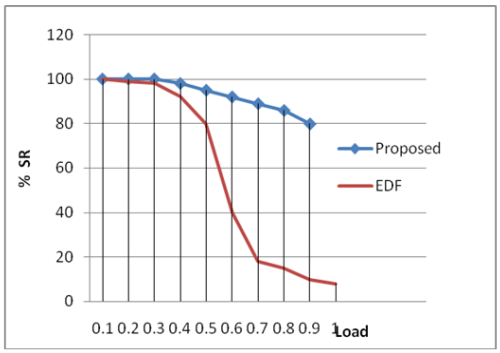
Overloading is a regular problem in real time system and it is wastage of valuable resources. In the proposed algorithm the system will never enter into the overloaded state and by this way it can avoid extra pressure of overload. Failure of any process proved costly for most of the real time system. For these type of system overloading avoidance algorithm is very essential, time consuming and efficient. Although 100% resource utilization is not possible according to this algorithm but still it is more cost effective than any other algorithm as it does not need to pay extra resources due to overload. In near future algorithm will be updated to improve resource utilization.



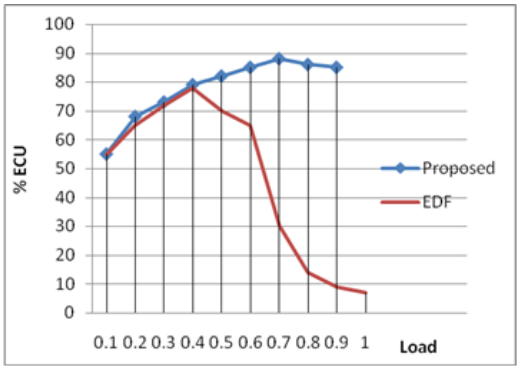
Figute 2: Load Vs %SR when number of CPU= 2



Figute 3: Load Vs %ECU when number of CPU= 2



Figute 4: Load Vs %SR when number of CPU= 5



Figute 5: Load Vs %ECU when number of CPU=5

References

[1] Kargahi M.; Movaghar A., "A method for performance analysis of earliest-deadline-first scheduling policy" In Dependable Systems and Networks, 2004 International Conference on 28 June-1 July 2004, pp. 826-834.

[2] Buttazzo G.,.Spuri M. and Sensini F., "Value Vs. Deadline scheduling in overload conditions", In proceeding of RTSS, 1995, pp. 90-95.

[3] Apurva Shah, Ketan Kotecha"Efficient Scheduling Algorithm for Real-Time Distributed System", In 2010 1st International Conference on Parallel, Distributed and Grid Computing (PDGC - 2010).

[4] K.Kotecha and A. Shah, "Efficient dynamic scheduling algorithms for real-time multiprocessor systems", In proceedings of HPCNCS 08, Fl, 2008, pp. 21-25.

[5] C.D. Locke., "Best Effort Decision Making for Real-Time Scheduling", Ph.d.thesis, Computer Science Department Carnegie-Mellon University, 1986.

[6] Liu C.L and Layland. "Scheduling algorithms for multiprogramming in a hard real-time environment". Journal of ACM. Vo. 20, 1973, pp. 46-61.

[7] Maen Saleh, "Comparing FCFS & EDF Scheduling Algorithms for Real-Time Packet Switching Networks". IEEE jopurnal, 2010, pp. 698-703.

[8] J.W.S. Liu, Real-time systems, Pearson Education, India, 2001

[9] K.Ramamritham, Stankovik J.A. and Shiah P.F, "Efficient scheduling algorithms for real-time multiprocessor systems", IEEE Transaction on Parallel and Distributed Systems,1(2), April1990.

A. F. M. Suaib Akhter who born in Rajshahi, Bangladesh, has completed B. Sc. in the Department of Computer Science and Information Technology (CIT) from Islamic University of Technology(IUT), Gazipur, Bangladesh. With keen interest in distributed system and network security have done several research in these fields. Now he is working as a Lecturer in Computer Science and Engineering(CSE) dept. in Dhaka International University, Dhaka, Bangladesh.

Mahmudur Rahman Khan who born in Rajshahi, Bangladesh, has completed Bachelor of Computer Applications from Bangalore University, India and PGDIT degree from Institute of Information Technology(IIT) under University of Dhaka, Bangladesh.