International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

# Improved AED Scheduling Algorithm for Real-Time System

**Sohel A. Bhura [1], Dr. A. S. Alvi [2]**

[1]Babasaheb Naik College of Engineering, Amravati University, Pusad. Maharashtra, India

[2]Prof Ram Meghe Institute of Technology & Research, Amravati University, Badnera, Maharashtra, India

**Abstract:** ***The design and implementation of real-time database presents many new challenging problems. Compared with conventional database, real-time database have distinct features: they must maintain the coherent data while satisfy the timing constraints associated with transaction. In addition, real-time database must adapt to changes in operating environment and guarantee the completion of critical transaction with favoring changes in the system. With evolution of Earliest Deadline First (EDF) in 1973 by LIU and LAYLAND, laid the path for development of RTDB, it is very inefficient in overloaded workload conditions. Adaptive Earliest Deadline (AED) improves the performance which uses feedback control mechanism to detect overloaded condition. There prevails the risk of losing transaction with extremely high value may cause severe losses to system. A extension of AED called Hierarchical Earliest Deadline (HED) provide solution by establishing the value based bucket hierarchy thus ensuring the completion of high value transaction. Where value assigned reflects the return expected to receive if the transaction commits before its deadline. In this paper we present the comparison of EDF, AED under different workload conditions with calculated examples. We are also proposing a method which will focus on using analysis of arrival-time of transaction, seek-time and transaction size to determine best scheduling algorithm for the current workload, switching and tuning algorithm as necessary to improve performance.***

**Keywords:** AED, EDF, Deadline, HED, Real-Time

## 1. Introduction

Traditionally, A Real time system has two notations of correctness: logical and temporal. In particular, in addition to producing correct outputs (logical correctness), such systems needs to ensure that these outputs are produced at correct time (temporal correctness). However, selecting appropriate methods for scheduling activities is one of the important considerations in the design of a real time system; such methods are essential to ensure that all activities are able to meet their timing constraints. These timing constraints are usually specified using a deadline, which corresponds to the time by which a specific operation must be completed [1].

Real time systems can be broadly classified as hard or soft depending on the criticality of deadlines. In hard real time all deadlines must be met; equivalently, a deadline miss results in an incorrect system. On the other hand, in soft real-time system, timing constraints are less stringent; occasional deadline misses do not affect the correctness of the system.

A real-time system is typically composed of several or sequential tasks with timing constraints. In most real-time systems, tasks are invocated repeatedly: each invocation of task is referred to as a job; and the corresponding time of invocation is referred to as the job's release time or job's deadline [2].

A Real-Time Database System (RTDBS) is a transaction processing system that is designed to handle transactions with the timing constraint. Several previous RTDBS as in studies had been done to address the issue of scheduling transactions with the objective of minimizing the number of miss transaction. A common observation of these studies has been that, if we assigning priorities to transactions according to an Earliest Deadline policy minimize the number of miss transactions in systems operating under low or moderate levels of workload condition. This is due to earliest Deadline giving the highest priority to transactions that have the least remaining time in which to complete. These studies have also observed that the performance of Earliest Deadline steeply degrades in an overloaded system. This is because, under heavy loaded workload condition transactions gain high priority only when they are close to their deadlines. Giving highest priority to the transaction which is close to miss its deadline may cause delay and may not leave sufficient time for transactions to complete before their deadline.

Hence, a question arises, how to improve the performance in overloaded workload conditions, how to determine the overloaded and under-loaded condition and take the appropriate measure to ensure the performance? There was a need of algorithm which would respond to the different condition and completing the transactions with favouring changes in the system. Adaptive earliest deadline first (AED) is the priority assignment algorithm which stabilize the overloaded conditions. It uses cost-effective feedback control mechanism to achieve performance guarantees in unpredictable environments. While early research on real-time scheduling was concerned with guaranteeing complete avoidance of undesirable effects such as overload and deadline misses, adaptive real-time systems are designed to handle such effects dynamically. There are real-time database applications that may assign different values to transaction. Where the value of a transaction reflects the return the application expects to receive if the transaction commits before its deadline. The primary goal of the RTDBS is to maximize the value realized by the in-time transactions and minimizing the number of miss transaction in the system is secondary concern. Hierarchical earliest Deadline (HED) is an algorithm which uses both values and deadline characteristic of the transaction to schedule them [3].

In this paper we introduces a new algorithm I-AED i.e. Improved Adaptive Earliest Deadline which uses the transaction parameter to determine the workload condition using newly proposed method. In this paper we restrict our attention toward the study of different workload condition and uses the appropriate measure to adapt these conditions.

## 2. Proposed Algorithm

The proposed algorithm I-AED will use a method which will focus on using analysis of arrival-time of transaction, seek-time and transaction size to determine best scheduling algorithm for the current workload, switching and tuning algorithm as necessary to improve performance. This algorithm will use proposed method to determine the overloaded, under-loaded condition at the time when transactions will enter in the system and utilize the algorithms accordingly.

### 2.1 Definition of Problem

#### a) Overload Condition

A real-time system is a processing system that is designed to handle workloads whose transactions have completion deadlines. The objective of the real-time system is to meet these deadlines; that is, to process tasks before their deadlines expire. Therefore, in contrast to the conventional computer systems where the goal is on satisfying the timing constraints of tasks. Under ideal condition, transactions never miss deadlines and this behaviour would be as expected. In reality, however, unanticipated emergency conditions may occur where the processing required to handle the emergency exceeds the system capacity, thereby resulting in missed deadlines. The system is then said to be in overload workload condition. If this happens, it is important that the performance of the system degrades [4].

**Example.** To illustrate the overloaded condition we are using this example in which five transactions T1, T2, T3, T4 and T5 arrives in the system at arrival time zero (0) with its relative deadline 90, 60, 120, 150 and 210 respectively. Average execution time for T1 is 45, T2 is 30, T3 is 60, T4 is 75 and T5 is 105.

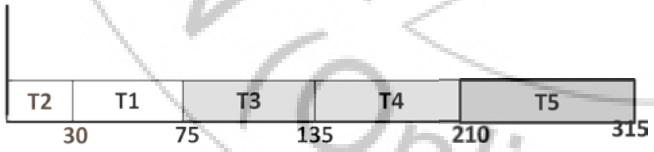


**Figure 1:** Timing Diagram

#### b) Timing Diagram

According to Figure 1 timing diagram for EDF explained above we observed that transaction T2 is executed first with HIT, T1 executed next with HIT, followed by T3 which is MISS and subsequently all transactions MISS as they all missed their deadline. From this we conclude that if "Arrival Rate" of system increases in small instance of time then number of miss increases and hence it depict the overload situation.

### 2.2Proposed Model

As we proposed, our model will utilize:

1) Arrival Time: Time at which transaction request arrives.
2) Seek Time: Time require by the disk head to reach the desired bock.
3) Transaction size: Number blocks per transaction request.

To check the Feasibility of the transaction and calculate the Effective Disk Utilization which will be used for determination of overload and under-load workload conditions.

Parameter Related to I-AED:

**Table 1:** Parameters considered

| *Parameter* | *Description* |
|---|---|
| Tid | Unique ID assigned to transaction request |
| It | Random Integer value assigned to transaction individually |
| TranSize | Transaction Size ie. No. of blocks in the transaction |
| AvgExt | Average execution time of the transaction |
| St | Seek time |
| TTT | Total transaction time |
| T.a | Request time or arrival time |
| T.e | Execution Time |
| T.d | Deadline Assigned to transaction |
| T.s | Slack Factor |
| T.er | Remaining time for execution of transaction |
| Du | Disk Utilization Factor |
| Feasibility% | Percentage Ratio of Feasible transaction in system |
| Mean-Du | Mean of Disk Utilization Factor |

### 2.3 Feasibility Test of Transaction:

In our model, each input transaction T is independent of all other transaction and is completely characterized by three attributes:

1) T.a = the request time
2) T.e = the execution requirement
3) T.d = the relative deadline, often called as the deadline

The significance of these parameters is that transaction T, for successful completion, needs to be allocated the processor for T.e units of time during the interval [T.a, T.a + T.d).

We assume that the system has knowledge of transaction parameters only at the instance when it makes the service request at time (T.a). Transactions that complete execution by their deadlines are of value to the user application; that is, all deadlines are firm.

Slack Factor = T.s = T.d/T.e

Slack factor of transaction T is defined to be the ratio T.d/T.e and is denoted by T.s; it is a quantitative indicator of the tightness or slackness of the transaction deadline. It is niggling to see that it is necessary $T.s \geq 1$ for it to be at all possible to complete a transaction before its deadline. In this study, we consider transaction sets where it is known a priori

that all transaction in the transaction set will have a slack factor of at least f, where f≥1 is a calculated constant.

A task T is said to be active at time-instant t if:

1) It has requested service by time t (i.e., T.a≤ t),
2) Its service is not complete (i.e., T.er > 0, where T.er is the transaction's remaining service requirement), and
3) Its deadline has not expired (i.e., t < T.a + T.d).

An active task T is feasible at time t if T.er ≤ (T.a + T.d - t); that is, it is still possible to meet the task's deadline.

**2.4 Disk Utilization Factor**

This factor is used to indicate the ratio of time for which the disk is busy in seeking to access the desired data as compare to accessing the required data.

EXAMPLE: If the current position of head is at location 30 and has to access the transaction of block size 5 is present at the location 3456, then time required by disk head to reach to desired block at location 3456 from location 30 will very large as compare to the time required for accessing the transaction block[5].

Hence, in a system if majority of transaction posses same condition then large amount of time is require for seeking as compare to accessing and processing of transaction. So there is chance of increase in number of transaction which will miss their deadline, resulting to overload workload condition.

Disk utilization Factor is denoted by: Du

$$Du = \frac{seek\ time}{average\ execution\ time}$$

**2.5 I-AED Algorithm**

Assumptions: Basic parameter; that is; arrival time, seek time and transaction size are known when transaction enters into the system.

The main steps of algorithm are outlined below:

Step 1: unique transaction Id is assigned to transaction when it enters in system.
Step 2: Determine the Feasibility of transaction and Disk Utilization Factor for each transaction in the system.

a) Feasibility Test Of Transaction
IF (T.er ≤ (T.a + T.d - t) then
Transaction is Feasible.
ELSE
Transaction is not feasible

b) Disk Utilization Factor

$$Du = \frac{seek\ time}{average\ execution\ time}$$

Step 3: Compute the percentage of feasible transaction.

$$Feasibility\% = \frac{No.of\ feasible\ transaction}{Total\ no.of\ transaction} * 100$$

Step 4: Compute the Mean disk utilization factor

$$Mean\text{-}Du = \frac{sum\ of\ Du\ of\ all\ transactions}{Number\ of\ transaction}$$

Step 5: Determining the workload condition and applying the algorithm accordingly

```
IF (Feasibility% < 95) then
 {
 System is in overload condition
 Use AED algorithm.
 }
 ELSE
 {
 IF (Mean-Du > 1)
 {
 System is in overloaded condition
 Use AED algorithm.
 }
 ELSE
 {
 Use EDF
 }
}
```

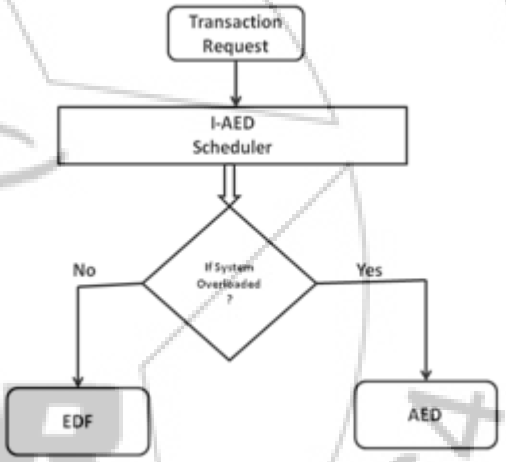


**Figure 2:** I-AED Scheduling Model

I-AED scheduler as explained in the figure-5 uses the result of feasibility test of transactions in the system and disk utilization factor and determines the workload condition i.e. under-load and overload condition. I-AED algorithm initially tests the feasibility of all transactions in the system and computes their disk utilization factors, is used to calculate the Feasibility percentage and mean disk utilization factor. In fifth step of the algorithm, if Feasibility percentage less than 95 percent, then system is declared to be overloaded and use AED algorithm. Else it check for Mean-Disk-Utilization factor if it is greater than 1 then system is declared overloaded then use AED algorithm. If above both condition are not satisfied then system is said to be under-loaded and EDF is used to schedule the transactions.

## 3. Conclusion

In this paper our study was focused to understand the different workload situation in the system and explore the method to adapt to these conditions, so that it would result in favoring changes in the system. We have addressed the issue of stabilizing the overload performance of Earliest Deadline in the real-time algorithm for applications with firm

deadline. In earlier studies we have observed that in under-loaded real-time database systems, using an Earliest Deadline policy to schedule tasks results in the fewest missed deadlines. When the real-time system is overloaded, however, an Earliest Deadline schedule performs worse than most other real-time scheduling policies.

We studied the AED algorithm which delivered the better overall performance in both of the workload conditions. The feedback control mechanism of AED accurately estimated the number of transactions that could be sustained under an ED schedule. AED's policy of restricting the use of the Earliest Deadline approach to the HIT group delivered stabilized performance at high loads. In some real-time applications, different transactions may be assigned different values. Setting tradeoff between value and priority is difficult task, it is addressed by HED algorithm where the goal here is to maximize the sum of the values of those transactions that commit by their deadline, and minimizing the number of missed deadlines becomes a secondary concern.

We proposed I-AED algorithm to address this issue, which uses the analysis of transaction parameters, arrival-time, seek-time and transaction-size to determine Feasibility of transaction and disk utilization factors which was used to determine the different workload condition and depending on which EDF and AED where switched to schedule the transaction. I-AED scheduled the transaction in under-loaded condition using EDF and switched over AED whenever overloaded condition was be encountered. We trust to incorporate the method to determine the seek pattern while executing the transaction and use the seek optimization algorithm to address the disk utilization problem. We hope to address the limitation in this paper in our future work.

## References

[1] C. Liu and J.W. Layland, "Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment," Journal. ACM, pp. 104-112,Jan. 1973.
[2] Chenyang Lu, John A. Stankovic, Gang Tao and Sang H. Son "Feedback Control Real-Time Scheduling: Framework, Modeling, and Algorithms" Journal of Real-Time Systems, Special Issue on Control-Theoretical Approaches to Real-Time Computing
[3] Sanjoy K. Baruah,and Jayant R. Haritsa,"Sheduling for Overload in Real-Time Systems "IEEE TRANSACTIONS ON COMPUTERS, VOL. 46, NO. 9, pp. 1034-1039, SEPTEMBER 1997
[4] Jayant R. Haritsa Miron Livny Michael J. Carey" Earliest Deadline Scheduling for Real-Time Database Systems"Computer Sciences Department University of Wisconsin Madison, WI 53706
[5] R. Abbot and H. Garcia-Molina."Scheduling Real-Time Transactions" , SIGMOD Record,Vol. 17,No. 1,March 1988
[6] R. Abbot and H. Garcia-Molina."Scheduling Real-Time Transactions: A Performance Evaluation" ,Proceedings of the 14th VLDB Conference ,Los Angeles, California, March 1988
[7] Ben Kao and Hector Garcia-Molina," Ovierview of Real-Time Database System" Princton University,Princton NJ 08544 USA, Standford University, Standford, CA, 94305 USA (1991)