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

Scheduling of Update Transactions for Real-time Data Freshness using EDF & DSEDF

Gauri Chavan[1], Vidya Gogate[2]

[1]M.E. Student, Mumbai University, Shah & Anchor Kutchhi Engineering College, Chembur, Mumbai, India

[2]Assistant Professor, Shah & Anchor Kutchhi Engineering College, Chembur, Mumbai, India

Abstract: ***The real time data is stored in real time database. To maintain the temporal validity of real time data different real time scheduling algorithms are used. The valid data is further used for different operations in real time sensing and control system or in process control applications. The real time data values are stored in real time database at every instant of time. So previous data value becomes stale. The new data value is called as fresh data. To store the data values at every instant update transactions are used. Each update transaction is having number of update jobs. These jobs are used to store current data values in real time database. The scheduling algorithms like Earliest Deadline First (EDF) and Deferrable scheduling with Earliest deadline first (DS-EDF) are used to maintain the real time data valid. In this paper we show the scheduling of number of update transactions using Horn's algorithm for earliest Deadline first and Deferrable scheduling with Earliest Deadline First algorithms.***

Keywords: real-time data; data freshness; temporal validity; update transactions

1. Introduction

In real time scheduling the update transactions are used to maintain the validity of real time data. Examples of real time data is like the sensor data in process control system. This data is updated at every instant and stored the values in real time database. These data values are used to monitor the current status of the process control system. For example different temperature values in process control system. The update transactions capture the latest status of real time entities & store these values in real time database [1]. Each update transaction is having number of jobs. The problem for update transaction is designed using the validity interval. Previously different algorithms are used to maintain the validity of real time data. These algorithms are half-half, More-Less, Deferrable scheduling with fixed priority. These algorithms are static scheduling algorithms. In this paper we see the scheduling using dynamic scheduling algorithms like Earliest Deadline First (EDF) and Deferrable scheduling with Earliest Deadline First (DS-EDF) [2]. EDF is very basic dynamic scheduling algorithm and if we use EDF with deferrable scheduling then it becomes DS-EDF. To get the schedule using EDF and DS-EDF we use Horn's algorithm [6].

The paper is organized as follows; section 2 reviews the concept of real time data freshness and validity interval. In section 3, we explain Horn's algorithm. Section 4 defines the problem for number of update transactions and explains EDF and DSEDF algorithms.

2. Review of Real-Time Data Freshness

2.1 Real-Time Data Freshness

The sensor update transactions sample the values of real world entities at each instant. The value sampled at that instant is valid then that data is called as fresh data or valid data. It is necessary to refresh the real time data before it becomes invalid. It means the next update transaction should occur before the validity interval expires. Each update transaction is having its validity interval. If next update transaction occur before the validity interval expires then that data is called as valid data and the process of getting valid data is called as real time data freshness.

A data value D_i is sampled at any time t is valid upto $t+V_i$, where V_i is the validity interval. So the real time data value is temporally valid if the sampling time $r_{i,j}$ plus the validity interval length V_i of the data object is not less than t,

$r_{i,j}+V_i \geq t$ (1)

2.2 Feasibility and Optimality Criterion

A valid schedule is a feasible schedule if every job completes by its deadline or in general it meets its timing constraints. The set of jobs is schedulable according to a scheduling algorithm. The real time scheduling algorithm is optimal if the algorithm always produces a feasible schedule if the given set of jobs has feasible schedules [4].

It means if the optimal algorithm fails to find a feasible schedule, we can conclude that the given set of jobs cannot feasibly be scheduled by any algorithm. Commonly used performance measures include the maximum and average tardiness, lateness response time. The right choice of performance measure depends on the objective of the scheduling [4]. The lateness of a job is the difference between its completion time and its deadline. The lateness of a job completes late is positive. The tardiness is zero if completion time is equal to deadline of job; otherwise it is the difference between completion time and deadline time. Response time is the difference between the completion time of a job and the time at which the job is ready for execution [5].

3. Horn's Algorithm

Horn's Algorithm is used to solve the problem of scheduling with a set of n independent tasks, when tasks may have

dynamic arrivals and preemption is allowed. If tasks are not synchronous but can have arbitrary arrival times i.e. tasks can be activated dynamically during execution, then preemption becomes an important factor. In preemptive scheduling algorithm, the scheduler interrupts the currently executing task in order to meet its own deadline.

Horn's Algorithm: Given a set of n independent tasks with arbitrary arrival times, any algorithm that at any instant exceeds the task with the earliest absolute deadline among all the ready tasks is optimal with respect to maximum lateness [6]. The problem object for Horn's Algorithms is as given below:

p = problem ('1|pmtn, r_j | L_{max}) (2) where L_{max} is the optimality criterion that is maximum lateness. '1', indicates it is single processor system, 'pmtn' is used for preemption indicates all jobs are preemptive.

TORSCHE is Time optimization of resources, scheduling. This is Matlab based scheduling toolbox. This toolbox offers a collection of Matlab routines that allow the user to formalize the scheduling problem [6].

4. EDF and DS-EDF Algorithms

4.1 Problem Derived for Real Time Data Freshness

If there are three update transactions like T_{u1}, Tu2, Tu3, where i = 1 to 3. Each update transaction is having number of update jobs denoted as $J_{i,j}$, where i is the number of update transaction and j is number of job. Suppose T_{u1} has 6 jobs like J_{10}, J_{11}, J_{12}, J_{13}, J_{14}, J_{15}. T_{u2} has 3 jobs like J_{20}, J_{21}, J_{22}. T_{u3} has 1 job J_{30}. All jobs in same transactions have same execution time. All first jobs have zero release time. The current job has release time $r_{i,j}$ and deadline time $d_{i,j}$. The deadline time of next update job is derived from this formula, $d_{i,j+1} = r_{i,j} + V_i$. The next deadline $d_{i,j+1}$ is derived from the release time and validity time of current job so that update transactions validity criteria get satisfied i.e. $r_{i,j}+V_i \geq t$. The sampling time of next update job should be less than $d_{i,j+1}$.

4.2 Earliest Deadline First (EDF)

EDF is one of the basic dynamic priority algorithms. In dynamic priority algorithm, the priority of a task can change during its execution. In EDF the priority of a job is inversely proportional to the absolute deadline i.e. the highest priority of a job is the one with the earliest deadline [5].

If two tasks have the same absolute deadlines then chose one of the two at random if release times are same. If release times of two jobs are different then arrange the job which has earliest deadline. The priority is dynamic since it changes for different jobs of the same task. An optimality criterion of EDF is based on maximum Lateness L_i [4]. To get the schedule of Earliest Deadline First we can use Horn's Algorithm. The illustration of EDF is given in section 5.

4.3 Deferrable Scheduling with Earliest Deadline First (DS-EDF)

If we use deferrable scheduling with Earliest Deadline First then it is called as DS-EDF algorithm. This is again a dynamic priority algorithm. In DS-EDF the new or deferred release time is calculated backwards from its deadline [3].

- In DS-EDF the release time of all the first jobs of each update transaction is initialized to zero.
- First update job's deadline of each transaction is considered as its respective validity interval.
- The update jobs are enqueued in ascending order of their deadlines. The job with earliest deadline is always scheduled first.
- Each update job's new release time is calculated by considering its deadline time minus the computation time and if there is any high priority preemption from high priority jobs.
- So the calculated release time is given by the following formula:
  $r'_{i,j}$ = Deadline time – Execution time – High priority preemption. (4)

We get the new release time from above formula. By using new release time and deadline time for each update job we can plot the schedule for DS-EDF using Horn's Algorithm The illustration for DS-EDF is given in Section 5.

5. Result and Analysis

5.1 Example I

If there are 3 update transactions T_{u1}, T_{u2}, T_{u3}, with validity intervals as V_1=6,V_2=15,V_3=47 respectively. The computation time of each update transaction is $C_1 = 2$, $C_2 = 3$, $C_3 = 3$. There are three jobs in first transaction T_{u1} are J_{10}, J_{11}, J_{12}. Two jobs in second transaction T_{u2} are J_{20}, J_{21} and one job in third transaction T_{u3} is J_{30}. The release time and deadline time of each update job is given as below:

J_{10} (0, 6), J_{11} (4, 6), J_{12} (8, 10)
J_{20} (0, 15), J_{21} (10, 15)
J_{30} (0, 47)

5.2 Example II

J_{10} (0, 6), J_{11} (4, 6), J_{12} (8, 10), J_{13} (12, 14), J_{14} (16, 18), J_{15} (20, 22)
J_{20} (0, 15), J_{21} (10, 15), J_{22} (22, 25)
J_{30} (0, 47)

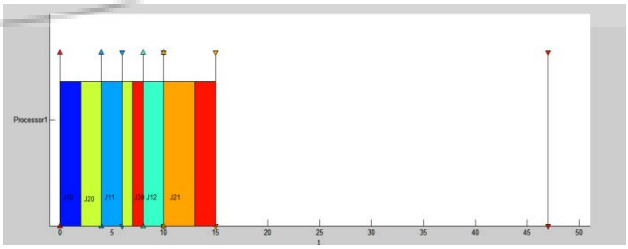

Figure 1: EDF Scheduling for Example I

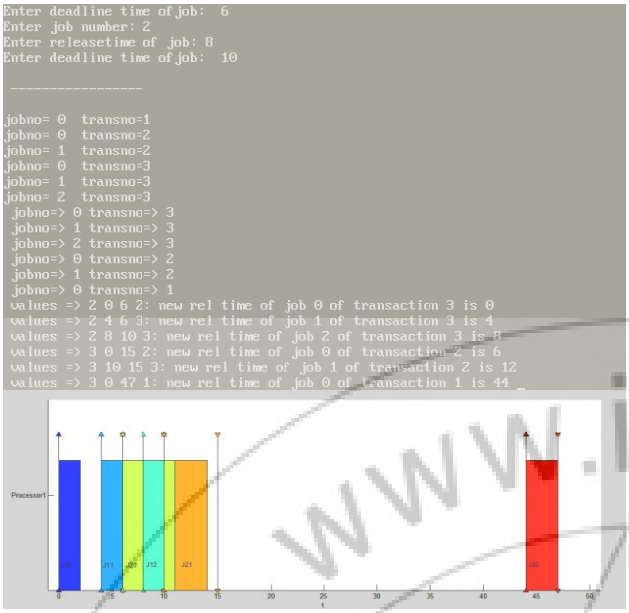

Figure 2: DS-EDF Scheduling for Example I

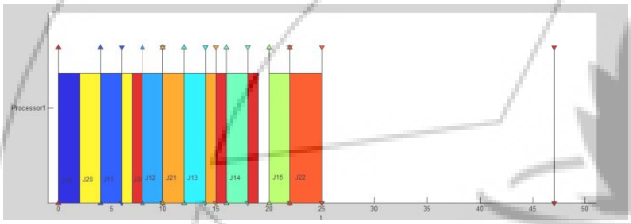

Figure 3: EDF Scheduling for Example II

The schedule of Earliest Deadline First Algorithm for example I is shown in Figure 1. In this figure EDF requires that, each time a new ready task arrives, it is inserted into a queue of ready tasks, sorted by their deadlines. If newly arrived task is inserted with less absolute deadline than the currently executing task then currently executing task is preempted. All jobs in figure 1 are meeting its deadline by using Earliest Deadline First approach. Figure 2 shows the schedule for Deferrable scheduling with Earliest Deadline First Algorithm (DS-EDF). In this figure by using DS-EDF algorithm, the new release time is calculated by considering the deadline constraints and jobs are deferred to meet its deadline. Job, J_{30} has its new release time as 44 because it does not have any other job and all jobs of first and second update transactions are completed before 15 time units. In example II we have increased number of jobs of first update transaction, T_{u1} from 3 to 6 update jobs and in second update transaction, T_{u2} from 2 to 3 update jobs. The schedule of EDF algorithm for Example II is shown in figure 3. All update jobs meet its deadline by using EDF algorithm and maintain the real time data freshness. Figure 4 gives the scheduling of Example II by using DS-EDF algorithm. By using DS-EDF algorithm, the new release time is calculated in such ways that all update jobs meet its deadline and we can maintain the data validity.

5.3 Example III

J_{10} (0, 6), J_{11} (4, 6), J_{12} (8, 10), J_{13} (12, 14), J_{14} (16, 18),
J_{15} (20, 22), J_{16} (21, 26)
J_{20} (0, 15), J_{21} (10, 15), J_{22} (22, 25)
J_{30} (0, 47)

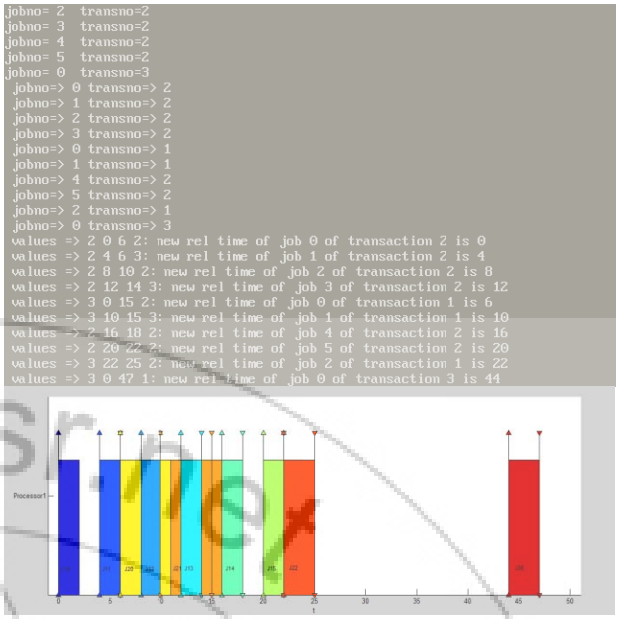

Figure 4: DS-EDF Scheduling for Example II

The figure 5 gives the schedule for Example III using DS-EDF approach. In update transaction first, T_{u1} total 6 update jobs are used. Job J_{16} missed its deadline as the actual release time of this job is less than the deadline time of the previous job. As the update job missed its deadline the real time data is not valid and we cannot maintain the data freshness. This problem is overcome by using deferrable scheduling with Least Actual Laxity First Algorithm. In this algorithm the laxity time of each job will be needed to schedule the job whose deadline is missing.

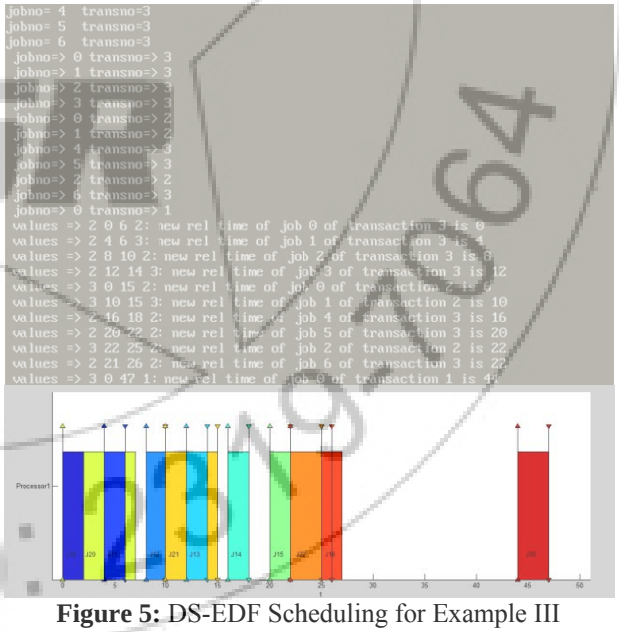

Figure 5: DS-EDF Scheduling for Example III

6. Conclusion & Future Scope

EDF & DS-EDF algorithms are used to schedule the different update transactions without missing its deadline. So that the update transactions scheduling for the defined problem keep the real time data fresh. If we compare the schedule of EDF and DS-EDF then we can say that in EDF all tasks are scheduled one after the other without any free time slot by considering the earliest absolute deadline. While

in DS-EDF the new release time calculated is deferred the update jobs in such a way that it minimizes the processor workload.

In Example III one of the update jobs is missing its deadline. This problem can be overcome by using other scheduling technique that is Deferrable scheduling with Least Actual Laxity First (DS-LALF). In this algorithm we will use the laxity time of the update job so that the update job which is missing its deadline will be completed before its deadline time or exactly on the deadline time to get the valid real time data.

References

[1] M. Xiong, S. Han, K.-Y. Lam, and D. Chen, "Deferrable "Scheduling for Maintaining Real-Time Data Freshness: Algorithms, Analysis, and Results," IEEE Trans. Computers, vol. 57, no. 7, pp. 952-964, July 2008.
[2] S. Han, D. Chen, M. Xiong, K.-Y. Lam, A.K. Mok, and K.Ramamritham, "Schedulability Analysis of Deferrable Scheduling Algorithms for Maintaining Real-Time Data Freshness,Technical Report TR-11-38. http://apps.cs.utexas.edu/tech_reports/reports/tr/TR-2055. pdf, 2011.
[3] "On Co-Scheduling of Update and Control Transactions in Real-Time Sensing and Control Systems: Algorithms, Analysis, and Performance" by Song Han, Member, IEEE, Kam-Yiu Lam, Member, IEEE, Jiantao Wang, Student Member, IEEE, Krithi Ramamritham, Fellow, IEEE, and Aloysius K. Mok, Member, IEEE. IEEE TRANSACTIONS OCTOBER 2013.
[4] "Real-Time Systems" by Jane W.S. Liu, Pearson Education.
[5] "Real-Time Systems" by C.M.Krishna & Kang G. Shin, Tata McGraw- Hill.
[6] M. Kutil, P. Sucha, M. Sojka, and Z. Hanzalek, TORSCHE Scheduling Toolbox Manual, February 2006. http://rtime.felk.cvut.

Author Profile

Gauri Chavan pursuing M.E. degree in Electronics Engineering from Shah and Anchor Kutchhi Engineering College, Chembur, Mumbai, Maharashtra, India.

Prof. Vidya Gogate Assistant Professor in Electronics Engineering Department in Shah and Anchor Kutchhi Engineering College, Chembur, Mumbai, Maharashtra, India