International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169
Volume: 4 Issue: 5 247 – 250

Application of Fuzzy Logic to Real Time Systems for Task Scheduling

Shyamal G. Mundada
Assistant Professor, Department of Computer Science Engineering
Ramdeobaba College Of Engineering and Management
Nagpur, India
shyamal.laddha@gmail.com

Abstract— Many scheduling algorithms have been proposed to ensure the time constraints of real-time processes. Scheduling decision of these algorithms is usually based on various timing properties such as execution time, deadline which are assumed to be crisp. Traditional real-time scheduling algorithms often consider that these properties of the tasks are completely known a priori. This, however, is usually not the case. A natural approach to handle the uncertainty is the use of fuzzy techniques. The vagueness of these parameters can be solved by making use of fuzzy logic to decide in what order the tasks should be executed for better utilization of the system and as a result reduce the chance of a task deadline being missed.

Keywords- *Real Time Systems, Task Scheduling, Fuzzy Logic*

*****

I. INTRODUCTION

Real Time Systems are the computing Systems for which a timely response to the external situation within a specified time frame is a must. Real time computing has become an important part in Telecommunications systems, factories, defense systems, aircraft, airports, medical equipment, space stations and underwater vehicles. Time is the most important resource in Real Time Systems. The purpose of a real-time system is to have a physical effect within a chosen time-frame. Computing applications of real time systems (time critical systems) require that it should satisfy explicitly time constraints of the tasks and these timing constraints play an important role for safe operation of Real Time Systems.

In this paper an idea of fuzzy techniques and use of these techniques for real time scheduling purpose is discussed. Section 2 will give a very brief summary of Real Time Task Model. In section 3, traditional real time scheduling techniques are discussed. In Section 4 & 5 how the ideas of fuzzy models and fuzzy set theory could be applied to real-time scheduling is discussed.

II. TASK MODEL

A Real Time System is represented as a general model consisting set of tasks

$$T = \{T_i \mid i = 1,2,3, \ldots . n\}$$

Each task happening in a real-time system has some timing properties. These properties should be measured while scheduling tasks on a real-time system. Timing properties of a real time task are defined as follows

1. *Release time:* Time at which the task is ready for execution.
2. *Deadline*: Time by which execution of the task should be finished, after the task is released.
3. *Worst case execution time*: Maximum time taken to finish the task, after the task is released. The worst case execution time is also referred to as the *worst case response time*.
4. *Execution time*: Time taken without interruption to complete the task, after the task is released.
5. *Period:* Time quantum to repeat if the task is Periodic task.

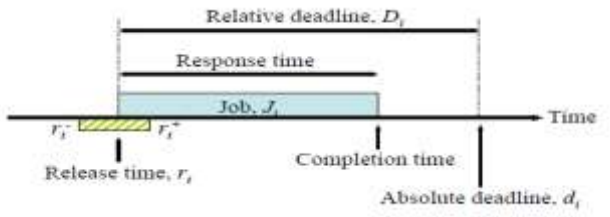

Fig. 1 Timing Properties of Task

The execution of a task may or may not be interrupted over the set of tasks depending on priority which constrains the order of execution. Specially, the execution of a task will begin only when the execution of all its predecessors is completed. Predecessors are identified according to the precedence constraints. The system on which the jobs are to be executed is characterized by the amounts of resources available. In advanced real-time systems, these points are preferred:

- Combining hard, firm, and soft real-time activities, to get optimum result by applying dynamic scheduling policies.
- Task scheduling for dynamically adaptive, reconfigurable, reflexive real-time system.
- System should be reliable, secure and safe.

For any given real-time system, certain properties of the system and the tasks occurring in it are to be considered. Based on timing properties real time tasks are classified as

1. *Soft/Hard/Firm real-time tasks*

The real-time tasks are classified as hard, soft and firm real-time tasks. Tasks having hard deadline are hard real time tasks and with soft deadline as Soft real time Tasks. A deadline is said to be hard if the consequences of not meeting it can be disastrous: causing breakdown of entire system or generating even loss of human lives. A deadline is said to be soft if the consequences of not meeting it makes a utility function decreasing over time.

2. *Periodic/Aperiodic/Sporadic tasks*

Periodic tasks are real-time tasks which are released regularly at fixed intervals. A periodic task is one that repeats after a certain fixed time interval. The precise time instants at which periodic tasks re cur are usually demarcated by clock interrupts. For this reason, periodic tasks are sometimes referred to as clock-driven tasks. The fixed time interval after which a task repeats is called the period of the task.

An aperiodic task is a stream of jobs arriving at irregular intervals. A sporadic task is an aperiodic task with a hard deadline and a minimum inter-arrival time. .

3. *Preemptive/Non-preemptive tasks*

In some real-time scheduling algorithms, a task can be preempted if another task of higher priority becomes ready for execution. In contrast, the execution of a non-preemptive task should be completed without interruption, once it is started even is higher priority task comes.

4. *Fixed/Dynamic priority tasks*

In priority driven scheduling, a priority is assigned to each task. Priorities can be assigned statically or dynamically while the system is running.

III. REAL TIME SCHEDULING

Basically, the scheduling problem is used to find out a schedule for the execution of the tasks so that all the tasks must be completed before the overall deadline. Goals to be considered in scheduling a real-time system:

- Should Meet all the timing constraints of the system
- Should Prevent real-time access to shared resources and devices
- Should attain a higher degree of utilization while satisfying the timing constraints of the system
- Should reduce the cost of context switch caused due to preemption
- Should Reduce the communication cost in real-time distributed systems by finding the optimal way to decompose the real-time application into smaller portions in order to have the minimum communication cost between mutual portions (each portion is assigned to a computer).

For given set of tasks, the general scheduling problem is used to determine an order according to which the tasks are to be executed so that various constraints are satisfied. For scheduling a real-time system, information, such as deadline, minimum delay, maximum delay, run-time, and worst case execution time of each task is required. A majority of systems assume that much of this information is available at early stage and, hence, are based on static design. However, some of the real-time systems are designed to be dynamic and flexible.

Real-Time scheduling can be categorized into hard vs soft. Hard real-time scheduling can be used for soft real-time scheduling. Figure 2 given below represents taxonomy of real-time scheduling algorithms.

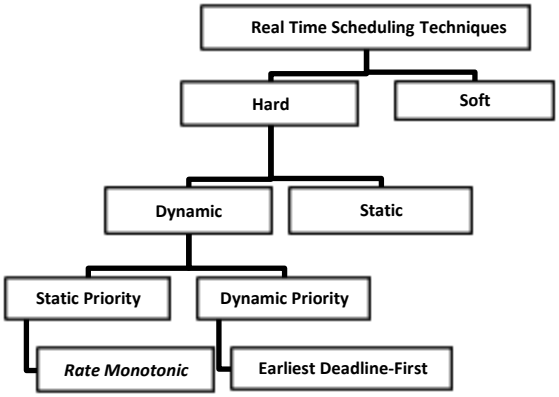

Fig. 2 Taxonomy of Real Time Scheduling Algorithms

Hard real-time scheduling can be classified into two types: static and dynamic. In static scheduling, the scheduling decisions are made statically at compile time and run-time schedule is generated off-line based on the previous knowledge of task-set parameters, e.g., release times, maximum execution times, precedence constraints, and deadlines. So run-time overhead is small. On the other hand, in dynamic scheduling decisions are made dynamically at run time, selecting one out of the current set of ready tasks. Dynamic schedulers are flexible and adaptive. But significant overheads can be incurred because of run-time processing.

A. Dynamic Scheduling Algorithms

Schedulability test is used by dynamic schedulers to determine whether a given set of ready tasks can be scheduled or not to meet their deadlines. Some of the scheduling algorithms and their Schedulability criteria are discussed below.

1.Rate Monotonic Algorithm (RMA)

Rate monotonic algorithm comes under dynamic preemptive scheduling based on static priorities. Static priorities are assigned in rate monotonic algorithm based on task periods. Here task period is the time quantum after which the tasks repeat. Task arrival rate is defined as inverse of period. For example, a task with a period of 10ms repeats itself after every 10ms time quantum. The task having shortest period has the highest priority, and the task having longest period has the lowest static priority. At run time, the dispatcher selects the task with the highest priority for execution. According to RMA a set of periodic, independent task can be scheduled to meet their deadlines, if the sum of their utilization factors of the n tasks is given as below.

$$\sum \left(\frac{C_i}{T_i}\right) \leq U(n) = n\left(2^{1/n} - 1\right)$$

$C_i =$ *Worst case execution time of task*

$T_i =$ *Period of task*

$U(n) =$ *Utilization bound for n tasks*

2.*Earliest Deadline-First Algorithm (EDF)*

Earliest Deadline-First algorithm is an optimal dynamic preemptive algorithm based on dynamic priorities. In this

after any significant event occurs, the task with the earliest deadline is assigned the highest priority. A significant event in a system can be blockage of a task, invocation of a new task, completion of a existing task etc. The processor utilization can be up to 100% with Earliest Deadline-First algorithm, even when the task periods are not multiples of the smallest periods. The dispatcher operates in the same way as the dispatcher for the rate monotonic algorithm.

Schedulability test for EDF is:

$$\sum_{i=1}^{n} {}^{e_i}/_{T_i} \leq 1$$

Where

$$e_i = Execution\ time\ of\ task$$
$$T_i = Period\ of\ task$$

B. Static Scheduling Algorithms

In static scheduling, scheduling decisions are made statically during compile time. In this type of scheduling, it is assumed that parameters of all the tasks are known a priori and a schedule is built based on this. After finalizing a schedule, it cannot be modified online. Static scheduling is not suitable for dynamic systems. This scheduling is beneficial for Process control applications, where sensor data rates of all tasks are known at early design stage. In static scheduling. priority is of no use. Based on task arrival pattern a time line is built and embedded into the program and no change in schedules are possible during execution.

IV. Fuzzy Scheduling

Scheduling of task is basically done by using timing properties or characteristics of task which are assumed to be crisp or firm. These characteristics of any task are known at early phase of real time system design. But in some circumstances these characteristics are uncertain, means only an approximate idea available at early phase i.e uncertainty or impreciseness is associated with the timing constraints of any real time task. To address this uncertainty in timing constraints, several modeling techniques are proposed. These modeling techniques provide more realistic information than the simpler crisp analysis.

Various Modeling Techniques considering Uncertainty are

1. Probabilistic Approach[3]
2. Fuzzy Approach
3. Interval Approach[4]

In case of Probabilistic Modeling, probabilistic information is not available early phases of system design and hence the strong mathematical requirements of probabilistic analysis are not fulfilled. While in fuzzy modeling, computations are simpler and expert's knowledge can be utilized for task modeling. Also, it is bit faster in computations and bit more flexible in modeling as compared to other techniques. It is also possible to model parameters by considering wide variety of fuzzy membership functions without increasing complexity while analyzing the system.

Two kinds of real time characteristics or timing properties of task can be considered for fuzzification [1]

1. Execution time of Tasks
2. Real time constraints

Execution time of task represents a measure of a particular task characteristic and it is one of the most concrete and obvious place where vagueness has to be considered. It is often assumed that details of task execution time is known a priori and precisely used in the scheduling algorithms. But unfortunately, this hypothesis is unrealistic and sometimes, completely wrong. By modeling the execution time of a task as a fuzzy number, the uncertainty of execution time can be taken into consideration.

The nature of the **Real time constraints** is completely different from the previous one because the aim is not to measure a task characteristic, but is to define the context and the way to execute the task. Fuzzy numbers can be used to represent real time constraints such as deadline, release time or period of tasks. All these constraints are precisely known but they might be better described by a fuzzy number than an exact value.

For better results, the real time constraints can be expressed by use of linguistic terms such as "2 second periods", "less than 0.1 second response time", etc. For large and complex application, the hypothesis of constraints precision is often arbitrary. For example, a display task with a period of 10 seconds can support wide perturbations if globally the average of the display period value remains at 10 seconds. Here typically the size of the fluctuations is subjective and a good solution could be to capture this size by introducing the notion of fuzzy periods.

V. Fuzzy Inference System

To build a fuzzy system, inputs and output(s) to it must be first selected. After that, partition these inputs and outputs into appropriate conceptual categories which actually represents a fuzzy set on a given input or output domain. The conceptual partitions developed for the input and output dimensions are used to create a fuzzy rule set to determine the behavior of the fuzzy system which is being designed. Change in the input variables causes change in the corresponding membership functions accordingly. Fuzzy inference system (FIS) is abstractly very simple. It consists of an input component, a processing unit, and an output component. The input unit maps the inputs to the appropriate membership functions and truth values. The processing unit executes each appropriate rule and generates a corresponding result and then the results are combined. Finally, the output component converts the combined result back into a specific output value. The 5 steps of fuzzy inference system are as follows:

1. Fuzzification of Inputs
2. Application of fuzzy operators
3. Application of implication methods
4. Aggregation of Outputs
5. De-fuzzification of results

There are two common inference processes [10]. First is called Mamdani's fuzzy inference method proposed in 1975 by Ebrahim Mamdani [12] and the other is Takagi-Sugeno-Kang, or simply Sugeno, method of fuzzy inference introduced in 1985 [13]. These two methods are the same in many respects, such as the procedure of fuzzifying the inputs and fuzzy operators. The main difference between Mamdani and Sugeno is that the Sugeno output membership functions

are either linear or constant but Mamdani's inference expects the output membership functions to be fuzzy sets.

VI. CONCLUSION

Real time task scheduling is a fast expanding application domain. It involves increasingly complex applications where program execution time evaluation is very difficult. Moreover, in these applications, even if the real time constraints have to be globally respected, they are not so critical and precise as in the classical inboard real time applications. For these reasons, it is very interesting to develop scheduler able to take into account their inherent imprecision. The Fuzzy set theory is very pertinent both for capturing human constraints' formulation and imprecise task execution time evaluation. Its use to tackle real time scheduling is then very natural.

REFERENCES

[1] F Terrier and Z Chen, "Fuzzy Calculus Applied to Real Time Scheduling", IEEE Conference on Fuzzy Systems,1994

[2] F Terrier, L Rioux and Z Chen "Real time Scheduling under Uncertainty", IEEE Conference on Fuzzy Systems,1995

[3] G. Sun J., M.K. Gardner and J.W.S. Liu, "Bounding Completion Times of Jobs with Arbitrary Release Times, Variable Execution Times, and Resource Sharing" ,IEEE Transaction on Software Engineering, Vol. 23, No. 10, 1997, pp. 603-615.

[4] Tia T.-S., et al. "Probabilistic Performance Guarantee for Real-Time Tasks with Varying Computation Times" ,IEEE Real Time Technology and Applications Symposium,1995, pp. 164-173.

[5] M.Kaladevi and Dr.S.Sathiyabama "A Comparative Study of Scheduling Algorithms for Real Time Task" International Journal of Advances in Science and Technology, Vol. 1, No. 4, 2010.

[6] Girish Saini "Application of Fuzzy Logic to Real-Time Scheduling" Real time Conference,14 th IEEENPSS,2005.

[7] Pettersson, Lennart. "Real-time Scheduling using Fuzzy Techniques," http://citeseer.ist.psu.edu/5487.html

[8] Lee, J. et al. "A Fuzzy Rule-Based Approach to Real-Time Scheduling," Proceedings of the Third IEEE Conference on Fuzzy Systems: IEEE

world Congress on Computational Intelligence, Vol. 2, Pages 1394 - 1399, 26-29 June 1994.

[9] Mojtaba Sabeghi, Mahmoud Naghibzadeh "A fuzzy Algorithm for Real –Time Scheduling of Soft Periodic Tasks", International Journal of Computer Science and Network Security, VOL. 6 No. 2A, February 2006

[10] Wang Lie-Xin, A course in fuzzy systems and control, Prentice Hall, Paperback, Published August 1996.

[11] Liu C. L., Layland J. W., Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment. Journal of the ACM, 20(1):46-61, 1973.

[12] Mamdani E.H., Assilian S., An experiment in linguistic synthesis with a fuzzy logic controller, International Journal of Man-Machine Studies, Vol. 7, No. 1, pp. 1-13, 1975.

[13] Sugeno, M., Industrial applications of fuzzy control, Elsevier Science Inc., New York, NY, 1985.