A FUZZY APPROACH FOR TASK SCHEDULING IN A REAL TIME DISTRIBUTED SYSTEM

Sagar Gulati *

Neha Arora **

KamalDeep ***

ABSTRACT

All practical real-time scheduling algorithms in distributed systems present a trade-off between performance and computational complexity. In real-time distributed systems, tasks have to be performed correctly and timely. The research till date on task scheduling has primarily focused upon computation time, laxity, priority etc. Further all existing task scheduling algorithms are based upon Boolean Arithmetic. Introduction of Fuzzy theory in scheduling algorithms can really make the study very interesting. Finding an optimal schedule in distributed systems, with real-time constraints is shown to be NP-hard. Deterministic and reliable behavior is an important characteristic for system's robustness analysis. The intrinsic uncertainty in dynamic real-time systems increases the difficulties of scheduling algorithms. To alleviate these deficiencies, we have proposed a fuzzy scheduling approach to arrange real-time periodic and non-periodic tasks with reference to optimal utilization of distributed processors. The present piece of research has been simulated on MATLAB 7.0.4 Mamdani Fuzzy Inference Engine to evaluate the performance of the proposed methodology. Experimental results have shown that the proposed fuzzy scheduler creates feasible schedules for homogeneous and heterogeneous tasks.

Keywords: *Fuzzy Scheduling, CPU Time, Laxity, Priority, Deadline, System Utilization, Feasible Schedule.*

* Assistant Professor, CSE Department, Technology Education & Research Integrated Institutions, Kurukshetra, Haryana.

** Software Developer, Appnetix Pvivate Techno Limited, Noida.

*** Kurukshetra University, Kurukshetra.

INTRODUCTION

Scheduling real time systems involves allocation of resources and CPU-time to tasks in such a way that certain performance requirements are met. In real-time systems scheduling plays a more critical role than non-real-time systems because in these systems having the right answer too late is as bad as not having it at all [1]. Such a system must react to the requests within a fixed amount of time which is called deadline. In general, real-time systems can be categorized into two important groups: hard real-time systems and soft real-time systems. In hard real-time systems, meeting all deadlines is obligatory, while in soft real-time systems missing some deadlines is tolerable. In both cases, when a new task arrives, the scheduler is to schedule it in such a way that guaranties the deadline to be met. Scheduling involves allocation of resources and time to tasks in such a way that certain performance requirements are met. These tasks can be classified as periodic or aperiodic. A periodic task is a kind of task that occurs at regular intervals, and aperiodic task occurs unpredictably. The length of the time interval between the arrivals of two consecutive requests in a periodic task is called period.

Schedulability of periodic, preemptive, soft real-time tasks on uniprocessor systems is well understood; simple and efficient algorithms are available and widely used [2, 3, 4, 5]. Nevertheless, for distributed systems these algorithms are not promising. Meeting the deadlines of realtime tasks in a distributed system requires a scheduling algorithm that determines, for each task in the system, on which processor it must be executed, and in which order with respect to the other tasks, it must start its execution.

In the global scheme, processors repeatedly execute the highest priority tasks available for execution. It has been shown that using this approach with common priority assignment schemes such as rate monotonic (RM) and earliest deadline first (EDF) can give rise to arbitrarily low processor utilization [6]. In this approach a task can migrate from one processor to another during execution. In both cases researchers made some significant contributions by those results in better multiprocessor scheduling algorithms. Although, these algorithms focus on timing constraints but there are other implicit constraints in the environment, such as uncertainty and lack of complete knowledge about the environment, dynamicity in the world, bounded validity time of information and other resource constraints. In real world situations, it would often be more realistic to find viable compromises between these parameters. For many problems, it makes sense to partially satisfy objectives. The satisfaction degree can then be used as a parameter for making a decision.

One especially straightforward method to achieve this is the modeling of these parameters through fuzzy logic. The same approach is also applied on uniprocessor real-time scheduling in [7, 8].

The scope of the paper is confined to scheduling of soft periodic tasks in multiprocessors real-time systems. In the proceeding section of the paper, fuzzy inference systems, proposed model, experimental results, Conclusion and future works are debated respectively.

FUZZY INFERENCE ENGINE

Fuzzy logic [9, 10] is a superset of conventional Boolean logic and extends it to deal with new aspects such as partial truth and uncertainty. Fuzzy inference is the process of formulating the mapping from a given input set to an output using fuzzy logic. The basic elements of fuzzy logic are linguistic variables, fuzzy sets, and fuzzy rules [11]. The linguistic variables' values are words, specifically adjectives like "small," "little," "medium," "high," and so on. A fuzzy set is a collection of couples of elements. It generalizes the concept of a classical set, allowing its elements to have a partial membership. The degree to which the generic element "x" belongs to the fuzzy set A (expressed by the linguistic statement x is A) is characterized by a membership function (MF), fA(x). The membership function of a fuzzy set corresponds to the indicator function of the classical sets. It can be expressed in the form of a curve that defines how each point in the input space is mapped to a membership value or a degree of truth between 0 and 1. The most common shape of a membership function is triangular, although trapezoidal and bell curves are also used. This operation normalizes all inputs to the same range and has a direct effect on system performance and accuracy. U is the whole input range allowed for a given fuzzy linguistic variable. All fuzzy sets related to a given variable make up the term set, the set of labels within the linguistic variable described or, more properly, granulated. Fuzzy rules form the basis of fuzzy reasoning. They describe relationships among imprecise, qualitative, linguistic expressions of the system's input and output. Generally, these rules are natural language representations of human or expert knowledge and provide an easily understood knowledge representation scheme.

A typical conditional fuzzy rule assumes a form such as

IF DEADLINE is "Catastrophic" AND LAXITY is "Low" THEN Priority is "High".

Deadline is "Catastrophic" and laxity is "Low" is the rule's premise; while Priority is "High is the consequent. The premise predicate might not be completely true or false, and its degree of truth ranges from 0 to 1. We compute this value by applying the membership functions of

the fuzzy sets labeled "Low" and "Catastrophic" to the actual value of the input variables Deadline and Laxity. After that, fuzzification is applied to the conclusion; the way in which this happens depends on the inference model.

There are two types of fuzzy inference models:

1. Mamdani [12],
2. TSK or Sugeno [13].

Interpreting an if-then rule involves two distinct parts: first evaluating the antecedent and then applying results to the consequent (known as implication) [14, 15]. In the case of two-valued or binary logic, if-then rules do not present much difficulty. If the premise is true, then the conclusion is true, whereas with fuzzy approach, if the antecedent is true to some degree of membership, then the consequent is also true to that same degree.

Mamdani-type [12] inference expects the output membership functions to be fuzzy sets. After the aggregation process, there is a fuzzy set for each output variable that needs defuzzification. It is possible, and in many cases much more efficient, to use a single spike as the output's membership function rather than a distributed fuzzy set. This is sometimes known as a singleton output membership function, and it can be thought of as a pre-defuzzified fuzzy set. It enhances the efficiency of the defuzzification process because it greatly simplifies the computation required by the more general Mamdani method, which finds the centroid of a two-dimensional function. Rather than integrating across the two-dimensional function to find the centroid, Sugenotype systems use weighted sum of a few data points. In general, Sugeno-type systems can be used to model any inference system in which the output membership functions are either linear or constant.

THE PROPOSED MODEL

In the proposed model, the input stage consists of three input variables i.e. CPU Time, Laxity and Deadline, as shown in Fig. 1. CPU Time is the actual amount of time a task requires on CPU to get executed, Laxity is the time how much a task can wait before getting a chance to get executed where as Deadline represents the final time limit before what, a task has to get terminated. The three input parameters combinely decide the highest priority of the task from the task queue, to get executed at a sequence according to their priorities.

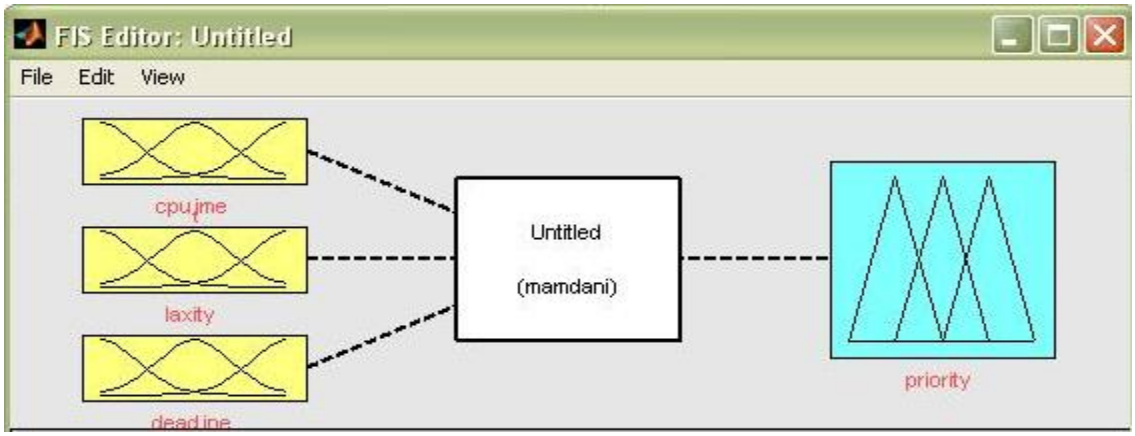

Fig. 1 The Proposed Fuzzy Inference Engine

Membership functions describe the degree to which each input parameter represents its association. Linguistic variables are assigned to each input parameter, to represent this association. CPU Time is categorizes as Low, Medium and High. Similarly Laxity is defines in the same way. However, Deadline is defined as tolerable, sensitive and catastrophic as depicted in Fig. 2, Fig. 3 and Fig. 4.

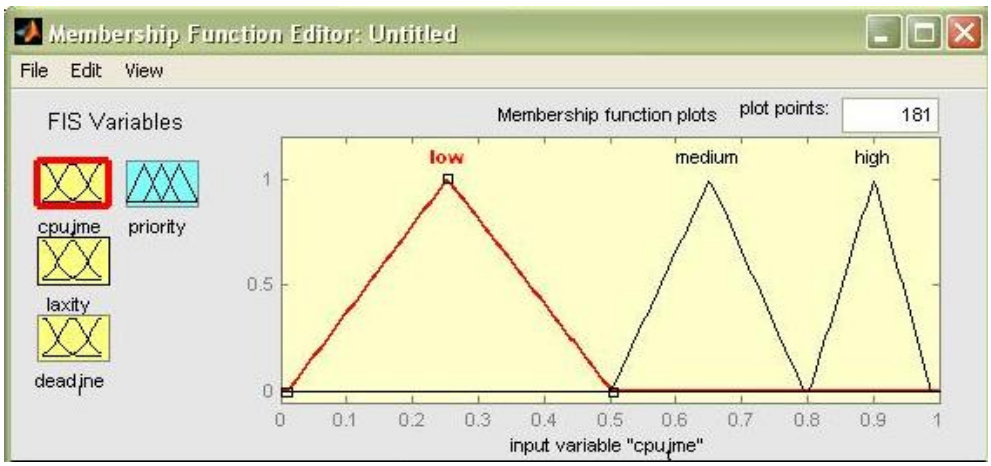

Fig. 2 Fuzzy Set corresponding to CPU Time

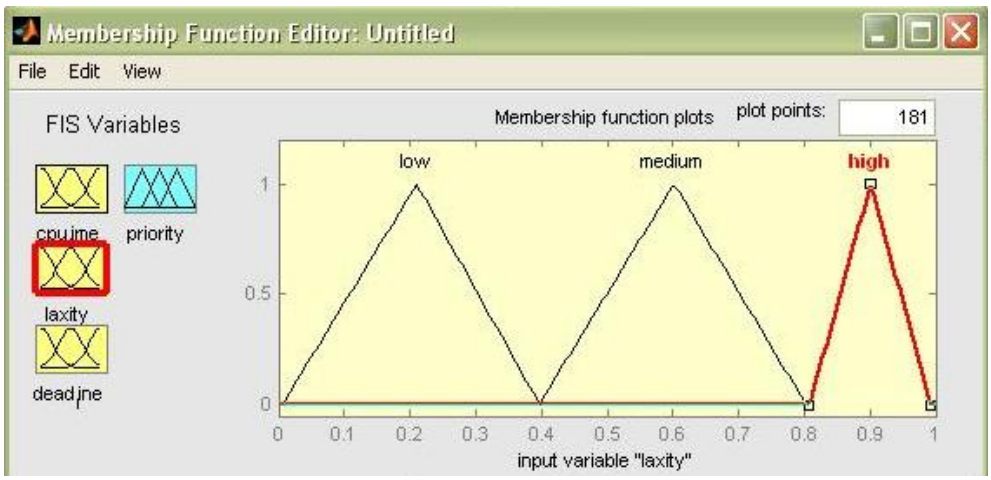

Fig. 3 Fuzzy Set corresponding to Laxity

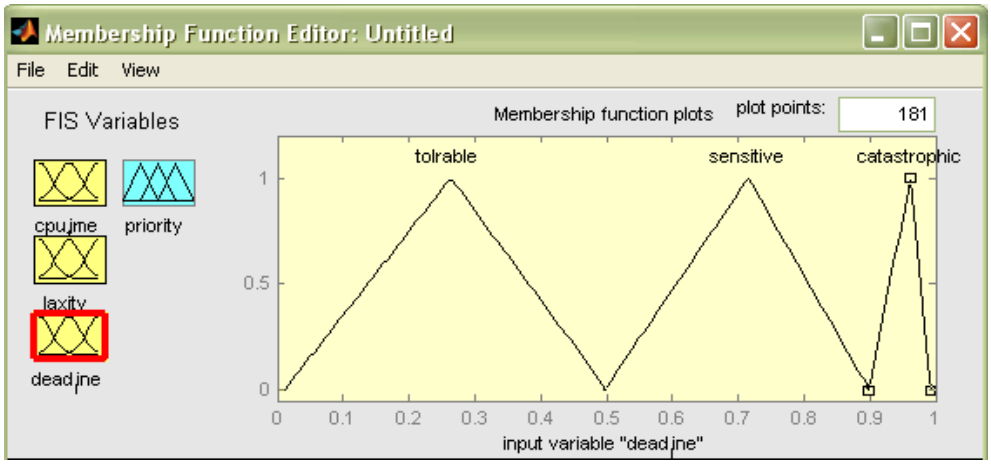

Fig. 4 Fuzzy Set corresponding to Deadline

Fuzzy rules try to combine these parameters as they are connected in real worlds. Some of these rules are mentioned here:

- If (CPU Time is Low), (Laxity is Low) and (Deadline is Tolerable), then (Priority is Medium).
- If (CPU Time is Medium), (Laxity is High) and (Deadline is Tolerable), then (Priority is Low).
- If (CPU Time is High), (Laxity is Medium) and (Deadline is Catastrophic), then (Priority is High).

In fuzzy inference systems, the number of rules has a direct effect on its time complexity. Therefore, having fewer rules may result in a better system performance.

In our proposed algorithm as shown in Fig.5, a newly arrived task will be added to the input queue. This queue consists of the remaining tasks from last cycle that has not yet been assigned.

Loop

For each processor in the distributed system, do the following:

1. *For each ready task, feed its CPU Time, Laxity and Deadline into the inference engine. Consider the output of inference module as priority of the task T.*
2. *Execute the task with highest priority until a scheduling event occurs. (a running task finishes, a new task arrives)*
3. *Update the system states.*

End Loop

Fig. 5 Proposed Algorithm

EXPERIMENTAL RESULTS

Choosing number of rules and membership functions directly affects system accuracy while performance of the system increases with rule size decrease. There are some techniques for adjusting membership functions however; in this paper we did not consider these approaches.

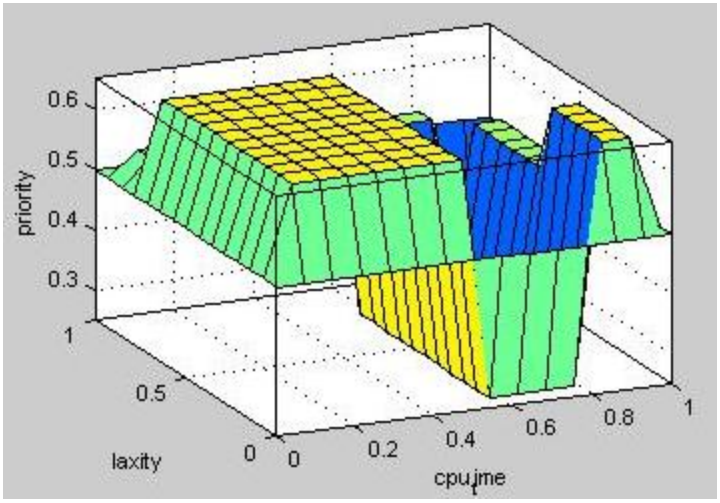

Fig. 6 The decision surface corresponding to Inference rules

The corresponding decision surface to these rules and membership functions is illustrated in Fig. 6.

CONCLUSION AND FUTURE WORK

The scheduler, which is proposed in this paper, has low complexity due to the simplicity of fuzzy inference engine. Consequently, its computation complexity and response time is constant and by increasing, the number of processors will not increase. This model is efficient when system has heterogeneous tasks with different constraints.

In the future, we will conduct a deeper simulation and compare the results of fuzzy approach with the other algorithms.

REFERENCES

1. Ramamritham K., Stankovic J. A., 1994. Scheduling algorithms and operating systems support for real-time systems. Proceedings of the IEEE, Vol. 82(1), pp55-67.
2. Goossens J., Richard P., 2004. Overview of real-time scheduling problems. Euro Workshop on Project Management and Scheduling
3. Liu C. L., Layland J. W., 1973. Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment. Journal of the ACM, Vol. 20, No.1, pp 46-61.
4. Hong J., Tan X., Towsley D., 1989. A Performance Analysis of Minimum Laxity and Earliest Deadline Scheduling in a Real-Time System. IEEE Trans. on Comp., Vol. 38, No. 12.

5. Sha, L. and Goodenough, J. B., 1990. Real-Time Scheduling Theory and Ada. IEEE Computer, Vol. 23, No. 4, pp. 53-62.
6. Andersson B., Jonsson J., 2000. Fixed-priority preemptive multiprocessor scheduling: to partition or not to partition. Seventh International Conference on Real-Time Computing Systems and Applications (RTCSA'00)
7. Sabeghi M., Naghibzadeh M., Taghavi T., 2005. A Fuzzy Algorithm for Scheduling Soft Periodic Tasks in Preemptive Real-Time Systems. International Joint Conferences on Computer, Information, and Systems Sciences, and Engineering (CISSE).
8. Sabeghi M., Naghibzadeh M., Taghavi T., 2006. Scheduling Non-Preemptive Periodic Tasks in Soft Real- Time Systems Using Fuzzy Inference. 9th IEEE International Symposium on Object and component-oriented Real-time distributed Computing (ISORC).
9. L. A. Zadeh, "Fuzzy sets versus probability," Proc. IEEE, vol. 68, pp. 421-421, March 1980.
10. L. A. Zadeh, "Fuzzy logic, neural networks, and soft computing," Commun. ACM, vol. 37, pp. 77-84, March 1994.
11. W. Pedrycz and F. Gomide, An introduction to fuzzy sets: analysis and design: The MIT Press, 1998.
12. E. H. Mamdani, "Application of fuzzy algorithms for the control of a dynamic plant," Proc. IEE, vol. 121, pp. 1585-1588, Dec 1974.
13. T. Takagi and M.Sugeno, "Fuzzy identification of systems and its applications to modeling and control," IEEE Trans. Syst., Man, Cybern., vol. 15, pp. 116-132, 1985.
14. H. Surmann and A. P. Ungering, "Fuzzy rule-based systems on general-purpose processors," IEEE Micro, vol. 15, pp. 40-48, Aug 1995.
15. G. Ascia and V. Catania, "A general purpose processor oriented to fuzzy reasoning," in Proc. 10th IEEE International Conf. Fuzzy Systems, Melbourne, Australia, 2001, pp. 352-355.