Fuzzy Priority CPU Scheduling Algorithm

Bashir Alam[1], M.N. Doja[1], R. Biswas[3], M. Alam[4]

**[1]Department of Computer Engineering, Jamia Millia Islamia
New Delhi-110025, India**

**[2]Department of Computer Engineering, Jamia Millia Islamia
New Delhi-110025, India**

**[3]Department of Computer Science , Jamia Hamdard
New Delhi-110062, India**

**[4]Department of computer Science, Jamia Millia Islamia,
New Delhi-110025, India**

Abstract

There are several CPU scheduling algorithms like FCFS, SRTN,RR , priority etc. Scheduling decision of these algorithms are based on parameters which are assumed to be crisp. However, in many circumstances these parameters are vague. The vagueness of these parameters suggests that scheduler should use fuzzy logic in scheduling the jobs. A fuzzy priority CPU scheduling algorithm has been proposed. This proposed algorithm improves the priority based CPU scheduling algorithm as obvious from simulation results.

Keywords:- FIS, Priority CPU Scheduling, Fuzzy Logic

1. Introduction

When a computer is multiprogrammed , it frequently has multiple processes competing for the CPU at the same time. When more than one process is in the ready state and there is only one CPU available, the operating system must decide which process to run first. The part of operating system that makes the choice is called short term scheduler or CPU scheduler. The algorithm that it uses is called scheduling algorithm. There are several scheduling algorithms. Different scheduling algorithms have different properties and the choice of a particular algorithm may favor one class of processes over another. Many criteria have been suggested for comparing CPU scheduling algorithms and deciding which one is the best algorithm[1]. Some of the criteria include (i)Fairness(ii)CPU utilization(iii)Throughput(iv)Turnaround time(v)Waiting time(vi)Response time

It is desirable to maximize CPU utilization and throughput, to minimize turnaround time, waiting time and response time and to avoid starvation of any process.[1,2] Some of the scheduling algorithms are briefly described here. FCFS: In First come First serve scheduling algorithm the process that request first is scheduled for execution[1,2,3]SJF: In shortest Job first scheduling algorithm the process with the minimum burst time is scheduled for execution.[1,2]SRTN: In shortest Remaining time next scheduling algorithm , the process with shortest remaining time is scheduled for execution.[3]Priority: in Priority Scheduling algorithm the process with highest priority is scheduled for execution. Round-robin: In this the CPU scheduler goes around the ready queue allocating the CPU to each process for a time interval of up to one time quantum. [1,2,3]Multilevel queue scheduling: In this the ready queue is partitioned into several separate queue. The processes are permanently assigned to one queue generally based on some property of the process such as memory size, process priority or process type. Each queue has its own scheduling algorithm. There is scheduling among the queues, which is commonly implemented as fixed-priority preemptive scheduling . Each queue has absolute priority over low priority queues.[1]Multilevel feedback-queue scheduling:-

This allows a process to move between queues.[1] Fair share Scheduling: Fair share scheduler considers the execution history of a related group of processes, along with the individual execution history of each process in making scheduling decision. The user community is divided into a fair- share groups. Each group is allocated a fraction of CPU time. Scheduling is done on the basis of priority of the process, Its recent processor usage and the recent processor usages of the group to which the process belongs. Each process is assigned a base priority. The priority of a process drops as the process uses the processor and as the group to which process belongs uses the processor.[3]Guaranteed scheduling:-In this a ratio of actual CPU time a process had and its entitled CPU time is calculated. The process with this lowest ratio is scheduled[2] Lottery Scheduling:-The basic idea is to give processes lottery tickets for CPU time. Whenever a scheduling decision has to be made, a lottery ticket is chosen at random and the process holding the ticket gets the CPU.[2] HRRN :- In this response ration is calculated for each process. The process with the highest ratio is scheduled for execution. [3] In all the these scheduling algorithms the parameters used are crisp. However, in many circumstances these parameters are vague. To exploit these vagueness we have used fuzzy logic in our proposed scheduling algorithm.

2. Fuzzy Inference Systems and Fuzzy Logic

A fuzzy inference system (FIS) tries to derive answers from a knowledgebase by using a fuzzy inference engine. The inference engine which is considered to be the brain of the expert systems provides the methodologies for reasoning around the information in the knowledgebase and formulating the results. Fuzzy logic is an extension of Boolean logic dealing with the concept of partial truth that denotes the extent to which a proposition is true. Whereas classical logic holds that everything can be expressed in binary terms (0 or 1, black or white, yes or no), fuzzy logic replaces Boolean truth values with the degree of truth. Degree of truth is often employed to capture the imprecise modes of reasoning that play an essential role in the human ability to make decisions in an environment of uncertainty and imprecision. 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. The input space is sometimes referred to as the universe of discourse [4]. Fuzzy Inference Systems are conceptually very simple. An FIS consists of an input stage, a processing stage, and an output stage. The input stage maps the inputs, such as deadline, execution time, and so on, to the appropriate membership functions and truth values. The processing stage invokes each appropriate rule and generates a result for each. It then combines the results of the rules. Finally, the output stage converts the combined result back into a specific output value [4]. The processing stage, which is called the inference engine, is based on a collection of logic rules in the form of IF-THEN statements, where the IF part is called the "antecedent" and the THEN part is called the "consequent". Typical fuzzy inference subsystems have dozens of rules. These rules are stored in a knowledgebase. An example of fuzzy IF-THEN rules is: IF *Remaining Time* is *Extremely short* then *priority* is *very high*, in which *Remaining Time* and *priority* are linguistics variables and *Extremely short* and *very high* are linguistics terms. The five steps toward a fuzzy inference are (i) fuzzifying inputs(ii) applying fuzzy operators(iii) applying implication methods(iv) aggregating outputs(v)defuzzifying results

Bellow is a quick review of these steps. However, a detailed study is not in the scope of this paper. Fuzzifying the inputs is the act of determining the degree to which they belong to each of the appropriate fuzzy sets via membership functions. once the inputs have been fuzzified, the degree to which each part of the antecedent has been satisfied for each rule is known. If the antecedent of a given rule has more than one part, the fuzzy operator is applied to obtain one value that represents the result of the antecedent for that rule. The implication function then modifies that output fuzzy set to the degree specified by the antecedent. Since decisions are based on the testing of all of the rules in the Fuzzy Inference Subsystem (FIS), the results from each rule must be combined in order to make the final decision. Aggregation is the process by which the fuzzy sets that represent the outputs of each rule are processes into a single fuzzy set. The input for the defuzzification process is the aggregated output fuzzy set and the output is then a single crisp value [4]. This can be summarized as follows: mapping input characteristics to input membership functions, input membership function to rules, rules to a set of output characteristics, output characteristics to output membership functions, and the output membership function to a single crisp valued output. There are two common inference methods [4]. The first one is called Mamdani's fuzzy inference method proposed in 1975 by Ebrahim Mamdani [5] and the second one

is Takagi-Sugeno-Kang, or simply Sugeno, method of fuzzy inference introduced in 1985 [6]. 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's output membership functions are either linear or constant but Mamdani's inference expects the output membership functions to be fuzzy sets. Sugeno's method has three advantages. Firstly, it is computationally efficient, which is an essential benefit to real-time systems. Secondly, it works well with optimization and adaptive techniques. These adaptive techniques provide a method for the fuzzy modeling procedure to extract proper knowledge about a data set, in order to compute the membership function parameters that best allow the associated fuzzy inference system to track the given input/output data. The third, advantage of Sugeno type inference is that it is well-suited to mathematical analysis.

3. The Proposed Model

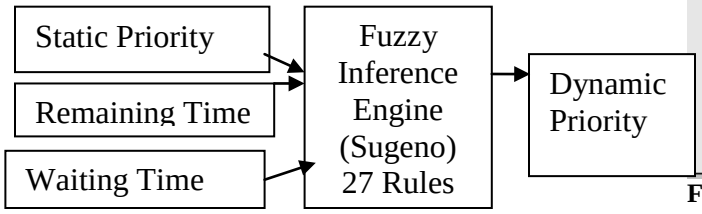

Figure 1: Block diagram of Fuzzy Inference System

The block diagram of the proposed fuzzy inference system is given in figure1. In the proposed model, the input stage consists of three linguistic variables. The first one is the static priority that is assigned to the process before its execution. The second is the expected remaining time of the process. The third input is the waiting time of the process. The output stage consists of one linguistic variable called Dynamic priority.

The input and out variables are mapped into fuzzy sets using appropriate membership functions. Membership functions are given below

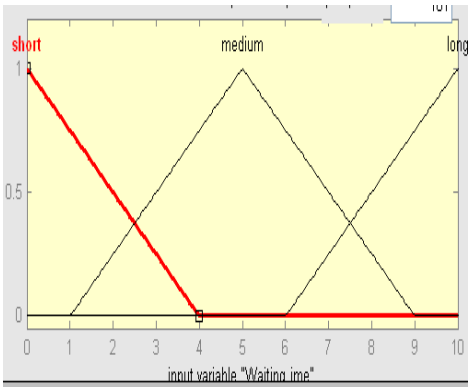

Figure 2: Membership Function for Waiting Time

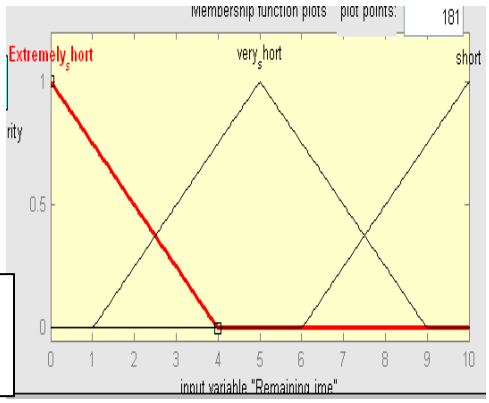

Figure 3 : Membership Function for remaining Time

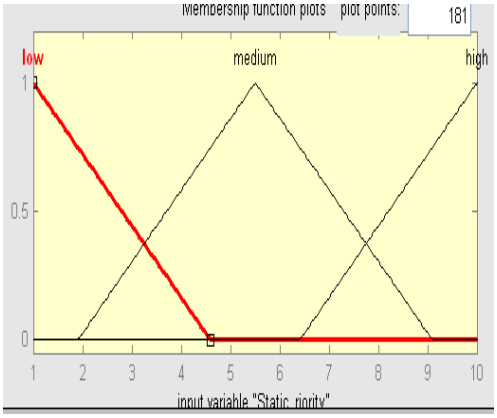

Figure 4: Membership Function for Static Priority

The shape of the membership function for each linguistic term is determined by the expert. Adjusting these membership functions in an optimal mode is very difficult. However, there are some techniques for adjusting membership functions [6, 8].

Twenty seven rules are formulated and a Sugeno type fuzzy Inference system is built. Some of the rules are

listed here : if the static priority is 'low' and remaining time is 'extremely short' and waiting time is 'long' then the dynamic priority is 'very high'. if the static priority is 'low' and remaining time is ' short' and waiting time is 'short' then the dynamic priority is 'very low'. if the static priority is 'medium' and remaining time is ' extremely short' and waiting time is 'long' then the dynamic priority is 'very high'. if the static priority is 'medium' and remaining time is ' short' and waiting time is 'short' then the dynamic priority is 'medium'.

Table 1: Rule base for Fuzzy Inference System

S.No.	Static Priority	Remaining Time	Waiting Time	Dynamic Priority
1.	Low	Extremely short	Short	Very High
2.	Low	Extremely short	Medium	Very High
3.	Low	Extremely short	Long	Very High
4.	Low	Very short	Short	Very low
5.	Low	Very short	Medium	Low
6.	Low	Very short	Long	High
7.	Low	Short	Short	Very low
8.	Low	Short	Medium	Low
9.	Low	Short	Long	High
10.	Medium	Extremely short	Short	Very high
11.	Medium	Extremely short	Medium	Very high
12.	Medium	Extremely short	Long	Very high
13.	Medium	Very short	Short	Medium
14.	Medium	Very short	Medium	Medium
15.	Medium	Very short	Long	Very High
16.	Medium	Short	Short	Medium
17.	Medium	Short	Medium	Medium
18.	Medium	Short	Long	High
19.	High	Extremely short	Short	Very high
20.	High	Extremely short	Medium	Very high
21.	High	Extremely short	Long	Very high
22.	High	Very short	Short	High
23.	High	Very short	Medium	High
24.	High	Very short	Long	Very High
25.	High	Short	Short	High
26.	High	Short	Medium	High
27.	High	Short	Long	Very High

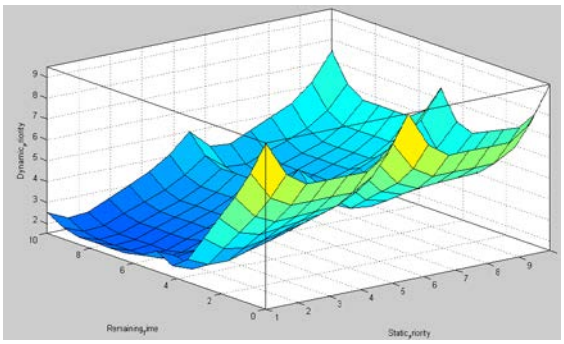

Figure 5: Surface view of FIS for Priority scheduling

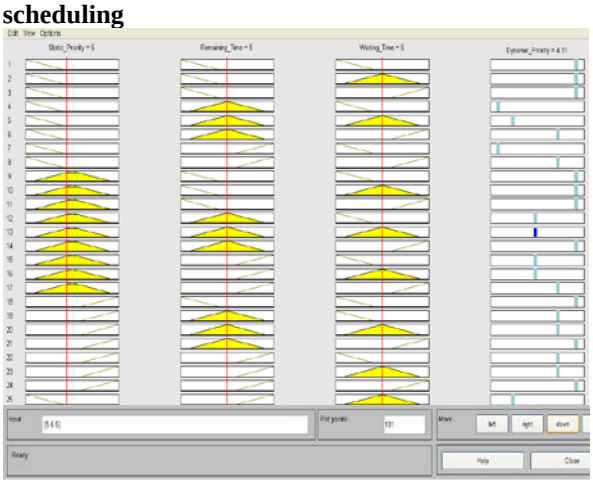

Figure 6: Rule View of FIS for Priority Scheduling

4. Proposed Algorithm

The parameters of process are stored in table called Process Control Block (PCB). Each process has its own PCB. The structure of the Process Control Block is given in Table II.

Table 2: Structure of Process control Block for priority scheduling

Process
Process name="bash" Process identifier=100 State=Ready CPU reserve {CPU Burst Time= CPU Remaining Time= Priority= Waiting time= Arrival Time= Start time= … …. }

The parameters remaining time Rt_i , static priority sp_i , dynamic priority dp_i and waiting time wt_i of process P_i are stored in Process Control Block PCB_i . The proposed algorithm is as follows:

i. For each process P_i in ready queue fetch its parameters Rt_i, sp_i, and wt_i from PCB_i and give them as input to FIS and then set dp_i to the output of FIS
ii. Schedule the process P_i with the highest value of dp_i for execution.
iii. If the scheduled process finishes and no new request arrives go to step ii
iv. If new request arrives go to step first .

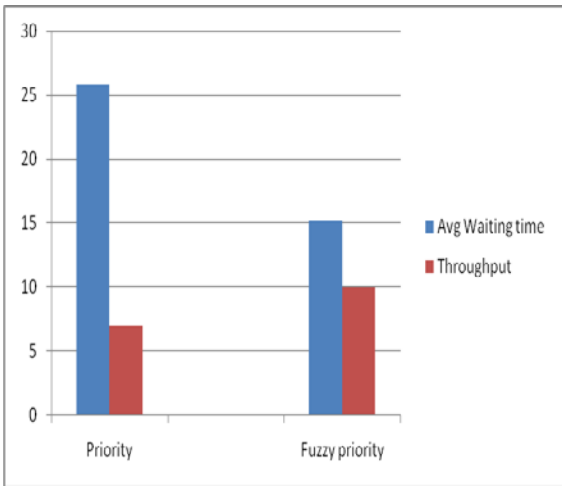

Figure 7 : Performance of Priority and Fuzzy Priority Scheduling Algorithms

4. Performance

For comparing the performance of Priority CPU Scheduling Algorithm and Fuzzy Priority CPU Scheduling algorithm, we did simulation on 1000 processes in groups of ten each. We assumed random burst time of processes and random arrivals. Max burst time of a process should not exceed 10 ms. Throughput and average waiting time of the processes in a group was computed and then average was taken over all groups to give average throughput and average waiting time. The performance is shown in the column chart given in figure 7 above.

6. Conclusion

We have proposed fuzzy priority CPU scheduling algorithm based on FIS. This algorithm is having benefits of shortest remaining time next (SRTN) as well as Priority scheduling algorithm and is capable of removing the starvation problem of priority scheduling algorithm. This proposed algorithm also improves the system performance by not pre empting the process that requires extremely less fraction of CPU time. Fuzzy priority scheduling algorithm gives better throughput and lesser waiting time than traditional Priority Scheduling as obvious from the figure 7.

References

[1]Silberschatz, A., Peterson, J. **L.,** and Galvin, B.,***Operating System Concepts,*** Addison Wesley, 7th Edition, **2006.**

[2]Andrew S. Tanenbaum , and Albert S. Woodfhull , Operating Systems Design and Implementation,Second Edition,2005

[3] William Stallings, Operating Systems Internal and Design Principles, 5th Edition ,2006

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

[5]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, 1975.

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

[7]Jang, J.-S. R., ANFIS: Adaptive-Network-based Fuzzy Inference Systems, IEEE Transactions on Systems, Man, and Cybernetics, Vol. 23(3), 685, May 1993.

[8] Simon D, Training fuzzy systems with the extended Kalman filter, Fuzzy Sets and Systems, Vol. 132, No. 2, 1, December 2002.