2012 International Conference on Computer Technology and Science (ICCTS 2012)
IPCSIT vol. 47 (2012) © (2012) IACSIT Press, Singapore
DOI: 10.7763/IPCSIT.2012.V47.3

Efficient CPU Scheduling Algorithm Using Fuzzy Logic

Puneet Kumar Varshney [1], Nadeem Akhtar [1 +] and M Faridul Haque Siddiqui [1]

[1] Zakir Hussain College of Engineering and Technology
Aligarh Muslim University, Aligarh, India

Abstract. For using an operating system's resources more efficiently, multiprogramming plays an important part. And for multiprogramming to take place, scheduling plays a pivotal role. This policy of deciding which processes to run at a given time should attempt to maximize throughput, minimize latency, prevent process starvation etc. Various techniques are present to perform the said task. In this paper a new improved scheduling algorithm technique based on Fuzzy Logic has been proposed. The proposed algorithm has been implemented and compared with the existing FCFS and Round Robin. Here Fuzzy Logic has been used to decide a value for time quantum that is neither too large nor too small such that every process has reasonable response time and the throughput of the system is not decreased due to unnecessary context switches.

Keywords: Multiprogramming, scheduling, process, Fuzzy Logic, FCFS, Round Robin

1. Introduction

In a multiprogramming environment, the processes that are loaded into the memory compete for processor time. While a process is being executed by a processor, others wait for I/O to be performed or for some other events to take place. Scheduling, a key concept in operating system design, determines which process will progress and which will wait. Some of the objectives that scheduling function should satisfy in order to be effective include fairness, efficient use of processor time, response time, turnaround and throughput [1]. Operating systems may feature up to three distinct types of scheduling: long-term scheduling, medium-term scheduling and short-term scheduling.

Long-term scheduling determines which jobs or programs are admitted to the system for processing. The admitted program is transformed to a process and then added to the queue for the short-term scheduling. The more processes created, the smaller is the time that each process can be executed on a processor. In other words, this type of scheduling may limit the degree of multiprogramming to offer reasonable services to the existing set of processes. The scheduler may decide to add new jobs every time a job terminates or if the fraction of time that the processor is unoccupied exceeds a certain threshold [1, 2]. This type of scheduler executes relative infrequently and decides whether or not to take on a new process and which one to take.

Several criteria have been suggested for comparing CPU scheduling algorithms and deciding which one is the best algorithm. Some of the criteria include (i) Fairness (ii) CPU utilization (iii) Throughput (iv) Turnaround time (v) Waiting time (vi) Response time [1, 3]. 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.

Our motive was to develop an algorithm using fuzzy logic which decreases average response time without compromising with the average waiting time and average turnaround time.

Some of the scheduling algorithms are briefly described below:

[+] Corresponding author. Tel.: +91-9450658150;
E-mail address: nadeemalakhtar@gmail.com.

2. Short-term process scheduling algorithms

A scheduling algorithm gives rights to processes to use the processor time. The simplest algorithm is First Come First Served (FCFS). When the currently running process ceases to execute, the oldest process in the ready queue is selected for running. This technique is non-preemptive [1, 2, 3].Another straightforward and starvation-free process scheduling algorithm is Round Robin (RR). It assigns time slices to each process in equal portions and in order, handling all processes without priority. Shortest Process Next (SPN) is another non-preemptive algorithm that selects the waiting process with the shortest execution time to execute next. Shortest Remaining Time (SRT) is the preemptive version of Shortest Process Next (SPN). It permits a process that enters the ready list to preempt the running process if the time for the new process is less than the remaining time for the running process. Multilevel Feedback Queues (MFQ) scheduling is intended to give preferences to short jobs and I/O bound processes. Instead of focusing on the time remaining to execute, it focuses on the time spent in execution so far.

3. Introduction to fuzzy logic

In the real world, information is often hazy or imprecise. When we state that it is warm today, the context is necessary to approximate the temperature. A warm day in February may be 10 degree Celsius, but a warm day in July may be 32 degrees. Human way of thinking interprets information in order to reach at conclusions. Although machines cannot yet handle inexact information in the same ways that humans do, computer programs with fuzzy logic are becoming quite useful.

Fuzzy Logic is a generalization of standard logic, in which a concept can possess a degree of truth anywhere between 0.0 and 1.0. It allows intermediate values to be defined between conventional evaluations like true/false, yes/no, high/low, etc.

Fuzzy logic is a convenient way to map an input space to an output space. Between the input and the output we'll put a black box that does the work. Some examples where fuzzy logic is used such as automobile and other vehicle subsystems, air conditioners, cameras, rice cookers, dishwashers, elevators, washing machines etc.

3.1. Fuzzy Inference Systems

A fuzzy inference system (FIS) tries to derive answers from a knowledge base 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. 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.

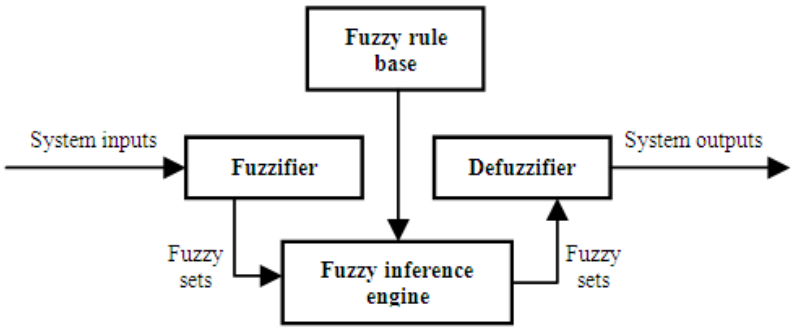

Fig.1- Structure of a fuzzy system

The input space is sometimes referred to as the universe of discourse. 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 five steps toward a fuzzy inference are as follows:

(i) Fuzzifying inputs

(ii) Applying fuzzy operators

(iii) Applying implication methods

(iv) Aggregating outputs

(v) Defuzzifying results [5]

There are two common inference methods. The first one is called Mamdani's fuzzy inference method proposed in 1975 by Ebrahim Mamdani [6] and the second one is Takagi-Sugeno-Kang, or simply Sugeno, method of fuzzy inference introduced in 1985 [7]. 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.

4. FIS for finding Time Quantum

The Fuzzy Inference System for finding the time quantum has got 2 inputs and one output. First input is N that specifies the number of user/ processes in the system and second input is the average burst time of the processes in the ready queue. Time quantum is the output of the FIS.

The following three membership functions for average burst time, number of processes and for finding time quantum are as follows.

4.1. Membership Function for N (Number of Processes)

Type- Triangular, Range: 1-10, low-[0, 2, 4], medium-[3, 5.5, 8], High-[7, 8.5, 10]

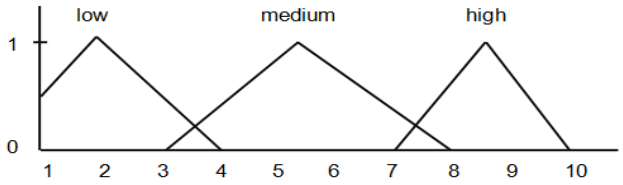

Fig.2- Membership function for processes

To Fuzzify number of processes we have used the method as described in Fig.2. We have divided it in three triangular ranges low, medium and high. Depending upon number of processes result will lies in these ranges with value from 0 to 1. For example membership function for N will be [0.0, 1.0, 0.0] if N=4.

4.2. Membership Function for the Average Burst Time

Type- Triangular, Range:0-10, low-[-4,0,4], medium-[3,5,7] High:-[6,10,16]

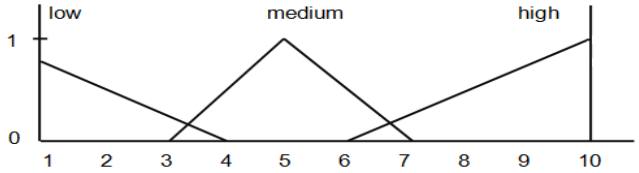

Fig.3- Membership function for ABT

To Fuzzifying Average burst time we have used the method as described in Fig.3. We have also divided it in three triangular ranges low, medium and high. We have span the burst times in the scale of 1 to 10 starting from minimum to maximum burst time further depending upon ABT result will lies in these ranges with value from 0 to 1.

4.3. Membership Function for Time Quantum

Type- Triangular, Range:1-5, low-[0,1,2], medium-[1,2.5,4] High :-[3,5,7]

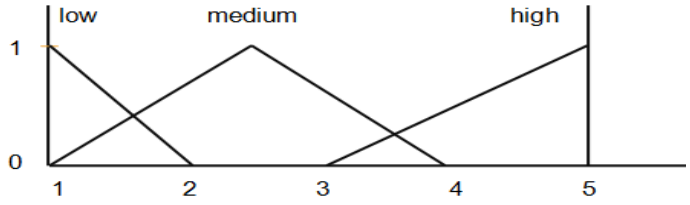

Fig.4- Membership function for Time Quantum

To Fuzzifying time quantum we have also divided it into three parts as shown in Fig.4. We have span the given time quantum on the scale of 1 to 5 and calculated the estimated time quantum on the basis of Rule base table as discussed in next section.

4.4. Rule Base for FIS

1. If (N is low) and (ABT is low) then (TimeQuantum is low) (1)
2. If (N is low) and (ABT is medium) then (TimeQuantum is medium) (1)
3. If (N is low) and (ABT is high) then (TimeQuantum is high) (1)
4. If (N is medium) and (ABT is low) then (TimeQuantum is medium) (1)
5. If (N is medium) and (ABT is medium) then (TimeQuantum is medium) (1)
6. If (N is medium) and (ABT is high) then (TimeQuantum is medium) (1)
7. If (N is high) and (ABT is low) then (TimeQuantum is low) (1)
8. If (N is high) and (ABT is medium) then (TimeQuantum is medium) (1)
9. If (N is high) and (ABT is high) then (TimeQuantum is medium) (1)

Fig.5- Rule base for FIS

In Fig.5 we have shown the rule base table for fuzzy inference system using this we have calculated estimated time quantum.

4.5. Flowchart of proposed Scheme

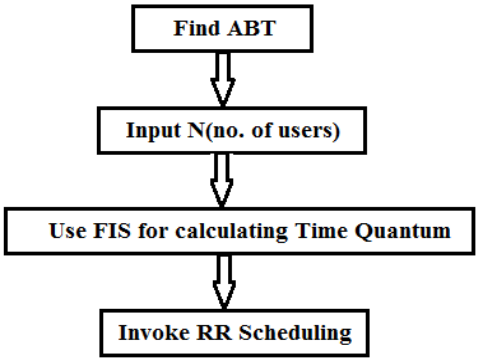

Fig.6- Flow chart of the proposed scheme

4.6. Results and comparisons

The simulator has been designed to use and carry out experiments. Simulation of algorithms generated useful data that has been used to draw results. One of these results is discussed here. Table.1 shows the experiment data with its results when applied on proposed scheme of scheduling.

Table.2 shows the comparison of proposed scheme with existing FCFS and Round robin scheduling policies.

Table.1- Results of proposed scheme

Name	Arrival	Burst	Response	Waiting	CompleteAt
Job1	0	80	0	116	196
Job2	1	55	25	135	191
Job3	2	20	50	48	70
Job4	3	41	70	117	161

Membership N [0.0, 1.0, 0.0]

Membership ABT [0.0, 0.5625, 0.03125]

Table.2- Comparison with existing algorithms

	FCFS	Round Robin	Proposed Scheme
Average Response Time	92.5	67.5	36.25
Average Waiting Time	91.0	116.5	104.0
Average Turnaround Time	141.5	167.0	154.5

Membership tq [0.0, 0.03125, 0.0]

Time quantum [given=50, fractional=2.5, estimated=25.0] Estimated TQ=25

Here given time quantum was 50 and as shown above after applying the proposed scheme the estimated new time quantum is 25.

Let us see another example-

Table.3- Ex.2 Results of proposed scheme

Name	Arrival	Burst	Response	Waiting	CompleteAt
Job1	0	20	0	144	164
Job2	1	35	9	208	244
Job3	2	60	18	348	410
Job4	3	51	27	320	374
Job5	4	67	36	370	441
Job6	5	34	45	239	278
Job7	6	48	54	332	386
Job8	7	290	63	553	850
Job9	18	245	72	549	812

Table.4- Ex.2 Comparison with existing algorithms

	FCFS	Round Robin	Proposed Scheme
Average Response Time	197.33	111.11	36.0
Average Waiting Time	192.22	314.78	340.33
Average Turnaround Time	292.77	414.33	439.89

Membership N [0.0, 0.0, 1.0]

Membership ABT [0.75, 0.13, 0.0]

Membership tq [0.75, 0.13, 0.0]

Time quantum [given=30, fractional=1.65, estimated=9.87]

Estimated TQ=9

Here given time quantum was 30 and as shown above after applying the proposed scheme the estimated new time quantum is 9.

4.7. Simulator designed

We have designed a simulator using JAVA language with a view to develop a software tool which can be used for study and simulation of CPU scheduling algorithms.

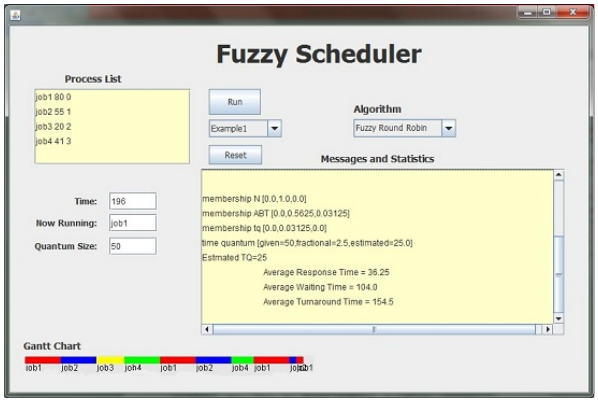

Fig.7- Simulator designed for comparisons

5. Acknowledgements

We express our gratitude towards almighty who has enlightened our minds to help us work in this direction. He has helped us to explore this vast topic in an organized manner and blessed us to think on all the ideas on how to work towards a research - oriented venture.

We would also like to thank our family members and friends who were always there in the need of the hour and provided with all the help and facilities

6. References

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

[2] Stallings, W., “Operating Systems: internals and design principles”, 3rd edt., Prentice Hall, Inc., 1998, pp. 394

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

[4] Jeffay, K.; Donelson Smith, F.; Moorthy, A.; Anderson, J.;" Proportional share scheduling of operating system services for real-time applications", Real-Time Systems Symposium, 1998. Proceedings. The 19th IEEE, 2-4 Dec. 1998 Page(s):480 – 491

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

[6] Mamdani E.H., Assilian S., An experiment in linguistic synthesis with a fuzzy logic controller, International Journal of Man-Machine Studies, Vol.7

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

[8] No. 1, 1975.L.A. Zadeh, "Making computers think like people," IEEE. Spectrum, 8/1984, pp. 26-32