Journal of Computer Science 9 (8): 1079-1085, 2013
ISSN: 1549-3636

doi:10.3844/jcssp.2013.1079.1085 Published Online 9 (8) 2013 (http://www.thescipub.com/jcs.toc)

FUZZY ROUND ROBIN CPU SCHEDULING ALGORITHM

Bashir Alam

Department of Computer Engineering,
Faculty of Engineering and Technology, Jamia Millia Islamia New Delhi-110025, India

Received 2013-02-11, Revised 2013-06-06; Accepted 2013-07-09

ABSTRACT

One problem in Round Robin CPU Scheduling is that if the time required for the running process is slightly more than time quantum even by a fraction value, then process gets preempted and context switch occurs. This causes more waiting time for that process and more overheads due to unnecessary context switch. Another problem with RR scheduling is the value of time Quantum. If it is too large, RR algorithm degenerate to FCFS and if it is too short frequent context switches occurs which results into more overheads which in turn degrade the performance. In this work a Fuzzy Round Robin scheduling algorithm has been proposed that tries to remove these two problems using fuzzy technique. Simulation has been done to compare the performance of this algorithm with its non fuzzy counterpart.

Keywords: FIS, Fuzzy Logic, Scheduling, Round Robin

1. INTRODUCTION

When a computer is multiprogrammed, it has several processes competing for the CPU at the same time. When more than one process is in the queue of the processes in ready state and are waiting for CPU allocation, the operating system must decide which process to run first and allocate the CPU to that process. The part of operating system that makes this choice is called short term scheduler or CPU scheduler. The algorithm used to make the choice is called scheduling algorithm. Several scheduling algorithms exists. Each scheduling algorithms have their own features and the choice of a particular algorithm may favour one class of processes over another. For comparing CPU scheduling algorithms and deciding which one is the best algorithm, several criteria have been suggested (Silberschatz *et al*., 2008). Some of the criteria include (i) Fairness(i) CPU utilization (iii) Throughput (iv) Turnaround time (v) Waiting time (vi) Response time. Sceduling algorithm should try to (i)maximize CPU utilization and throughput, (ii)to minimize turnaround time, waiting time and response time and (iii)to avoid starvation of any process (Silberschatz *et al*., 2008; Andrew and Albert, 2006). Some of the scheduling algorithms are briefly described below.

FCFS: In First come First serve scheduling algorithm the process that request first is scheduled for execution (Silberschatz *et al*., 2008; Andrew and Albert, 2006; Stallings, 2008). SJF: In shortest Job first scheduling algorithm the process with the minimum burst time is scheduled for execution (Silberschatz *et al*., 2008; Andrew and Albert, 2006). SRTN: In shortest remaining time next algorithm, a process is scheduled for execution whose remaining execution time is shortest (Stallings, 2008). Priority: In Priority Scheduling algorithm the process with highest priority is scheduled for execution (Silberschatz *et al*., 2008; Andrew and Albert, 2006; Stallings, 2008). Multilevel queue scheduling: In this the ready queue is partitioned into several different queues. Assignment of processes to one queue permanently are generally based on some property of the process such as size of memory, priority of process or type of process. Queues are free to have their own scheduling algorithm. Scheduling among the queues, is commonly

implemented as fixed-priority preemptive scheduling. Every queue has got absolute priority over low priority queues (Silberschatz *et al*., 2008). Multilevel feedback-queue scheduling: This is like multilevel queue scheduling but allows a process to move between queues (Silberschatz *et al*., 2008). Fair share Scheduling: Fair share scheduler considers the execution history of processes of a related group, along with the execution history of each individual process in making decision of scheduling. Fair- share groups are constructed within the user community. A fraction of CPU time is allocated to each group. Scheduling is done on the basis of process priority, its usage of recent processor and the usages of recent processor of the group to which the process belongs. A base priority is assigned to each process. The process priority drops as the process uses the processor and as the group to which process belongs uses the processor (Stallings, 2008). Guaranteed scheduling:- This algorithm calculates a ratio of actual CPU time a process had and its entitled CPU time is calculated and then schedule the process having this lowest ratio (Andrew and Albert, 2006). Lottery Scheduling: The basic idea is to give processes lottery tickets for allocation of CPU time. A lottery ticket is chosen at random at the time of scheduling decision and the process holding the ticket gets the CPU (Andrew and Albert, 2006). HRRN:-In this response ratio is calculated for each process. The process with the highest response ratio is scheduled for execution (Stallings, 2008). Fuzzy HRRN: In this algorithm FIS has been used to improve the performance of basic Highest Response Ration Next (HRRN) CPU Scheduling algorithm (Alam *et al*., 2011). Fuzzy Fair Share Scheduling: In this CPU Scheduling algorithm fuzzy logic has been used to improve the performance of basic fair share scheduling algorithm (Alam *et al*., 2009). 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. If the process does not complete its execution within the time quantum, the process goesto the end of ready queue and process switch occurs where state of the running process is put onto stack and the state of the next process is taken from the stack and its execution resumes. If the time requirement of the running process is slightly more than time quantum, even then process is preempted and context switch happenss. This causes more waiting time for that process and more overheads due to unnecessary context switch. Another problem with RR scheduling is the value of time Quantum. If it is too large, RR algorithm degenerate to FCFS and if it is too short frequent context switches occurs which results into more overheads which in turn degrade the performance. In this work an algorithm using fuzzy logic has been proposed that tries to remove these two problems.

1.1. 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 provides the methodologies for reasoning around the information in the knowledgebase and results formulations. Fuzzy logic deals with the concept of partial truth that denotes the extent to which a proposition is true. In classical logic everything can be expressed in binary terms (0 or 1, black or white, yes or no). Fuzzy logic replaces boolean truth values with the truth's degree. Truth's degree is often employed to capture the imprecise modes of reasoning that play an essential role in the human ability to make decisions in uncertain and imprecise environment. The membership function of a fuzzy set is analogous to the indicator function of the classical sets. Curves are used to express the membership functions. curve shape defines how each point in the input space is mapped to a membership value or a truth's degree between 0 and 1. Triangular is the most common shape of a membership function. Other curve shape like trapezoidal and bell are also used. Uuniverse of discourse is the input space (Wang, 1997). Fuzzy Inference Systems (FIS) consists of three stages namely input, processing and output. In input stage the inputs, such as deadline, execution time and so on are mapped to the appropriate membership functions and truth values. In processing stage each appropriate rule is invoked each of them generates a result. The results of the rules are then combined. Finally, in the output stage the combined result is converted back into a specific output value (Wang, 1997). The processing stage, called the inference engine, works with the help of 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". Fuzzy inference systems have several rules. Knowledgebase stores these rules. An example of fuzzy IF-THEN rules is: IF Remaining Time is short then priority is high, in which Remaining Time and priority are linguistics variables and short and

high are linguistics terms. The five steps of a typical a fuzzy Inference System are as follows:

- Fuzzifying inputs
- Applying fuzzy operators
- Applying implication methods
- Aggregating outputs
- Defuzzifying results

These steps are review quickly below. Fuzzifying the inputs is the act of determining the degree to which they belong to each of the appropriate fuzzy sets with help of membership functions. Once the inputs have been After fuzzification, the degree of satisfaction to which each part of the antecedent for each rule is found out. For rules with antecedent having several parts, fuzzy operator is applied to obtain one value that represents the result of the antecedent for that rule. The output fuzzy set is modified by the implication function to the degree specified by the antecedent. The results from each rule in FIS are combined using aggregation process into a single fuzzy set representing the output of each rule. The defuzzification process takes the aggregated output fuzzy set as input and gives a single crisp value as output (Wang, 1997). There are two common inference methods (Wang, 1997). The first one is called Mamdani fuzzy inference method Mamdani and Assilian (1999) and the second one is Takagi-Sugeno-Kang, or simply Sugeno, fuzzy inference method (Sugeno, 1985). These two methods are the same in many respects. The main difference between these two methods (Hammam and Georganas, 2008) 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. The input and output variables are mapped into fuzzy sets using appropriate membership functions. Expert determine shape of the membership function for each linguistic term. It is very difficult to adjust these membership functions in an optimal mode. However, membership functions may be adjusted using some available techniques (Jang, 1993; Simon, 2002). These techniques cannot be covered in this study. They can be further studied in a separate paper.

1.2. Fuzzy Round Robin CPU Scheduling Algorithm

In Round Robin Scheduling the time quantum is fixed and then processes are scheduled such that no process get CPU time more than one time quantum in one turn. Too Large time quantum increases the response time of the processes too much which may not be tolerated in interactive environment. Too small time quantum causes unnecessarily frequent context switches leading to more overheads resulting in less throughput. In this work a method using Fuzzy Logic has been proposed that decides a value that is neither too large nor too small such that every process has got reasonable response time and the throughput of the system is not decreased due to unnecessarily context switches.

In Round Robin scheduling CPU scheduler goes around the ready queue allocating the CPU to each process for a time interval of up to one time quantum. If the process do not complete its execution within the time quantum, the process go to the end of ready queue and process switch occurs where state of the running process is put onto stack and the state of the next process is taken from the stack and its execution is restarted. If the time required for the running process is slightly more than time quantum even by a fraction, even then Scheduler prempt the process resulting into context switch. This causes more waiting time for that process and more overheads due to unnecessary context switches. In this work an algorithm named fuzzy round robin scheduling algorithm has been proposed that tries to remove these two problems. This algorithm uses two FIS one for finding the time quantum value and another for deciding the preemption.

1.3. FIS for Finding Time Quantum

The Fuzzy Inference System for finding the time quantum has got two 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. Block diagram, rules base, surface view and rule view of the FIS designed are shown below in **Fig. 1**, **Table 1**, **Fig. 2 and 3** respectively. This FIS solves the first problem.

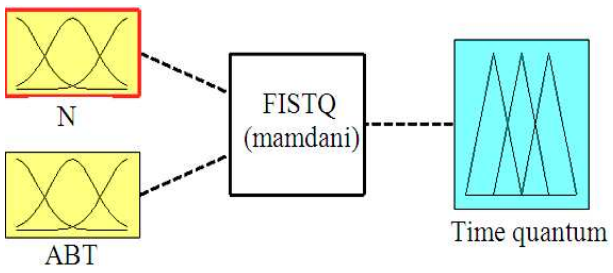

Fig. 1. FIS for Finding Time Quantum (FISTQ)

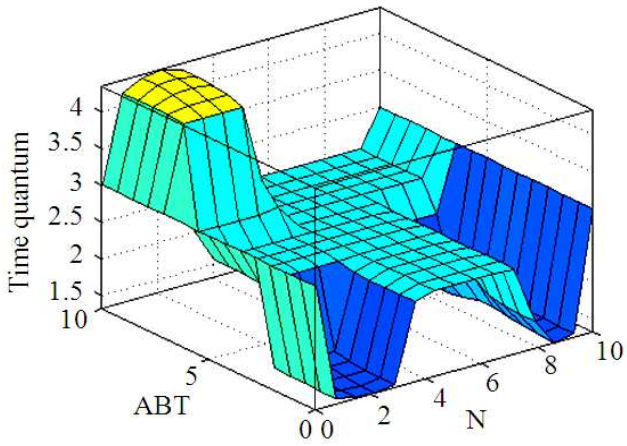

Fig. 2. Surface View of FIS for Finding Time Quantum

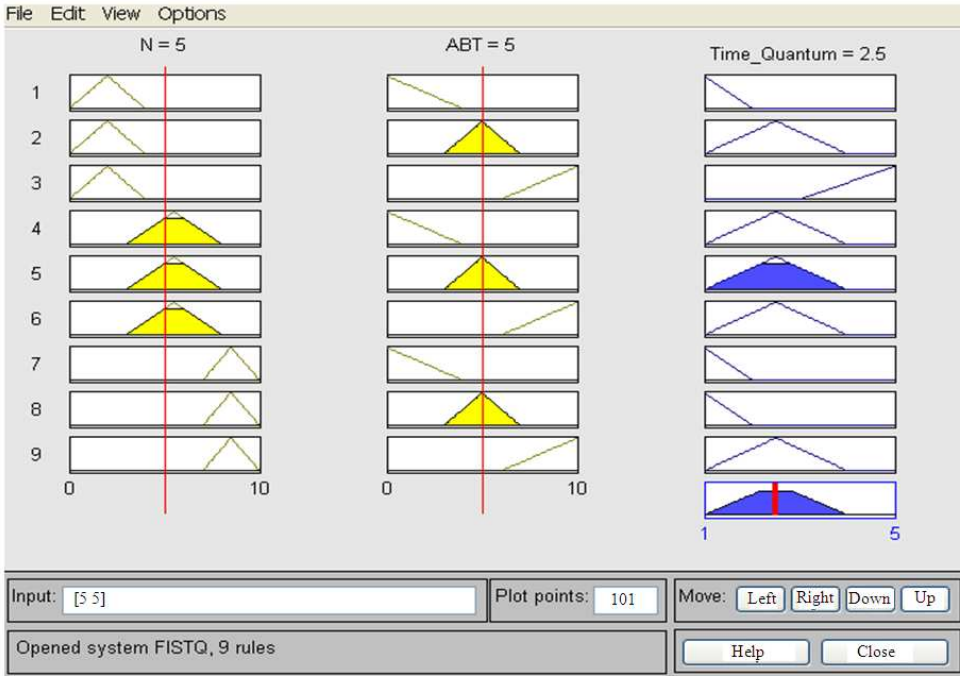

Fig. 3. Rule View of FIS for Finding Time Quantum

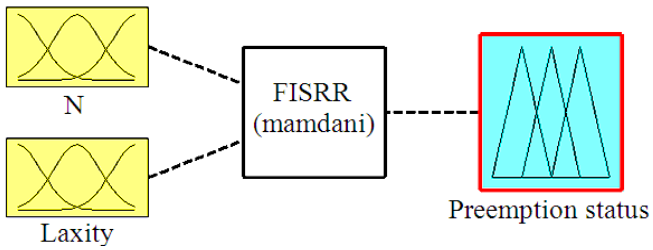

Fig. 4. FIS for Preemption (FISRR)

Inputs and outputs membership functions are given below:

Membership Function for N
Type: - Triangular,
Range: - 1 - 10,
Low: - [0, 2, 4],
Medium: - [3, 5.5, 8] and
High: - [7, 8. 5, 10]

Membership Function for Average Burst Time
Type: - Triangular,
Range: - 0 – 10,
Low: - [-4, 0, 4],
Medium: - [3, 5, 7]
High:- [6, 10, 16]

Membership Function for Time Quantum
Type: - Triangular,
Range: - 1 - 5,
Low: - [0, 1, 2],
Medium: - [1, 2.5, 4] and
High: - [3, 5, 7]

1.4. FIS for Deciding Preemption

The main problem in Round Robin Scheduling is that the process is preempted on the expiry of time quantum even if the process needs a few fraction of time quantum to complete its execution. The unnecessary overheads and waiting time may be reduced if the process is given time slightly more than the time quantum so that it may complete its execution in that turn itself. The time required for any process is not very clear and fuzzy logic is suitable for vague thing. So, Fuzzy logic may be used to delay the preemption of the process. A fuzzy Inference system with two inputs and one output has been designed. Laxity and N, the number of Ready Processes are inputs and Preemption Status is the output. Block diagram, rules base, surface view and rule view of the FIS designed for deciding premption are shown above in **Fig. 4, Table 2, Fig. 5 and 6** respectively. This FIS solves the second problem.

The membership functions for these fuzzy variables are given below:

Membership Function for N
Type- Triangular,
Range: 1-10,
Low- [0, 2, 4],
Medium- [3, 5.5, 8] and
High- [7, 8.5, 10]

Membership Function for Laxity
Type- Triangular,
Range: 0-TQ,
Low- [0, TQ/4, 3/8TQ],
Medium- [3/8TQ, TQ/2, 5/8TQ] and
High- [5/8TQ, 3/4TQ, TQ]

Membership Function for Preemption status
Type- Triangular,
Range: 0-1,
No Preempt- [0, 0.3, 0.6] and
Preempt- [0.5, 0.75, 1.0]

1.5. Proposed Fuzzy Round Robin CPU Scheduling Algorithm

Proposed Fuzzy Round Robin Scheduling Algorithm is described below:

Step1: Select the first process p in the ready queue for execution and remove it from the ready list.
Step2: Find ABT, the average burst time of the processes
Step3: Give N, the number of users and ABT to the FIS for Time Quantum.
Step4: Take output of FIS as the time quantum and load it into the interval timer
Step5: Start execution of process P on the CPU
Step6: If P initiates an I/O operation or completes its execution, go to step 1 to schedule another process for execution.
Step7:

- When a timer interrupt occurs, do not preempt the process but start another counter to measure laxity/ relaxation given to the process
- If process completes go to step 1 to schedule another process for execution
- Give Laxity and N, the number of ready processes to the input of FIS for deciding Preemption and take its output as preemption status
- If preemption status is greater than or equal to 0.5, preempt the process and put it at the end of the ready queue and go to step 1 to schedule another process for execution

Else
If process completes, go to step 1 to schedule another process for execution.

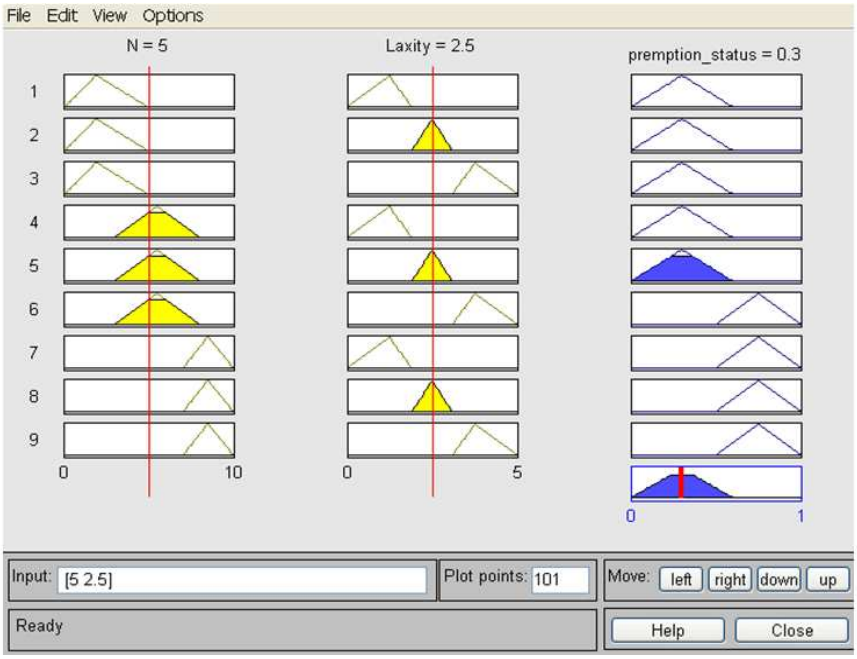

Fig. 5. Rule view of FIS for Preemption

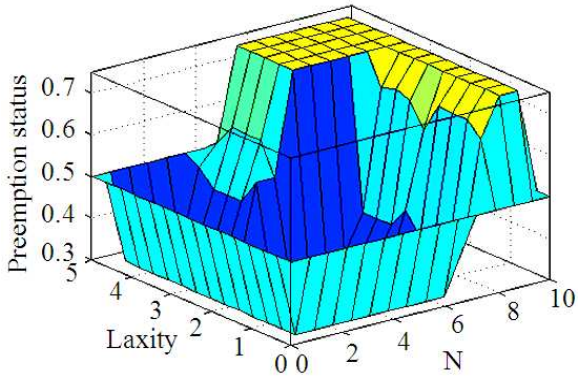

Fig. 6. Surface view of FIS for Preemption

Table 1. Rule Base of FIS for Finding Time Quantum

S.No.	N	ABT	Time Quantum
1	Low	Low	Low
2.	Low	medium	Medium
3.	Low	High	High
4.	Medium	Low	Medium
5.	Medium	medium	Medium
6.	Medium	High	Medium
7.	High	Low	Low
8.	High	medium	Low
9.	High	High	Medium

Table 2. Rule base for FIS for Deciding Preemption

S.No.	Laxity	No of processes	Preemption Status
1	Low	Low	No Preemption
2.	Low	Medium	No Preemption
3.	Low	High	No Preemption
4.	Medium	Low	No Preemption
5.	Medium	Medium	No Preemption
6.	Medium	High	Preemption
7.	High	Low	Preemption
8.	High	Medium	Preemption
9.	High	High	Preemption

1.6. Performance

For comparing the performance of Round Robin CPU scheduling algorithm with Fuzzy Round Robin CPU Scheduling algorithm, we did simulation on 1000 processes in groups of ten each.

We assumed random burst time of processes and their inter arrivals time are random. 10ms is assumed to be the Max burst time of a process. 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 column chart given in **Fig. 7** given above compares the performance of the proposed algorithm with its counterpart.

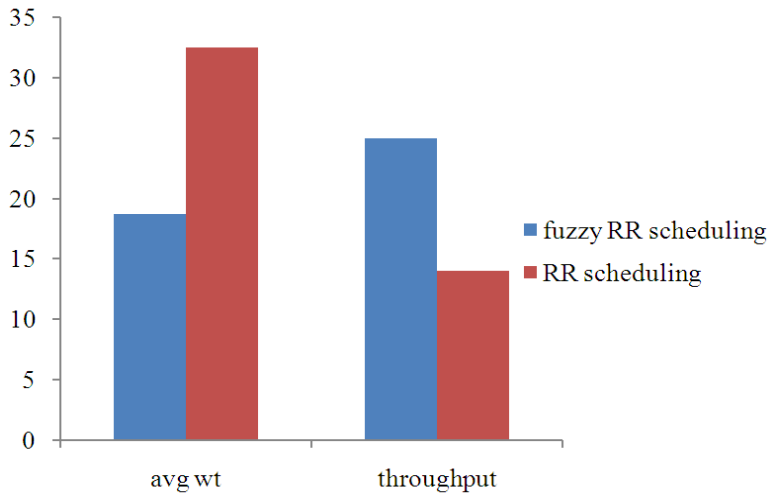

Fig. 7. Performance comparison of Fuzzy RR CPU scheduling and RR CPU scheduling

The average waiting time of Fuzzy Round Robin Scheduling is found to be lesser than the same for non fuzzy counterpart. The average throughput of Fuzzy Round Robin Scheduling is found to be more than the same for non fuzzy counterpart.

2. CONCLUSION

In this study two FIS has been constructed one for deciding the value of time quantum and another for deciding the preemption. The proposed algorithm using these has been presented. The rule base and membership functions may be fine tuned further to give much better performance

3. REFERENCES

Alam, B., M.N. Doja and R. Biswas, 2009. Improving the performance of Fair Share Scheduling algorithm using Fuzzy logic. Proceedings of the International Conference on Advances in Computing, Communication and Control, Jan. 23-24, ACM Press, Mumbai, Maharashtra, India, pp: 567-570. DOI: 10.1145/1523103.1523216

Alam, B., M.N. Doja and R. Biswas, 2011. Fuzzy HRRN CPU Scheduling Algorithm. Int. J. Comput. Sci. Inform. Sec., 9: 120-124.

Andrew, S.T. and S.W. Albert, 2006. Operating Systems: Design and Implementation. 3rd Edn., Pearson Prentice Hall, Upper Saddle River, N.J., ISBN-10: 0131429388, pp: 1054.

Hammam, A. and N.D. Georganas, 2008. A comparison of Mamdani and Sugeno fuzzy inference systems for evaluating the quality of experience of Hapto-Audio-Visual applications. Proceedings of the IEEE International Workshop on Haptic Audio visual Environments and Games, Oct. 18-19, IEEE Xplore Press, Ottawa, Ont., pp: 87-92. DOI: 10.1109/HAVE.2008.4685304

Jang, J.S.R., 1993. ANFIS: Adaptive-network-based fuzzy inference system. IEEE Trans. Syst. Man Cybernet., 23: 665-685. DOI: 10.1109/21.256541

Mamdani, E.H. and S. Assilian, 1999. An experiment in linguistic synthesis with a fuzzy logic controller. Int. J. Human-Comp. Stud., 51: 135-147. DOI: 10.1006/ijhc.1973.0303

Silberschatz, A., P.B. Galvin and G. Gagne, 2008. Operating System Concepts. 8th Edn., Wiley, ISBN-10: 0470128720, pp: 992.

Simon, D., 2002. Training fuzzy systems with the extended Kalman filter. Fuzzy Sets Syst., 132: 189-199. DOI: 10.1016/S0165-0114(01)00241-X

Stallings, W., 2008. Operating Systems: Internals and Design Principles. 6th Edn., Prentice Hall, Harlow, ISBN-10: 6014226184, pp: 822.

Sugeno, M., 1985. Industrial Applications of Fuzzy Control. 3rd Edn., North-Holland, Amsterdam, ISBN-10: 0444878297, pp: 269.

Wang, L.X., 1997. A Course in Fuzzy Systems and Control. 1st Edn., Prentice Hall PTR, Upper Saddle River, ISBN-10: 0135408822, pp: 424.