CPU SCHEDULING POLICIES USING FUZZY LINGUISTIC VARIABLES IN REAL WORLD ENVIRONMENT

Jaspreet kaur*,Silky Miglani**T.P.Singh***
*Research scholar (CSE) Yamuna Institute of Engg. & Tech. Gadholi
**Assistant Prof.Yamuna Institute of Engg. & Tech. Gadholi,Yamuna Nagar.
***Professor (Maths and Computer Science)Yamuna Institute of Engg. & Tech. Gadholi,Yamuna Nagar.

ABSTRACT

The performance of CPU depends on the scheduling policies adopted. These policies provide a schedule framework for execution of processes waiting in ready queue. Various scheduling algorithms as FCFS, SJF, Priority, Round Robin are already existing in the literature for operating system. In an interactive environment, these scheduling policies differ in their efficiency depending upon the characteristics and criteria to be considered .Fuzzy logic can be integrated to decide the task to be implemented according to sequential order. In the present paper, we propose the integration of fuzzy logic with existing Scheduling policies. Fuzzy logic concept is used to select among different values by using reasoning which is approximate and vague. By Fuzzy Logic we are able to decide on the basis of already known knowledge that in which order instructions are given by the user. The inference system in fuzzy logic enables the scheduler to invoke the order of task as in the beginning rather waiting for high and low priorities .In this paper we have designed a C++ simulator to compare various CPU parameters. With Linguistic variables introduced through Fuzzy Inference Rule , the efficiency of different scheduling policies have been evaluated.

Keywords-Round Robin, Fuzzy logic, Ready Queue, Fuzzy Inference System.

1.INTRODUCTION

The CPU Scheduling is one of the important research area of Operating System. It acts as a backbone in any computer system as it decides to allocate resources among different processes efficiently. Scheduling algorithms play an important role in computing systems as a means of meeting user demands as per number of processes. Various scheduling algorithms such as priority, FCFS, SJF, RR, etc. are available but none is sufficient for real time task. In FCFS algorithm, the shortcomings are throughput can be low since long processes can hold the CPU, secondly turnaround time, waiting time, response time can be high for the same reason no prioritization occurs as the system has trouble to meet deadlines. The SJF algorithm selects the process with lowest burst time but the shortcoming of the algorithm is that it is very difficult to know the burst time of next CPU request. Secondly, it cannot be implemented at the level of short term CPU Scheduling. In Priority scheduling algorithm, the process with highest priority is assigned CPU first and so on. The low priority process gets interrupted by incoming of higher priority. Hence creates indefinite blocking and starvation of lowest priority process due to large number of high priority. Round Robin algorithm has high context switching rates and less throughput along with large waiting time, response time and turn around time. The efficiency of Round Robin is totally depends on the Quantum. If the quanta is too small, every process executes for a small amount time which causes frequent context switching. This puts overhead over CPU. Further if Quanta is too large, the process executes for long duration and hence it maximize waiting time. These shortcomings can be improved by adding fuzzy logic as an inference of decision with FCFS, SJF, Priority and Efficient Round Robin. In this paper, we discuss scheduling policies under fuzzy inference system rule which better suits for a real time environment.

2.LITERATURE SURVEY

Alexander[1999] stated that Multimedia applications have unique requirements that must be met by networks and operating system components. In context of multimedia applications, the CPU scheduler determines quality of service rendered. Efficient soft real-time processing is proposed by C. Lin et al[2004]. The more CPU cycles scheduled to a process, the more data can be produced faster ,which results in better quality ,more reliable output. Terry Regner & Craig Lacey[2005] introduced the concepts and fundamentals of the structure and functionality of operating systems .Ajit Singh[2010] developed a new approach for Round Robin scheduling which helps to improve the CPU efficiency in real time and time sharing operating system..Many researchers have tried to implement fuzzy logic to schedule the processes .Mamdani et al [1975] demonstrated an experiment through an experiment in linguistic variables with a fuzzy logic controller.A fuzzy based CPU scheduling algorithm is proposed by Shata J. kadhim et. Al[2010]. An Improved fuzzy-based CPU Scheduling(IFCS) algorithm for real time systems is proposed by H.S. Behera[2012]. Recently, Silky and T.P Singh [2014] extended the work of earlier researchers on the basis of quick sort by considering the CPU burst time in fuzzy environment. In this paper we are studying the fuzzy CPU scheduling algorithms on the basis of bubble sort in wide context through linguistic synthesis in fuzzy inference rule by using Mamdani model.

3.FUZZY LOGIC TERMINOLOGY:

3.1.Fuzzy logic:- A fuzzy logic was developed by Zadah(1965). Crisp set theory is based on bivalent logic where an object is either a member of a set or not. But with fuzzy logic an object can be a member of multiple sets with different degree of membership in each set. The main idea is that there are many cases where TRUE or FALSE or ON and OFF fail to describe a given situation. These cases require a sliding scale where variables can be measure as partially TRUE or mostly true and partially FALSE. Infact,fuzzy logic is the use of fuzzy set in linguistic form. A fuzzy logic system consists of 4 major modules such as fuzzification module, inference engine, knowledge base and defuzzification module. The process of fuzzy logic is first to collect the crisp set of inputs and convert it into fuzzy sets using fuzzy linguistic variables and membership functions. This is known as Fuzzification. Then an inference is made on the basis of set of rules. Finally, the resulting fuzzy output is mapped to a crisp output using the membership functions, in the defuzzification module(diagram 1).

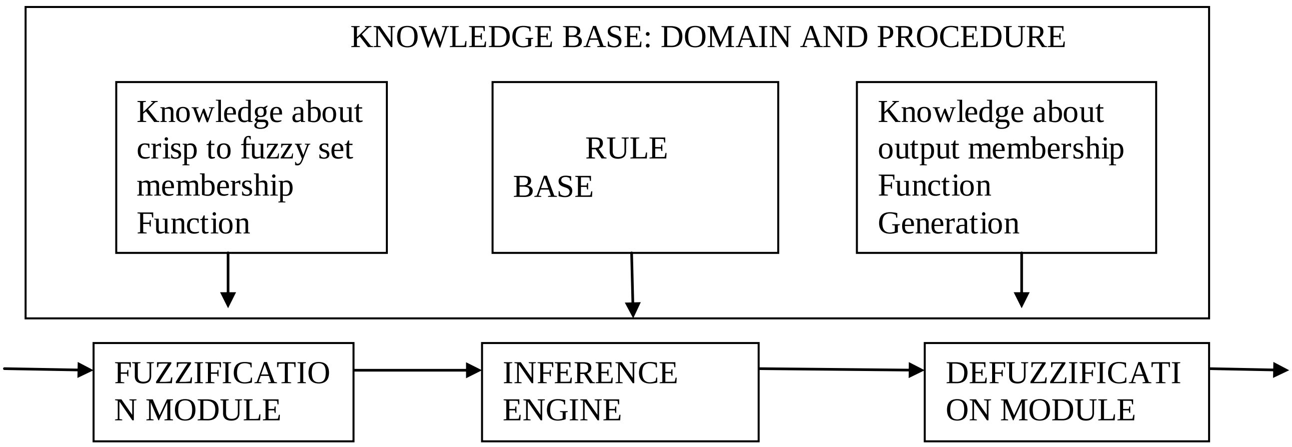

Diagram1: showing fuzzy logic based system

3.2 Linguistic Variables:- These are the inputs or output variables of the system whose values are non-numeric. The values may be words derived from natural language. **In our paper, different linguistic variables are used**. For average waiting time and average turnaround time, linguistic variables may be low, medium and high.

3.3 Membership Functions:- Membership functions are used in the Fuzzification and defuzzification steps of a FLS, to map the non-fuzzy input values to fuzzy linguistic terms and vice-versa. A membership function is used to quantify a linguistic term. In our paper, we consider 4 membership functions in input named algorithm as FCFS, SJF, Priority and RR. In input named sorting, membership functions are ascending, descending and random.

3.4 Fuzzy Inference System(FIS):- Fuzzy inference is the process of formulating the mapping from a given input to an output using fuzzy logic. The mapping provides a concept for decision making.An FIS consists of an input stage, a processing stage, and an output stage. The input stage maps the inputs, such as algorithm used, sorting technique applied, and MF to the appropriate membership functions and truth values. There are two common FIS:- (i) Mamdani's fuzzy inference method proposed in 1975 by Ebrahim Mamdani. (ii) Takagi-Sugeno-Kang, or simply Sugeno, method of FIS introduced in 1985.These two methods are similar as the procedure of fuzzifying the inputs and fuzzy operators. The difference between the two is that the Sugeno's output membership functions are either linear or constant but Mamdani' FIS gives the output membership functions to be fuzzy sets. In our paper, we used Mamdani' inference system.

4.Result Analysis:

We have designed simulator using C and C ++ for different scheduling algorithms to calculate their respective average waiting time and average turnaround time. First come First serve (FCFS), SJF, Priority, Round Robin have been calculated for the 12 processes on the basis of designed scheduling programs. These scheduling algorithms have been implemented through FIS. The results obtained after comparative study have been presented in tables (1-8) taking variable time quanta for round robin algorithm. After the aggregation process we find fuzzy variables for each output variables which needs defuzzification. The defuzzification has been done by Yager and Chang graded mean integration formulae The analytical study has been made for both triangular and trapezoidal functions for the defuzzified value of burst time. The Mamdani model is used as a processing model in order to generate the inference rules for fuzzy inference system(FIS). This model expects the output parameters to be fuzzy. The fuzzy model design for comparative study of CPU scheduling policies has been depicted in the following figures through MATLAB (fig 1-8(c)).

Table 1:CPU Burst time(Triangular Membership function)

S. No.	N	Burst time T (a)-ns	Burst time T (b)-ns	Burst time T (c)-ns	Final values(Nanoseconds) CPU BURST(TRIANGULAR)
1	1000	384.170	407.453	582.076	473.5555
2	2000	954.6056	931.3226	1164.153	1001.17173
3	3000	1653.098	1676.381	1804.438	1726.82767
4	4000	2735.76	3003.515	2852.175	2941.4266667
5	5000	3573.59	3853.347	3958.121	3981.524
6	6000	4994.217	5261.973	6170.012	5653.9046667
7	7000	5809.125	6193.295	6612.39	6461.05
8	8000	4330.65	7986.098	8381.903	9336.5156667
9	9000	9790.529	10838.27	15646.22	12790.167
10	10000	13527.46	13911.63	17997.81	15401.7466667
11	12000	19057.19	19825.53	20232.98	20217.46
12	15000	22398.31	23351.74	23818.58	23825.1633333

Table 2:CPU Burst time(Trapezoidal Membership function)

S.No.	N	Burst time T (a)-ns	Burst time T (b)-ns	Burst time T (c)-ns	Burst time T (d)-ns	Final values(Nanoseconds) CPU BURST(TRAPEZOIDAL)
1	1000	384.170	407.453	582.076	593.718	492.812667
2	2000	954.6056	931.3226	1164.153	1199.078	1057.4391333
3	3000	1653.098	1676.381	1804.438	1874.287	1748.1705
4	4000	2735.76	3003.515	2852.175	3713.649	3026.7981667
5	5000	3573.59	3853.347	3958.121	3966.479	3860.5008333
6	6000	4994.217	5261.973	6170.012	7520.43	5896.4361667
7	7000	5809.125	6193.295	6612.39	7706.694	6521.1981667
8	8000	4330.65	7986.098	8381.903	10721.85	7964.750333

9	9000	9790.529	10838.27	15646.22	19802.25	13760.2931667
10	10000	13527.46	13911.63	17997.81	18137.51	15913.975
11	12000	19057.19	19825.53	20232.98	20861.63	20005.9733333
12	15000	22398.31	23351.74	23818.58	24563.63	23550.43

Table 3: Comparison of Average Waiting & Turnaround Time in Triangular Fuzzy Environment. Total Processes=12(ASCENDING SORTED CPU BURST)

O/P Parameters	First Come First Serve	Shortest Job First	Priority	Round-Robin(TQ-50ns)
Average Waiting Time(ns)	22927.536458	22927.53645	22927.544271	55764.56250
Average Turnaround Time(ns)	31578.40	31578.419922	31578.41	65165.437502

Table 4: Comparison of Average Waiting & Turnaround Time in Triangular Fuzzy Environment Total Processes=12(RANDOM SORTED CPU BURST)

O/P Parameters	First Come First Serve	Shortest Job First	Priority	Round-Robin(TQ-50ns)
Average Waiting Time(ns)	57758.519531	22927.544922	31640.330078	45885.453125
Average Turnaround Time(ns)	66409.320312	31578.41992	39331.136719	54536.253906

Table 5: Comparison of Average Waiting & Turnaround Time in Triangular Fuzzy Environment Total Processes=12(DESCENDING SORTED CPU BURST)

O/P Parameters	First Come First Serve	Shortest Job First	Priority	Round-Robin(TQ-50ns)
Average Waiting Time(ns)	72232.085938	22927.544922	23149.703125	46158.7968
Average Turnaround Time(ns)	80882.960938	31578.41992	31800.578125	54809.6757

Table 6: Comparison of Average Waiting & Turnaround Time in Trapezoidal Fuzzy Environment Total Processes=12(ASCENDING SORTED CPU BURST)

O/P Parameters	First Come First Serve	Shortest Job First	Priority	Round-Robin(TQ-75ns)
Average Waiting Time(ns)	22993.711484	22993.763672	22993.76367	45812.515625
Average Turnaround Time(ns)	31643.669922	31643.662109	31643.662109	54462.417969

Table 7: Comparison of Average Waiting & Turnaround Time in Trapezoidal Fuzzy Environment Total Processes=12(RANDOM SORTED CPU BURST)

O/P Parameters	First Come First Serve	Shortest Job First	Priority	Round-Robin(TQ-25ns)
Average Waiting Time(ns)	57785.667969	22993.763672	37363.207031	45985.433594
Average Turnaround Time(ns)	66435.562500	31643.662109	46013.105469	54635.324219

Table8:Comparison of Average waiting time and turn around time in trapezoidal fuzzy system Total Processes=12(DESCENDING SORTED CPU BURST)

O/P Parameters	First Come First Serve	Shortest Job First	Priority	Round-Robin(TQ-50ns)
Average Waiting Time(ns)	72155.117188	22993.763672	22993.763672	46114.597656
Average Turnaround Time(ns)	80805.007812	31643.662109	31643.662109	54764.50000

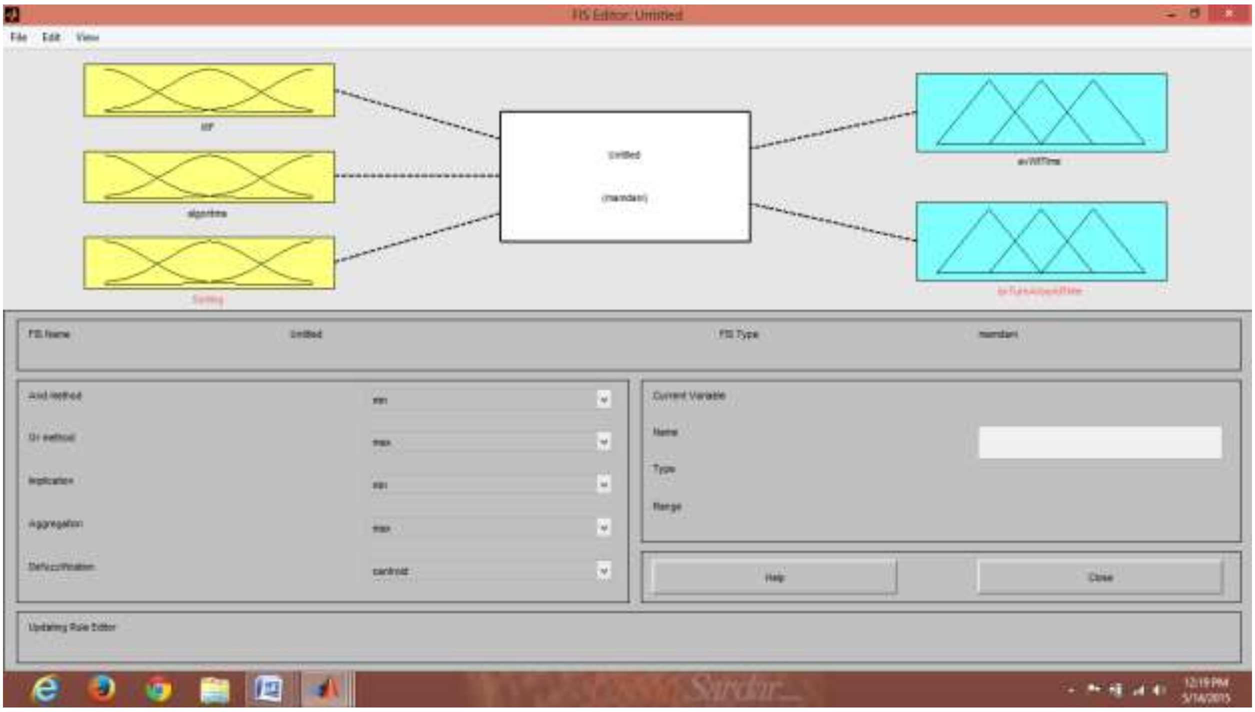

Fig1: FIS showing I/O parameters of the Model

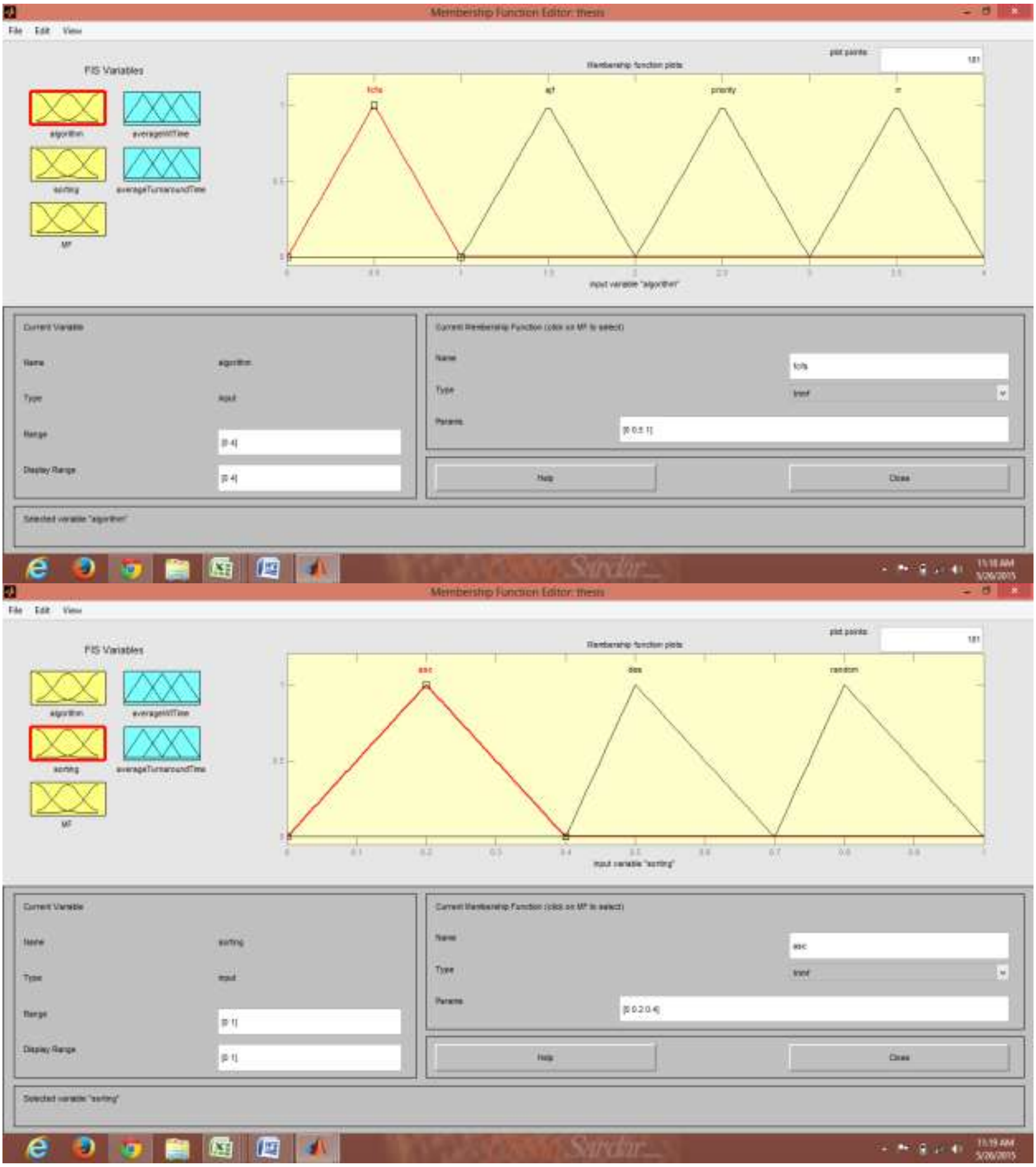

Fig2 : showing membership functions of input named "algorithm"

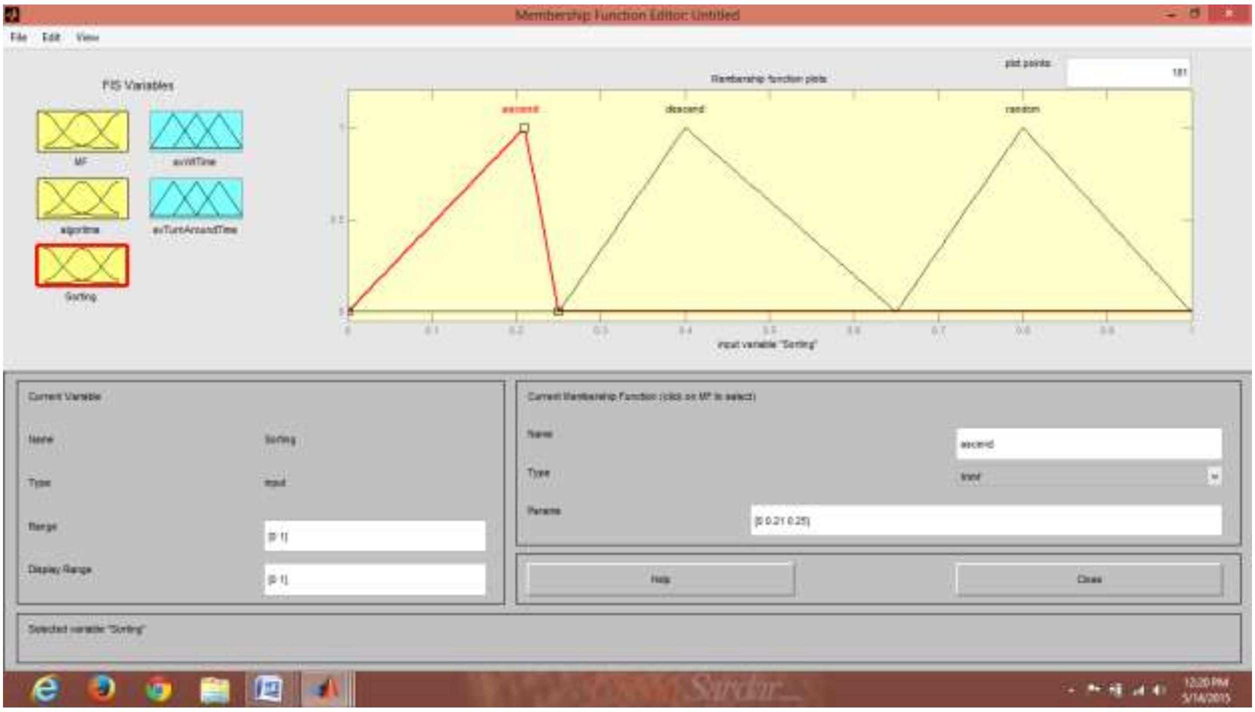

Fig3: FIS showing MF in the input variable "Sorting".

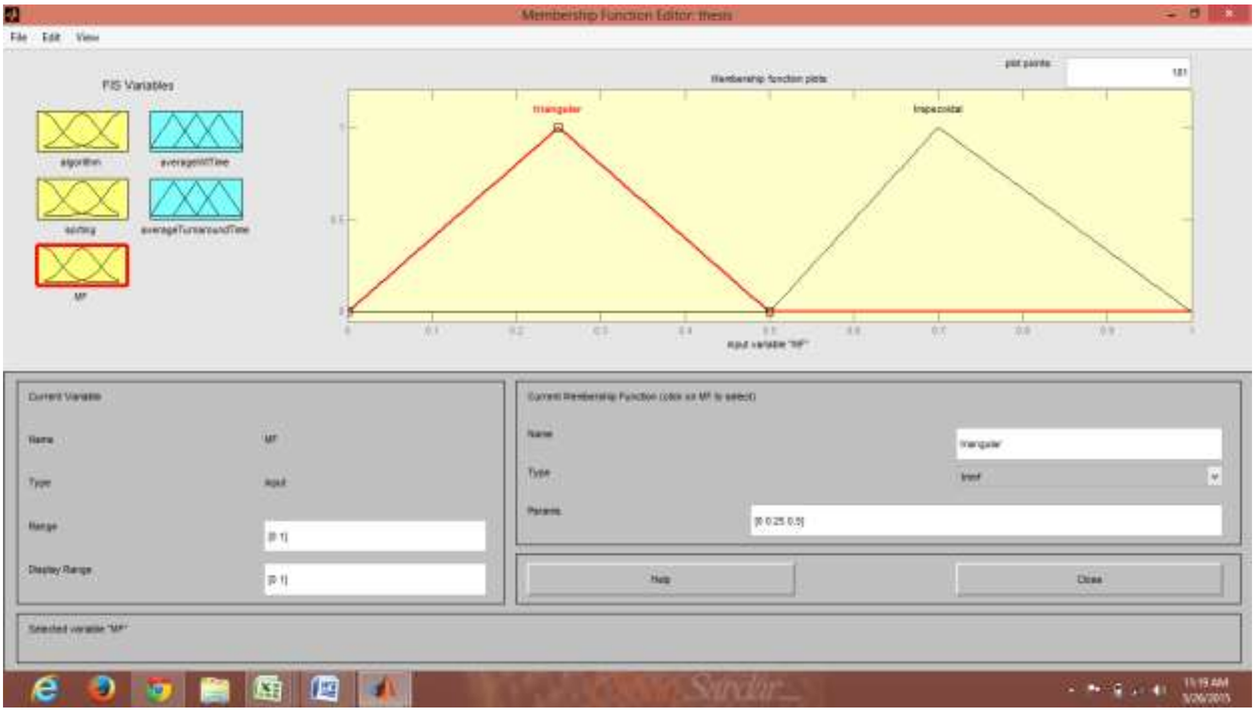

Fig4: showing membership functions of input named"MF"

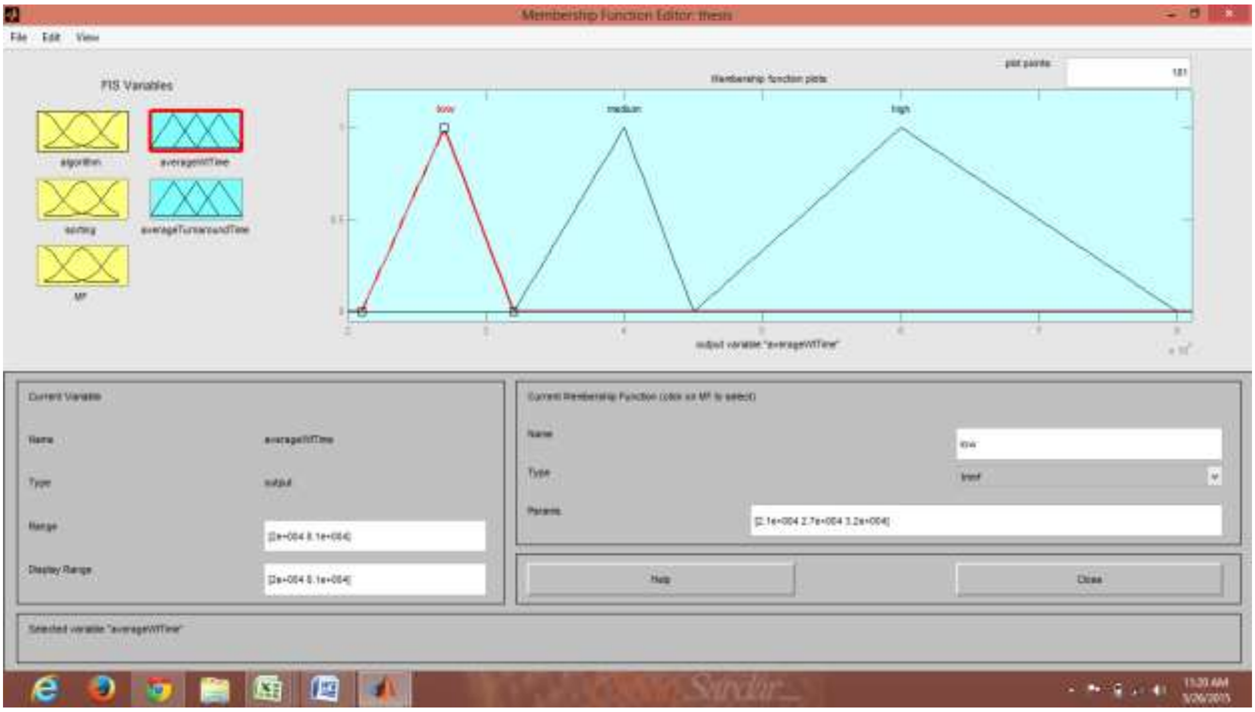

Fig 5: showing membership functions of output named"averagewttime"

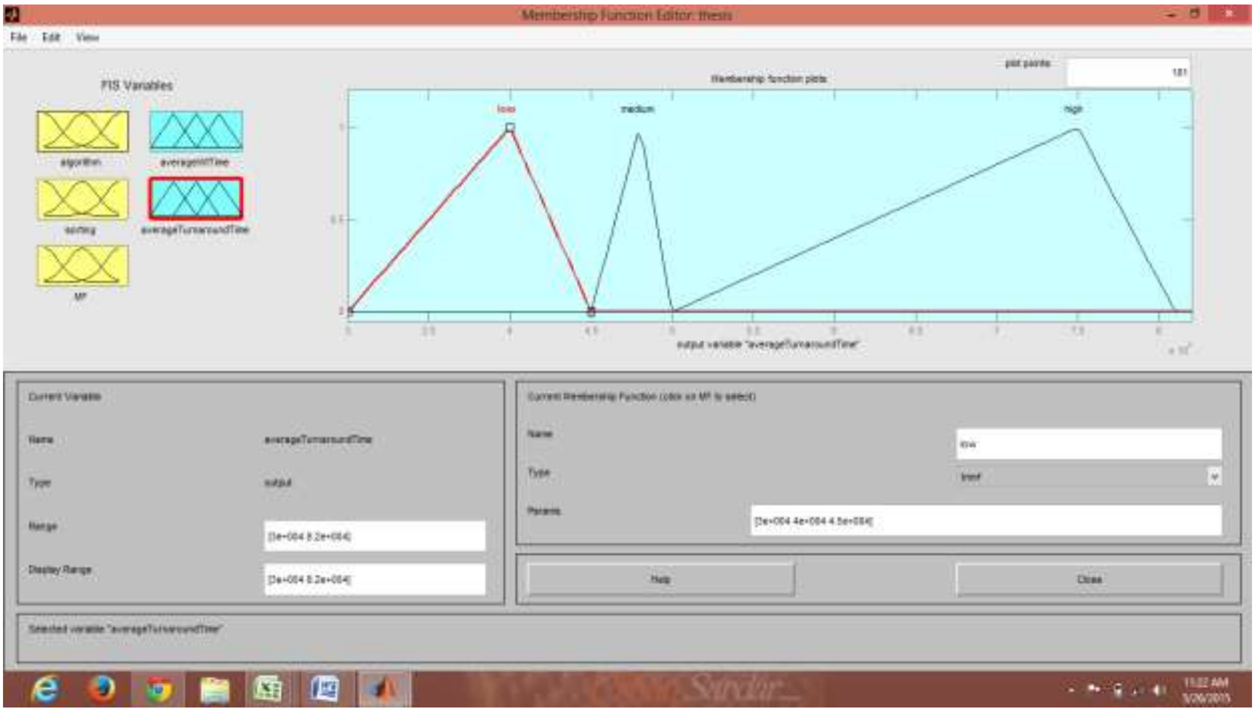

Fig 6: showing membership functions of output named"averageTurnaroundTime"

Fuzzy Inference Rules

The different CPU scheduling algorithms are analyzed through Fuzzy Inference System where the following parameters are taken:

1.INPUT PARAMETERS

i. MF(MEMBERSHIP FUNCTION)

Range: [0-1]

Values: triangular[0-0.5],trapezoidal[0.5-1]

ii. ALGORITHM

Range: [0-4]

Values: FCFS[0-1],SJF[1-2],PRIORITY[2-3],RR[3-4]

iii. SORTING

Range: [0-1]

Values: ASC[0-0.4],DES[0.4-0.7],RANDOM[0.7-1]

2.OUTPUT PARAMETERS

i. AveragewtTime

Range: [20000-80000]

Values: low[21000-32000],medium[32000-45000],high[45000-80000

ii. AverageTurnroundTime

Range: [30000-82000]

Values: low[30000-45000],medium[45000-50000],high[50000-81000]

Different rules are given below:

1.If (MF is triangular)and(sorting is ascending)and(algorithm is FCFS)then(averagewttime is low)(averageturnaroundtime is low)

2.If (MF is triangular)and(sorting is descending)and(algorithm is FCFS)then(averagewttime is high)(averageturnaroundtime is high)

3.If (MF is triangular)and(sorting is random)and(algorithm is FCFS)then(averagewttime is high)(averageturnaroundtime is high)

4.If (MF is triangular)and(sorting is ascending)and(algorithm isSJF)then(averagewttime is low)(averageturnaroundtime is low)

5.If (MF is triangular)and(sorting is descending)and(algorithm is SJF)then(averagewttime is low)(averageturnaroundtime is low)

6.If (MF is triangular)and(sorting is random)and(algorithm is SJF)then(averagewttime is low)(averageturnaroundtime is low)

7.If (MF is triangular)and(sorting is ascending)and(algorithm is priority)then(averagewttime is low)(averageturnaroundtime is low)

8.If (MF is triangular)and(sorting is descending)and(algorithm is priority)then(averagewttime is low)(averageturnaroundtime is low)

9.If (MF is triangular)and(sorting is random)and(algorithm is priority)then(averagewttime is low)(averageturnaroundtime is medium)

10.If (MF is triangular)and(sorting is ascending)and(algorithm is rr)then(averagewttime is high)(averageturnaroundtime is high)

11.If (MF is triangular)and(sorting is descending)and(algorithm is rr)then(averagewttime is high)(averageturnaroundtime is high)

12.If (MF is triangular)and(sorting is random)and(algorithm is rr)then(averagewttime is high)(averageturnaroundtime is high)

13.If (MF is trapezoidal)and(sorting is ascending)and(algorithm is FCFS)then(averagewttime is low)(averageturnaroundtime is low)

14.If (MF is trapezoidal)and(sorting is descending)and(algorithm is FCFS)then(averagewttime is high)(averageturnaroundtime is high)

15.If (MF is trapezoidal)and(sorting is random)and(algorithm is FCFS)then(averagewttime is high)(averageturnaroundtime is high)
16.If (MF is trapezoidal)and(sorting is ascending)and(algorithm isSJF)then(averagewttime is low)(averageturnaroundtime is low)
17.If (MF is trapezoidal)and(sorting is descending)and(algorithm is SJF)then(averagewttime is low)(averageturnaroundtime is low)
18.If (MF is trapezoidal)and(sorting is random)and(algorithm is SJF)then(averagewttime is low)(averageturnaroundtime is low)
19.If (MF is trapezoidal)and(sorting is ascending)and(algorithm is priority)then(averagewttime is low)(averageturnaroundtime is low)
20.If (MF is trapezoidal)and(sorting is descending)and(algorithm is priority)then(averagewttime is low)(averageturnaroundtime is low)
21.If (MF is trapezoidal)and(sorting is random)and(algorithm is priority)then(averagewttime is medium)(averageturnaroundtime is medium)
22.If (MF is trapezoidal)and(sorting is ascending)and(algorithm is rr)then(averagewttime is high)(averageturnaroundtime is high)
23.If (MF is trapezoidal)and(sorting is descending)and(algorithm is rr)then(averagewttime is high)(averageturnaroundtime is high)
24.If (MF is trapezoidal)and(sorting is random)and(algorithm is rr)then(averagewttime is high)(averageturnaroundtime is high).

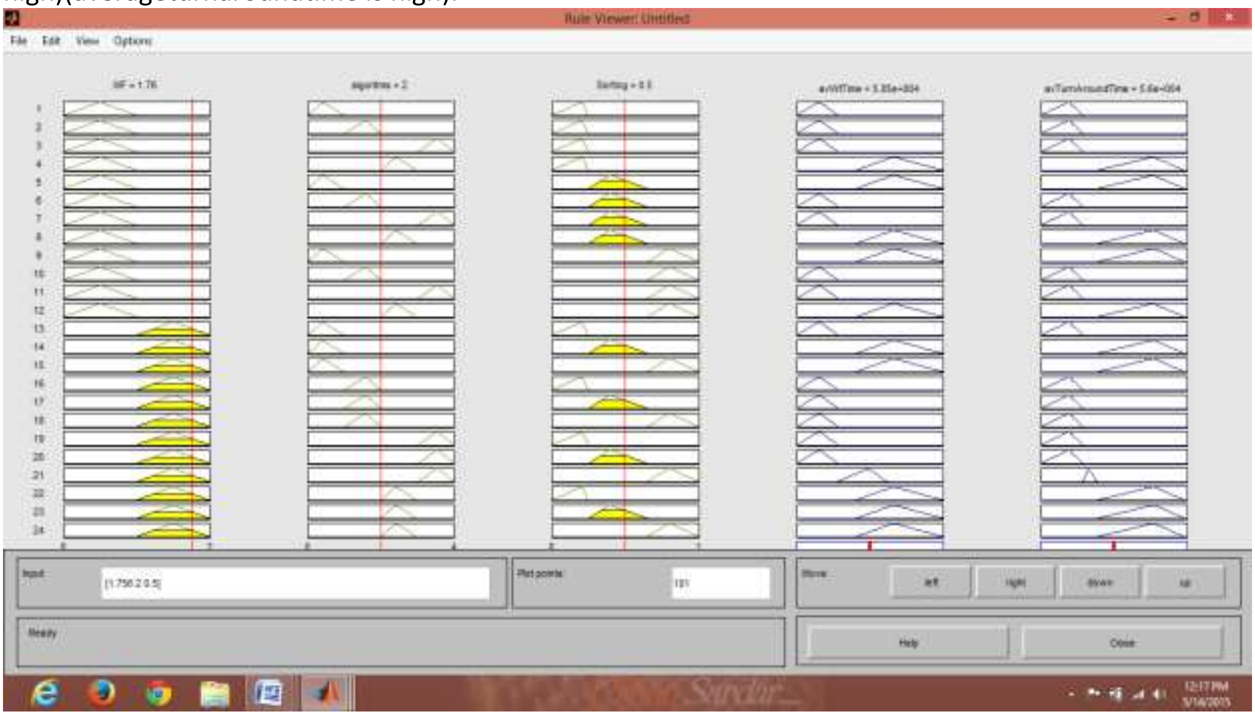

Fig7: Inference Rules through Mamdani Model

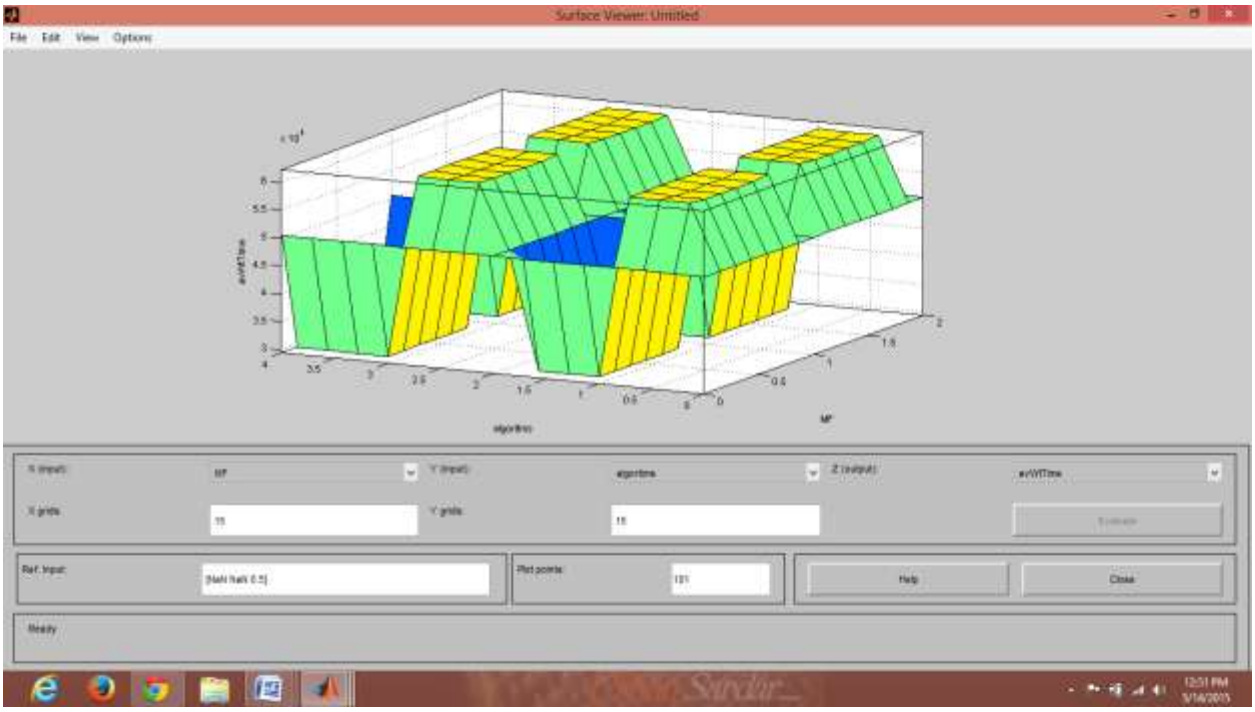

Fig8(a): Surface view of average waiting time vs average turnaround time

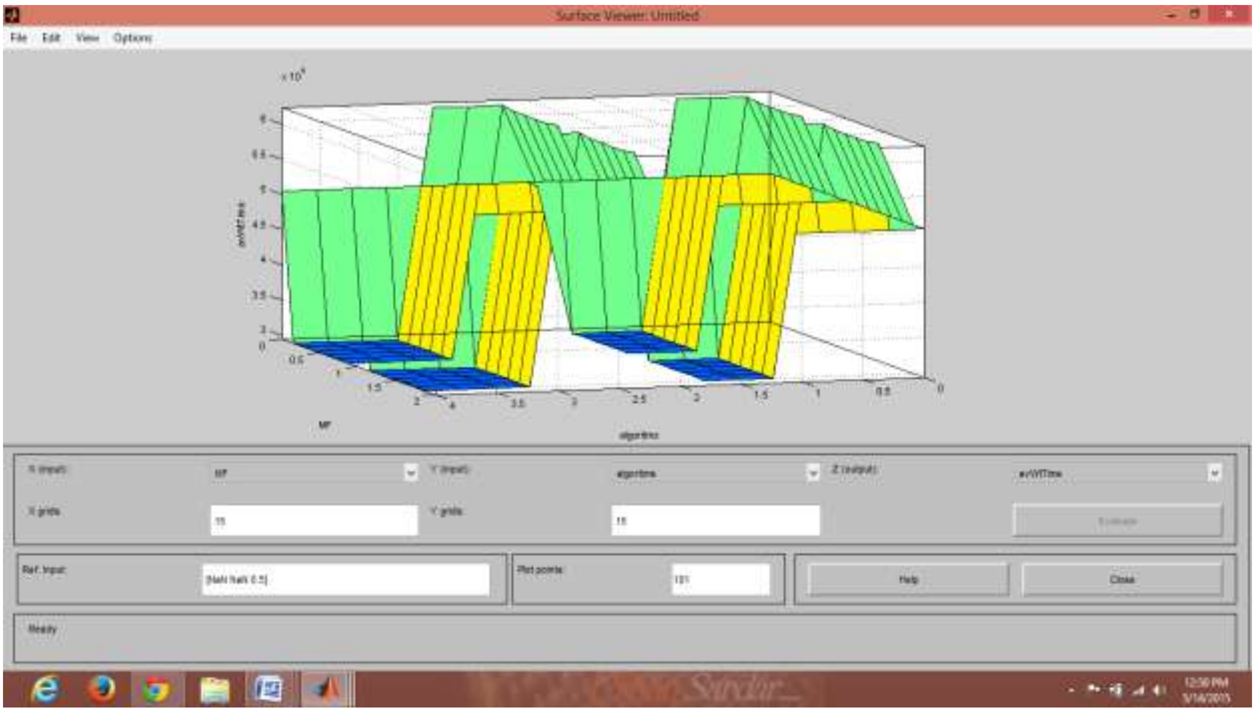

Fig8(b):Surface View for average waiting time vs average turnaroundtime.

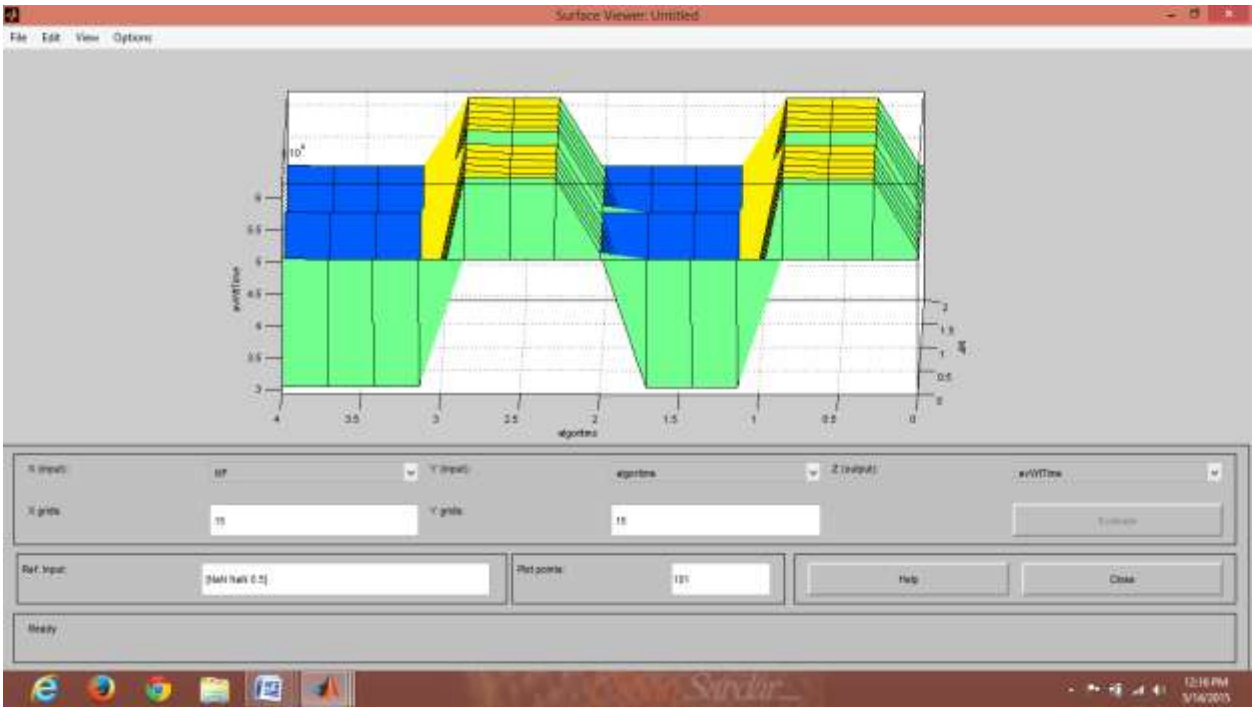

Fig8(c):Surface View for average waiting time vs average turnaroundtime.

Discussion on simulation results

It is clear from various case studies presented above that SJF and Priority scheduling policies show almost same results in terms of average turnaround time and waiting time and lower than Round Robin and FCFS. Further SJF policy gives almost a constant value of average waiting time and turnaround time in both triangular and trapezoidal cases which is minimum than other policies. It also maximize the CPU throughput. We conclude that SJF is best policy while Round Robin takes more time than other policies.These results shows that system performance is far improved when compared to round robin and FCFS.But FCFS is more advantageous than others as it reduces extra burden of calculations of response time.

Conclusion

The analytical study for various policies through fuzzy inference system shows that the average waiting time as well as average turnaround time of scheduling policies from FCFS, SJF, Priority and Round Robin gradually increases as depicted through FIS Model. The fuzzy CPU scheduling algorithm reduces the cost of calculating response ratio.The CPU scheduling comparison can be further improved by improving fuzzification process.A new fuzzy neuro based CPU scheduling algorithm or dynamic programming algorithm can be generated which may further brings an improvement.The work can be further improved by taking more accurate formula for evaluating fuzzy membership value which can reduced the waiting time and turnaround time.

ACKNOWLEDGEMENT:

The first author would like to thanks my friend Ms.Ramanpreet kaur for her valuable suggestion in improving the paper.

References:

[1]. L.A. Zadah," Fuzzy set and system ", System theory Brooklyn N.Y.pp(29-37),1965.

[2].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.

[3]. Alexander Taranovsky, " CPU Scheduling in Multimedia Operating Systems", Research Report,1999.

[4]. C.Lin and S. A. Brandt," Efficient soft real-time processing l in an integerated system " in Proc. 25th IEEE Real-Time Systems Symp,2004

[5]. Terry Regner, craig lacy,"An Introductory Study of Scheduling Algorithms ",CPSC 321: A Project Assignment, Feb 2005.

[6].Ajit Singh, Priyanka Goyal," An Optimized Round Robin Scheduling Algorithm for CPU scheduling Algorithm for CPU Scheduling" (IJCSE)International Journal on Computer Science and Engineering,Vol.02,No.07,2010.

[7].Shata J.Kadhim, Kasim M. Al-Aubidy, "Design and Evaluation of a Fuzzy Based CPU scheduling Algorithm", Springer-verlag Berlin Heidelberg 2010,CCIS 70,pp.45-52,2010.

[8]. H.S. Behera, Ratikanta Pattanayak and Priyabrata Mallick," An improved Fuzzy-Based CPU Scheduling(IFCS) Algorithm for Real Time Systems", International Journal of Soft Computing and Engineering (IJSCE),Volume-2,Issue-1,March 2012.