THINK INDIA JOURNAL

ISSN: 0971-1260

Vol-22-Issue-17-September-2019

A Fuzzy Built Methodology towards Predicting the Next CPU-Burst For SJF Scheduling Algorithm.

Parul Prakram[#1],Mohit Prakram[*2]

[#] *Department of CSE,LPU,Jalandhar.* [*]*Department of CSE,LPU,Jalandhar.*

[1]*mohit.15284@lpu.co.in*

[2]parul.16466@lpu.co.in

***Abstract*— The objective of multiprogramming is to maximize the CPU utilization. Impression is to retain many processes in main memory at all times. when a single procedure in the system is looking for the admittance of the central processing unit for its total execution ,then at that spell, CPU is engaged from that procedure by the OS and specified to another procedure, which is correspondingly looking for same.**

The process is then choosen by a central processing unit scheduling algorithm, and altogether this is completed to retain the CPU demanding at all the period and to in order to devising this perception , SJF is most popular CPU job scheduling algorithm, that describes that on every occasion when a procedure has to interval, then in this moment, the Central processing unit will be accessible to the smallest procedure in the memory.

This algorithm had a few flows as when an operating system is doing the task of executing a process, it is not able to predict the execution time of the process which it is executing at that time. The precise implementation time of that procedure would visible thereafter the complete execution. The time of a process to execute is estimated before running, there were several algorithms has introduced.

The unique method is to approximate SJF scheduling. Since process execution is assortment of a cycle of CPU execution along ith I/O wait, so we accept that the subsequent CPU burst is analogous in extent to the preceding CPU burst . The exponential average of estimated lengths of previous CPU bursts, the next CPU burst is usually expected.This algorithm is identified as Exponential Average Algorithm (EAA).In this paper a Fuzzy-built Algorithm for the calculation of next CPU burst has been proposed.

The algorithm mentioned in this research article reveals the intellectual fuzzy system employed to examine the performance time occupied by the operating system for its accomplishment that depends upon the performance of the previous CPU burst. The core or the main idea of the system is a database that shelters "if-then rules" and the comparative analysis of the Exponential Average Algorithm . The Fuzzy oriented systems reveals that the proposed approach is more optimal, hence it predicts much nearest values to the actual CPU-burst if matches with "exponential average algorithm".

Index Terms—Fuzzy system, Operating systems, CPU scheduling algorithms, SJF algorithm,

I. INTRODUCTION

So , to fit all the jobs, the main memory is pointlessly small hence the jobs are initially or in the beginning are set aside on the disk in the job pool or in the job queue ,Where the pool includes all the job which needs to be accomplished or the processes that is on the disk waiting to get the access of the main memory storage for their task to be completed or execution.

We are then put in a work queue, where processes who are waiting enter the system. Few investigation study could be engaged in the main storage from the task group.

The above mentioned process is identified as task scheduling and is accomplished through a "long-term-scheduler"or a task scheduler. Hence, the sequence of tasks in the memory could be a subgroup of the tasks reserved in the task group. The procedures that exists in theprime storage and are availa ble for completing and intialy the processes are kept in the availabl queue.

The OS choose one of the procedure which are in the available queue to operate. The above mentioned technique is identified as CPU job scheduling and is accomplished by a "short term scheduler" .

The system can ultimately have to wait to complete any task such as I / O service. The central processing unit would be indolent in a non-multitasking procedures. Due to , multitasking, the OS ,just logon to, and process another task that is chosen by a central processing unit task scheduling algorithm and forward on.

It is important to know that at any time there is only one syst em that can operate on any device and there are many applications that are ready and waiting for it however.

ISSN: 0971-1260
Vol-22-Issue-17-September-2019

Dispatcher is prime element which is intricate in the central processing unit scheduling. The dispatcher provides administration to the central processing unit for the method selection by the "short-term scheduler". So it is important for the dispatcher to be as profligate as possible, because it is started when each process changes. The time consumed by the transmitter to end a process and initiate next running process is termed as dispatch latency.
CPU scheduling contracts with the issue of the
methods to be assigned to the CPU in the ready queue.
Following are some CPU scheduling algorithms.

1.First-come First-Served Scheduling
2.Shortest Job First Scheduling
3.Priority Scheduling
4.Round Robin Scheduling
5.Multilayer queue scheduling
6.Multilayer response queue scheduling

II OVERVIEW

According to the SJF algorithm, it is allocated to the system in the memory which is aimed and is looking for
the least next CPU burst every time the CPU is free and If for the next two process, CPU bursts are the same then
the CPU Scheduling algorithm First Come First Serve is use d to break the tie. In[1]Rajesh,Kumar Garg et al. compared Round Robin efficiency, preventive priority and non-preventive scheduling policies on the
basis of average waiting times and average turn around times for a variety of processes.
The author found that starvation is possible in case of priority scheduling whereas the average waiting time under RR scheduling is quite long.

In [2] S. N. Mehmood Shah et al. used a self made CPU scheduling simulator to carry out widespread experiment using , Windows Vista operating system on an Intel-Core2 Duo. All the prominent CPU scheduling algorithms were simulated and a comparison of these algorithms was done on the basis of three parameters – Average-Waiting Time, Average-Turn Around Time, and Average-Response Time. The results prove that the Shortest Job First algorithm (specially the preemptive version) was very optimal.
It has been long known that Shortest, residual processing time (preemptive SJF) algorithm has the, least mean response time of any scheduling policy, given any arrival schedule and job sizes [3, 4]. These papers prove that, queuingdiscipline,that always operates with the shortest rema ining processing time minimizes the number of jobs in the ne -twork
The SJF scheduling algorithm is known to be efficient in that it gives a given set of system the minimum average waiting t ime[5].Shifting a small process before a long one decreases the short,process's waiting time more than it increases the lon g process's waiting time. The average waiting time hence decreases.
In [6] Nikhil Bansal et al. stated that the meaning of starvation of a process is stated as unfairness or non availability of SRPT on the jobs which are lengthy or taking more time for the execution.
Long jobs are often thought to have worse average performa nce under SRPT than under other policies, since SRPT favor s small jobs.The argument given is that if a scheduling plan s ucceeds in increasing the reaction time for small jobs, then th e response time for large jobs would need to be significantly increased.This argument applies to planning policies that do not take advantage of scale, see the popular Kleinrock Conse rvation Law[7],[8,page197].
This argument applies to policies that do not take advantage of the scale, see Kleinrock's common law on conservation
The main difficulty in using SJF .If we talk about CPU scheduling algorithm in a actual computer system then it means that whenever a procedure originates in a system, its completion time is not known before. There is no way by which we can find about the length or duration of the next CPU burst. Only when we are done with the execution of the real CPU-burst then only it would appear [9,10].
Exponential Average Algorithm is one approach that is used in order predict the next CPU-burst of a process [11]. Where We might not identify the length of next CPU-burst, however we may be able to expect it based on the concept of past history.
In [12] Abdolghader Pourali et al. stated that
The time series is a group of interpretations ordered by time. Forecasting of the next CPU-burst is also a type of time series and is done through "exponential average algorithm". This task is finished by gathering all the previous CPU-bursts time , which were completed in the processor before
The author has also proposed a Fuzzy built Scheduling-Algorithm for estimation of next CPU-burst time to implement Shortest Process Next CPU scheduling algorithm. This algorithm forecasts the succeeding CPU-burst time of process, using the known time series and fuzzy system.
The author explains that by use of intelligent systems such as Fuzzy systems, it is likely to evaluate a lot of time series along with the CPU-burst time series with required accuracy.

In [13] Bashir Alam et al. proposed a Fuzzy based algorithm for finding the time quantum of Round Robin CPU scheduling algorithm. The author describes that a Fuzzy inference system (FIS) tries to evaluate answers from a

ISSN: 0971-1260
Vol-22-Issue-17-September-2019

information base using a fuzzy inference engine. The brain of the expert is Fuzzy inference engine system .

It provides the procedures for reasoning around the information in the knowledge-base and formulating the results. The authors suggested to use triangular membership functions and suggested to use two Fuzzzy inference engines.Which are Mamdani's inference engine [14], proposed by Ebrahim Mamdani in 1975 and the Sugeno inference engine implemented in 1985 [15]. The author has also provided the basic rule that need to be used.

It can be clear until you know that the average exponential al gorithm is one of the approaches used to predict the next CP U burst.Another solution is the fuzzybased algorithm suggest ed in [12] to predict the next CPU-burst.. But as no rule base has been suggested and no results are provided hence, it is nothing more than an approach or an idea. In this paper, a simple Fuzzy-based Algorithm has been proposed. This algorithm uses the rule base much similar to the one suggested in [13] and the membership function suggested in [12].

III CURRENT METHODOLOGY

The formula below shows the next CPUburst time's exponent ial average:

$$\tau_{n+1} = \alpha t_n + (1-\alpha)\tau_n$$

Where,

i. t_n is the actual length of nth CPU-burst.

ii. τ_n is the projected length of past CPU-burst.

iii. τ_{n+1} is the projected value of next CPU-burst.

iv. α is the forgetting coefficient on the range of [0, 1] i.e $0 \le \alpha \le 1$. The parameter α controls the relative weight of recent and past history in our prediction. The expected value can be tilted either to near or far from previous values by adj usting the value of α.

We consider three cases for the value of α.

i. If $\alpha = 0$, then

$$\tau_{n+1} = \tau_n$$

This means that the current history on which the work has been done till now has no impact and it is also presumed that current conditions of the case are also temporary.

ii. If $\alpha = 1$, then

$$\tau_{n+1} = \alpha t_n$$

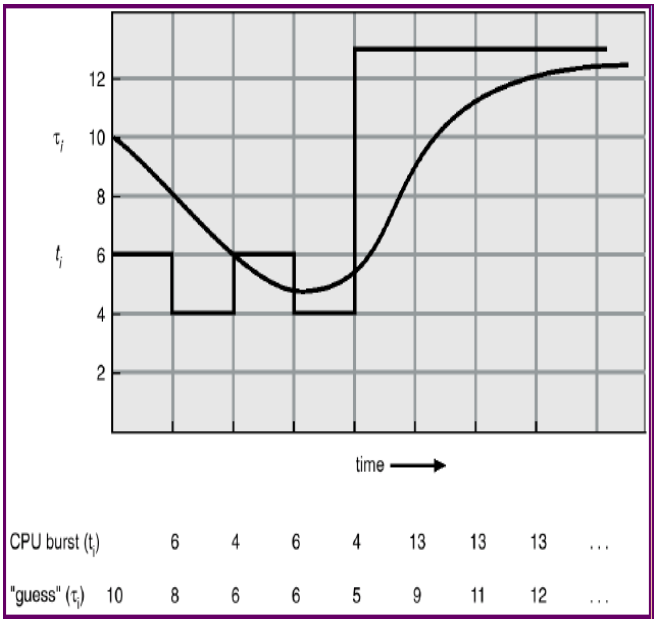
This means that the most recent CPU-burst matters and the history is expected to be old and irrelevant.

iii. If $\alpha = ½$ then

$$\tau_{n+1} = 1/2\,(t_n) + 1/2(\tau_n)$$

This means latest history and previuos history are equally weighted.

You may describe the initial value of 0 as a constant or an ag gregate average.Let us consider an example to understand this algorithm. We consider $\alpha = ½$ and $\tau_0 = 10$. Fig below displays an sample of this forecasting by use of exponential average algorithm. As the value of $\alpha = ½$, so case iii. is applied.



The proposed method uses a Fuzzy system for the prediction of the next process that wants to be scheduled for the CPU-burst time.The input given to the scheme, is the value of the former CPU burst The cost of the expected next CPU burst time for that process is the CPUburst of a process and t he performance of the device.Figure 1 shows a Fuzzy system' s main structure. In this structure, there are four sections of a Fuzzy system:Fuzzifier, Inference engine, Rule Base, Defuzzifier [16].

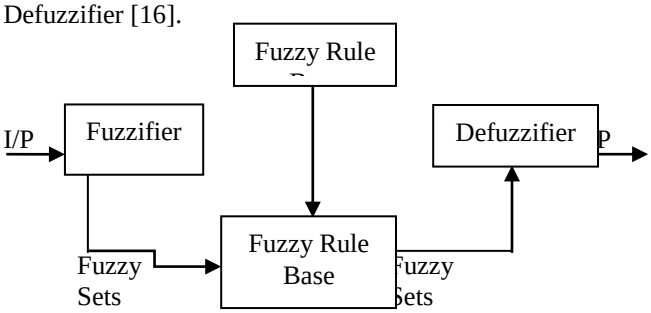


Fig 1: Fuzzy System

ISSN: 0971-1260
Vol-22-Issue-17-September-2019

IV PROJECTED METHODOLOGY

Following algorithm has taken the values of the process's of the past two actual CPU burst times as its input and then produces the next predicted CPU burst time for that process as the output..

This algorithm is using the idea of a triangular membership function, which is well-defined by a lower limit ' a' , an upper limit ' b' , and a value ' m' , where a < m < b.

Then in that case ,three membership tasks for each input as well as for the output would be nominated as:

Category- Triangular, Range 0-20

Low [0,5,10]

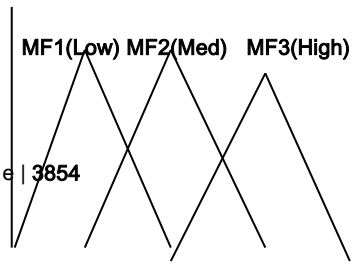
Medium [5,10,15]

High [10,15,20]

The equation given below is used to fuzzify the inputs.

$$\mu_A(x) = \begin{cases} 0 & x <= a \\ (x-a)/(m-a) & a < x <= m \\ (b-x)/(b-m) & m < x < b \\ 0 & x > b \end{cases}$$

Figure 2 shows a triangular membership function. In case we have more than one input variable then in that case(infact, the case we have)



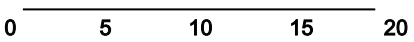


Fig 2: Triangular Membership function.

RULE BASE FOR FIS

Table 1 shows the rule base used for the proposed FIS. This is a set of some required logic rules, which are in the form of Ifthen statements where the IF part is said to the "antecedent" and the THEN part is said to be the "consecutive." There are hundreds of rules for com mon fuzzy inference subsystems.

Table 1: Rule Base for FIS

S.No	Input 1	Input 2	Output
1.	Lower	Lower	Lower
2.	Lower	Medium	Lower
3.	Lower	Higher	Higher
4.	Medium	Lower	Medium
5.	Medium	Medium	Medium
6.	Medium	Higher	Medium
7.	Higher	Lower	Lower
8.	Higher	Medium	Lower
9.	Higher	Higher	Higher

FUZZY INFERENCE ENGINE

Then the degree of membership for the output value will be defuzzified by using Mamdani's defuzzification method.this ,method,hasgivenmanymethods.The centroid method is used in this paper to defuzzify the efficiency.

The formula for centroid method is given below.

ISSN: 0971-1260
Vol-22-Issue-17-September-2019

$$X^* = \frac{\sum A\overline{x}}{\sum A}$$

Where,

X* is the desired crisp value (Next Predicted CPU-burst)

A is the area of the segment of aggregated fuzzy set.

$\overline{X}$, is the corresponding centroid

After defuzzification the result will be a crisp value which will be the next predicted CPU-burst time for a process. If the result comes out to be in fractions, then it is rounded off.

PROPOSED FUZZY BUILT ALGORITHM

Step 1 – Find the result of previous two CPU-bursts for a process may be using Exponential Average algorithm.

Step 2 – Give these values as input to the FIS designed above.

Step 3 – Take the output of FIS as the estimated next CPU-burst of that process.

Step 4 – Repeat this for all those processes which are in wait for the CPU in the ready queue.

Step 5 – Invoke SJF scheduling algorithm.

SIMULATION METHODOLOGY

MATLAB is used to simulate the suggested Fuzzy-based algorithm.Matlab (matrix laboratory)which is a programmin g language of the fourth generation and numerical computing environment. Fuzzy built algorithm is computer-generated by making use of Fuzzy logic tool box by making use of the Matlab.

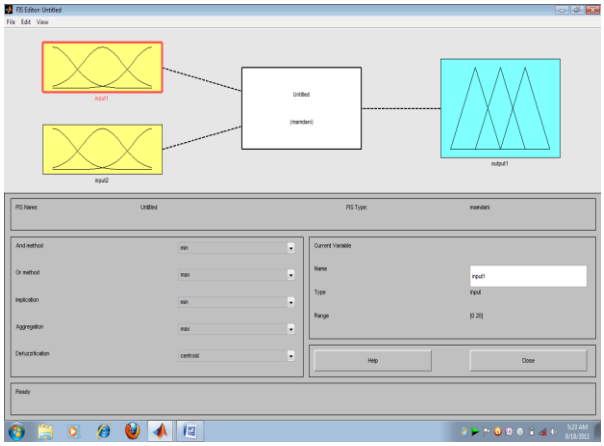
Following toolbox helps you to use simple logic rules to mod el complex system behavior and then apply these rules in a F uzzy Inference System (FIS).

Fuzzy Logic Toolbox give tools create and change fuzzy inference systems within the framework of MATLAB. Also one can fit in your fuzzy systems into recreations with Simulink..

v OBSERVATIONS & RESULT

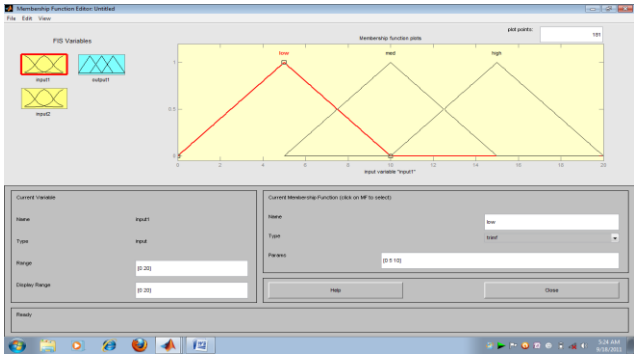
We applied the Fuzzy built algorithm for the same task which was used to determine the Exponential Average Algorithm.

As, the proposed algorithm requires the values of previous two real CPU-bursts of a process, so while designing the fuzzy system it was necessary to choose two inputs and one output. This is shown in the screenshot below.

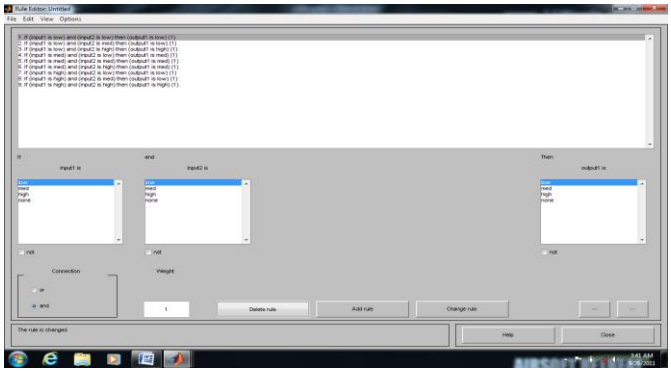


Then the membership function for each Input and Output was defined. Fuzzy logic tool box has a list of prominent membership functions, you just have to choose the one you want to use. We have used triangular membership function in our work. This is shown in the screenshot.

ISSN: 0971-1260
Vol-22-Issue-17-September-2019



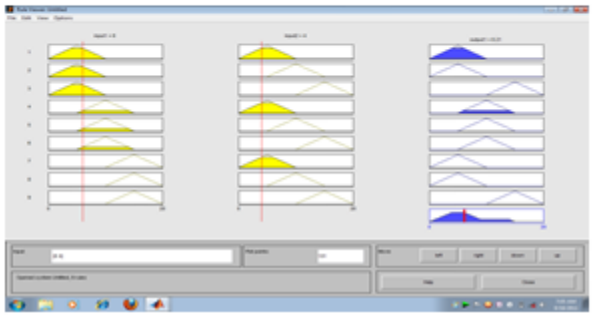
After defining the membership functions, the rule base to be used was designed or added to the fuzzy system. Fuzzy logic toolbox allows you to simply add rules to the designed system. Figure below shows the rule base added. It has a total of nine rules defined in it.



Now, when the Fuzzy system was designed and ready to be used, we supplied the values of the previous two real CPU-bursts. Following results were obtained.

FIRST PREDICTION:

Value of Previous two CPU bursts supplied as Input to the designed FIS : 6 and 4
Value of next CPU burst predicted by EAA : 6
Value of next CPU burst predicted by FBA: 6
Real CPU burst came to be : 6



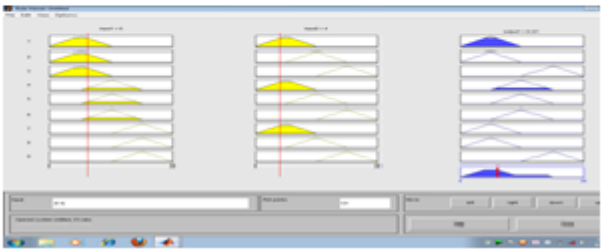
SECOND PREDICTION:

Value of Previous two CPU bursts supplied as Input to the designed FIS : 4 and 6
Value of next CPU burst predicted by EAA: 6
Value of next CPU burst predicted by FBA : 5
Real CPU burst came to be : 4

ISSN: 0971-1260
Vol-22-Issue-17-September-2019

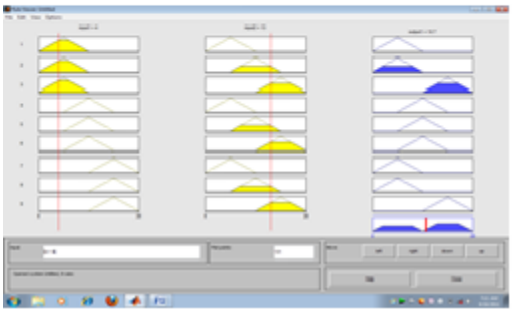
THIRD PREDICTION:

Value of Previous two CPU bursts supplied as Input to the designed FIS : 6 and 4
Value of next CPU burst predicted by EAA: 5
Value of next CPU burst predicted by FBA: 6
Real CPU burst came to be : 13



FOURTH PREDICTION:

Value of Previous two CPU bursts supplied as Input to th designed FIS : 4 and 13
Value of next CPU burst predicted by EAA : 9
Value of next CPU burst predicted by FBA : 11
Real CPU burst came to be : 13



FIFTH PREDICTION:

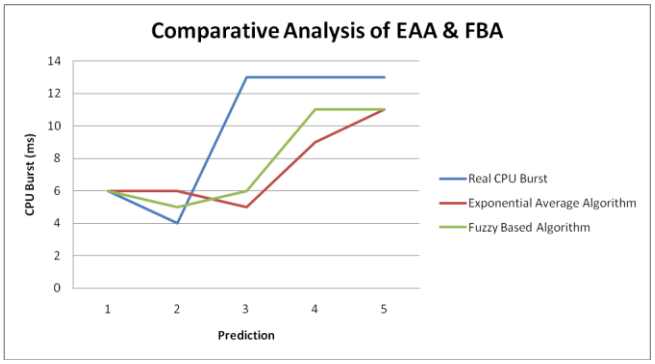
Value of Previous two CPU bursts supplied as Input to the designed FIS : 13 and 13
Value of next CPU burst predicted by Exponential Average Algorithm : 11
Value of next CPU burst predicted by Fuzzy Based Algorithm : 11
Real CPU burst came to be : 13

After implementing both the algorithms, we summarize the results of both the algorithms in Table 2. Rows in the table specify the number of prediction i.e first prediction or second prediction. Columns specify the values of real CPU burst and those specified by Exponential Average algorithm and Fuzzy based algorithm.

Table 2 : Results of EAA & FBA

Prediction	Real burst	EAA	FBA
1	6	6	6
2	4	6	5
3	13	5	6
4	13	9	11
5	13	11	11

It can be easily noticed that there is no such prediction in which the Exponential Average Algorithm has given better results than our fuzzy based algorithm. Each time the fuzzy based algorithm has proven its optimality. The Fuzzy-based algorithm has shown an overall improvement of 16% over the Exponential Average algorithm. Figure below performs a comparative analysis of both the approaches. It can be easily seen that the Fuzzy-based approach is more optimal when compared with Exponential Average Algorithm, as it is projecting much closer values to the real CPU burst as being projected by the EAA.



VI CONCLUSION

ISSN: 0971-1260
Vol-22-Issue-17-September-2019

One of the basic jobs that an operating systems kernel does is to schedule the jobs in order to get maximum CPU utilization. This process is known as CPU scheduling. Shortest Job First (SJF) CPU scheduling algorithm, which is one of the best CPU scheduling algorithms, always has some difficulties in real world implementation.

The real problem in implementing the SJF CPU scheduling algorithm in real environment is knowing the length of the next CPU-burst of the process. This is because when a process arrives in the method its execution time is unknown.Only when it has executed the exact CPU-burst would appear. Literature review explains that Exponential Average algorithm is one of the approach that is used to predict the next CPU-burst of the process.

In this paper a Fuzzy based approach to predict the next CPU-burst time of a process has been proposed. This approach is based on the Fuzzy Inference System (FIS), which uses a fuzzy inference engine to derive answers from the knowledge base. The input given to this system is the value of the previous two CPU-bursts of the process and the output is the predicted next CPU-burst time of that particular process.

After implementing EAA and the Fuzzy based approach, this thesis evaluated and compared the performance of both the algorithms. The simulation results clearly state that the fuzzy based algorithm is more optimal than the exponential average algorithm in that it predicts more closer values to the real CPU-burst. Results prove that the Fuzzy based algorithm has shown an improvement of 16% over the EAA.

As literature review states that, SJF is one of the best CPU scheduling algorithm but still it cannot be used in real world due to the problem of knowing the exact execution time of a process before it has actually executed. Therefore, using the proposed Fuzzy based approach, running a SJF scheduling algorithm which requires to know the upcoming CPU-burst time of tasks in a real computer system will come true.

VII FUTURE WORK

In this paper, only the non-preemptive version of the Shortest Job First (SJF) CPU scheduling algorithm is considered. So, future scope in this regard can be to consider the preemptive version of SJF, while doing the simulation. Further, the work carried out in this thesis assumes the value of CPU burst in the range of 0 to 20 ms. So, it would be interesting to see the results if this range is extended. Also, this thesis requires values of previous two CPU bursts to predict the new CPU burst. This can be a further scope to extend the thesis and see what effect does changing the number of inputs have on the results obtained.

Also the choice of membership function can play a major role in predicting more close results. In this paper, the triangular membership function is considered, so an idea can be to use some other membership function like trapezoidal or Gaussian, and then analyze the results. Another scope can be to use some other defuzzification methods like center of area (COA), center of gravity (COG), weighted average deffuzification techniques etc. It will be interesting to note the results and compare them with the results of this paper, which has used centroid method for defuzzification.

ISSN: 0971-1260
Vol-22-Issue-17-September-2019

REFERENCES

[1] Rajesh Kumar Garg, Vikram Singh, "Simulator for Performance Evaluation of Process Scheduling Policies For Embedded Real-Time OS", *International Journal of Computer Applications(0975-8887)*, Vol. 12, No. 3, Nov. 2010.

[2] S.N. Mehmood Shah, A.K.B. Mahmood, Alan Oxley, "Hybrid Scheduling And Dual Queue Scheduling", *International Conference on Computer Science and Information Technology*, ICCSIT, Aug. 2009, pp. 539-543.

[3] D.R. Smith, "A New Proof of the Optimality of the Shortest Remaining Processing Time Discipline", *Operations Research*, Vol. 26, 1976, pp. 197 – 199.

[4] L.E. Schrage, "A Proof of the Optimality of the Shortest Processing Remaining Time Discipline", *Operations Research*, Vol. 16, 1968, pp. 678-690.

[5] A. Silberschatz, P.B. Galvin, G. Gagne, "Operating System Concepts", Seventh edition, *John Willey & Sons*, 2005.

[6] Nikhil Bansal, Mor Harchol-Balter, "Analysis of SRPT Scheduling : Investigating Unfairness", *In Proceedings of 2001 ACM SIGMETRICS, International Conference on Measurement and Modeling of Computer Systems*, Vol. 21 Issue 1, Jun. 2001, pp. 279-290.

[7] L. Kleinrock, R.R Muntz, and J. Hsu, "Tight bounds on average response time for time-shared computer systems", *In Proceedings of IFIP congress*, Vol. 1, 1971, pp. 124 – 133.

[8] Leonard Kleinrock, "Queueing Systems", volume II, Computer Applications, *John Willey & Sons*, 1976.

[9] Lupetti, S. and D. Zagorodnov, "Data Popularity and Shortest-Job-First Scheduling of Network Transfers", *Proceeding of the International Conference on Digital Telecommunications*, ICDT, Aug. 2006, pp. 26.

[10] Sandmann. W, "Benefits of Alternating FCFS/SJF Service Order", *Proceedings of the 6th WSEAS International Conference on Applied Informatics and Communications*, World Scientific and Engineering Academy and Society, USA, Aug. 2006, pp. 194-199.

[11] Chi-Hong Hwang, Allen C.-H. Wu, "A Predictive System Shutdown Method for Energy Saving of Event-Driven Computation", *IEEE/ACM International Conference of Computer-Aided Design*, Nov. 1997, pp. 28–32.

[12] Abdolghader Pourali, Amir Masoud Rahmani, "A Fuzzy-Based Scheduling Algorithm for Prediction of Next CPU-Burst Time to Implement Shortest Process Next", *International Association of Computer Science and Information Technology - Spring Conference*, IACSIT-SC, Apr. 2009, pp. 217.

[13] Bashir Alam, M.N. Doja, R.Biswas, "Finding Time Quantum of Round Robin CPU Scheduling Algorithm Using Fuzzy Logic", *International Conference on Computer and Electrical Engineering*, ICCEE, Dec. 2008, pp. 795-798.

[14] Zadeh, L.A., "Fuzzy Sets", *Information and Control*, Vol. 1.8, 1965, pp. 338-353.

[15] M. Sugeno, "Industrial Applications of Fuzzy Control", Elsevier Science Pub. Co., 1985.

[16] Wang Lie-Xin, "A Course in Fuzzy Systems and Control", *Prentice Hall*, August 1996.