International Journal of Computer Sciences and Engineering
Vol.**11**, Issue.**10**, pp.**19-28**, October **2023**
ISSN: 2347-2693 (Online)
Available online at: www.ijcseonline.org

ISSN: 2347-2693 (E)

Research Paper

Neuro Fuzzy Xavier Technique for optimization of Time Quantum in Scheduling Algorithm

Rajeev Sharma[1], Atul Kumar Goel[2], M.K. Sharma[3*]

[1]Dept. of Computer Science, IIMT Engineering College, Meerut, India
[2]Dept. of Mathematics, A.S. (PG) College, Mawana, Meerut, India
[3]Dept. of Mathematics, Ch. Charan Singh University, Meerut-250004, India

**Corresponding Author: drmukeshsharma@gmail.com*

Received: 03/Sept/2023; **Accepted**: 06/Oct/2023; **Published**: 31/Oct/2023. **DOI**: https://doi.org/10.26438/ijcse/v11i10.1928

Abstract: We drop-shipped a novel Round Robin Neuro-Fuzzy System (RRNFS) model for the decision makers, based on neuro-fuzzy system for the processing of scheduling in a batch operating system. TS fuzzy model (Takagi & Sugeno, 1985) implemented in the RRNFS proposed model to identify the ideal Time Quantum. Our proposed RRNFS model takes two inputs: the total number of processes and the average burst time (ABT) of each process that is presented in the ready queue, fuzzifying the input values, activate the necessary rules of the proposed neuro fuzzy controller, and then determines the best Time Quantum for each process in the ready queue. Once Time quantum is calculated, every process will run on central processing unit as per the allocated Time quantum reduces the context switching, turnaround and waiting time. We bespoke the performance of the proposed model over date sets and compared our results with the classical round robin policy and modified round robin using fuzzy logic scheduling algorithms. We developed the neuro fuzzy technic for Xavier Normal function to minimize the error of the proposed model with the targeted time quantum.

Keywords: Neuro-fuzzy system (NFS), Time Quantum (TQ), Round robin (RR) scheduling, Ready queue (RQ), Xavier Normal function.

1. Introduction

Multitasking [1] is an analytical extension of multiprogramming [2]. In multitasking, several processes usually competed for the CPU in the similar time. This state arises when more than two process are running concurrently in the ready state. The operating system component that makes the decision is known as the scheduler, and the algorithm it adopts is known as the scheduling algorithm [3]. There are many situations when scheduling algorithms are needed. First, when a process exists. Second, when a process blocks on input/output or semaphore. Third, when an I/ O interruption occurs. Fourth, when a clock interruption occurs. The main job of the scheduling algorithms is to keeps up the CPU busy at all times and fair allocation of processor time to every process. Whenever CPU is ideal, it selects the process in RQ with the help of Short Term schedular [4]. The Short Term Schedular select the process from RQ and allocate to CPU burst time, so switching of the process from running state to RQ or from waiting state to RQ are called Preemptive Scheduling. Contrarily, termination of a process or switching from RQ to waiting state is called Non-Preemptive Scheduling [5]. Resources are allotted to a process in preemptive scheduling for a short period of time. We can state that it is priority-based scheduling since low priority processes may starve if a high priority process frequently enters in the ready queue [6]. Under Non-Preemptive Scheduling, A process retains the CPU once it is assigned to it until its termination its burst period or by moving to the waiting state. It is not flexible in nature and not expensive.

Numerous algorithms are available for deciding the allocation of process to CPU from ready queue. Different CPU scheduling algorithms have different properties which helps to selection of algorithm in particular situations. Some of important algorithms are: First-come first-served (FCFS); allocation of CPU to processes as per their arrival in ready queue [7]. Implementation of this policy is managed by FIFO (first in first out) queue. Shortest-Job-First scheduling (SJF); this policy is based on execution time of process on CPU [8]. Minimum execution time process will run first other than highest execution time process. It is also called shortest-next-CPU-burst-algorithm. Priority scheduling algorithm; In this approach, each process has a priority, and only the runnable processes with the highest priority are permitted to run. Processes with equal priority are scheduled in FCFS order. Multilevel queue scheduling; It is appropriate for processes that are categorized into various forms [9]. This approach splits the ready queue into a number of queues according to the different properties of processes like process type,

memory size and process priority. Highest Response Ratio Next the scheduling technique, the process with the highest response ratio will be scheduled next [10]. Round Robin scheduling is the best suitable for time sharing systems, in which time quantum (TQ) play a significant role for the CPU allocation time to process [11]. Processes are selected from the ready queue (RQ) and assigned to the CPU. Ready queue works as a circular queue in this approach. With round-robin scheduling, each task shares an equal amount of CPU time. Alam B. et al. designed a FIS to select the value of TQ by using two inputs (number of process and ABT) and one output (Time Quantum) [12]. Aburas A. A. et al. reduced the jobs' average lateness with respect to the internal deadline in uniprocessor scheduling by using the fuzzy logic [13]. Preemption of the process was the main issue with RR algorithm, even if the process needs few amounts of TQ to complete its execution so Alam B. gave the solution of this issue by using the FIS for Preemption (FISRR) for process [14]. Datta L. developed a modified RR algorithm using fuzzy logic for employment in real time and embedded systems based on each process's externally determined priority, comparative remaining CPU burst time, and comparative waiting time, a new priority is given to it [15]. Lim S. et al. planned an intelligent CPU process scheduling algorithm using FIS [16]. This study divided processes into batch, interactive, and real-time and used the FIS to determine the priority. Granam B. et al. employed the burst time and static priority accessible to the CPU scheduler, which dynamically classifies the priority of processes, to feed a FIS [17]. Kalas and Deshpande studied the sepsis detection in newborn infants using fuzzy inference system [18]. The results showed that the average waiting time for the fuzzy-priority scheduler was less than that for the priority and round-robin schedulers. This study is proposed by Atique M. et al., in which the use of ANFIS as a multimedia operating system's support for the execution of both conventional and multimedia programmes was presented [19]. Based on prior decisions, this scheduler makes decisions. By enhancing throughput, reducing waiting times, and reducing turnaround times for a process, Trivedi J. A. et al. improved the efficiency of RR scheduling [20]. This is accomplished by combining neuro-fuzzy approach with already used scheduling techniques. Nagargoje and Baviskar discussed opportunities and challenges in handling big data analytics in uncertain environment [21]. Benhammadi F. et al. investigate the use of neuro-fuzzy and Bayesian reasoning to estimate CPU load [22]. Sharma R. et al. proposed a modified RR algorithm [23]. In which fuzzy rules are used to discover the value of TQ but execution of process from RQ to CPU is based on: after run the one cycle as per RR method, arranged the processes in ascending order with their remaining execution time and select the process with very short remaining execution time to run compare to others.

The length of TQ is the most interesting issue with RR. Some other issues which are also faced by this are as follow:

(1) It takes a certain amount of time to complete the administrative tasks involved in switching from one process to another process. It is also term as context switching.
(2) It does not give higher priority for the urgent tasks.
(3) In this approach, determining the correct time quantum is a quite challenging task.
(4) For processes with longer burst times, it causes the starvation because they must repeatedly complete the cycle.
(5) RR degrades to FCFS if the TQ is excessively large.
(6) This method spends more time on context switching.

In (RRNFS) using the Xavier normal function to improve response time, average waiting time and average turnaround time [24]. The major problem in RR is selection of TQ. Here, neuro-fuzzy technique is used to estimate the predicted optimal value of TQ. Neuro-fuzzy system is used the Takagi & Sugeno approach to find the TQ [25]. Fuzzy logic controller is also developed to represent the structure using a Neural Network and this network is trained using the Back propagation algorithm. Xavier Normal function is also defined to reduce the error of RRNFS.

The following aspects shed light on the uniqueness of the work done in this;

(1) A RRNFS model is designed to calculate the value of TQ which reduces the context switch, waiting time and turnaround time.
(2) Xavier normal function is used to calculate the optimum value of weights.
(3) We designed a data set using the fuzzy inference system (FIS) to train the Neural Network structure.
(4) In this proposed work, we gave the mathematical formulation of RRNFS model.
(5) Numerical examples are used to show the applicability of planned policy.
(6) We provided a comparison of the results of proposed and the existing policies.

The present research work is categorized into eight segments. In the first segment, we gave the primary introduction, limitations of traditional RR and the uniqueness of proposed model. We gave some basic terminology of proposed model in second segment. In third segment, we described the RRNFS model with its data flow diagram. In the fourth segment, we defined the mathematical formulation and error calculation of proposed model. Fifth segment illustrate the data collection method for sample data. In sixth section, we described the numerical computations and Comparison of proposed with Existing Algorithms. Comparison the outcome of proposed model with Existing Algorithms mentioned in seventh segment. Conclusion of entire work and future work of the proposed model given in eighth segment.

2. Basic terminology

2.1 Artificial Neural Networks

Artificial neural networks are also referred to concept of artificial intelligence with neural network [26]. ANN is an artificial intelligence technique that trains the computers to analyse data similarly with the human brain functioning. It is multi-layered structure network, having an input layer, one or

more hidden layers, and an output layer. Every neuron or node connected with others neurons that's have an associated weights and bias value [27]. To learn and gradually increase its accuracy, it relies on training data. Consider every node to be a separate linear regression model, composed with input data, weights, a bias (or threshold), and an output. The equation (1) would resemble approximately like this:

$$\sum w_i \alpha_i + bias = w_1\alpha_1 + w_2\alpha_2 + w_3\alpha_3 + bias \quad (1)$$

$$\text{output} = f(\alpha) = 1 \text{ if } \sum w_1\alpha_1 + b \geq 0; 0 if \sum w_1\alpha_1 + b < 0$$

Artificial neural networks can be categorized as (1) Feedforward neural networks (2) Convolutional neural networks (3) Recurrent neural networks. Figure 1. Shown the architecture of ANN.

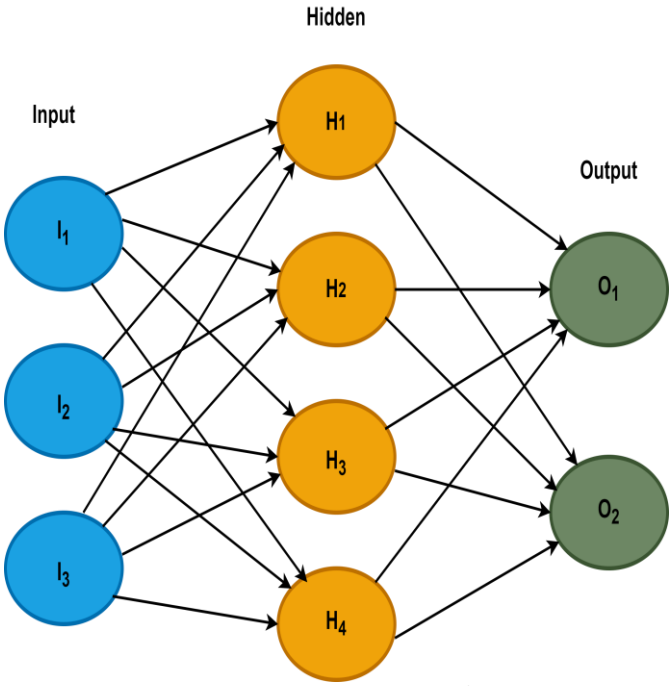

Figure 1. Architecture of ANN

2.2 Neural Fuzzy System

A fuzzy system that learns its parameters (fuzzy sets and fuzzy rules) by processing data samples using a learning algorithm based on or inspired by neural network theory is known as a neuro-fuzzy system shown in figure 2. Neural networks are excellent at pattern identification, whereas Fuzzy logic handles imperfect or incomplete data well so it can also deal with both structured and unstructured data. A NFS can be seen of as a three-layers feedforward neural network [28], where the first layer represents input variables, the middle layer, which is hidden, represents fuzzy rules, and the third layer represents output variables. (Fuzzy) connection weights are used to encode fuzzy sets. The knowledge utilised by the system, having the ability to learn, takes the form of IF-THEN fuzzy rules. The predefining rules allow the system to learn more quickly.

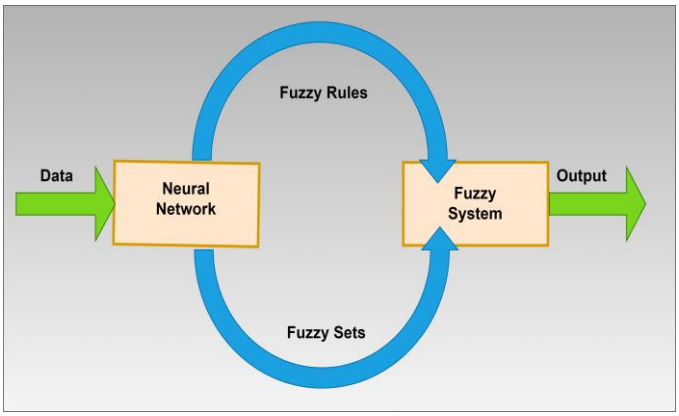

Figure 2. Architecture of Neural Fuzzy System

Batch mode of training, Back Propagation and Genetic algorithm are used to train the Neuro-fuzzy system.

2.3 ANFIS (Adaptive Neural Fuzzy Inference System)

The Takagi and Sugeno approach-based neuro fuzzy system is often referred to as ANFIS [29]. It uses a hybrid learning algorithm. With a rule R_n represented as:

$$R_n = IF \mu_{S_i}(x) AND \mu_{T_i}(y) THEN\ f = P_n x + Q_n y + C_n \quad (2)$$

Here n is total number of rules. Equation (2) if-then conditions for n rules. S_i and T_i are n fuzzy membership functions which is symbolized by μ in the antecedent part of the rule R_n and P_n, Q_n, C_n are the linear parameters of consequent part of the n^{th} rule.

2.4 Xavier Normal function

The performance of neural network played a vital role in AI and deep learning etc. so to reach the optimal performance levels, it needs a large amount of training. Wights are used to reached at optimal performance levels and training of neural network so initialization of weight value is crucial decision because weights should not be very large or very small and same. Weights should have good variance. Many techniques are used to initialization of weights, Xavier Normal function is one of them. It is often referred to as the "Xavier-Glorot initialization" or the "Glorot normal initialization." Xavier Glorot and Yoshua Bengio presented the Xavier normal function in 2010 [24]. A layer's weights are initialised by the Xavier normal function by selecting them from a Gaussian distribution with a zero mean and a variance that is based on the layer's number of input and output units. The variance is computed specifically as follows in equation (2):

$$\sigma = \sqrt{\frac{2}{fan_{in} + fan_{out}}} \quad (3)$$

Where, σ represents the variance, fan_{in} denotes the input on neurons and fan_{out} indicates the output from neurons.

In short, Xavier technique is very effective in training of neural network and widely used in practice.

3. Proposed model to find the optimal time quantum

We have proposed a Round Robin Neuro-Fuzzy System (RRNFS) model to find the optimal Time Quantum using the Neuro fuzzy expert system. NFS has a multi-layered structure. The RRNFS model used the concept of Neuro fuzzy system, based on Takagi & Sugeno approach [30]. In proposed model has multi-layer structure; layer one used two inputs as (a) number of process (b) average burst time of all process, layer two fuzzified the inputs with two membership functions (for input one we have consider three linguistic term as low, medium, high and for the second input which is ABT consider three linguistic term as small, large, very large so nine possible combinations of the input parameters), layer three calculate the firing strength of rules, layer four normalized the weight value, layer five computed the output ($y_i = a_iI_1+b_iI_2+c_i$) of each neuron; we used the Xavier Normal technique to find the range for a_i, b_i and c_i in layer five, layer six compute the final output of the model.

3.1 Data flow diagram of proposed model

Figure 3. represented the flow of data in proposed policy.

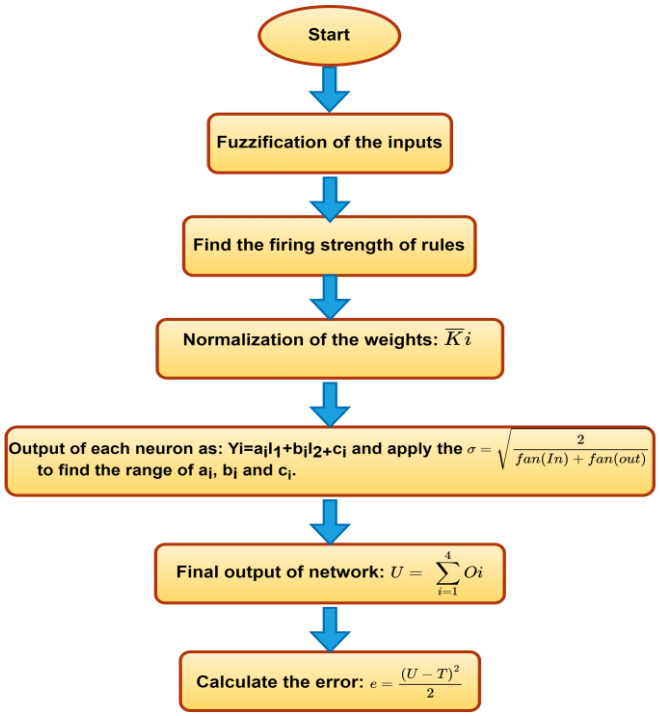

Figure 3. Flow diagram for proposed model

4. Mathematical formulation of RRNFS

Let α_1 and α_2 be the two inputs and M_i be the result for each of the values of i=1, 2, 3, 4, 5, 6, 7, 8 and 9. Let i^{th} IF-THEN rule of a RRNFS can be stated as;

R_i: IF α_1 is I_1 and α_2 is I_2 then M_i is $y_i = a_iI_1 + b_iI_2 + c_i$ for i=1, 2, 3, 4, 5, 6, 7, 8 and 9. Where I_1 and I_2 are alienated in to two groups which is shown in figure 5.

Layer 1(Input layer): Layer 1 is referred to as the input layer, we used a non-linear system that is a composite of numerous linear or non-linear systems. Here we taken two inputs with two membership function, which is shown in fig for the inputs.

Layer 2 (Fuzzification): Layer 2 is referred to as a layer of fuzzification [28] in which neurones are received an input and these inputs than fuzzify this layer with some degree of membership [13, 29]. Layer 1 and layer 2 are connected with some connecting weights. The connecting weights are lies between 0 and 1 so this connecting weight expressed in normalized scale may having some other value in the real scale and that particular value in real scale is going to represent, that all the particular triangular membership function distribution. This layer will work as input layer of next layer which is layer 3 after the process of fuzzification. Simply put, these are the inputs' membership values.

We used Triangular membership function in our work, Membership function for N number of processes and average burst time (ABT) inputs are shown in figure 4.

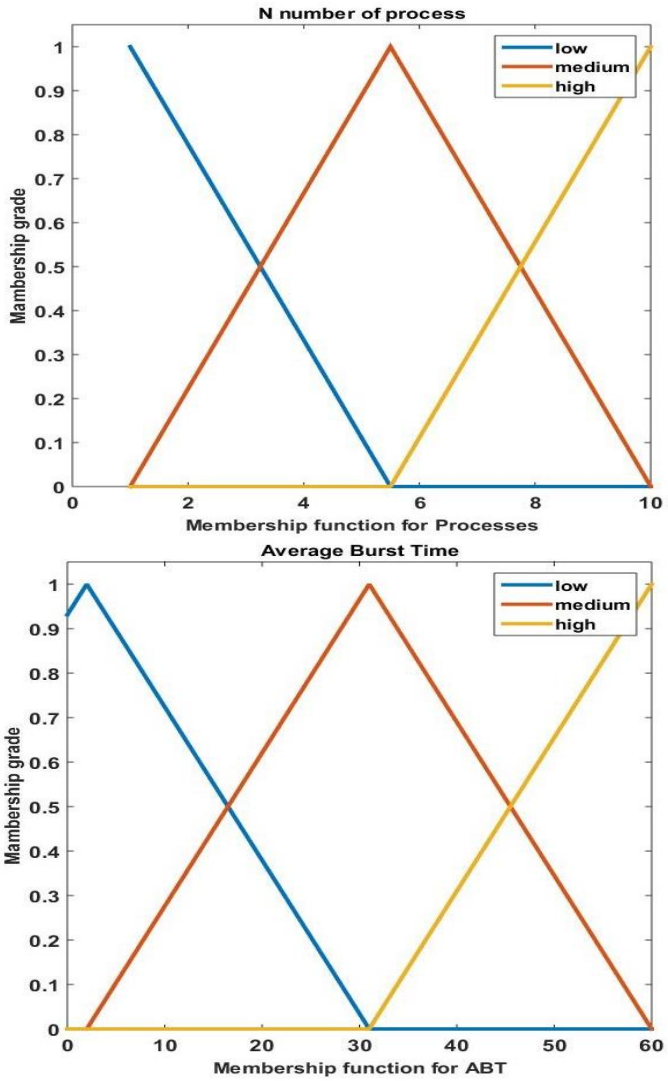

Figure 4. linguistic categorization of triangular membership function for inputs

For N number of processes:

$$P_{LW}(\gamma)=\begin{cases}\dfrac{\gamma+3.5}{4.5} if\ -3.5\le\gamma\le1\\ \dfrac{5.5-\gamma}{4.5} if\ 1\le\gamma\le5.5\\ 0 if\ \gamma\le-3.5 and\ x\ge5.5\end{cases}$$

$$P_{M}(\gamma)=\begin{cases}\dfrac{\gamma-1}{4.5} if\ 1\le\gamma\le5.5\\ \dfrac{10-\gamma}{4.5} if\ 5.5\le\gamma\le10\\ 0 if\ \gamma\le5.5 and\ x\ge10\end{cases}$$

$$P_{H}(\gamma)=\begin{cases}\dfrac{\gamma-5.5}{4.5} if\ 5.5\le\gamma\le10\\ \dfrac{15-\gamma}{5} if\ 10\le\gamma\le15\\ 0 if\ \gamma\le5.5 and\ x\ge15\end{cases}$$

For ABT:

$$P_{SM}(\gamma)=\begin{cases}\dfrac{\gamma+27}{29} if\ -27\le\gamma\le2\\ \dfrac{31-\gamma}{29} if\ 2\le\gamma\le31\\ 0 if\ \gamma\le-27 and\ x\ge31\end{cases}$$

$$P_{LG}(\gamma)=\begin{cases}\dfrac{\gamma-2}{29} if\ 2\le\gamma\le31\\ \dfrac{60-\gamma}{29} if\ 31\le\gamma\le60\\ 0 if\ \gamma\le2 and\ x\ge60\end{cases}$$

$$P_{VL}(\gamma)=\begin{cases}\dfrac{\gamma-31}{29} if\ 31\le\gamma\le60\\ \dfrac{90-\gamma}{30} if\ 60\le\gamma\le90\\ 0 if\ \gamma\le31 and\ x\ge90\end{cases}$$

Layer 3 (AND operation layer): In third layer we got the nine possible combinations of the input parameters. Now we will work for the implementation the neurons lying in this layer.

For example: 3_1 and 3_5 are the inputs of first neuron lying on third layer so inputs are nothing but can say that μ_{lw} and μ_{sm} will be used as actually the inputs so these two inputs values will be multiplied just to find the firing strength or the output of this layer $x_1= \mu_{31}* \mu_{35}$ so by following this principal we can find the firing strength of all neurons;

$$x_2= \mu_{32}* \mu_{37}$$
$$x_3= \mu_{33}* \mu_{36}$$
$$x_4= \mu_{34}* \mu_{38}$$

so, the values of firing strength will be lying between 0 and 1. Figure 5.0 shown the working of proposed RRNFS model.

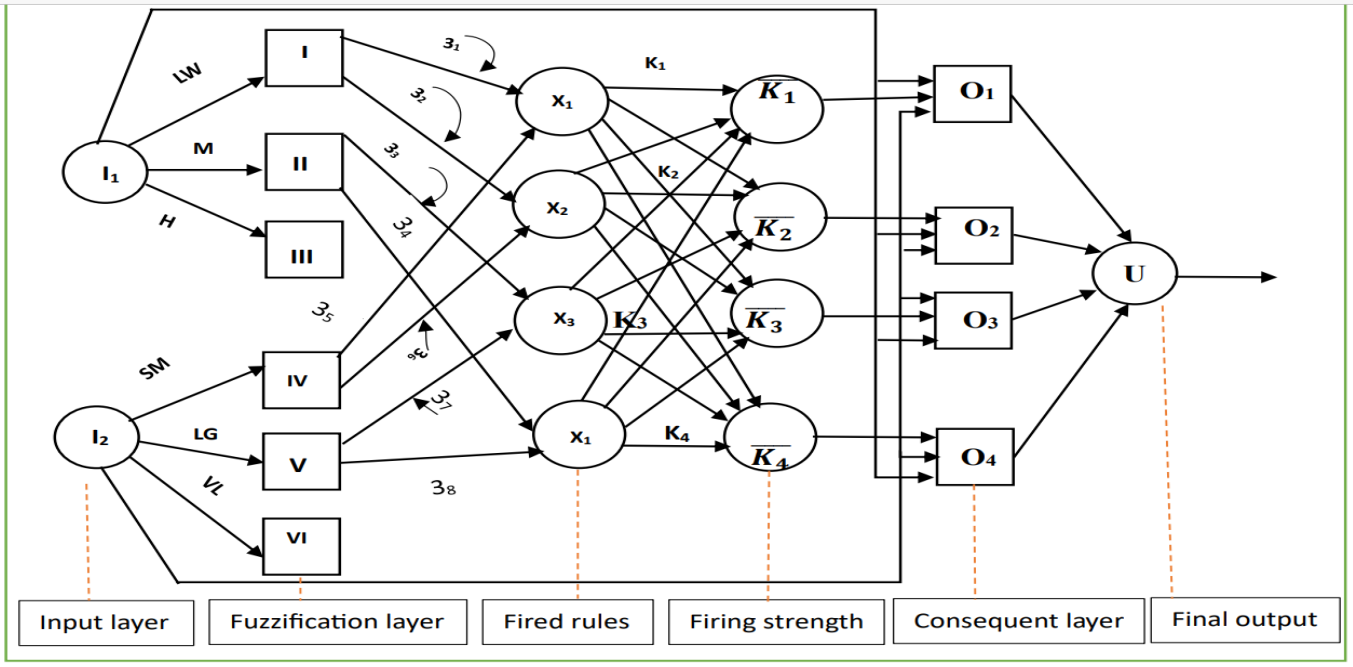

Figure 5. Proposed structure of RRNFS model

Layer 4 (Normalized the weights): This layer normalized the weight value which is received from the layer 3. The normalised firing strength of a rule is determined by each node in this layer and is designated by $\overline{K_1}$, $\overline{K_2}$, $\overline{K_3}$ *and* $\overline{K_4}$. That is, the j^{th} node determines the following formula to express the firing strength [29] of the j^{th} rule in relation to all other rules:

$$\overline{\mathbf{K}_1} : \frac{\mu 31^* \mu 35}{\mu 31^* \mu 35 + \mu 32^* \mu 37 + \mu 33^* \mu 36 + \mu 34^* \mu 38}$$

$$\overline{\mathbf{K}_2} : \frac{\mu 32^* \mu 37}{\mu 31^* \mu 35 + \mu 32^* \mu 37 + \mu 33^* \mu 36 + \mu 34^* \mu 38}$$

$$\overline{\mathbf{K}_3} : \frac{\mu 33^* \mu 36}{\mu 31^* \mu 35 + \mu 32^* \mu 37 + \mu 33^* \mu 36 + \mu 34^* \mu 38}$$

$$\overline{\mathbf{K}_4} : \frac{\mu 34^* \mu 38}{\mu 31^* \mu 35 + \mu 32^* \mu 37 + \mu 33^* \mu 36 + \mu 34^* \mu 38}$$

Layer 5 (Output of each neuron): In fifth layer we will find out, what should be the output of each of these particular rules? Equation (4) play pivot role in to find the optimum output. The output is nothing but $y_i = a_i I_1 + b_i I_2 + c_i$ (4)

Once you find the value of a_i, b_i and c_i then the output can also be calculated easily so use the Xavier Normal technique to find the range of values for a_i and b_i:

$$\text{Wij~N (0, σ) where } \sigma = \sqrt{\frac{2}{fan_{in} + fan_{out}}} ;$$

$$fan_{in} = No.of\ input\ on\ neuron,$$

$$fan_{out} = No.of\ output\ from\ neuron$$

After finding the value of y_i, now calculate the output for i^{th} neurons of fifth layer:

$O_i = \overline{K_i} * Y_i$ for i=1,2,3……..., where K_i is the output of layer 4 and Y_i is the the consequent parameter set.

So, for the RRNFS model the output will be-

$$O_1 = \overline{K_1} * Y_1,$$

$$O_2 = \overline{K_2} * Y_2,$$

$$O_3 = \overline{K_3} * Y_3,$$

$$O_4 = \overline{K_3} * Y_3$$

Layer 6(Overall output): This layer used for find the final output of the network for one set of input parameters. The sixth layer's function is to determine the net membership grade U that represented the value of TQ, and this layer will be used to finish the subsequent part of the fuzzy rule which is as follow:

$$U = \overline{K_1} * y_1 + \overline{K_2} * y_2 + \overline{K_3} * y_3 + \overline{K_4} * y_4$$

or

$$U = \sum_i^4 o_i , \quad \text{for i=1, 2, 3, 4.}$$

4.1 Error calculation in the RRNFS model

We used a method to calculate the error of received data from layer six in equation (5);

$$e = \frac{(U - T)^2}{2} \quad (5)$$

Where, U is the final output that is represent the optimal value of TQ and T is the targeted value, to reduce the error of proposed model, we have to update the parameter

Table 1. sample data set for train the model

Number of Processes(N)	ABT	TQ		Number of Processes(N)	ABT	TQ
1	5	2.5		3	10	2.5
3	52	6.94		3	12	2.5
4	40	4.5		3	17	2.76
3	27	4.5		9	20	4.5
10	2	4.5		3	38	5.98
2	10	2.5		7	10	3.39
7	20	4.5		3	40	6.64
2	20	4.5		3	45	6.78
9	30	4.5		3	49	6.87
2	47	6.83		1	10	2.5
10	20	4.5		3	57	6.97
2	55	6.85		8	20	4.5
2	60	6.85		4	10	2.5
4	12	2.5		6	38	5.98
7	35	4.5		5	59	6.97
4	18	3.81		6	50	2.5
4	20	4.5		6	20	4.5
5	35	6.7		1	10	4.5
7	60	5.73		6	30	4.5
4	40	6.64		6	40	6.64
7	48	5.73		4	37	4.98
9	15	4.5		6	48	6.85
4	57	7.01		3	32	4.5
4	59	7.02		8	55	4.5
6	45	6.7		7	15	3.47
5	7	2.5		7	18	3.81
5	12	2.5		4	45	6.78
8	60	4.5		7	30	4.5
5	25	4.5		5	17	2.76
4	17	2.76		7	38	5.55
5	40	6.85		7	40	5.58
2	15	2.5		4	25	4.5
5	59	6.97		6	37	4.98
8	10	4.5		8	57	4.5
8	15	4.5		4	5	2.5
2	43	6.72		9	60	4.5
8	25	4.5		9	10	4.5
8	30	4.5		4	50	6.9
5	48	6.97		2	5	2.5
8	50	4.5		9	25	4.5
6	55	6.85		6	17	2.76
1	40	6.64		9	35	4.5
3	39	6.61		2	50	6.85
10	30	4.5		1	15	2.5

$y_i = a_i I_1 + b_i I_2 + c_i$ i.e., a_i, b_i and c_i for each i=1, 2, 3, 4.

5. Data Collection Method

In this paper, we prepared the data set to train the model. We designed the FIS using MATLAB to prepare the sample data set. Two inputs like; N number of processes and average burst time assigned to FIS for calculate the value of TQ. The value of TQ is influenced by both input parameters. Different inputs created the different value of TQ. The sample data set is described in table 1. We used the more than 120 different inputs for the formulation of small sample data set. In this, we used the minimum number of processes; 1, maximum number of processes;10 and minimum ABT; 2, maximum ABT; 60 for generating the value of TQ. However, the limit of number of processes and ABT can increased to generate the large sample data. Sample data help to train the model using the neuro fuzz designer in MATLAB to find the optimum value of TQ.

6. Numerical Computations

Layer1: No. of process (N)= 7 and average burst time (ABT) = 30 are the inputs for Time Quantum.

Layer 2: After extracting the input values from the given data, we must fuzzified them in order to perform further calculations.
So, the membership values of given inputs are as follows:
For No. of process: μ_{lw} = 0.33 and μ_m = 0.67

For ABT: μ_{lw} = 0.034 and μ_{sm} = 0.97
Only two rules, out of all available rules, correspond to these two inputs, will be applied in this situation. The fired rules are as follow:

(1) If No. of process is low (LW) and ABT is large (LG) then $y_1 = a_1 I_1 + b_1 I_2 + c_1$.
(2) If No. of process is low (LW) and ABT is small (SM) then $y_2 = a_2 I_1 + b_2 I_2 + c_2$.
(3) If No. of process is medium (M) and ABT is large (LG) then $y_3 = a_3 I_1 + b_3 I_2 + c_3$.
(4) If No. of process is medium (M) and ABT is small (SM) then $y_4 = a_4 I_1 + b_4 I_2 + c_4$.

Layer 3: The weights or the strength of the rules can be determined as follows:

x_1: (μ_{31}* μ_{35}) 0.33*0.034= 0.01122
x_2: (μ_{32}* μ_{37}) 0.33*0.97= 0.34
x_3: (μ_{33}* μ_{36}) 0.67*0.034= 0.023
x_4: (μ_{34}* μ_{38}) 0.67*0.97= 0.65

Layer 4: Calculate the normalised firing strength of $\overline{K_i}$ for i=1, 2, 3, and 4 is as follow:

$\overline{K_1} = 0.0112$, $\overline{K_2} = 0.332$, $\overline{K_3} = 0.0225$ *and* $\overline{K_4}$ = 0.635.

Layer 5: The different neuronal outputs of this layer can be obtained as:
(We used Xavier Normal function to find the range of values for a_i and b_i which is: 0 to 1 and the bias value= 1 for all c_i)

O_1: $\overline{K_1}$ * $a_1 I_1 + b_1 I_2 + c_1$ = 0.0112 *(0.01*7+0.1*30+0.1) = 3.17

O_2: $\overline{K_2}$ * $a_2 I_1 + b_2 I_2 + c_2$ = 0.332 *(0.2*7+0.1*30+0.2) = 4.6

O_3: $\overline{K_3}$ * $a_3 I_1 + b_3 I_2 + c_3$ = 0.0225*(0.05*7+0.1*30+0.3) = 3.6

O_4: $\overline{K_4}$ * $a_4 I_1 + b_4 I_2 + c_4$ = 0.635*(0.2*7+.11*30+0.4) = 5.1

Layer 6: calculate the final output of the network for one set of input parameters as:

$$U = \sum_i^4 o_i, \quad \text{for i=1, 2, 3, 4.}$$

So, U = 4.9

Now the find the value of optimal TQ = 4.9

For error calculation use the e= $\frac{(U-T)^2}{2} = \frac{(4.9-4.5)^2}{2}$ = 0.02

7. Comparison of the results

The given TQ was 45 but when we implemented the RRNFS model, it provides the optimal value of TQ as 49. The response time, turnaround time, and waiting time in table 2. determined using the optimal TQ=49. The result implementation of proposed model is represented in the table 2. and the comparison result of RRNFS model with RR and Modified RR using fuzzy logic is demonstrated in Table 3 To organize, manage, and track tasks in a system, a Gantt chart offers a graphical representation of process scheduling. The Gantt chart illustrates how the processes are executed successfully. Chart 1. represented the performance evaluation of proposed model with existing policies.

Table 2. Outcome implementation of proposed model

Process	Arrival time	Burst Time	Response Time	Turnaround Time	Waiting Time
G1	0	12	0	12	0
G2	1	47	12	58	11
G3	2	19	57	76	57
G4	3	40	75	115	75
G5	4	16	114	130	114
G6	5	30	129	159	129
G7	6	46	158	204	158

Gantt Chart of proposed problem

G1	G2	G3	G4	G5	G6	G7
0	12	59	78	118	134	164	210

Average Response Time (ART) : 77.857
Average Turnaround Time (ATAT): 107.714
Average waiting Time (AWT): 77.7143

Table 3. Compression of result of RRNFS with existing algorithms using first experimental data

Scheduling approaches	Round Robin	Modified RR using Fuzzy logic	RRNFS
Average Response Time	61.28	76.571	**77.857**
Average Turnaround Time	126.28	128	**107.714**
Average Waiting Time	96.286	98	**77.714**

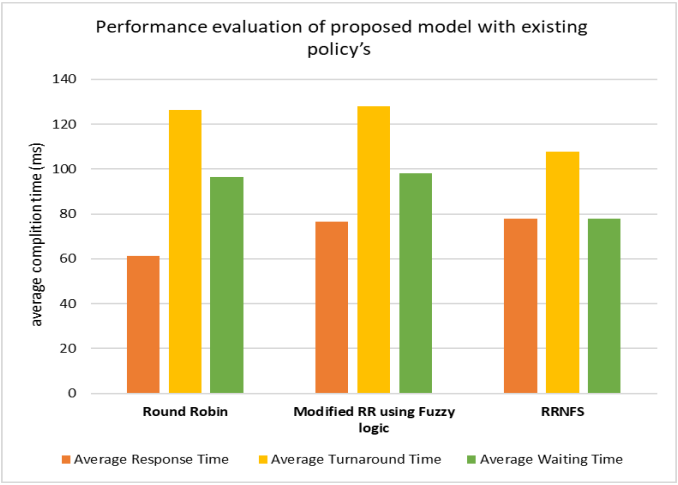

Chart 1: Performance evaluation of proposed policy with existing policy's

7.1 Optimum TQ value on different inputs with error

Table 4. TQ value with different inputs

Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6	Targeted value	error
N=3 ABT=27	μ_M=0.56 μ_{lw}=0.45 μ_{SM}=0.14 μ_{LG}=0.86	x_1=0.074 x_2=0.486 x_3=0.063 x_4=0.387	$\overline{K_1}$ =0.0776 $\overline{K_2}$ =0.4768 $\overline{K_3}$ =0.063 $\overline{K_4}$ =0.383	$\overline{K_1}$ *(0.2*3+0.3*27+0.1) =8.8 $\overline{K_2}$ *(0.1*3+0.2*27+0.2) =5.9 $\overline{K_3}$ *(0.3*3+0.2*27+0.3) =7.1 $\overline{K_4}$ *(0.25*3+0.56*27+0.4) =3.84	U=5.41	T=4.5	0.417
N=4 ABT=25	μ_M=0.33 μ_{lw}=0.67 μ_{SM}=0.21 μ_{LG} = 0.8	x_1=0.0693 x_2=0.264 x_3=0.1407 x_4=0.536	$\overline{K_1}$ =0.069 $\overline{K_2}$ =0.261 $\overline{K_3}$ =0.14 $\overline{K_4}$ = 0.530	$\overline{K_1}$ *(0.21*4+0.12*25+0.3) =4.14 $\overline{K_2}$ *(0.15*4+0.13*25+0.1) =3.95 $\overline{K_3}$ *(0.24*4+0.3*25+0.25) =8.71 $\overline{K_4}$ *(0.31*4+0.1*25+0.13) =3.87	U=4.63	T=4.5	0.008
N=5 ABT=48	μ_M=0.11 μ_{lw}=0.89 μ_{VL}=0.59 μ_{LG}=0.41	x_1=0.065 x_2=0.046 x_3=0.53 x_4=0.37	$\overline{K_1}$ =0.064 $\overline{K_2}$ =0.045 $\overline{K_3}$ =0.52 $\overline{K_4}$ = 0.37	$\overline{K_1}$ *(0.11*5+0.1*48+0.1) =10.2 $\overline{K_2}$ *(0.12*5+0.1*48+0.2) =5.6 $\overline{K_3}$ *(0.2*5+0.05*48+0.3) =3.7 $\overline{K_4}$ *(0.25*5+0.08*48+0.4) =9.81	U=6.46	T=6.97	0.130
N=6 ABT=17	μ_H=0.11 μ_M=0.89 μ_{SM}=0.52 μ_{LG}=0.48	x_1=0.0572 x_2=0.0528 x_3=0.4628 x_4=0.4272	$\overline{K_1}$ =0.0572 $\overline{K_2}$ =0.0528 $\overline{K_3}$ =0.4628 $\overline{K_4}$ = 0.4272	$\overline{K_1}$ *(0.1*6+0.05*17+0.1) =1.55 $\overline{K_2}$ *(0.01*6+0.1*17+0.1) =1.86 $\overline{K_3}$ *(0.1*6+0.07*17+0.1) =1.89 $\overline{K_4}$ *(0.2*6+0.1*17+0.1) =3	U=2.34	T=2.76	0.087
N=8 ABT=10	μ_M=0.56 μ_{lw}=0.44 μ_{LG}=0.724 μ_{SM}=0.28	x_1=0.41 x_2=0.16 x_3=0.32 x_4=0.12	$\overline{K_1}$ =0.41 $\overline{K_2}$ =0.16 $\overline{K_3}$ =0.32 $\overline{K_4}$ = 0.12	$\overline{K_1}$ *(0.1*8+0.2*10+0.2) =3 $\overline{K_2}$ *(0.2*8+0.15*10+0.1) =3.2 $\overline{K_3}$ *(0.3*8+0.2*10+0.16) =4.56 $\overline{K_4}$ *(0.15*8+0.3*10+0.25) =4.45	U=3.76	T=4.5	0.292

N=9 ABT=35	μ_M=0.22 μ_H=0.78 μ_{VL}=0.14 μ_{LG}=0.86	x_1=0.0308 x_2=0.19 x_3=0.11 x_4=0.67	$\overline{K_1}$ =0.031 $\overline{K_2}$ =0.19 $\overline{K_3}$ =0.11 $\overline{K_4}$ = 0.67	$\overline{K_1}$ *(0.1*9+0.02*35+0.1) =1.7 $\overline{K_2}$ *(0.2*9+0.1*35+0.2) =5.5 $\overline{K_3}$ *(0.12*9+0.1*35+0.2) =4.78 $\overline{K_4}$ *(0.1*9+0.12*35+0.3) =5.4	U=5.24	T=4.5	0.273

Table 4 is used to show the result of calculated TQ value as per assigned different-different inputs (number of processes and ABT)

8. Conclusion and Future Work

The final consideration that we used the RRNFS model to calculate the optimum value for TQ. When we applied the optimum value of TQ in given experimental data then found that the result of proposed model is far better than the RR and Modified RR using Fuzzy logic. The proposed model reduces the context switching, turnaround and waiting time. Xavier Normal function played a key role to calculate the variance of input neurons and output neurons for each layer that increased the performance of proposed model. The complete study of this paper provided evidence for the following points:

i. RRNFS model is used to calculate the best values for TQ, preventing unnecessary context switching of process schedular.
ii. We used the Xavier Normal function to calculate the optimum value of weights for increased the performance level in training neuro fuzzy inference system.
iii. We used the FIS to prepared the sample data set.
iv. We have compared the result with classic RR and Modified RR using Fuzzy logic in the parameters of average response time, average turnaround time and average waiting time.
v. The suggested model has a high throughput, avoids starvation and provides more complete and effective solutions than the previous ones.

To enhance the performance of proposed model, we can use some other parameters as inputs for neuro fuzzy inference system in future. In short, we can say that proposed model creates the new path with merging some other fuzzy techniques to create an innovative hybrid model.

Conflicts of Interests: Authors have no conflict of interests.

Funding Declaration: No funding is granted for this research work.

Author Contribution Declaration

Rajeev Sharma: Contributed to the conceptualization, design, and creation of the technique, as well as the writing of the first draft. was in charge of the verification of the mathematical models and carried out the basic analysis of the algorithm.

Atul Kumar Goel: Played a crucial part in putting the developed approach to use and testing it, helped analyse the results, and helped write and edit the manuscript.

M. K. Sharma: Provided fuzzy framework experience, oversaw the research, helped with the numerical examples, and significantly revised the final text. ensured the work's overall cohesion and scientific quality.

Each author has reviewed and given final approval to the text, and they all agree to be responsible for all parts of the work, ensuring that any concerns about the accuracy or integrity of any portion of the work are duly investigated and addressed.

Acknowledgements; This work has been carried out under the University Research Scheme Ref. number Dev. /1043 dated 29.06.2022.

References

[1]. A. Silberschatz, P. B. Galvin, and G. Gagne, "Operating system principles," *Wiley India Edition*, 7th edition, **2006**. ISBN: 978-81-265-0962-1.

[2]. Y. A. Adekunle, Z. O. Ogunwobi, A. S. Jerry, B. T. Efuwape, S. Ebiesuwa, and J. P. Ainam, "*A comparative study of scheduling algorithms for multiprogramming in real-time systems*," International Journal of Innovation and Scientific Research, Vol.**12**, No.**1**, pp.**180-185, 2014**. ISSN: 2351-8014.

[3]. N. Goel and R. B. Garg, "*A comparative study of CPU scheduling algorithms*," arXiv preprint arXiv: 1307.4165, **2013**. https://doi.org/10.48550/arXiv.1307.4165.

[4]. E. Kondili, C. C. Pantelides, and R. W. Sargent, "*A general algorithm for short-term scheduling of batch operations—I. MILP formulation*," Computers & Chemical Engineering, Vol.**17**, No.**2**, pp.**211-227, 1993**. https://doi.org/10.1016/0098-1354(93)80015-F.

[5]. W. Li, K. Kavi, and R. Akl, "*A non-preemptive scheduling algorithm for soft real-time systems*," Computers & Electrical Engineering, Vol.**33**, No.**1**, pp.**12-29, 2007**. https://doi.org/10.1016/j.compeleceng.2006.04.002.

[6]. C. Keerthanaa and M. Poongothai, "*Improved priority-based scheduling algorithm for real-time embedded systems*," in 2016 International Conference on Circuit, Power and Computing Technologies (ICCPCT), pp.**1-7, 2016.** DOI: 10.1109/ICCPCT.2016.7530188.

[7]. B. Nie, J. Du, G. Xu, H. Liu, R. Yu, and Q. Wen, "*A new operating system scheduling algorithm*," in Advanced Research on Electronic Commerce, Web Application, and Communication: International Conference, ECWAC 2011, Guangzhou, China, April 16-17, 2011. Proceedings, Part I, Springer Berlin Heidelberg, pp.**92-96, 2011.** ISSN 1865-0929.

[8]. M. Hamayun and H. Khurshid, "*An optimized shortest job first scheduling algorithm for CPU scheduling*," J. Appl. Environ. Biol. Sci, Vol.**5**, No.**12**, pp.**42-46, 2015**. ISSN: 2090-4274.

[9]. V. Chahar and S. Raheja, "*Fuzzy based multilevel queue scheduling algorithm*," in 2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp.**115-120, 2013.** DOI: 10.1109/ICACCI.2013.6637156.

[10]. A. Moallemi and M. Asgharilarimi, "*A fuzzy scheduling algorithm based on highest response ratio next algorithm*," in Innovations and Advanced Techniques in Systems, Computing Sciences and Software

Engineering, Springer Netherlands, pp.75-80. DOI: 10.1007/978-1-4020-8735-6_15.
[11]. A. Singh, P. Goyal, and S. Batra, "*An optimized round robin scheduling algorithm for CPU scheduling*," International Journal on Computer Science and Engineering, Vol.**2**, No.**7**, pp.**2383-2385, 2010.** ISSN: 0975-3397.
[12]. B. Alam, "*Finding time quantum of round robin CPU scheduling algorithm using fuzzy logic*," in 2008 International Conference on Computer and Electrical Engineering, pp.**795-798, 2008.** DOI: 10.1109/ICCEE.2008.89.
[13]. A. A. Aburas and V. Miho, "*Fuzzy logic-based algorithm for uniprocessor scheduling*," in 2008 International Conference on Computer and Communication Engineering, pp.**499-504, 2008.** DOI: 10.1109/ICCCE.2008.4580654.
[14]. B. Alam, "*Fuzzy Round Robin CPU Scheduling Algorithm*," J. Comput. Sci., Vol.**9**, No.**8**, pp.**1079-1085, 2013.**
[15]. L. Datta, "*A new RR scheduling approach for real-time systems using fuzzy logic*," International Journal of Computer Applications, Vol.**119**, No.**5, 2015.**
[16]. S. Lim and S. B. Cho, "*Intelligent OS process scheduling using fuzzy inference with user models*," in New Trends in Applied Artificial Intelligence: 20th International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2007, Kyoto, Japan, June 26-29, 2007. Proceedings 20, Springer Berlin Heidelberg, pp.**725-734, 2007.** DOI: https://doi.org/10.1007/978-3-540-73325-6_72.
[17]. B. Granam and H. ElAarag, "*Utilization of Fuzzy Logic in CPU Scheduling in Various Computing Environments*," in Proceedings of the 2019 ACM Southeast Conference, **2019**. doi.org/10.1145/3299815.3314463.
[18]. M.S. Kalas, Nikita D. Deshpande, "*Sepsis Detection in newborn infants - Diagnosis using fuzzy inference system- A Review*", International Journal of Computer Sciences and Engineering, Vol.**9**, Issue.**5**, pp.**43-46, 2021.** https://doi.org/10.26438/ijcse/v9i5.4346
[19]. M. Atique and M. S. Ali, "*A novel adaptive neuro fuzzy inference system-based CPU scheduler for multimedia operating system*," in 2007 International Joint Conference on Neural Networks, pp.**1002-1007, 2007.** DOI: 10.1109/IJCNN.2007.4371095.
[20]. J. A. Trivedi and P. S. Sajja, "*Improving efficiency of round robin scheduling using neuro fuzzy approach*," International Journal of Research and Reviews in Computer Science, Vol.**2**, No.**2**, pp.**308, 2011.**
[21]. Priya Nagargoje, Monali Baviskar, "*Uncertainty Handling In Big Data Analytics: Survey, Opportunities and Challenges*", International Journal of Computer Sciences and Engineering, Vol.**9**, Issue.**6**, pp.**59-63, 2021**. https://doi.org/10.26438/ijcse/v9i6.5963
[22]. F. Benhammadi, Z. Gessoum, and A. Mokhtari, "*CPU load prediction using neuro-fuzzy and Bayesian inferences*," Neurocomputing, Vol.**74**, No.**10**, pp.**1606-1616, 2011.** https://doi.org/10.1016/j.neucom.2011.01.009.
[23]. R. Sharma, A. K. Goel, M. K. Sharma, N. Dhiman, and V.N. Mishra, "*Modified Round Robin CPU Scheduling: A Fuzzy Logic-Based Approach*," in Applications of Operational Research in Business and Industries: Proceedings of 54th Annual Conference of ORSI, Singapore: Springer Nature Singapore, **2023**. https://doi.org/10.1007/978-981-19-8012-1_24.
[24]. X. Glorot and Y. Bengio, "*Understanding the difficulty of training deep feedforward neural networks*," in Proceedings of the thirteenth international conference on artificial intelligence and statistics, pp.**249-256, 2010.**
[25]. D. Nauck, "*Neuro-fuzzy systems: review and prospects*," in Proceedings of Fifth European Congress on Intelligent Techniques and Soft Computing (EUFIT'97), pp.**1044-1053, 1997**. URL: fuzzy.cs.uni-magdeburg.de/nauck.
[26]. A. Krogh, "*What are artificial neural networks*?" Nature biotechnology, Vol.**26**, No.**2**, pp.**195-197, 2008.** https://doi.org/10.1038/nbt1386.
[27]. R. Fullér, "Neural fuzzy systems," ISSN 0358-5654, 1995.
[28]. C. T. Lin and C. G. Lee, "*Neural fuzzy systems: a neuro-fuzzy synergism to intelligent systems*," Prentice-Hall, Inc., **1996**. https://dl.acm.org/doi/abs/10.5555/230237.
[29]. M. N. M. Salleh, N. Talpur, and K. Hussain, "*Adaptive neuro-fuzzy inference system: Overview, strengths, limitations, and solutions*", in Data Mining and Big Data: Second International Conference, DMBD 2017, Fukuoka, Japan, July 27–August 1, 2017, Proceedings 2, Springer International Publishing, pp.**527-535, 2017.** https://doi.org/10.1007/978-3-319-61845-6_52.
[30]. L. A. Zadeh, G. J. Klir, and B. Yuan, "*Fuzzy sets, fuzzy logic, and fuzzy systems: selected papers*," Vol.6, World scientific, ISBN 9810224214.

AUTHORS PROFILE

Mr. Rajeev sharma is the Assistant professor in IIMT Engineering college, Meerut. He has more than 15 years teaching as well as administrative experience. He has a conference Proceeding on Modified CPU Scheduling Process to his Credit published by Springer nature. He has attended many conferences, workshops, FDP's and symposium and presented research papers.

Dr. Atul Kumar Goel is the Professor of Mathematics in A.S. (P.G.) College Mawana, Meerut, affiliated to Chaudhary Charan Singh University, Meerut. He has more than 23 years teaching as well as administrative experience. He qualified JRF. He was awarded Ph.D. in Mathematics. More than 40 research papers in reputed journals, book chapters in edited books are to his credit. He has attended many conferences, workshops and symposium and presented research papers. Prof. Atul goel has delivered many invited talks and chaired sessions in India and Abroad.

Prof. Mukesh Kumar Sharma: Prof. Mukesh Kumar Sharma is Professor of Mathematics in Chaudhary Charan Singh University, Meerut. He has more than 20 years teaching as well as administrative experience. He was a merit holder. He qualified NET, JRF and GATE exam in 2001, 2002 respectively. He was awarded Ph.D. in 2007 on the topic "A Study of Fuzzy aspect to System Reliability. He has guided seven Ph.D. students and eleven M. Phil. Projects. More than 100 research papers in reputed journals, book chapters in edited books in Springer, Taylor & Francis and proceedings are to his credit. He has also completed two research projects awarded by U.P Government under Research & Development and Centre of Excellence Awarded by U.P. Government. He has many awards and fellowship to his credit. He has been the reviewers of many leading journals of IEEE, Elsevier, Springer and many more publication houses. He has organized many conferences, seminars and workshops on Fuzzy Logic, Optimization, MATLAB, Soft Computing and Artificial Intelligence. He has attended many conferences, workshops and symposium and presented research papers. Prof. Mukesh Kumar Sharma has delivered many invited talks and chaired sessions in 2015 India and Abroad.

