RESEARCH ARTICLE OPEN ACCESS

A Congestion Control Approach Based on Weighted Random Early Detection and Type-2 Fuzzy Logic System

Maha Salaheldeen Elbadawy Alhassan [1], Hani Hagras [2]

Department of Computer Science, College of Postgraduate Studies Sudan
University of Science and Technology-Sudan
The Computational Intelligence Centre, School of Computer Science and Electronic Engineering
University of Essex-U. K

ABSTRACT

Congestion is the greatest challenge facing the transmission of data packets across computer networks. Congestion in router buffer increases the packets loss. Active Queue Management (AQM) methods are able to detect congestion at an early stage and control it by packets dropping. The Weighted Random Early Detection (WRED) method, among many other AQM methods, gives a good performance to detect and control congestion, as well as preserve packet loss. This paper presents a methodology to improve the WRED by combining it with type-2 fuzzy logic system, where experiments are simulated under the discrete time queue in Java and OMNeT++ IDE to collect a dataset containing the queue sizes, average queue lengths and the number of packets dropped. The results show that the proposed type-2 Fuzzy Logic based Systems (FLS) outperformed type-1 FLS with WRED counterparts, according dropping and packet loss rates. The packet loss has been reduced at the rate of 32%compared to WRED, depending on the arrival probability.

Keywords: - Type-2 fuzzy logic, FLS, congestion, WRED, OMNeT ++.

I. INTRODUCTION

Network congestion control is a complex problem. This is becoming even more difficult with the increased demand to use the Internet where huge amounts of data are being transmitted and exchanged over the Internet for various purposes and applications (e.g. Voice over IP, video streaming, interactive games etc.), which are rapidly evolving with improved Quality of Service (QoS). The data is transmitted over different network topologies, which are interlinked by links and routers [1]-[3].

Congestion in router buffer comes with the increment in packets queued. If the number of stacked packets increases, these packets will be subject to delays [2], [3]. When the number of packets reaches the maximum capacity of the router buffer, the buffer will overflow and all the arriving packets will be lost. The delay and packet loss are considered major network problems and their control and reduction are objectives for various management techniques [3].

Generally, congestion control is classified into two main types, which are host centric algorithms (which uses TCP), and router centric algorithms based (which uses AQM) [2]-[4]. Active Queue Management (AQM) methods detect and control congestion at the early stage and start dropping packets early to reduce packet loss and delay. These methods depend on calculating dropping probability for each arrival packet in order to prevent congestion [5]. Examples of the existing AQM methods are Random Early Detection (RED), Weighted Random Early Detection (WRED)[5], [6]. Random Early Detection (RED) is the most widely deployed algorithm for congestion control which manages congestion before the router buffer overflow happens. The management is based on the computed average queue length and the calculated minimum and maximum thresholds values [6]. The minimum threshold is a position in the buffer and if it is not exceeded by average queue length (avg), the buffer can be considered having fair amount of packets in the queue. The maximum threshold parameter is another position if reached or exceeded by the avg it can be an indication of congestion [6], [7]. Therefore, the dropping probability increases when the queued packets reach maximum threshold. When packets arrive at router, the RED will calculate average queue length [6] as follows:

$$avg = (1 - w_q) * (avg + w_q * q) \quad (1)$$

Where w_q is the weight whose value range is from 0 to 1, q is the actual queue length. According to average queue length, the drop probability can be written as follows [6]:

$$DP = p_{max} * (avg - MIN_{th}) / (MAX_{th} - MIN_{th}) \quad (2)$$

$$P = \frac{DP}{(1 - count * DP)} \quad (3)$$

Where MIN_{th} is the minimum threshold and MAX_{th} is the maximum threshold of average queue length. P_{max} is the largest drop probability when the average queue length reaches MAX_{th}; count is the number of queued packets when avg stays between MIN_{th} and MAX_{th}.

The actual congestion control is implemented when the average queue length is between minimum and maximum thresholds [7]. Fig.1 illustrates the drop probability of RED algorithm.

RED method predicts congestion at the early stage as it responds by dropping packets to the increment of packet queuing in the buffer. The RED avoids dropping packets unnecessarily when a short heavy traffic is presented (false congestion), and avoids global synchronization by dropping packets randomly. However, the RED method suffers from insensitivity to current queue status in sudden congestion. It slowly adapts and results in packet loss as it uses average queue length instead of queue size. It cannot differentiate between traffic types. The algorithm that addresses this problem is called Weighted Random Early Detection (WRED) [6], [7].

The WRED is a queuing discipline for a network scheduler suited for congestion avoidance; it is an extension to random early detection (RED) [8]. The WRED combines the capabilities of the RED algorithm with Priority Queuing. The WRED must be complemented by a multilevel queue-scheduling algorithm, which determines the rates at which packets are forwarded from each queue to the output buffer based on pre-established priority values for each queue [7], [8]. The WRED can parse priority tags of packets according to the priority [9]. The drop probability of low priority packets is greater than high priority packets, so the high priority packets are not easy to be dropped, and are more likely to be sent to destination. Before congestion occurs, The WRED can reduce the chance of tail drop by selectively dropping packets. It avoids a large loss of packets as a result of buffer overflow, so that the risk of global synchronization is reduced [8]-[10]. Fig. 2 illustrates the drop probability of the WRED algorithm.

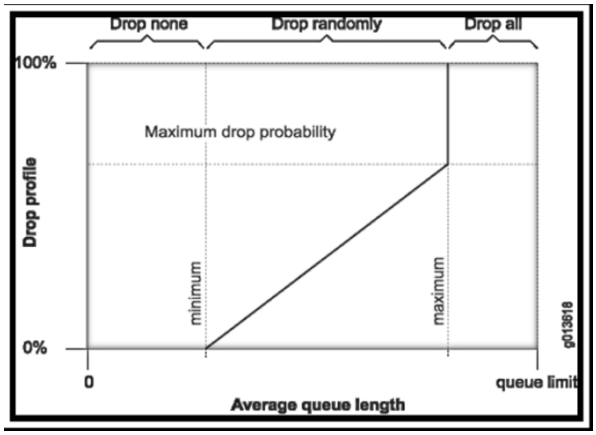

Fig. 1 The drop probability of the RED algorithm [6]

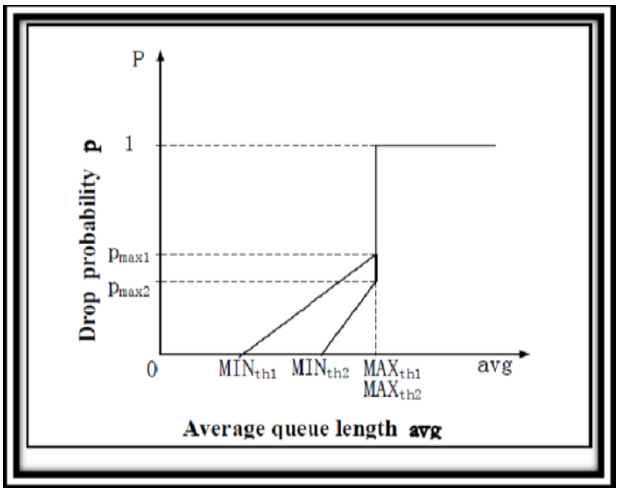

Fig. 2 The drop probability of the WRED algorithm [6]

This paper is organised as follows: Section II provides an overview on the Active Queue Management (AQM) methods. Section III provides an overview on the fuzzy logic **SYSTEMS. SECTION IV** provides the proposed congestion control approach based on Weighted Random Early Detection and type-2 Fuzzy Logic System (FLS). Section V provides experiments and results while Section VII presents the conclusions and future work.

II. AN OVERVIEW OF THE AQM METHODS

The Active Queue Management (AQM) methods focus on maximizing the network performance by detecting and , controlling congestion at early stage, and start dropping packets early to reduce packet loss and delay [11],[12]. The AQM methods can be classified into two classes based on the probability of each arrival packet. Different AQM methods have been developed for the congestion including non-Artificial Intelligence (AI) methods (such as Queue-based and Load-based) and AI methods (such as Support Vector Machine based (SVM), Particle Swarm Optimization (PSO) based, Neural Network based, Genetic Algorithm based (GA-based), and Fuzzy-based. Table 1 presents an evaluation of AQM Methods, together with Pros and Cons of each method.

TABLE I -EVALUATION OF AQM METHODS

Parameters					
Classification	**Examples**	**Congestion indicator**	**Pros**	**Cons**	**References**
Queue-based Methods	RED, WRED, ERED, and GRED	-average queue length -queue size	-avoid global synchronization - avoid dropping packets in heavy traffic - enhance delay measure	- fail response to sudden congestion	[11]-[16]
Load-based Methods	LUBA	-load factor	-avoid global synchronization - response to sudden congestion - improve delay	- drop packets in false congestion - require massive parameter initialization	[17]
Support Vector Machines	support vector based AQM (SAM) controller	-queue size	- higher predictive accuracy -	Bad non-linear process (in non-stable network traffic)	[18], [19]
Particle-based Methods	PSO-PID controller	-load factor	-generate a high quality solution within shorter calculation time -stable convergence characteristic than other stochastic methods.	- Very difficult to obtain a single solution.	[20]-[22]
Neuron-based Methods	NN-RED, Neuron PID, and AN AQM	-average queue length -queue size	- address the performance of TCP congestion control	- PID-controller cannot meet the Quality of Service (QoS) requirements.	[23]-[24],
Genetic algorithm-based Methods	IPAQM	Load factor	- optimal to control and detect congestion - derive optimal PID controller in stable queue length, low packet loss, and high link utilization	- complicated parameter initialization	[25], [26]
Fuzzy-based Methods	Fuzzy-RED, REDFL , AFRED, FAQM	-average queue length -queue size	- solve the parameter initialization problem	- fail to implement a congestion control - Most of these methods cannot differentiate between traffic types due to being based on RED algorithm	[27], [29]

III. A BRIEF OVERVIEW OF FUZZY LOGIC SYSTEMS

Fuzzy Logic is a multi-valued logic that allows intermediate values to be defined between conventional evaluations like true/false, yes/no, high/low, etc., where notions like rather tall, or very fast, can be formulated mathematically and processed by computers, in order to apply a more human-like way of thinking in the programming of computers [30], [31]. Fuzzy logic deals with degrees of truth that are provided in the context of fuzzy sets, which are called membership functions [32]. Fuzzy sets have laid the basis for a successful method of modeling uncertainty, vagueness and imprecision in a way that no other technique has been able to do [33]. The concept of the fuzzy set is only an expansion of the concept of a classical or crisp set. The classical set only considers a limited number of degrees of membership such as '0' or '1', or a range of data with limited degrees of membership [30-34]. The classical set has a sharp boundary, which means that a member either belongs to that set or it does not. This classical Fuzzy Logic is a multi-valued logic that allows intermediate values to be defined between classic evaluations like true/false, yes/no, high/low, etc., where notions like rather tall, or very fast, can be formulated mathematically and processed by computers, in order to apply a more human-like way of thinking in the programming of computers [30], [31]. Fuzzy logic deals with degrees of truth that are provided in the context of fuzzy sets, which are called membership functions [32]. Fuzzy sets have laid the basis for a successful method of modeling uncertainty, vagueness and imprecision in a way that no other technique has been able to do [33]. The classical set only considers a limited number of degrees of membership such as '0' or '1', or a range of data with limited degrees of membership [30-34]. The classical set has a sharp boundary, which means that a member either belongs to that set or it does not. Additionally, this classical set can be mapped to a function with two elements, 0 or 1 [30-34]. The fuzzy set is actually a fundamentally broader set compared with the classical or crisp set [33]. In a Type-1 fuzzy set, the membership grade for each element is a crisp number in (0, 1). Once the Type-1 membership functions have been chosen, the fact that the actual degree of membership itself is uncertain is no longer modeled in Type-1 fuzzy sets [34], [36].

In Type-1 Fuzzy Logic Systems (FLSs), the inference engine combines rules and gives a mapping from input Type-1 fuzzy sets to output Type-1 fuzzy sets. Multiple antecedents in rules are connected by the t-norm (corresponding to intersection of sets). Type-1 fuzzy sets handle the uncertainties associated with the FLS inputs and outputs by using accurate and crisp membership functions which, the user believes, would capture the uncertainties [35]-[36]. Multiple rules may be combined using the co-norm operation (corresponding to union of sets), or during defuzzification by weighted summation. The defuzzifier produces a crisp output from the fuzzy set that is the output of the inference engine, i.e., a crisp output is obtained from a Type-1 set [34].Fig 3 illustrates the Type-1 FLS. Type-1 Fuzzy Logic Systems (FLSs) have been applied with considerable success to many different applications. However, for the large majority of real-world applications, there is a need to contend with high levels of uncertainties
[34].

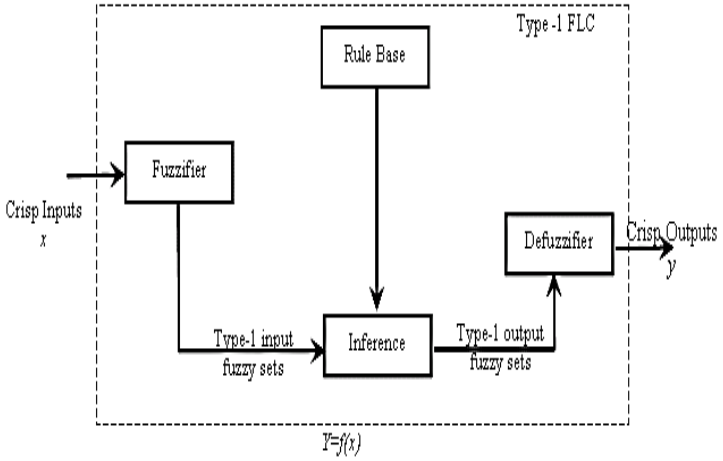

Fig. 3 Type-1 FLS [34]

Type-2 fuzzy sets and systems an expansion for Type-1 fuzzy sets and systems, so that more uncertainty can be handled, which is analogous to probability reducing to determinism when unpredictability vanishes [32-34]. A Type-2 fuzzy set is characterized by a fuzzy membership function, the membership value for each element of this set is a fuzzy set in [0,1], unlike a Type-1 fuzzy set where the membership value is a crisp number in [0,1] [32-35] .The membership functions of Type-2 fuzzy sets are include a footprint of uncertainty (FOU),FOU is a new third dimension of Type-2 fuzzy sets and the footprint of uncertainty that provide additional degrees of freedom that make it possible to directly model and handle uncertainties [35-37]. The denoted Type-2 fuzzy set is characterized by a type-2 membership function $\mu\tilde{A}(x, u)$ [35], where $x \epsilon X$ and $u \epsilon Jx \sqsubseteq [0, 1]$.

$$\tilde{A}= \{((x, u), \mu\tilde{A}(x, u))| \forall x \epsilon X \forall u \epsilon Jx \sqsubseteq [0, 1]\} \quad (4)$$

in which $0 \leq \mu\tilde{A}(x, u) \leq 1$.
The uncertainty in the primary memberships of a Type-2 fuzzy set consists of a bounded region that is called the footprint of uncertainty (FOU) [34-36]. It is the union of all primary memberships [35].
The FOU is described by its two functions, a Lower Membership Function (LMF) and an Upper Membership Function (UMF), both of which are Type-1 fuzzy sets. The FOU of a Type- 2 membership is functional, so it handles the rich variety of choices that can be made for a Type-1 membership function by using Type-2 fuzzy sets instead of Type-1 fuzzy sets. A Type-2 fuzzy set can be thought of as a large collection of embedded type-1 sets, each having a weight to associate with it.
This means it is possible to use Type-1 fuzzy set mathematics to characterize and work with interval Type-2 fuzzy sets [36]-[37]. Fig 4 illustrates an interval Type-2 fuzzy

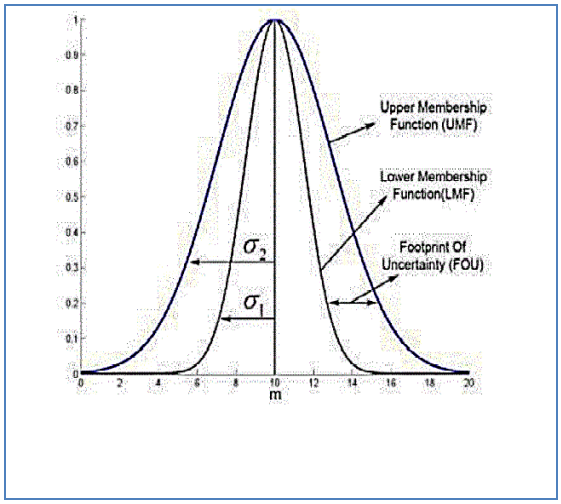

Fig 4.Interval Type-2 fuzzy set [38]

The Type-2 fuzzy sets membership functions can model and handle the numerical and linguistic uncertainties associated with the inputs and outputs of a fuzzy logic controller. Therefore, fuzzy logic systems that are based on Type-2 fuzzy sets have the potential to produce a better performance than Type-1 controllers [35] [36]. In the Type-2 fuzzy logic, an operation analogous to Type-1 defuzzification gives a Type-1 set from a Type-2 set. This process is called type-reduction rather than defuzzification. The type-reduced set can further be defuzzified to obtain a crisp output [35]. Fig 5 illustrates the operation of the Type-2 FLS.

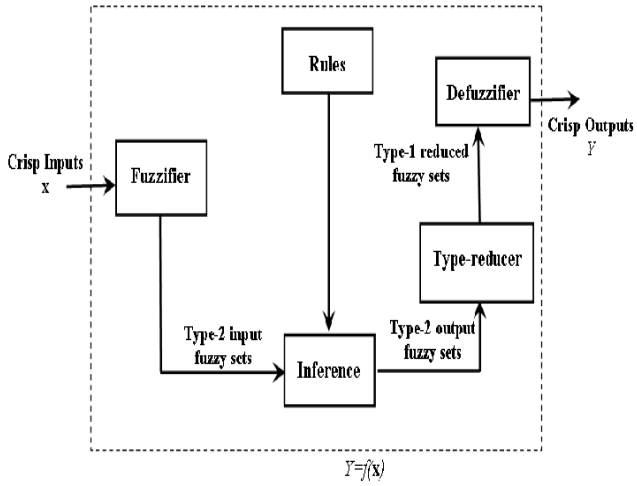

Fig. 5 Type-2 FLS [34]

IV. THE PROPOSED CONGESTION APPROACH BASED ON WEIGHTED RANDOM EARLY DETECTION AND TYPE-2 FUZZY LOGIC SYSTEMS

The proposed system is divided into two parts, in the first part the experiments are simulated under the discrete time queue in Java. And in the second part we have used Omnet ++ IDE at the simulation platform. For measuring, analysing and evaluating the performance of the WRED to control the congestion.

A. Simulation with Java

The programming code is developed in Net Beans 7.5 IDE. The discrete time queue, packet arrival and packet departure rates are established as probability values. Probability for arrival, and similarly for departure, with 0 values, means no packets will arrive at any time slot, whereas with a probability of 1.0 a packet will surely arrive at each time slot. For a value of 0.5, there is equal chance for the packet to arrive, or not arrive, at each time slot. The probability of the packet departure in the conducted experiments has been set to 0.5. The probability of packet arrival has been randomly assigned to a value between 0 and 1. When the arrival probability is below the departure, no congestion is expected, while congestion, or pre-congestion, is expected when the arrival rate is higher than the departure rate. The buffer size of 20 packets has been used MINth; MAXth, Pmax, and weight values are set to 5, 15, 0.5 and 0.5, respectively, as recommended in WRED. Moreover the proposed method involves other parameters that are the queue size and average queue length to calculate the number of packets dropped and packets lost. These parameters are set up empirically in the Type-1 fuzzy sets and Type-2 fuzzy sets, as shown the flowchart in Fig 6.

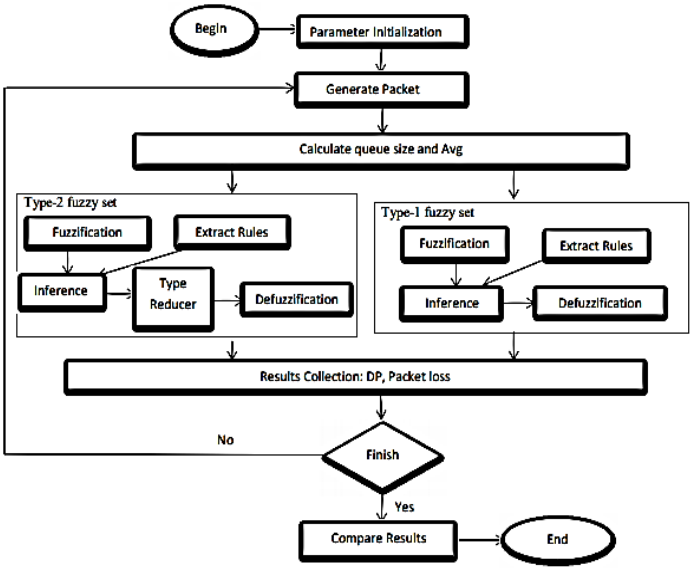

Fig 6: Flowchart of the proposed system

Based on the Type-1 fuzzy sets, the Type-1 fuzzy logic system has created the Fuzzy WRED method control congestion. This method relies on two input linguistic variables that are Queue size (q), and Average Queue Length (avg). The Queue size and Average Queue Length are classified into three fuzzy sets that are Low, Medium, and High of membership function as shown in Fig 7 and Fig 8. There is one output of this system, which is Drop Packets (DP). The Drop Probability is also classified into three fuzzy

sets, which are Low, Medium, and High of membership as shown in Fig 9.

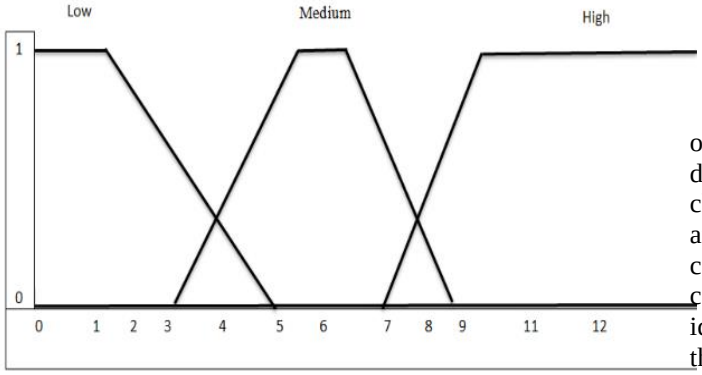

Fig 7. queue (q) Type- 1 fuzzy sets.

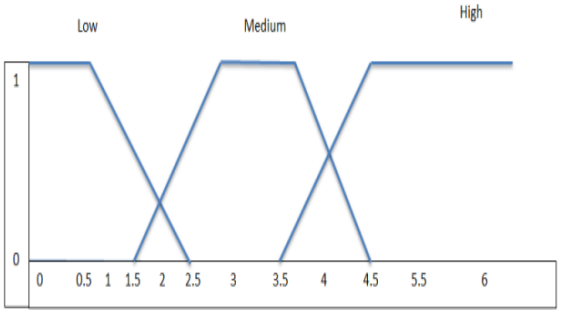

Fig 8. Average Queue Length (avg) Type-1 fuzzy sets.

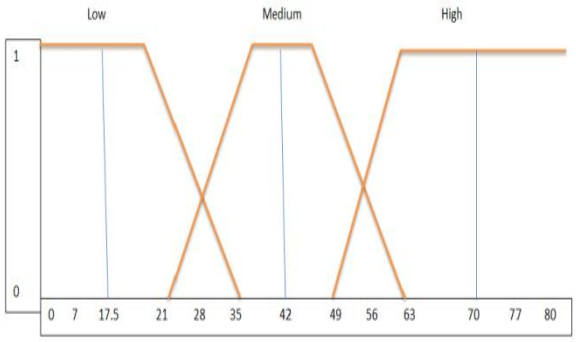

Fig 9. Drop Probability(DP) Type-1 fuzzy sets.

We have used Fuzzy C-Means (FCM) to extract the Type-1 fuzzy sets from data, as show in Fig 10, 11 and 12. FCM is a method of clustering which lets one part of data to belong to two or more clusters [39].

The clusters are created according to the distance between data points; and cluster centers are created for each cluster. It is based on minimization of the following objective function [39] [40]:

$$J_m = \sum_{i=1}^{N}\sum_{j=1}^{C} u_{ij}^{m} \left\| x_i - c_j \right\|^2, 1 \le m < \infty \qquad (5)$$

where m is any real number greater than 1, u_{ij} is the degree of membership of xi in the cluster j; xi is the i_{th} of d-dimensional measured data, c_j is the d-dimension center of the cluster, and ||*|| is any norm expressing the similarity between any measured data and the center [40]. Fuzzy partitioning is carried out through update of membership u_{ij} and the cluster centers c_j by assigning membership to each data point, identical to each cluster center, based on the distance between the center of the block and the data point.. As long as the data is near to the cluster center, the nearer is its membership to the particular cluster center [39]. The algorithm is composed of the following steps:

Step 1: Initialize the cluster membership values, μ_{ij}.

Step 2: Calculate the centers cj [40].

$$c_j = \frac{\sum_{i=1}^{N} u_{ij}^{m} x_i}{\sum_{i-1}^{N} u_{ij}^{m}} \qquad (6)$$

Step 3: Update membership values µij [39].

$$\mu_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{\left\| x_i - c_j \right\|}{\left\| x_i - c_k \right\|} \right)^{\frac{2}{m-1}}} \qquad (7)$$

Step4: Calculate the objective function, Jm.

Step 5: If µij || (k+1) - U (k) || <ε, then STOP; otherwise return to steps 2-4.

Step 6: This iteration will stop when [39]

$$max_{ij} = \left\{ \left| u_{ij}^{k+1} - u_{ij}^{k} \right| \right\} < \varepsilon$$

where ε is a termination criterion between 0 and 1, whereas k is the iteration step. This procedure converges to a local minimum or a saddle point of J_m [40].

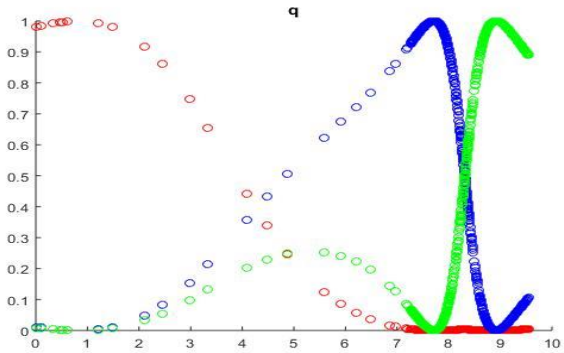

Fig10 .queue (q) Type-1 fuzzy sets.

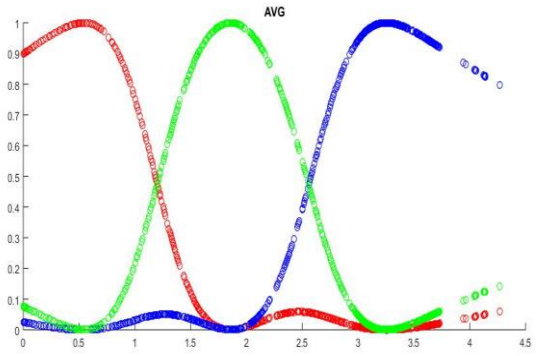

Fig 11 . Average Queue Length (avg) Type-1 fuzzy sets.

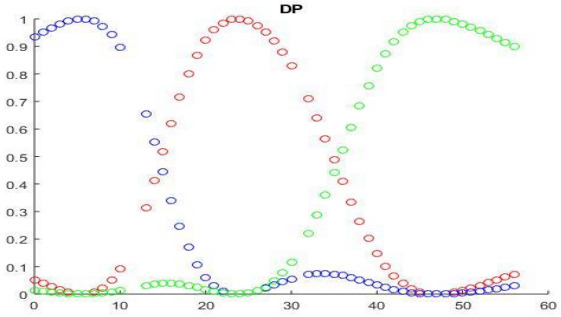

Fig 12 . Drop Packets (DP) Type-1 fuzzy sets.

Regarding the Type-2 fuzzy logic system, it is created by converting the Type-1 fuzzy sets to Type-2 fuzzy sets by introducing an uncertainty factor enabling the introduction of a Footprint of Uncertainty (FOU), encompassed between a Lower Membership Function (LMF) and an Upper Membership Function (UMF). Fig 13 illustrates the queue (q) Type-2 fuzzy set, Fig 14 illustrates the Average Queue Length (avg) Type-2 fuzzy set and Fig 15 illustrates the Drop Packets (PD) Type-2 fuzzy set.

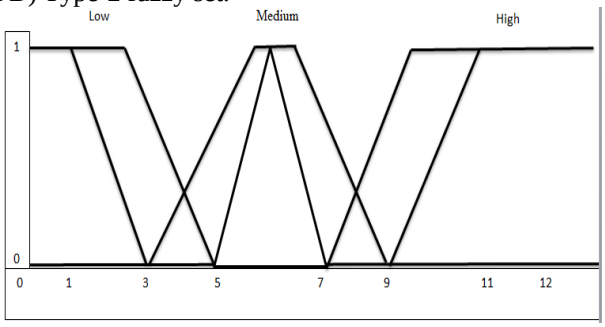

Fig 13. queue (*q*) Type- 2 fuzzy sets.

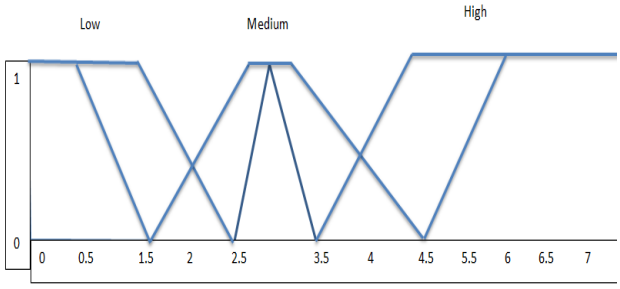

Fig 14. Average Queue Length (avg) Type-2 fuzzy sets.

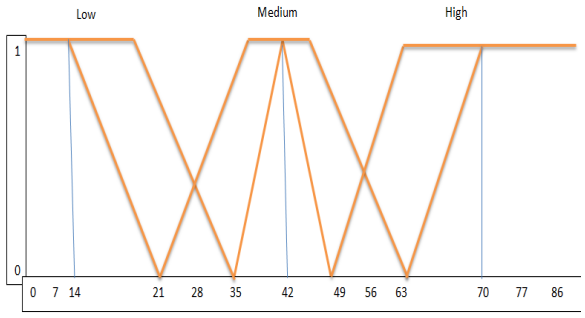

Fig 15. Drop Packets (DP) Type-2 fuzzy sets.

We have also used 10% FOU to encompass LMF and UMF, as shown in Fig 16, Fig 17 and Fig 18.

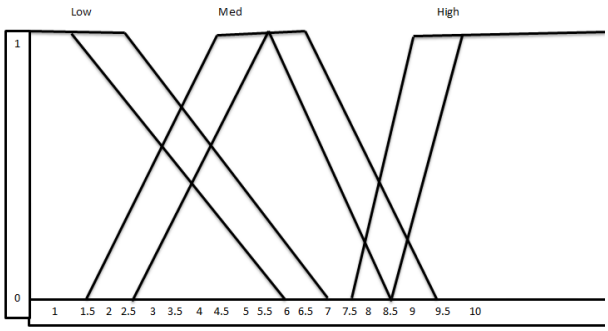

Fig 16. queue (q) Type- 2 fuzzy sets (10% FOU)

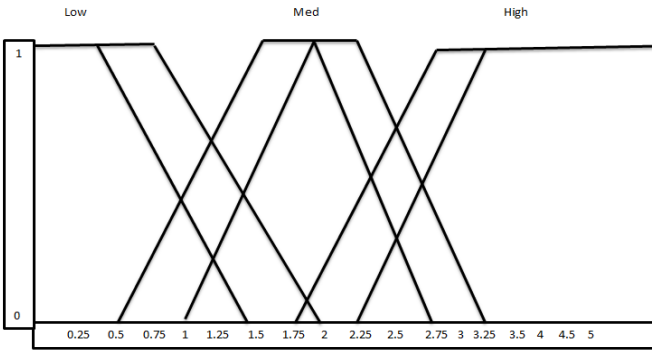

Fig 17. Average Queue Length (avg) Type-2 fuzzy (10% FOU).

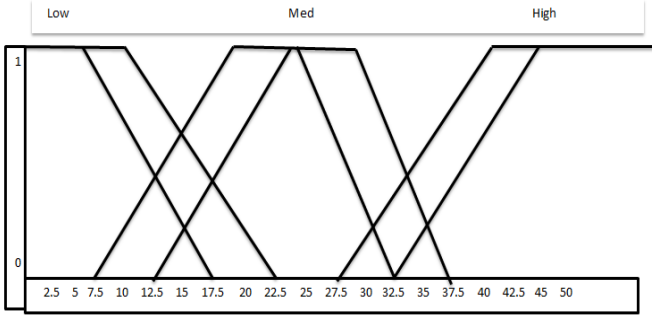

Fig 18. Drop Packets (DP) Type-2 fuzzy sets(10% FOU).

In all the Experiments on dataset, we have used 70% for training and 30% for system test. From the training data set, which contains numbers of rows, each one representing input - output pair (x (n), y (n)), n =1,...,N ,where N is the total number of training dataset. An example of the training dataset is shown in Table 2. We have used the following steps [34]:

Step1: In Type -1 FLS

- Computed the membership values $\mu_{\tilde{A}_s^z}$ for any antecedent fuzzy set z=1 ... k, (where k is the total number of fuzzy sets. representing the input pattern s where s=1…n.).
- Generated the rule base as described in [24] that can be extracted from each input-output pair (x(t), y(t)); each of them could be written as follows [34]:

IF x_1 is $\tilde{A}_1^t$ … and x_n is $\tilde{A}_n^t$ THEN y_1 is B_1^t. (8)

Step2: In Type -2 FLS

For any input-output pair (x (t), y (t)) in the training dataset, (t=1..N) , compute the upper membership values $\overline{\mu}_{\tilde{A}_s^z}^{cg}(x_s^{(t)})$ and lower membership values $\underline{\mu}_{\tilde{A}_s^z}^{cg}(x_s^{(t)})$ for each fuzzy set z=1..Z, and for each input variable s=1..n . Find z* ϵ {1...Z} such that [35]:

$$\mu_{\tilde{A}_s^{z^*}}^{cg}\left(x_s^{(t)}\right) \geq \mu_{\tilde{A}_s^z}^{cg}\left(x_s^{(t)}\right) \text{for all } z = 1,..zi \quad (9)$$

where $\mu_{\tilde{A}_s^z}^{cg}\left(x_s^{(t)}\right)$ is the center of gravity of the interval membership of $\tilde{A}_s^z$ at $x_s^{(t)}$ as followed by [35]:

$$\mu_{\tilde{A}_s^z}^{cg}\left(x_s^{(t)}\right) = \frac{1}{2}\left[\overline{\mu}_{\tilde{A}_s^z}^{cg}\left(x_s^{(t)}\right) + \underline{\mu}_{\tilde{A}_s^z}^{cg}\left(x_s^{(t)}\right)\right] \quad (10)$$

One rule is generated for each input–output data pair, where for each input the fuzzy set that achieves the maximum membership value at the data point is selected as the one in the IF part of the rule. The weight of the rule is computed as follows [35]:

$$w_i^{(t)} = \prod_{s=1}^{n} \mu_{\tilde{A}_s^z}^{cg}\left(x_s^{(t)}\right) \quad (11)$$

The weight of a rule $w_i^{(t)}$ is a measure of the strength of the points x (t) belonging to the fuzzy region covered by the rule.

$$IF x_1 is \tilde{A}_1^z ... and\ x_n is \tilde{A}_n^z\ THEN\ y\ is\ centered\ at\ y^{t_u^z} \quad (12)$$

- We repeated for all the t data points from 1 to N to obtain N data generated rules as described in [35].

Examples of the extracted rules are shown in Table 3

TABLE 2
EXAMPLES OF TRAINING DATASET

Value	*Q*	*Avg*	*PD*
1	1.56	0.62	6
2	3.47	1.52	18
3	4.35	2.86	31
4	5.97	2.58	37
5	7.86	3.29	44

TABLE 3
EXAMPLES OF THE EXTRACTED RULES

IF	*q*	*is*	*Low*	*and*	*avg*	*is*	*Low*	*THEN*	*PD is Low*
IF	*q*	*is*	*Low*	*and*	*avg*	*is*	*Med*	*THEN*	*PD is Low*
IF	*q*	*is*	*Med*	*and*	*avg*	*is*	*Low*	*THEN*	*PD is Low*
IF	*q*	*is*	*Med*	*and*	*avg*	*is*	*Med*	*THEN*	*PD is Med*
IF	*q*	*is*	*High*	*and*	*avg*	*is*	*Med*	*THEN*	*PD is Med*
IF	*q*	*is*	*High*	*and*	*avg*	*is*	*High*	*THEN*	*PD is High*

Table 3 indicates that if the q is in a Low fuzzy set, whatever the fuzzy set that the avg belongs to is, the PD will be in a Low fuzzy set. In case that the q is in either Med or High fuzzy set and the avg is in a medium fuzzy set, the PD will be in a Med fuzzy set. In case the q is in a High fuzzy set and the avg is in a High fuzzy set, the PD will be in a High fuzzy set.

B. *Simulation with OMNET++*

Regarding the second part of the systems shown in Fig 6, which is simulation part, we have created a network topology model based on two source nodes (s1, s2), two routers (r1, r2), and two destination nodes (s3, s4), which express different traffic types in OMNET++ as the simulation platform. The performance of the WRED to control the congestion has been analysed and evaluated according to congestion measures, such as queue size and the average queue length. A total of 2,000,000 slots of time were used to perform the experiments and three different arrival rates (0.92, 0.5, and 0.35). Fig 19 illustrates the network topology model. The outcome of this simulation is used to compare between Type-1 FLS and Type-2 FLS.

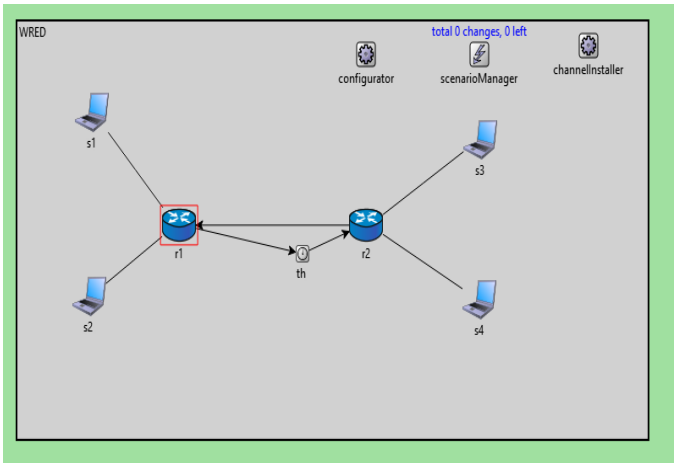

Fig 19. network topology model.

V. EXPERIMENTS AND RESULTS

To test and evaluate the Type-1, Type-2 fuzzy logic system and WRED method, we have used the test dataset (30%) acquired from the data explained in the previous section.

Type-2 FLS has been evaluated and compared with Type-1 FLS and WRED in three different scenarios in terms of packet loss, and packet dropping. Packet loss occurs because of buffer overflow. Packet dropping has been performed by a congestion method before a router buffer becomes full to avoid the ensuing congestion [11].

The first scenario assumed heavy congestion, in which the packet arrival rates was 0.92, whereas the departure rate was only 0.5. Fig 20 shows the Packet Losses (PL) and Packet Dropping (PD) ratios of WRED, Type-1 FLS, and Type-2 FLS for the first scenario. By comparison, Type-2 FLS displays the lowest packer loss of 12% followed by Type-1FLS having PL of 16% and WRED having PL of 34%. However, Type-2 FLS drops 32% followed by Type-1 FLS showing PD od 28% and finally WRED having PD= 20%, because Type-2 FLS predicts the congestion at an earlier stage than both methods under heavy congestion. Packet dropping shows the ability of a congestion method to predict the congestion at an early stage, control the number of dropped packets and from which source to drop. When both packet loss and packet dropping are aggregated, the total packets missed are nearly the same.

The second scenario assumed a light congestion, in which packet arrival and departure were both 0.5. This means the packet arrival and departure were equal, neither packet loss nor packet dropping should occur. However, packet loss and packet dropping occur because of the nature of the network traffic. Fig 21 shows the packet loss and packet dropping ratios, of WRED, Type-1 FLS, and Type-2 FLS for this scenario. Type-2 FLS shows a PD of 4% followed by Type-1 FLS having PD=3.4 and finally WRED having PD=3.1%. Furthermore the Type-2 FLS shows a PL of 2% followed by Type-1 FLS PL= 3% and WRED having PL= 4.4% and.

The third scenario assumed no congestion at all, in which the packet arrival and departure rates were 0.35 and 0.5, respectively. Fig 22 shows the packet loss and packet dropping ratios of WRED, Type-1 FLS, and Type-2 FLS for this scenario. WRED, Type-1 FLS, and Type-2 FLS show the same packet dropping and packet loss probabilities that are equal to 0, because the arrival rate was less than the departure rate, which indicated the lack of congestion.

Then we used the Root Mean Square Error (RMSE), as shown in Equation (13), to measure how much error there is between all outputs in Type-1FLS, Type-2 FLS and WRED method.

$$RMSE = \sqrt{\frac{1}{N}\sum[y_i - \overline{y}_i]^2} \qquad (13)$$

Where N is the total number of testing dataset, y is the actual output from the testing dataset, and $\overline{y}$ is the crisp output of the Type-1 FLS; Type-2 FLS and WRED method have been predicted.

The RMSE results of the Type-2 fuzzy based system as shown in Table 4 are better than Type-1 FLS and WRED method especially when using 10% FOU.

TABLE 4
TESTING DATA RESULTS

Method	*FOU*	*RMSE*
Type-1	Zero	0.970
Type-2	Empirical	0.935
Type-2	10%	0.829
WRED	-	1.684

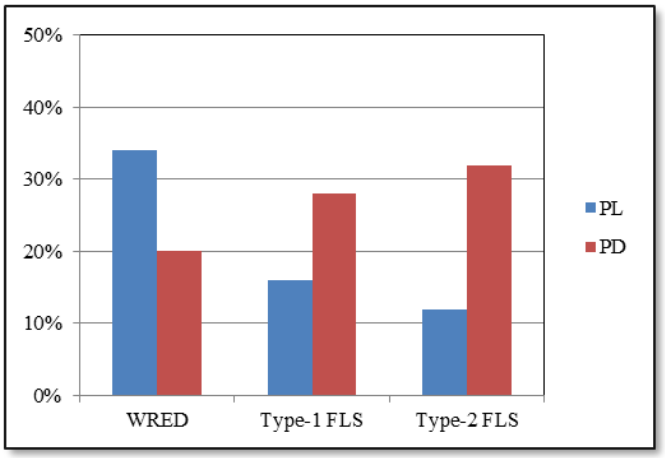

Fig 20. Heavy congestion: packet loss ratio and packet dropping ratio.

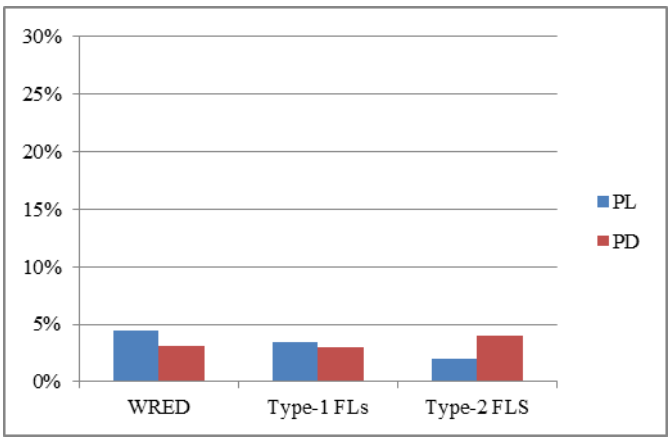

Fig 21. Light congestion:packet loss ratio, and packet dropping ratio.

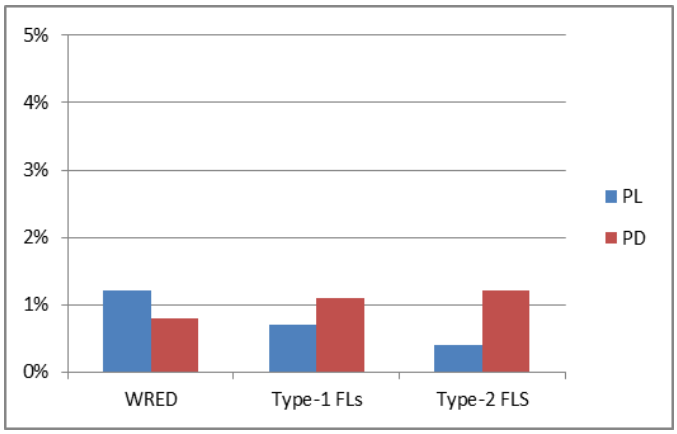

Fig 22 No congestion: packet loss ratio, and packet dropping ratio

VI. CONCLUSIONS AND FUTURE WORK

The WRED algorithm has been improved by combining it with the Type-2 fuzzy logic system.Type-2 FLS aimed to handle the nonlinear parameter uncertainties, and obtain a more satisfactory performance results in terms of packet loss in the event of heavy congestion. Experiments are simulated under the discrete time queue in Java and OMNeT++ IDE to collect a dataset containing the queue sizes, average queue lengths, and number of packets dropped.The results have been compared using RMSE, which showed that Type-2 FLS was better than Type-1 FLS and WRED method, because the packets loss decreased at by up to 25 % than Type-1 FLS and by up to 64% for WRED. Furthermore, the proposed Type-2 fuzzy based system increased the packets dropping by up to 12% for the Type-1 FLS and by up to 60% for the WRED based method. This is because the Type-2 FLS predicts the congestion at an earlier stage than both methods under heavy congestion, depending on the arrival probability and queue size. Generally, the queue size measure is helpful to avoid congestion, as well as packet loss.

The upcoming work will be using the Big Bang Big Crunch (BBBC) algorithm for optimizing the Type-2 fuzzy logic system to provide optimum choices for the Type-2 fuzzy sets parameters, as well as optimizing the rule base of the Type-2 FLS.

REFERENCES

[1] C. Hollot, "Analysis and design of controllers for AQM routers supporting TCP flows", IEEE Transactions on automatic control, Vol. 47, No. 6, pp. 945-959, 2002.

[2] K. Chitra, and G. Padamavathi. "Classification and performance of AQM-based schemes for congestion avoidance." arXiv preprint arXiv: 1005.4262 (2010).

[3] G. Thiruchelvi and J. Raja. "A survey on active queue management mechanisms", International Journal of Computer Science and Network Security, Vol.8, No. 12, pp. 130-145, 2008.

[4] C. Socrates, P. Devamalar and R. Kannamma Sridharan. "Congestion control for packet switched networks: A survey." International Journal of Scientific and Research Publications, Vol. 4, No. 12, pp.1-6, 2014

[5] K. Chhabra, M. Kshirsagar and A. Zadgaonkar. "Performance improvement of RED: Random early detection using input sensitivity with threshold modification." Proceedings of the 2015 International Conference on Electrical, Electronics, Signals, Communication and Optimization, 2015.

[6] I.Alshimaa et al. "Enhanced Random Early Detection (ENRED)." International Journal of Computer Applications, Vol. 92, No. 9, 2014.

[7] R. Jiang, Y. Pan, Y. Liu and X. Xue "Simulation Study of RED/WRED Mechanism Based on OPNET" Proceedings of the 2015 International Conference on Mechatronics, Electronic, Industrial and Control Engineering, 2015

[8] B. Lim et al. "RED and WRED performance analysis based on superposition of N MMBP arrival process." Advanced Information Networking and Applications (AINA), 2010 IEEE International Conference on advanced Information Networking and Applications pp.67-73, 2010.

[9] L. Mahmood, S. Jabbar, and G. Ma. "Performance evaluation of heterogeneous network based on RED and WRED" Indonesian Journal of Electrical Engineering and Computer Science, Vol. 3, No. 3 pp. 540-545, 2016

[10] C. Xu, J. Zhao, and G. Muntean. "Congestion control design for multipath transport protocols: a survey." IEEE Communications Surveys & Tutorials, Vol. 18, No. 4, pp. 2948-2969, 2016.

[11] R. Pan, B. Prabhakar, and K. Psounis. "CHOKe-a stateless active queue management scheme for approximating fair bandwidth allocation." Proceedings of 19th Joint Conference of the IEEE Computer and Communications Societies. Vol. 2, 2000.

[12] Z. Bing, and M. Atiquzzaman. "DSRED: an active queue management scheme for next generation networks", Proceedings of the 25th IEEE Local Computer Network conference, 2000

[13] J. Shahram, N. Alipasandi and B. Alipasandi. "An improvement over random early detection algorithm: a self-tuning approach." Journal of Electrical and Computer Engineering Innovations, Vol. 2, No. .2, pp. 57-61, 2014.

[14] A.Adeeb, et al. "Gentle-BLUE: A New Method for Active Queue Management." Proceeding of the 2014 IEEE International Conference on Advanced Computer Science Applications and Technologies pp.67-72, December 2014.

[15] K.Srisankar and R. Srikant. "End-to-end congestion control schemes: Utility functions, random losses and ECN marks." IEEE/ACM Transactions on networking, Vol. 11, No. 5, pp.689-702, 2003.

[16] L., Weidong, Xingwei Liu, and Jian Zhang. "SVM based analysis and prediction of network traffic" , Proceedings of the 2007 Intentional Conference on Intelligent Systems and Knowledge Engineering, 2007.
[17] M. Shah, M. Saleh, A. Wagan and M. Unar. "SAM: Support Vector Machine Based Active Queue Management" arXiv preprint arXiv: Vol 33, No.1, January 2014.
[18] X. Wang, et al. "PSO-PID: a novel controller for AQM routers", Proceedings of the 2006 IFIP International Conference on Wireless and Optical Communications Networks, 2006.
[19] S. Testouri, S. Saadaoui, and M. Benrejeb. "A particle swarm optimization approach for optimum design of first-order controllers in tcp/aqm network systems." International Journal of Computer Applications, pp. 33-38, 2012.
[20] S. Qaraawy, H. Ali, and Ali Mahmood. "Particle swarm optimization based robust controller for congestion avoidance in computer networks", Prceedings of the 2012 IEEE International Conference Future Communication Networks (ICFCN), 2012.
[21] B. Hariri, and N. Sadati. "NN-RED: an AQM mechanism based on neural networks." Electronics Letters, Vol. 43, No. 19, pp.1053-1055, 2007.
[22] C. Hyun, M. Fadali, and H. Lee. "Neural network control for TCP network congestion." Proceedings of the 2005 IEEE American Control Conference, 2005.
[23] R. Modjtaba, M. Tanhatalab and A. Rostami. "Nonlinear neural network congestion control based on genetic algorithm for TCP/IP networks" Proceedings of 2nd IEEE International Conference on Computational Intelligence, Communication and Networks, 2010.
[24] L. Xiao, Z. Wang, and X. Peng, "Research on congestion control model and algorithm for high-speed network based on genetic neural network and intelligent PID" Proceedings of the 2009 International Wireless Communications, Networking & Mobile Computing Conference, 2009.
[25] X. Fan, F. Du, and Z. Xie," Input-Rate Based Adaptive Fuzzy Neuron PID Control for AQM." In Advanced Materials Research , Vol. 846, pp. 3-8, 2014.
[26] F Li et al. "A comparative simulation study of TCP/AQM systems for evaluating the potential of neuron-based AQM schemes. Journal of Network and Computer Applications, Vol. 41, pp.274-299., 2014
[27] C. Chrysostomouet al. "Fuzzy logic controlled RED: congestion control in TCP/IP differentiated services networks." Soft Computing-A Fusion of Foundations, Methodologies and Applications, Vol 8, No. 2, pp.79-92, 2003.
[28] H. Hagras, V. Callaghan, M. Colley, "A Fuzzy-Genetic Based Embedded-Agent Approach to Learning and Control in Agricultural Autonomous Vehicles", Proceedings of the 1999 IEEE International Conference on Robotics and Automation, pp. 1005-1010, Detroit, , May 1999.
[29] F. Rivera-Illingworth, V. Callaghan and H. Hagras, "A Neural Network Agent Based Approach to Activity Detection in AmI Environments", Proceedings of the IEE International Workshop on Intelligent Environments, pp. 92-100, Colchester, UK, June 2005.
[30] T. Kumbasar and H. Hagras "Big Bang-Big Crunch Optimization based Interval Type-2 Fuzzy PID Cascade Controller Design Strategy Information Sciences, pp. 277-295, October 2014
[31] S. Helal, J. Woong Lee, S. Hossain, E. Kim, H. Hagras and D. Cook "Persim – Simulator for Human Activities in Pervasive Spaces" Proceedings of the 2011 International Conference on Intelligent Environments, Nottingham, UK, July 2011.
[32] T. Kumbasar and H. Hagras, "A Self-Tuning zSlices based General Type-2 Fuzzy PI Controller", IEEE Transactions on Fuzzy Systems, Vol.23, No.4, pp.991-1013, August 2015.
[33] A.Cara, C. Wagner, H. Hagras, I. Rojas and H. Pomares" Multi-objective Optimization and Comparison of Non-Singleton Type-1 and Singleton Interval Type-2 Fuzzy Logic Systems" IEEE Transactions on Fuzzy Systems, Vol. 21, No.3, pp. 459-476, June 2013.
[34] H. Hagras, M. Colley, V. Callaghan and M. Carr-West, "Online Learning and Adaptation of Autonomous Mobile Robots for Sustainable Agriculture", Autonomous Robots, Vol. 13, pp. 37-52, July 2002.
[35] H. Hagras, C. Wagner, "Towards the Widespread Use of Type-2 Fuzzy Logic Systems in Real World Applications" IEEE Computational Intelligence Magazine, pp.14-24, August 2012.
[36] B. Yao, H. Hagras, D. Alghazzawi and M. Al haddad, "A Big Bang-Big Crunch Type-2 Fuzzy Logic System for Machine Vision-Based Event Detection and Summarization in Real-world Ambient Assisted Living" IEEE Transactions on Fuzzy Systems, 2016
[37] A. Bilgin, H.Hagras, J. Helvert and D. Alghazzawi "A Linear General Type-2 Fuzzy Logic Based Computing With Words Approach for Realising an Ambient Intelligent Platform for Cooking Recipes Recommendation" IEEE Transactions on Fuzzy Systems, Vol. 24, No.2, pp. 306-326, 2016.
[38] A. Starkey, H. Hagras, S. Shakya and G. Owusu, "A Multi-Objective Genetic Type-2 Fuzzy Logic Based System for Mobile Field Workforce Area Optimization" Information Sceinces, Vol. 333, pp. 390-411, September 2016.
[39] T. Velmurugan, T. Santhanam, "Performance evaluation of k-means and fuzzy c-means clustering algorithms for statistical distributions of input data points". European Journal of Scientific, Vol.46, pp.320-330, 2010.
[40] B. Sharma, K. Venugopalan," Performance evaluation of k-means and fuzzy C-means clustering algorithms for identification of hematoma in brain CT scan images", international Journal of Advanced Research in Computer Science, Udaipur Vol. 3,pp.0976-5697, 2012.

[41] M. S. E. Alhassan and H. Hagras, "Towards Congestion Control Approach Based on Weighted Random Early Detection and Type-2 Fuzzy Logic System," 2018 10th Comput. Sci. Electron. Eng. Conf. CEEC 2018 - Proc., pp. 71–74, 2019.

[42] M. F. Masrom, N. M. Ghani, N. F. Jamin, N. A. A. Razali, "Control of Triple Link Inverted Pendulum on Two- Wheeled System Using IT2FLC", *In:* Proceedings of the Automatic Control and Intelligent Systems (I2CACIS), IEEE International Conference, pp:29-34, 2018.

[43] B. Yao, H. Hagras, D. Alghazzawi and M. Al haddad, "A Big Bang-Big Crunch Type-2 Fuzzy Logic System for Machine Vision-Based Event Detection and Summarization in Real-world Ambient Assisted Living". IEEE Transactions on Fuzzy Systems, Vol. 24, pp.1307 – 1319, 2016.

[44] H. A. Mohammed and H. Hagras, "Towards Developing Type 2 Fuzzy Logic Diet Recommendation System for Diabetes," 2018 10th Comput. Sci. Electron. Eng. Conf. CEEC 2018 - Proc., pp. 56–59, 2019.

[45] H. S. Yusuf and H. Hagras, "Towards Image Steganography Using Type-2 Fuzzy Logic and Edge Detection," 2018 10th Comput. Sci. Electron. Eng. Conf. CEEC 2018 - Proc., pp. 75–78, 2019.

[46] M. Antonelli, D. Bernardo, H. Hagras, and F. Marcelloni, "Multiobjective Evolutionary Optimization of Type-2 Fuzzy Rule-Based Systems for Financial Data Classification," *IEEE Trans. Fuzzy Syst.*, Vol. 25, No. 2, pp. 249-264 2016.

[47] J. Andreu-Perez, F.Cao, H. Hagras,G.Yang, "A self-adaptive online brain–machine interface of a humanoid robot through a general type-2 fuzzy inference system", IEEE Transactions on Fuzzy Systems, Vol. 26, No.1, pp. 101-116, Februray 2018.