International Journal of Computer Science Trends and Technology (IJCST) – Volume 9 Issue 3, May-Jun 2021

RESEARCH ARTICLE OPEN ACCESS

A Big-Bang Big-Crunch Type-2 Fuzzy Logic System for a Congestion Control Approach Based on Weighted Random Early Detection

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

[1] Department of Computer Science, College of Postgraduate Studies, Sudan University of Science and Technology - Sudan

[2] The Computational Intelligence Centre, School of Computer Science and Electronic Engineering University of Essex - U.K

ABSTRACT

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. In previous work we presented a methodology to improve the WRED by combining it with type-2 fuzzy logic system. This paper presents a Type-2 Fuzzy Logic System (FLS) which has been optimized by the Big-Bang Big-Crunch (BB-BC) approach to allow a best execution to detect and control congestion. The results show that the BB-BC Type-2 FLS outperformed Type-2 FLS, Type-1 FLS with WRED counterparts [4], according dropping and packet loss rates. The packet loss has been reduced at the rate of 35% compared to WRED.

Keywords: - Big-Bang Big-Crunch, Type-2 fuzzy logic System, FLS, congestion, WRED.

I. INTRODUCTION

Congestion occurs at the buffers of the network routers when the amount of incoming packets exceeds the available network resources and the buffer can no longer handle all incoming packets [2], [3]. When the number of packets reached the maximum capacity of the router buffer, the buffer will be overflowed and all the arriving packets are being lost. Packet loss is a major problem of network and their control and reduction are objectives for various management techniques [3], [4]. A congestion controls management mechanism is operated on the router to response to the congestion when it occurs [3]. Each router in the network uses two kinds of algorithms to congestion control. Queue Management algorithms to control the length of packet queues by dropping packets when appropriate. And scheduling algorithms to determine which packet to drop next and, which is to send [4].

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.

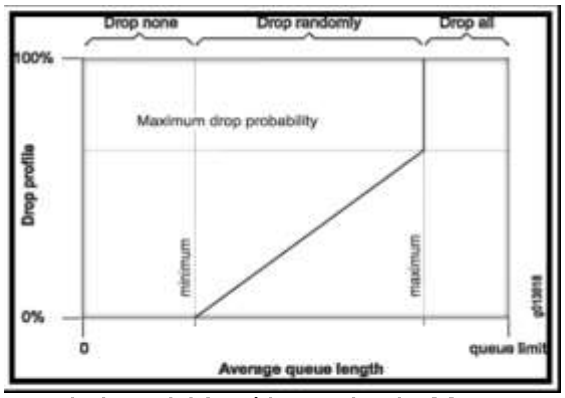


Fig. 1 The drop probability of the RED algorithm [6]

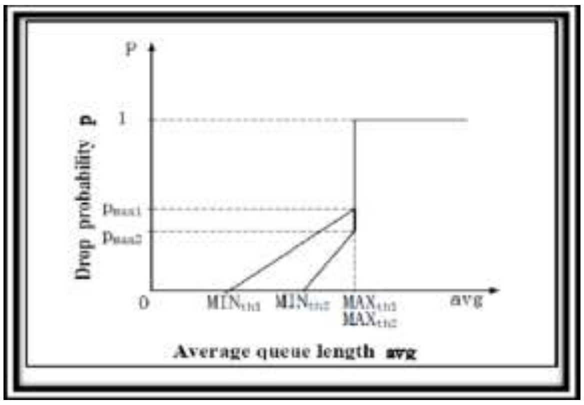


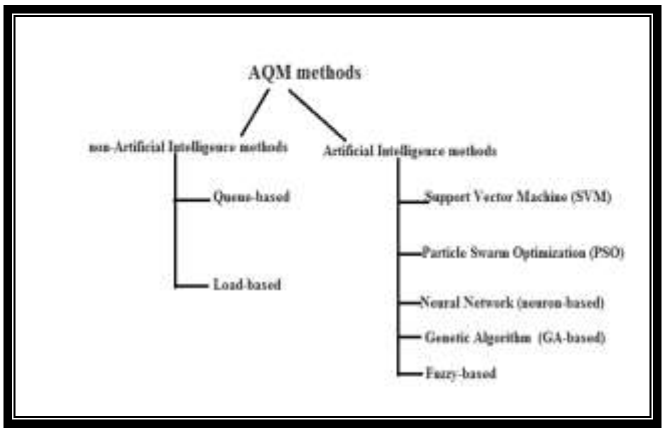
Fig. 2 The drop probability of the WRED algorithm [6]

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 over view of the Big Bang Big Crunch. Section V provides the proposed optimization congestion control approach based on Weighted Random Early Detection and type-2 Fuzzy Logic System (FLS). Section VI provides experiments and results while Section VII presents the conclusions.

II. AN OVERVIEW OF THE AQM METHODS

The aims of the congestion control process are to predict, detect and prevent congestion in early stage to decrease packet loss and average queue length, and increase throughput [11].

AQM methods are operated by gathering information about the performance measures, and maximize the network performance. AQM methods detect and control congestion in early stage and start dropping packets early to reduce packet loss and delay [11], [12]. Different AQM methods were developed for the congestion detection in early stage. These methods can be classified into two classes, first non-Artificial Intelligence methods such as Queue-based and Load-based. And second Artificial Intelligence methods such as Support Vector Machine based (SVM), Particle Swarm Optimization (PSO) based, Neural Network based (neuron-based), Genetic Algorithm based (GA-based), and Fuzzy-based. This classification is illustrated in Fig 3.


AQM methods
non-Artificial Intelligence methods
Artificial Intelligence methods
Queue-based
Load-based
Support Vector Machine (SVM)
Particle Swarm Optimization (PSO)
Neural Network (neuron-based)
Genetic Algorithm (GA-based)
Fuzzy-based
Fig. 3 AQM Methods Classification

A. *Non-Artificial Intelligence methods (Non-AI)*

These methods are including Queue-based and Load-based methods.

Queue-based methods use average queue length, queue size or packet loss as congestion indicator. These measures are then used for calculating dropping probability with each arrival packet; the calculated value is then used to decide on dropping packets. Several AQM methods have used average queue length as congestion indicator, such as RED, WRED, FRED, and GRED [10]-[12]; all these methods were built for the overall aim to enable stabilization of the average queue length under all circumstances in order to enhance network performance [11],[12].

Load-based methods use arrival rate or load factor as congestion indicator. Similar to queue-based methods avoid congestion by calculating dropping packets with each arrival packet, and use the calculated value to decide on dropping packets at early stage to avoid congestion, such as Link Utilization Based AQM (LUBA) [13], [14].

All methods, both queue-based and load-based have pros and cons. In the attempt to maximize the benefits of both types, several hybrid AQM methods were proposed with considers arrival rate and queue size as congestion indicators and calculates dropping packets in linear equation with the aim at stabilizing congestion control factors at router buffer, such as Robust Active Queue Management (RaQ) [14],[15].

B. Artificial Intelligence methods (AI)

There are many studies have used AI methods to controlling the Congestion to reduce packet loss and maintaining network throughput. Some of these methods are presented below.

1) Support Vector Machines (SVM)

SVM is an effective statistical learning method for pattern recognition [16], [17]. The goal of SVM is to produce a model that can predict the target values of the data instances in the testing set given the attributes only [16].

Several studies have dealt with SVM method to differentiating the cause of packet loss when the loss is caused by congestion.

The simulation process of the network running with AQM controllers are collected the being predicted traffic data, which is on the bottleneck router. The performance of the SVM controller has been compared with conventional AQM controllers [16], such as SAM is trained by using the SVM (Support Vector Machine) with the RBF (Radial Basis Function) kernel [17]. In terms of pros and cons, predictive model based on SVM could obtain higher predictive accuracy because SVM fit for the stable network traffic prediction, but they are not good in non-linear process. Therefore, the results of prediction would be bad if the network traffic is non-stable or the parameters are set unreasonable. The major drawback of the SVM is its higher computational cost for the constrained optimization programming.

2) Particle Swarm Optimization (PSO)

Particle Swarm Optimization (PSO) method was developed in 1995 [18]-[20] is biologically inspired solution motivated by social behaviour. PSO relies on the exchange of information between particles called swarm moving in the search space looking for the best solution [18], [19]. There are many studies have used the PSO to controlling the Congestion, and maintaining network throughput. In these studies are presented PI controllers using Particle Swarm Optimization (PSO) method as an Active Queue Management (AQM) for congestion avoidance in networks. The results shown that a faster response time as well as the regulation of the output to a constant value by the designed controller can be achieved [18]-[20].

3) Neuron- based methods

The neural network acts as a feedback controller to maintain the actual queue size close to a reference target. The neural AQM outperformed other AQM, several proposed active queue management methods based on neural networks such as NN-RED, Neuron PID, and AN AQM. All methods exhibited important attributes including fast convergence with high accuracy to a desired queue length [21]-[23]. In general the Neuron-based AQM methods address the performance degradations of TCP congestion control. But there are a few drawbacks of neuron adaptive PID-controller based AQM algorithms leading to poor performance like causing data retention dropping and oscillation when the time delay is large, which means that the existing neuron adaptive PID-controller cannot meet the Quality of Service (QoS) requirements [21],[22].

4) Genetic algorithm - based Methods

Genetic algorithm is an artificial optimization scheme developed in analogy to natural evolution performing an exploration of the search space. Therefore, some algorithms have combined GA and neural networks to improve the AQM Methods [24]-[26]. Genetic algorithm is optimal to control and detect congestion, especially for traffic that exceeds the nominal bandwidth causing severe overload on the node. Therefore, the improvement in terms of response time and link utilization is because the nonlinear controller has a variable gain that allows the AQM to recover faster from large variation in traffic loads [24], [26].

5) Fuzzy –based methods

Fuzzy logic (FL) is directed at thought patterns that are approximate rather than precise. It also provides a solution to non-linear control because it is closer to real world. Non-linearity is handled by rules, membership functions, and the inference process which results in improved performance [27], [28].

Several algorithms used of fuzzy logic (FL) in controlling congestion in the network, discussed FL based methods show improvement in increased in throughput, reduction in delays and packet loss. Fuzzy Logic can be effectively use in controlling congestion at core as well as at bottleneck router, such as Fuzzy-RED ,REDFL, FEM, AFRED,FAQM,GRED, [27]-[29]. In addition intelligent techniques based on Fuzzy Logic for congestion control and improved networking congestion control approach for TCP used ECN feedback mechanism based on Fuzzy Logic with satisfactory results [27]-[29].Fuzzy-based methods have solved the parameter initialization problem that presented in other AQM methods. However, fuzzy-based methods fail to implement a congestion control that can efficiently address the expected congestion cases encountered by the network resources, which in turn effect and waste the network resources. Generally fuzzy-based methods have solved the parameter initialization problem that presented in other AQM methods.

However, fuzzy-based methods fail to implement a congestion control that can efficiently address the expected congestion cases encountered by the network resources, which in turn effect and waste the network resources.

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]-[36]. 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]-[38]. 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 4 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].

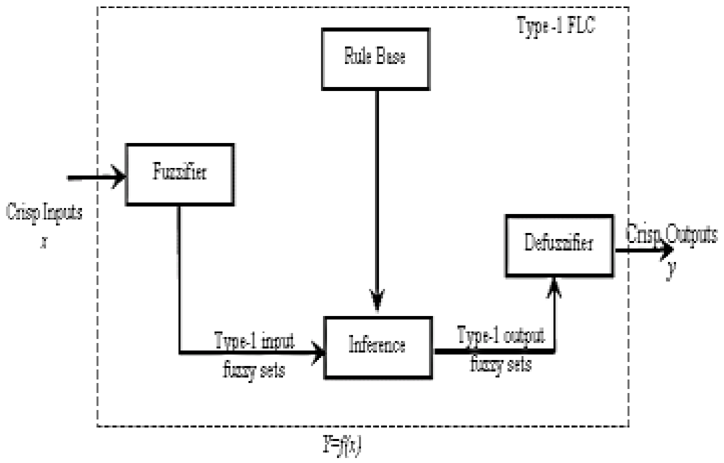


Fig. 4 Type-1 FLS [34]

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]-[36]. 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 $x \in X$ and $u \in Jx \sqsubseteq [0, 1]$.

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

in which $0 \leqslant \mu\tilde{A}(x, u) \leqslant 1$.

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]-[39]. Fig 5 illustrates an interval Type-2 fuzzy

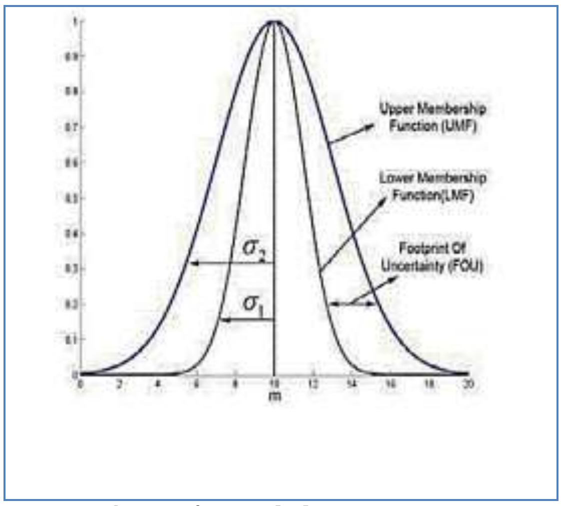


Fig 5.Interval Type-2 fuzzy set [38]

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 6 illustrates the operation of the Type-2 FLS.

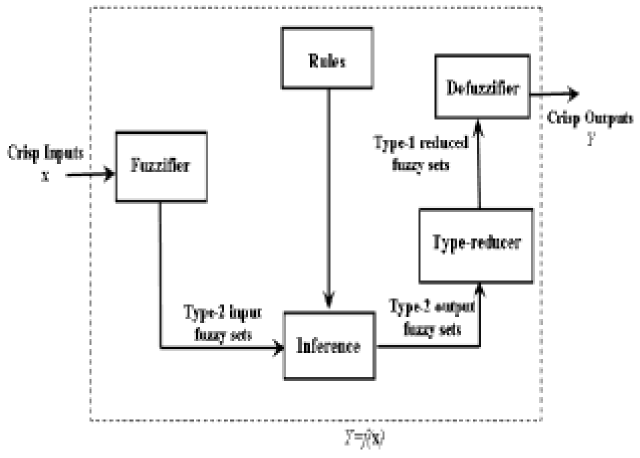


Fig. 6 Type-2 FLS [34]

IV. A BRIEF OVERVIEW OF THE BIG BANG BIG CRUNCH ALGORITHM

The BB-BC algorithm is a heuristic population-based evolutionary approach presented by Erol and Eksin [49]-[52]. It is derived from the theory of the birth of the universe in astrophysics, namely the Big-Bang Big-Crunch Theory. The key advantages of the BB-BC are its low computational cost, ease of implementation and fast convergence. The algorithm is similar to a Genetic Algorithm (GA) with respect to creating an initial population randomly. The creation of the initial random population is called the Big Bang phase. The Big Bang phase is followed by a Big Crunch phase which is akin to a convergence operator that picks out one output from many inputs via a centre of mass or minimum cost approach [49]-[51]. All subsequent Big Bang phases are randomly distributed around the output picked in the previous Big Crunch phase. The procedure of the BB-BC is as follows [50]:

Step 1(Big Bang Phase): Form an initial generation of N candidates in a random manner within the limits of the search space. Step 2: Calculate the fitness function values of all the candidate solutions.

Step 3(Big Crunch Phase): In this phase either the best-fit individual or the centre of mass is chosen as the centre point. The centre of mass is calculated as:

$$x_c = \frac{\sum_{i=1}^{N} \frac{x_i}{f_i}}{\sum_{i=1}^{N} \frac{1}{f_i}} \qquad (5)$$

Where x_c is the position of the center of mass, xi is the position of the candidate, fi is the cost function value of the i_{th} candidate, and N is the population size.

Step 4: Calculate new candidate solutions around the centre of mass by adding or subtracting a normal random number whose values decrease as the iterations elapse. This can be formalized as:

$$x_{new} = x_c + {}^{Lr}/_{k} \qquad (6)$$

Where x_c is the position of the centre of mass, L is the upper limit of the parameter, r is the random number and k is the iteration step. Then if new point x_{new} is greater than the upper limit l then x_{new} is set to L or if the new point x_{new} is smaller than the lower limit u then x_{new} is set to u.

Step 5: Check if stopping criteria are met. If M iterations are completed stop else return to Step 2.

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

In the earlier research [4], [6], we have proposed congestion Control Approach Based on Weighted Random Early Detection and Type-2 Fuzzy Logic System. We have started from the experiments are simulated under the discrete time queue in Java , and we involved 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. And then we have used Omnet ++ IDE at the simulation platform. For measuring, analyzing and evaluating the performance of the WRED to control the congestion [4].

C. Optimizing the Rule Base and Type-2 Membership of the Type-2 with BB-BC

To optimize the Type-2 FLS (Fuzzy WRED), the Membership Functions (MFs) and the rule base are optimized using the BB-BC optimization method [49]-[52]; to increase the accuracy of predicts congestion at the early stage.

The Membership Functions (MFs) for the Type-1 FLS is extracted using Fuzzy C Means Clustering (FCM) [42], and BB-BC is used to transform this MF into Type-2 MFs.

To optimize the Rule Base of the Type-2 FLS with BB-BC, for applying the BB-BC, the parameters of the rule base is encoded into the form of a population. The representation of a Type-2 rule base is shown in (7).

$$R_1^1 R_2^1, \dots R_a^1 R_{out}^1, \dots R_1^m R_k^m, \dots, R_a^m, R_{out}^m \qquad (7)$$

As shown in 7 R_k^m are the antecedents and R_{out}^m is the consequent of each rule respectively, where m= 1..., M, M is the number of rules and k = 1… a, a is the number of antecedents to be tuned. We use the following function as the cost function for BB-BC where RMSE is the Root Mean Square Error.

We generated different Type-2 FLS with different small rule bases 20-250 rules, and with a small number of antecedents per rules, such as 3 or 4 antecedents per rule. Which can speedy the system and each rule could be simply understood.

D. Generation of Type-1 and Type-2 Fuzzy Set from Data

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.

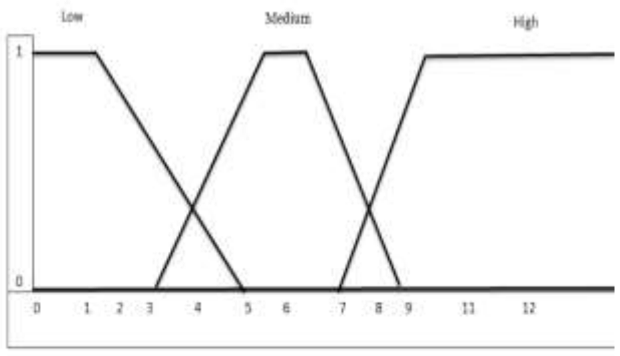


Fig 7. queue (q) Type- 1 fuzzy sets.

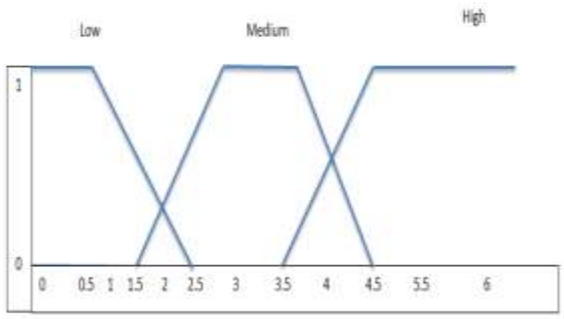


Fig. 8 Average Queue Length (avg) Type-1 fuzzy sets.

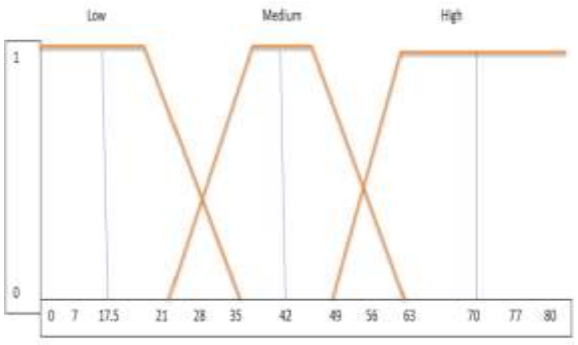


Fig .9 Drop Probability(DP) Type-1 fuzzy sets.

We have used Fuzzy C-Means (FCM) to extract the Type-1 fuzzy sets from data, as show in Fig 10 , 11and 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]:

$$J_m = \sum_{i=1}^{N}\sum_{j=1}^{N} u_{ij}^m \left\|x_i - c_j\right\|^2, 1 \le m \le \infty \qquad (8)$$

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 [43]. 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 [42]. The algorithm is composed of the following steps:

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

Step 2: Calculate the centers c_j [43].

$$c_j = \frac{\sum_{i=1}^{N} u_{ij}^m x_i}{\sum_{i-1}^{N} u_{ij}^m} \qquad (9)$$

Step 3: Update membership values µij [39].

$$m_{ij} = \frac{1}{\sum_{k-1}^{c} \left(\frac{\|x_i - c_j\|}{\|x_i - c_k\|} \right)^{\frac{2}{m-1}}} \qquad (10)$$

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 [42]

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

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 [43].

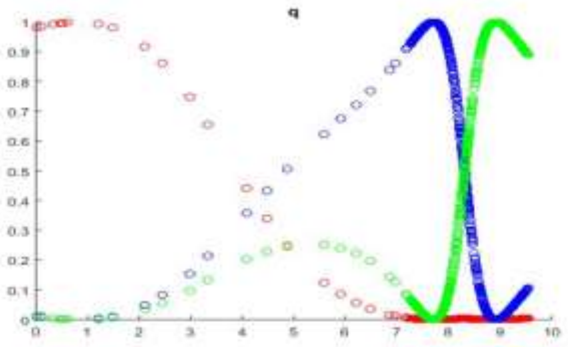


Fig 10 .queue (q) Type-1 fuzzy sets.

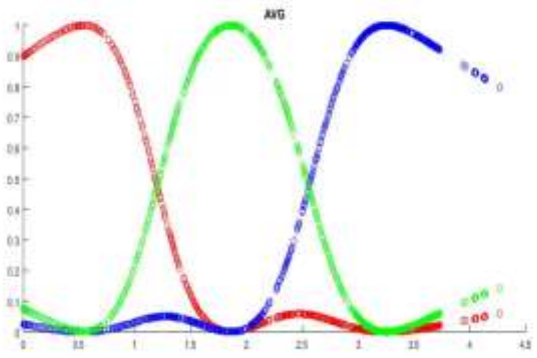


Fig. 11 .Average Queue Length (avg) Type-1 fuzzy sets.

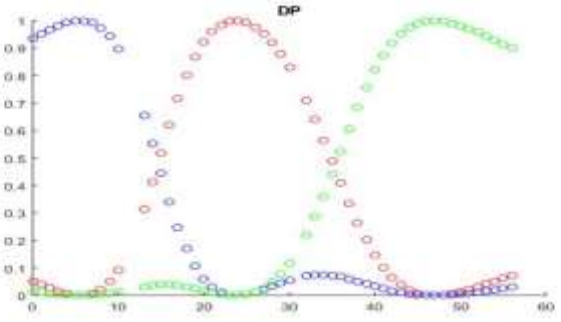


Fig 12 . Drop Packets (DP) Type-1 fuzzy sets.

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.

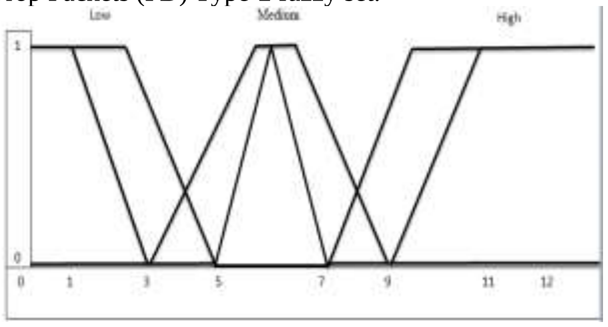


Fig. 13 queue (*q*) Type- 2 fuzzy sets.

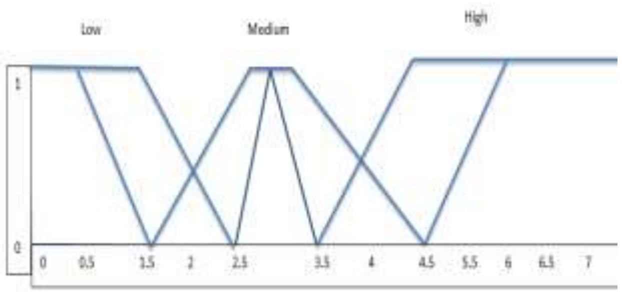


Fig.14 Average Queue Length (avg) Type-2 fuzzy sets.

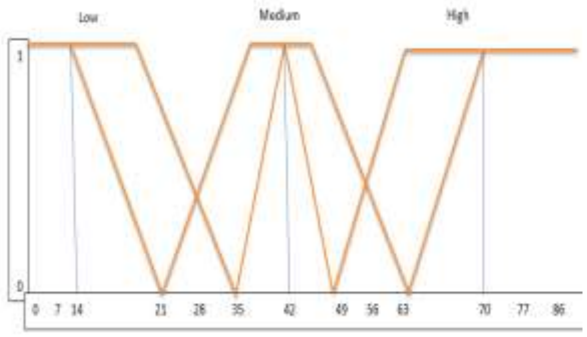


Fig 15. Drop Packets (DP) Type-2 fuzzy sets.

We have also used Fuzzy CMeans Clustering (FCM) [42], with 10% increment in FOU to encompass LMF and UMF, as shown in Fig 16, Fig 17 and Fig 18.

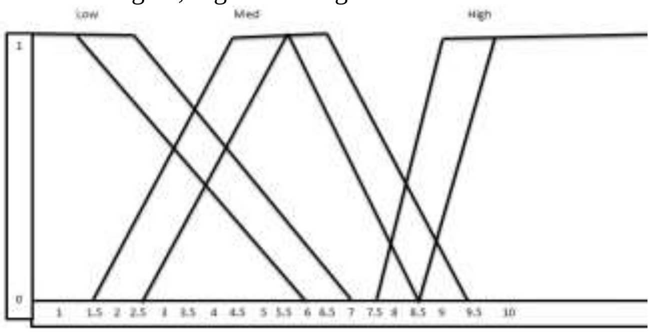


Fig .16 queue (q) Type- 2 fuzzy sets (10% FOU)

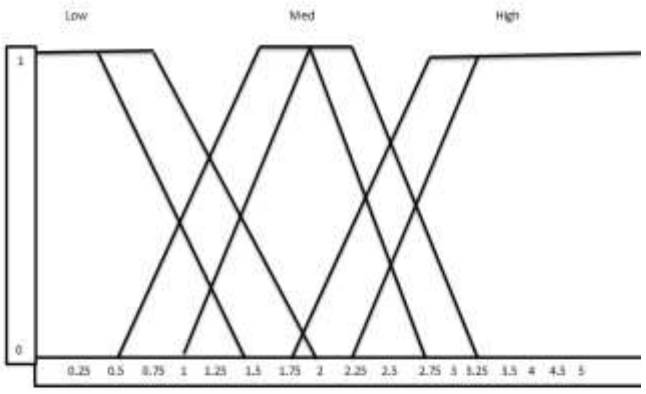


Fig. 17 Average Queue Length (avg) Type-2 fuzzy (10% FOU).

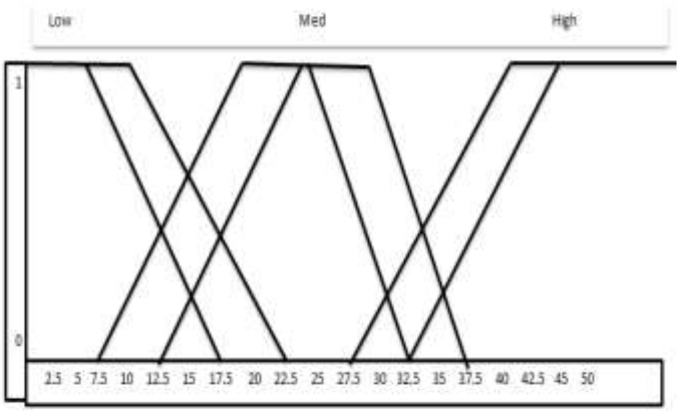


Fig 18. Drop Packets (DP) Type-2 fuzzy sets(10% FOU).

To implement the BB-BC for Type-1 & Type-2 fuzzy logic system, 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 [36]:

Step1: In Type -1 FLS

- Computed the membership values $\mu_{A_s^z}$ 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. (11)

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 $\bar{\mu}^{cg}_{\tilde{A}_s^z}(x_s^{(t)})$ and lower membership values $\underline{\mu}^{cg}_{\tilde{A}_s^z}(x_s^{(t)})$ for each fuzzy set z=1..Z, and for each input variable s=1..n . Find z* ϵ {1...Z} such that [35]:

$$\mu^{cg}_{\tilde{A}_s^{z*}}\left(x_s^{(t)}\right) \geq \mu^{cg}_{\tilde{A}_s^z} \quad for\ all\ z = 1 \dots z_i \qquad (12)$$

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

$$\mu^{cg}_{\tilde{A}_s^z}\left(x_s^{(t)}\right) = \frac{1}{2}\left[\bar{\mu}^{cg}_{\tilde{A}_s^z}\left(x_s^{(t)}\right) + \underline{\mu}^{cg}_{\tilde{A}_s^z}\left(x_s^{(t)}\right)\right] \qquad (13)$$

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]:

$$w_i^{(t)} = \prod_{s=1}^{n} \mu_{A_s^z}^{cg} \left(x_s^{(t)}\right) \qquad (14)$$

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.

$$IF\ x_1\ is\ \tilde{A}_s^z\ ...\ and\ \ x_n\ \ is\ \tilde{A}_s^z\ \ THEN\ y\ is\ centered\ at\ y^{t_u^z} \qquad (15)$$

- We repeated for all the t data points from 1 to N to obtain N data generated rules as described in [37].

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.

VI. EXPERIMENTS AND RESULTS

To implement the BB-BC for Type-1 and Type-2 fuzzy logic system, JAVA programming language has been used. This application can be configured with any dataset and any number of fuzzy sets. By using our previous proposed system [4], to evaluate and compared with in the Type-2 FLS, Type-1 FLS, and WRED method 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 [1], [4].

The first scenario assumed heavy congestion, in which the packet arrival rates was 0.92, whereas the departure rate was only 0.5. Fig 19 shows the Packet Losses (PL) and Packet Dropping (PD) ratios of BB-BC Type-1, BB-BC Type-2, Type-2 FLS, Type-1 FLS, and WRED for the first scenario. By comparison, BB-BC Type-2 displays the lowest packer loss of 10.3% followed by BB-BC Type-1 having PL of 11.4% and Type-2 FLS, Type-1FLS and WRED [4] are having PL of 12%, 16%, 34% respectively. However, BB-BC Type-2 drops 35% followed by BB-BC Type-1 PD of 33.6% and Type-2 FLS , Type-1 FLS and WRED are having PD of 32% , 28%,20% respectively [4].

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 20 shows the packet loss and packet dropping ratios of BB-BC Type-1, BB-BC Type-2, Type-2 FLS, Type-1 FLS, and WRED for this scenario. BB-BC Type-2 shows a PD of 6.2% followed by BB-BC Type-1 having PD=5.5% and Type-2 FLS,Type-1 FLS and WRED are having PD of 4%, 3.4%, 2% respectively [4] . Furthermore the BB-BC Type-2 shows a PL of 1.3% followed by BB-BC Type-1 having PD=1.6% and Type-2 FLS ,Type-1 FLS and WRED are having PL of 2%, 3% 4.4% respectively [4] .

Finally the third scenario assumed no congestion at all, in which the packet arrival and departure rates were 0.35 and 0.5, respectively. Fig 21 shows the packet loss and packet dropping ratios of BB-BC Type-1, BB-BC Type-2, Type-2 FLS, Type-1 FLS, and WRED, in this scenario 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.

$$RMSE = \sqrt{\frac{1}{N}\,[y_i - \bar{y}_i]^2} \qquad (16)$$

Where N is the total number of testing dataset, y is the actual output from the testing dataset, and $\bar{y}$ is the actual output of the FLS.

The RMSE result of the BB-BC Type-2 as shown in Table 4 is better than BB-BC Type-1, Type-2 FLS, and Type-1 FLS and WRED method.

TABLE 4
TESTING DATA RESULTS

Method	*FOU*	*RMSE*
WRED	-	1.684
Type-1	Zero	0.970
Type-2	Empirical	0.935
Type-2	10%	0.829
BB-BC Type-1	zero	0.797
BB-BC Type-2	10%	0.733

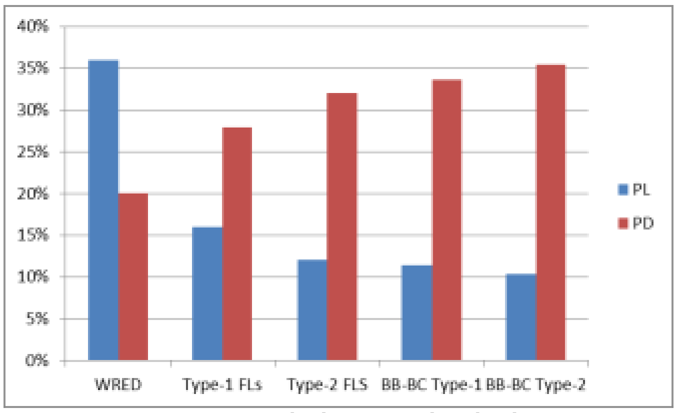


Fig 19. Heavy congestion: packet loss ratio and packet dropping ratio.

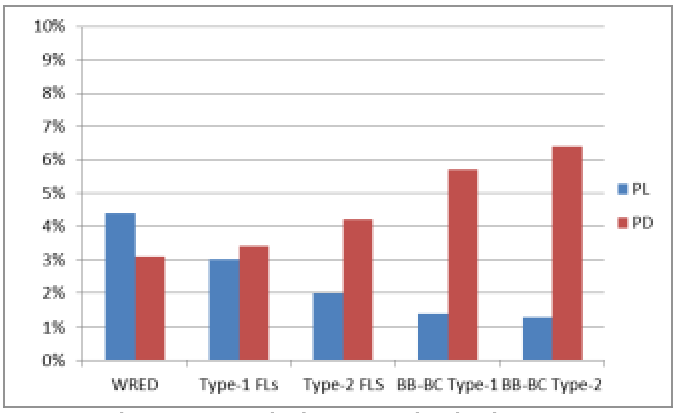


Fig 20. Light congestion:packet loss ratio, and packet dropping ratio.

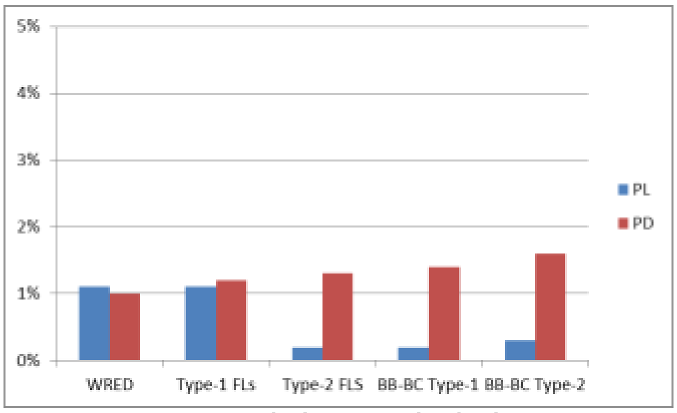


Fig 21. No congestion: packet loss ratio, and packet dropping ratio

VII. CONCLUSIONS

In this paper, we have used the Big Bang Big Crunch (BB-BC) algorithm for optimizing the Type-2 FLS to provide best choices for the Type-2 fuzzy sets parameters, as well as optimizing the rule base of the Type-2 FLS. By using our previous proposed system [4], to evaluate and compared with in the Type-2 FLS, Type-1 FLS, and WRED method.

The results have been compared using RMSE, which showed that BB-BC Type-2 FLS was better than results of our previous proposed system (Type-2 FLS, Type-1 FLS and WRED method) [4] , because the packets loss decreased at by up to 10% and increased the packets dropping by up to 35% . This is because the BB-BC Type-2 FLS predicts the congestion at an earlier stage than other methods under heavy congestion.

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] 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.

[4] M. S. E. Alhassan and H. Hagras, "A Congestion Control Approach Based on Weighted Random Early Detection and Type-2 Fuzzy Logic System." International Journal of Computer Science Trends and Technology (IJCST) 8, no. 4, pp.83-94, 2020.

[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] 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.

[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] 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.

[31] A.Bilgin, H. Hagras, A. Malibari, M. Alhaddad, D. Alghazawi, "Towards a linear general type-2 fuzzy logic based approach for computing with words", Soft Computing,pp.2203-2222, May 2013.

[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] D. Bernardo, H. Hagras, E. Tsang, "A genetic type-2 fuzzy logic based system for the generation of summarised linguistic predictive models for financial applications", Soft Computing, Vo.17, No. 12, pp. 2185-2201, August 2013.

[37] B. Yao, H. Hagras, M. Alhaddad, D. Alghazawi "A fuzzy logic-based system for the automation of human behaviour recognition using machine vision in intelligent environments", Soft Computing, Vol. 19, No.2 , pp.499- 506, April 2014.

[38] A. Sakalli, T. Kumbasar, E. Yesil, H. Hagras, "Analysis of the performances of type-1, self-tuning type-1 and interval type-2 fuzzy PID controllers on the Magnetic Levitation system". Proceedings of the 2014 IEEE International Conference on Fuzzy Systems, Beijing, China, July 2014.

[39] 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.

[40] 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.

[41] 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.

[42] 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.

[43] 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 SConference, pp:29-34, 2018.

[44] 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.

[45] 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.

[46] 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.

[47] 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.

[48] 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.

[49] Erol, Osman K., and Ibrahim Eksin. "A new optimization method: big bang–big crunch." Advances in Engineering Software 37, No. 2, pp. 106-111, 2006.

[50] 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

[51] Chimatapu, Ravikiran, Hani Hagras, Andrew Starkey, and Gilbert Owusu. "A big-bang big-crunch type-2 fuzzy logic system for generating interpretable models in workforce optimization." In 2018 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 1-8. IEEE, 2018.

[52] Saeed, Saeed Khalil, and Hani Hagras. "A Big Bang–Big Crunch Type-2 Fuzzy Logic Based System for Fraud-Detection in the Sudanese Financial Sector." Journal of Engineering and Computer Science (JECS) 21, no. 3, pp. 16-28, 2020.