ISSN 2277-3061

Congestion Control in Computer Networks with a New Hybrid Intelligent Algorithm

[1]Fatemeh Moosakhah, [2]Amir Massoud Bidgoli
[1]Islamic Azad University-Tehran North Branch
[2]Assistant Professor, B.Sc., M.Sc., Ph.D. (Manchester University), MIEEE
Islamic Azad University-Tehran North Branch

Abstract

With invention of computer networks, transferring data from one computer to another became possible, but as the number of computers that transfer data to each other increased and common communication channel bandwidth among them in a network limited, has led to a phenomenon called congestion, so that some of data packets would be dropped and never arrive to destination. Different algorithms have been proposed for overcoming congestion. These are divided into two general groups: 1- flow based algorithms and 2- class based algorithms. In present study, using class based algorithm with optimization of its control by fuzzy logic and new Cuckoo algorithm, we increased the number of packets that reach to destination and reduced the number of dropped packets considerably during congestion. Simulation results indicate a great improvement of efficiency.

Keywords:

Congestion; Flow- based; Class- based; Fuzzy control; Cuckoo optimization algorithm.

Council for Innovative Research
Peer Review Research Publishing System
Journal: INTERNATION JOURNAL OF COMPUTERS AND TECHNOLOGY
Vol. 13, No. 8
editorijctonline@gmail.com
www.ijctonline.com, www.cirworld.com

1. Introduction

Various algorithms have been proposed for addressing congestion phenomenon. Queuing algorithm is grouped into two classes: flow- based (Cisco, 2008) and class- based (Cisco, 2008). Flow- based algorithms are able to supply high quality services to one or more flow, because they reserve any resource requires in advance. However, these methods have one major problem and that is the necessity to do required adjustments in advance for each flow. Thus, in large scale, (i.e. when there are thousands or millions of flows) they lose their functionality. Therefore, another simple solution called class-based method is used for presenting high quality services without the need to do prior adjustments or determination of the whole path that can be implemented locally and distinctively in each router. In this part, we use class- based distribution queue that guarantees packets dispatch based on warranted guidance method and decisions concerning packets direction are made by fuzzy logic. In present study, we use class – based algorithm and improve its control through a new combination of fuzzy logic and Cuckoo algorithm that leads to substantial increase in the numbers of dispatched packets arrived to destination, so that the number of dropped packets are considerably reduced during a congestion period. We continue this discussion in following sections: in part two, performed studies concerning addressing congestion are investigated. In part three, we propose our theory and in part four we conduct simulation in order to analyze the results and conclude. Finally, in part 5 we propose future fields for further studies.

2. Conducted studies

2.1 Early identification in congestion (RED[1])

How a router can prevent global synchronization? This pattern known as early identification or early interruption or early disposal is usually referred to early identification. Each router maintains its mean queue length; whenever average length of queue for some of links exceeds a certain limit, that queue is faced with congestion and starts dropping by a special policy.

Randomness of early identification means that instead of waiting until overflow of queue and then stimulating several TCP links toward slow forwarding, a router slowly and randomly disposes datagrams with congestion increase. Early identification algorithm is used for calculating mean queue length and it uses two parameters, Min_{th}[2] threshold of queue length and Max_{th}[3] length of queue. Performance of early identification may be determined by three rules that determine the location of each input datagram as follows:

1- If number of datagrams of current queue is less than Min_{th} , add new datagram to the queue;
2- If the number of datagrams of current queue is more than Max_{th}, drop new datagram;
3- If number of datagrams of current queue is between the two Min_{th} and Max_{th} limits, drop the datagram randomly according to P probability.

This algorithm includes two main stages (Baraden, 1998):

2.1.1 Estimation of the average queue length

Short term increase in the length of queue due to burst traffic and or transient congestion has no effect on effective increase of mean length of queue. A parameter called avg that indicates mean length of queue is calculated by equation (1). In this relation, q represents length of queue once a packet enters and W_q, weight of each queue that is a number between zero and one usually being equal to 0.002.

1-Random Early Detection

2-Minimum threshold

3-Maximum threshold

Avg. = $(1-W_q)$ avg + $W_q q$ (1)

2.1.2 Decision making about packets disposal

Second step in the algorithm of early identification is disposing input packets. The main point in correct performance of early identification is selecting Min_{th} and Max_{th} limits and probability of P disposal. The limit must be big enough to insure that output link leads to upper output. Moreover, since early identification acts similar to tail disruption, when length of queue is more than Max_{th}, this value must be bigger than increase value in the length of queue during sweep time of TCP, i.e. more than Min_{th} (for example, adjusting Max_{th} with a value at least two times bigger than Min_{th}). Otherwise, early identification may induce fluctuations of global synchronization as tail interruption. This algorithm is implemented using two above mentioned thresholds as follows (Jacobson & Floyd, 1993):

```
For each packet arrival
        Calculate the average queue size avg.
        If $min_{th}$ <= avg. <$max_{th}$ calculate probability $P_a$
        With probability $P_a$:
              Mark the arriving packet
        Else if $max_{th}$ <= avg. mark the arriving packet
```

In this algorithm P_a is probability of each input packet that is a function of avg. mean of queue length. Calculation of probability of disposal of packets is the most complicated aspect of early identification. Instead of using a fixed number, a new value of P_a is calculated for each datagram. This value depends on the relationship between the current size of queue and threshold limit. When the length of queue is less than Min_{th}, early identification does not led to disposal of any datagram, so that probability of disposal becomes zero (P =0). When the length of queue is larger, early identification does not lead to disposal of all datagrams, so that probability of disposal becomes 1 (P =1). For values being between the length of queue, probability may be from zero to one in the form of a variable line. Probability is calculated with relation (2); in which P_b represents probability of each packet from zero to Max_{th} and Count Parameter (relation (3)) is the number of entering new packets when average length of queue is between two values of threshold. P_a value is directly proportional to Count (Jacobson & Floyd, 1993):

$$P_b = max_P\,(avg-min_{th})/\,(max_{th}-min_{th}) \qquad (2)$$

$$P_a = P_b/\,(1-count*P_b) \qquad (3)$$

Although linear pattern is the main principle for calculation of early identification probability but a change must be made for preventing sever reaction. In fact, the change is required considering a burst network traffic; because it may induce sever changes in router’s queue. If early identification follows a simple linear pattern, datagrams of the end of queue face with high probability of being disposed in each burst (because when they will arrive, the queue of router will have more entrees). However, a router should not drop datagrams when not required; because with this action a negative effect is suffered by operational throughput of TCP. Therefore, as data bursts are short term lived, it is not wise to dispose datagrams; because the queue never overflows. Of course, early identification could not delay disposal endlessly; because a long term burst would overflow the queue that would lead to tail interruption policy. Some of the problems of early identification are as follows (Safaeezadeh, 2008):

- Threshold limits can hardly be specified;
- Probability of disposal of packets is in the form of a curve;
- RED algorithm is suitable when mean length of queue is in steady state.

2.1.3 Early identification of congestion in distinct services based nodes (dsRED) [4]

RED expanded algorithm exists in distinct services based nodes and is presented by a set of routers located at a single managerial field. If a transmitter request for entering into network for receiving a type of distinct service, it’s sent packets carry the field of services when entering the field, so that some of them receive better services (Tanenbaum, 2008).

Architecture of distinct services provides traffic division for different groups. In this algorithm, there are physically four RED queues as shown in figure (1) in which each RED queue is divided virtually into three queues with different priority and each virtual queue has packets with the same priorities.

4- Differentiated Service RED (dsRED)

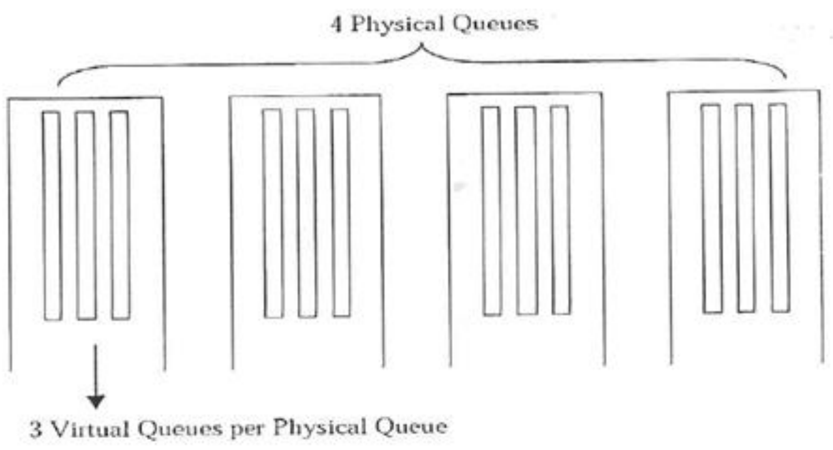


Fig 1: Typical queue of dsRED in distinct services based nodes (Pieda, 2000)

The method used for addressing congestion is as follows:

2.2 Differentiated Services Fuzzy Assured Forward Queuing (dsFAFQ) [5]

In this method, class- based distributed queuing is used based on fuzzy logic. In distributed queuing based on guaranteed guidance in nodes with distinct services, firstly, four guaranteed guidance classes are defined and each class receives its special resources and produced packets are categorized in any desired order in one of the four classes. Based on guaranteed guidance method, the packets are marked after being located in one of the four classes. In this step, three possibilities for packet elimination due to congestion is defined for marking the packets and the packets are marked with three types of packet elimination probability including low, average and high probability. Therefore, with four guaranteed guidance classes and three packet elimination probabilities, 12 classes of distinct services are developed. Then, buffer of a router is divided to two parts: edge and core. These two parts could be considered in two separate routers as well. In the edge, the router is grouped into 4 classes in order to separate output flows, so that if data sent faster by a flow, it would not be able to occupy the whole space of buffer and consequently, other flows would not be faced with resource shortage[6]. As can be seen in figure (2), each guaranteed guidance class is assigned with a physical queue and since three types of elimination probabilities are entered into each queue, for each elimination probability, one virtual queue[7] is assigned. In other words, each physical queue is assigned with three virtual queues.

5- Differentiated Services Fuzzy Assured Forward Queuing

6- Starvation

7-Precedence

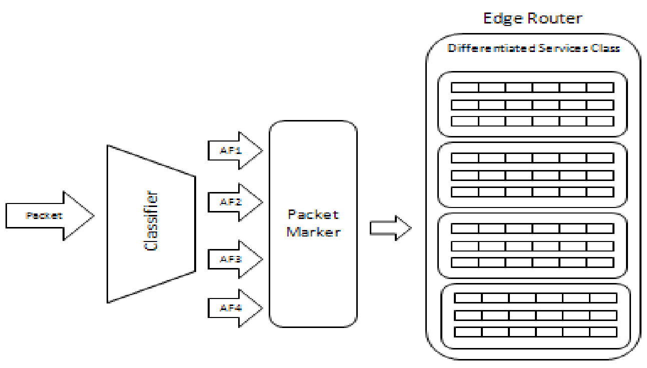


Fig 2: Classification of packets in distinct service based groups

The edge and core communicate with each other by fuzzy controller indicated in figure (3); namely, fuzzy controller checks conditions of existing queues in the core of router in certain times and specifies the percent of packets that must be transferred from the edge to the core of router and the packets are placed in the core of router inside physical and virtual queues; then the packets are sent from the core of router in a guaranteed manner.

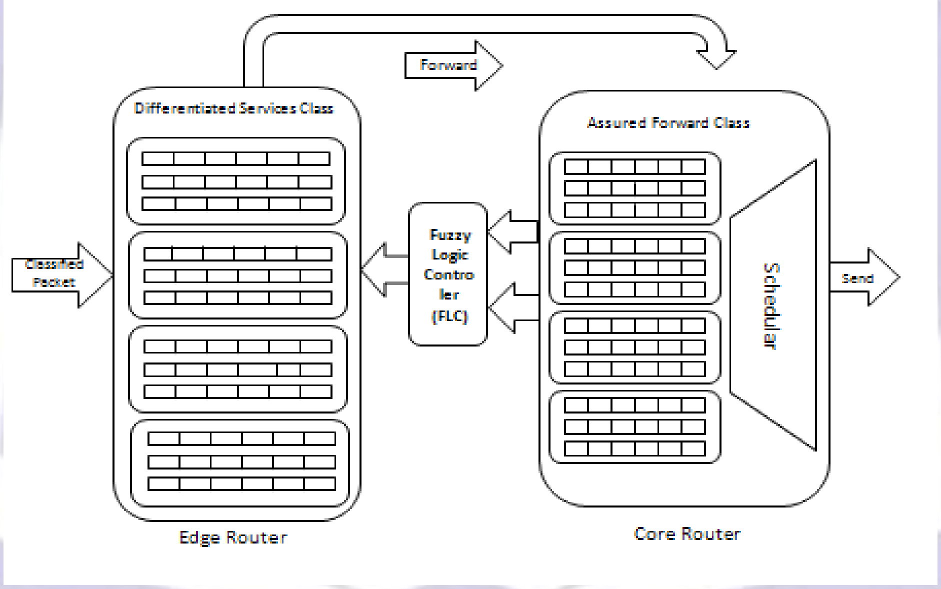


Fig 3: Dividing the space of router buffer in two parts: edge and core and their relationship

Arrival of packets into physical queues with respect to determined class and probability of elimination are determined in the edge of router. After determination of packets transfer percentage by fuzzy controller, the packets are transferred from the edge to the core router and timing is performed for servicing of packets exit. Service provision for different queues is also conducted in a cyclic manner; because if the packets are processed in the order they entered in a queue, an aggressive transmitter may occupy most of the capacity of router and so the quality of services to others will be reduced. When a network is not under congestion, any type of packet with any type of priority may enter the queues in the edge of router and receive services after being transferred to the core, but under congestion period, entering to queues in the core is reduced by fuzzy logic and the packets with high elimination priority are not allowed to enter to the edge of router (Yousefzadeh, 2011).

2.2.1 Fuzzy logic and its application for congestion control

Fundamental idea of fuzzy control is to make use of expert knowledge. In fuzzy science, any event relating to an object is defined as a fuzzy state. Fuzzy control has a deterministic behavior and can be applied to vague and indeterministic problems (Rajasekaran & Vijayalksmi, 2003).

Fuzzy systems are based on rules. The heart of a fuzzy system is a knowledge base consisted of IF-Then rules. A fuzzy If- Then rule is an If-Then statement that the If part is determined by the help of continuous membership functions. In summary, starting point of developing a fuzzy system is obtaining a set of fuzzy If-Then rules from the knowledge of expert people or the knowledge relating to a problem under investigation field. The next step is combining these rules in a unit system. Different fuzzy systems follow different methods and principles for combining these rules.

Fuzzy rules base indicates a set of fuzzy rules and fuzzy inference motor called Mamdani combines these rules into a mapping of fuzzy sets in entrance space of fuzzy sets in output space based on fuzzy logic principles (Rajasekaran & Vijayalksmi, 2003).

A fuzzy maker[8] in entrance converts the variables with real values to a fuzzy set and a fuzzy remover[9] converts a fuzzy set to a variable with real values in output by centroid method. Centroid method performs center of gravity and mean of maxima (MoM) performance on defuzzification by two ways the most known of which is center of gravity method (COG)[10]. In this method, fuzzy output is obtained for discrete values considering occupied space by fuzzy set in the form of equation (4), where x* represents defuzzified output, n is for the number of members, x_i represents members of a fuzzy input member and $\mu(x_i)$ represents membership function of each member (Rajasekaran & Vijayalksmi, 2003).

$$X^* = \frac{\sum_{i=1}^{n} \mathrm{xi}.\mu(\mathrm{xi})}{\sum_{i=1}^{n} \mu(\mathrm{xi})} \qquad (4)$$

8-Fuzzifier

9-Defuzzifier

10-Center of Sum (CoS)

ISSN 2277-3061

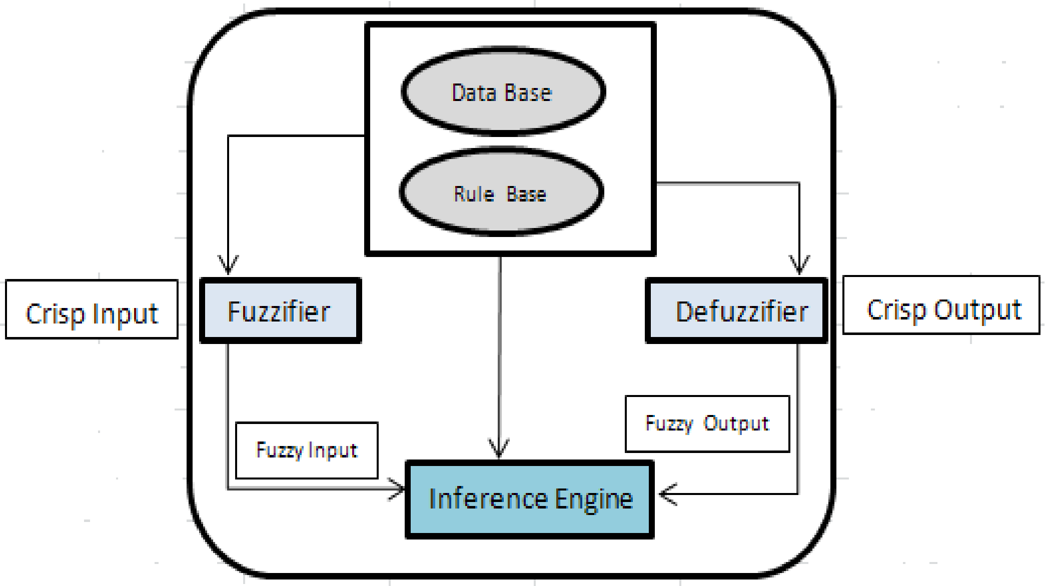


Fig 4: Designing fuzzy controller for guaranteed forward queuing based on fuzzy (Ronald, 1994)

Fuzzy controller in the router controls entrance of packets into the core of router with certain elimination priority such that the router and finally the network are not faced with congestion. It means that if the percentage of input packets entering available queues inside the core of router exceed specified threshold of queue, since corresponding queue is filled rapidly, the controller reduces the number of packets forwarded to that queue until the queue reaches an equilibrium state. In other words, the percentage of packets forwarded to the queue is proportional to current load of queue and mean time waiting in the queue. Controlling entrance of queues to classes with specified priority in the core of router is conducted by fuzzy controller indicated in figure (5). In fuzzy controller, two input variables and one output variable are considered.

Input and output variables are as follows:

-Queue load

It is the number of packets waiting in the queue inside router core to receive services that has seven fuzzy subsets:

Queue Load = {Empty, MEmpty, LHalf, Half, MHalf, Full, VFull}

Indicating Empty, More than Empty, Less than Half, Half, More than Half, Full and Very Full respectively.

-Waiting time

Waiting time is the time period during which a packet remains in the queue in order to receive services (this time does not include the time for performing servicing). For this input there are seven fuzzy subsets:

Waiting Time = {Small, Bsmall, SMed, Med, BMed, Large, VLarge}

Indicating Small, Bigger than Small, Smaller than Medium, Medium, Bigger than Medium, Large and Very Large respectively.

-Number of directed packets (Forwarded Packets)

This is the average number of packets sent to the queues of each class. Number of packets forwarded by fuzzy controller is an output variable for which seven fuzzy subsets is considered:

Forward = {VLow, Low, LNorm, Norm, MNorm, HigH, VHigh}

Indicating Very Low, Low, Less than Normal, Normal, More than Normal, High and Very High respectively (Yousefzadeh, 2011).

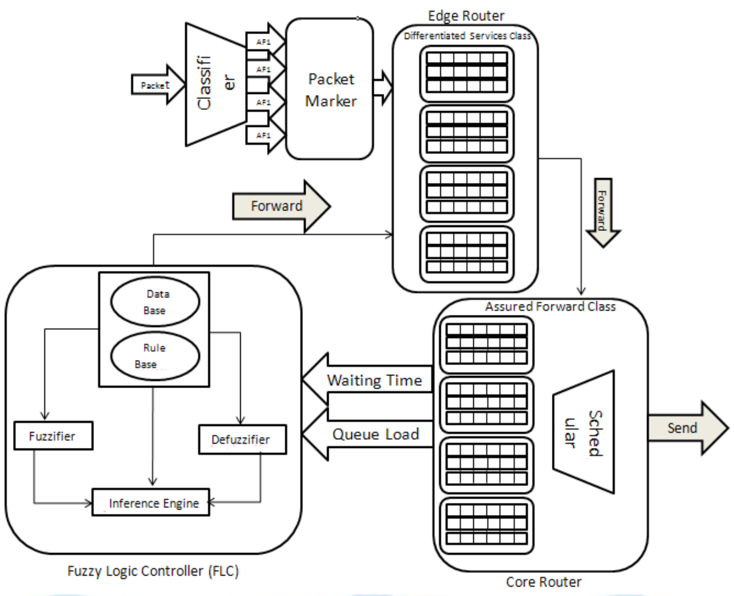
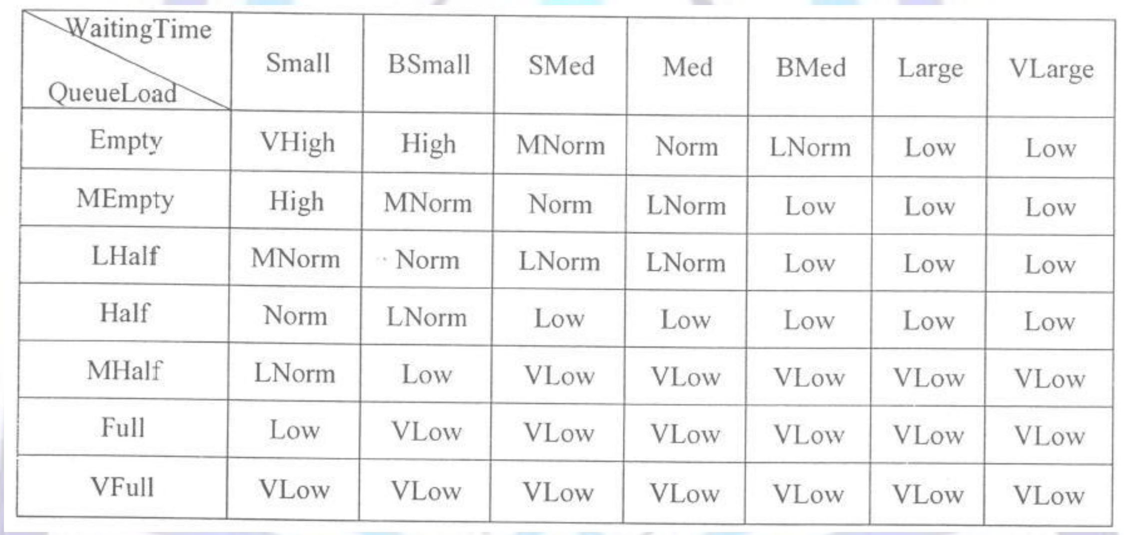


Fig 5: Proposed method in management of congestion

Table1. Database of used rules in proposed method for fuzzy control (Yousefzadeh, 2011)



WaitingTime QueueLoad	Small	BSmall	SMed	Med	BMed	Large	VLarge
Empty	VHigh	High	MNorm	Norm	LNorm	Low	Low
MEmpty	High	MNorm	Norm	LNorm	Low	Low	Low
LHalf	MNorm	Norm	LNorm	LNorm	Low	Low	Low
Half	Norm	LNorm	Low	Low	Low	Low	Low
MHalf	LNorm	Low	VLow	VLow	VLow	VLow	VLow
Full	Low	VLow	VLow	VLow	VLow	VLow	VLow
VFull	VLow	VLow	VLow	VLow	VLow	VLow	VLow

3. Proposed applied method (Fuzzy COA Related Random Early Drop (FCRred))

The new method used for addressing the congestion in network in a combinational and optimized manner is implemented by defining random early elimination (prediction of congestion) by fuzzy logic related to Cuckoo algorithm. As said before, defining fuzzy rules for controlling the congestion of distributed queues based on class is very complicated and it is achieved empirically by experts. Also neural networks applied widely in expert and fuzzy systems may be used for adjusting membership functions and fuzzy rules in knowledge base.

Indeed, we can say that defining fuzzy rules constitutes the heart of distributed queuing system so that if we wish distributed queues show good performance we must enhance their heart; it means that we must define and optimize fuzzy rules such that system output is highly matched to system input. By matching between output and input systems we mean that output accuracy in terms of inputs is in the best possible state; therefore, it is obvious that in the case of non-optimization of fuzzy control system, its performance may have good condition but this good performance is not necessarily the best response. How could we optimize a fuzzy control system, such as controller system of distributed queues congestion? In order to answer this question, we must continue our discussion specifically about congestion controller fuzzy system in distributed queues.

As explained before, congestion controller fuzzy system has two input parameters and one output parameter that specifies the value of output parameters based on input parameters value considering the table1 and similar number of packets are forwarded from the queue of edge router to the router of core. Two input parameters include: 1) length of queue at the time of control; 2) average waiting time during which the packets are waiting in queue and output parameter includes the number of packets that must be forwarded from edge router to core router. For example, if queue length during control is LHalf and average waiting time is Med, then output value is LNormal (table 1). One question emerges: is LNormal the best output for these two inputs? At first glance, it can be said that according to fuzzy rules database that the table 1 is its output, this answer is correct, but concerning being the best or optimized, we cannot claim anything without valid documents. In fact, this table is the heart of fuzzy system in this system and if its parameters are adjusted well, obtained results would be excellent and if it is adjusted badly, the result would be catastrophic. Suppose that VHigh replaces LNormal. Obviously, the result is increased congestion and loss of packets in queue of central router. Each rule in a fuzzy system has an assigned weight to it and it is a presumed weight value of 1. So, the main purpose of present study could be specified here by adjustment of the values of table of fuzzy rules database in an optimized manner (table 1) in distributed queues system so that the question could be answered as: system output is the best response. Now this question is presented: how could we optimize the values of rules table? In order to answer this question, we must say that experiences of knowledgeable persons and/or using optimization algorithms are the only way for this purpose. Since all people don't have the skill for adjustment of the values of this table, application of optimization algorithms may be the only way. There are different optimization algorithms for this purpose: genetic algorithm and Simulated Annealing algorithm. A new method proposed for optimization of fuzzy logic is Cuckoo optimization algorithm (Rajabioun, 2011). This algorithm has currently the highest speed in maximizing or minimizing target function (considered output: number of forwarded packets) from fuzzy controller in network router. For this purpose, we must become familiar with the performance of Cuckoo algorithm.

3.1 Cuckoo optimization algorithm and cuckoo's laying method (COA)

All the birds in the world have similar method for becoming a mother. All of them lay eggs. No bird gives birth to its offspring, but it lays egg and nurtures its chick outside its body. The bigger the eggs, the lower the possibility that female bird carries more than one egg inside its body, because bigger eggs make flying difficult. On the other hand, since the eggs are full of protein for other hunters, the birds need to find a safe place for laying. Finding a safe place for laying and nurturing them until they gain independence is an important issue that all birds have resolved it smartly. They use a sophisticated and complex engineering for this purpose. Most of birds build their nests in unknown place hidden under plants, so that prevent being identified by hunters. Some birds have freed themselves from parenthood and related duties and use a smart way. They are called brood parasites; they never build nest for themselves and instead put their eggs in the nests of other birds and wait to see those birds protect their eggs like theirs. Cuckoo is the most famous brood parasite who is specialized in deceiving. Mother cuckoo destroys one of the eggs of a host mother and puts her egg among other eggs in the nest of host and flies away rapidly. Cuckoos put their eggs inside the nests of different birds and in doing this they consider the color and pattern of eggs existing in each nest to ensure the similarity between her eggs to the eggs in host's nest. Any female cuckoo is specialized for a certain type of bird species. How female cuckoos mimic the eggs of a certain species with high precision is the secret of nature. In this regard, there are birds that recognize cuckoos' eggs in their nest and some of them even throw her egg out of their nest. Some birds even leave their nest and build a new one. In fact, cuckoos constantly improve their imitation method and host birds learn the ways by which they can identify alien eggs.

Most of cuckoo species are non-migrant, but there are various species that do seasonal migration. Some other species migrate periodically across domain of their habitat. For cuckoos living in elevated places, access to food requires them to migrate to warm areas in winter. Cuckoo chicks come out of egg sooner than that of host bird's chicks and grow faster. In most of cases, cuckoo chicks throw the host eggs and /or the chicks of host bird out of the nest. This is purely instinctive and cuckoo chicks have no opportunity to learn such behavior. Cuckoo chick enforces host bird to provide food for it in proportional to its growth and it demands food more often. Cuckoo chick shows its need for food by opening its peak, because open peak means for mother a stimulator for hunger of chicks. Female cuckoos are very skilled in producing eggs similar to the eggs of host bird.

3.2 Details of Cuckoo optimization algorithm

COA is a new method for thorough informed search inspired by a bird called cuckoo. Like other evolutionary algorithms, COA begins its operation with a primary population consisted of cuckoos. This population of cuckoos has a few eggs that would be placed in the nests of some host birds. Some of these eggs that are more similar to the eggs of host bird have higher chance for growing. Other eggs are identified by host bird and are destroyed. The number of grown eggs indicates suitability of the nests of that region.The more the viability of eggs in an area, the more the tendency for choosing that region by cuckoos. Therefore, the situation in which the highest number of eggs is saved is a parameter that COA intends to optimize it. Cuckoos seek for the best region for maximizing the number of eggs saved. As indicated in figure (6) each cuckoo randomly puts a few eggs in the nest of host bird located in their laying territory or ELR[11] (Rajabioun, 2011).

11-Egg Laying Radius (ELR)

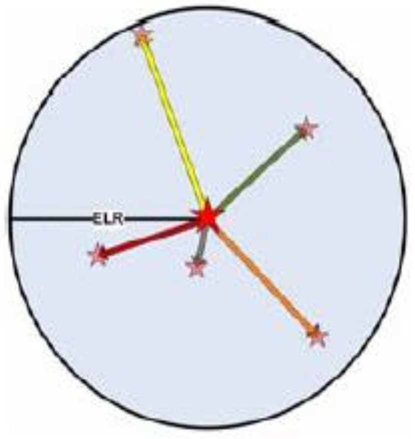


Fig 6: Displaying way of ELR determination and laying eggs in that area

ELR relationship for each cuckoo is determined as follows:

$$ELR = \alpha + \frac{Number\ of\ current\ cuckoo's\ eggs}{Total\ number\ of\ eggs} \times (var_{hi} - var_{low}) \qquad (5)$$

α is a variable that we'll define maximum ELR value with it.

var_{hi} is upper limit of the variables and var_{low} is lower limit of the variables.

When all cuckoos laid their eggs, some of eggs less similar to host bird's eggs are identified and thrown out of the nest. Therefore, after laying P% of all eggs (usually 10%) having lesser profit function are destroyed; the rest of chicks are fed in host nest and develop.

Other interesting point concerning cuckoo chicks is that only one egg has the chance to grow in each nest, because when chicks come out of eggs, throw out the eggs of host bird and if the chicks of host bird come out of egg sooner, cuckoo chick eats the largest portion of food brought by host bird (with its large body that is 3 times bigger than host bird chicks, it is dominant) and after several days host bird chicks die of hunger and only cuckoo chick survives.

When cuckoo chicks grow and get mature, they live for a period in their environments and groups, but when laying time arrives they migrate to better habitats where eggs have higher chance to survive. After forming some groups in different general habitats (searching space of problem) the group having the best situation is selected as target point for other cuckoos for the purpose of migration. When mature cuckoos live in nearly most of regions of the world, it is hard to specify each cuckoo belongs to which group. For solving this problem, the cuckoos are grouped through K-means method (one k between 3 to 5 suffices). When cuckoo groups are formed, mean profit of group is calculated so that relative optimality of habitat of that group is obtained. Then the group with highest average profit (optimality) is selected as target group and other groups move toward it. During migration toward target point, the cuckoos don't go through all paths toward target location; they only travel part of path and they deviate at that way. This type of traveling is evident in figure (7).

As it is indicated in figure (7), each cuckoo travels just λ% of the whole way toward ideal target location and there is a deviation with φ radiant. These two parameters help them to search larger area. λ is a random number between 0 and 1 and φ is a number between $^{\pi}/_{6}$ and $^{-\pi}/_{6}$. When all cuckoos migrated toward target destination and new habitats were specified, each cuckoo lays a few eggs.

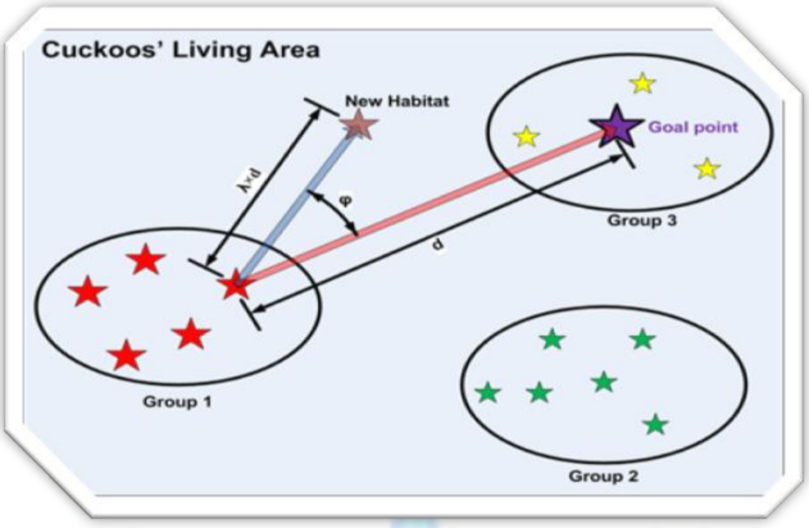


Fig 7: Displaying way of migration of birds (Rajabioun, 2011)

Given the number of eggs of each cuckoo, an ELR is determined for it and then laying begins. Given this fact that there is always a balance between bird populations in nature, a number like Nmax controls and limits the number of cuckoos may live in an environment. This balance is due to food limitation, being hunted by hunters and difficulty in finding suitable nests for eggs. After several repetitions all populations reach to an optimum point with maximum similarity between eggs and the eggs of host birds and also achieve the location of highest food resource. This location has highest total profit and in which minimum number of eggs would be destroyed. Convergence of more than 95% of all cuckoos to one point leads COA to its end.

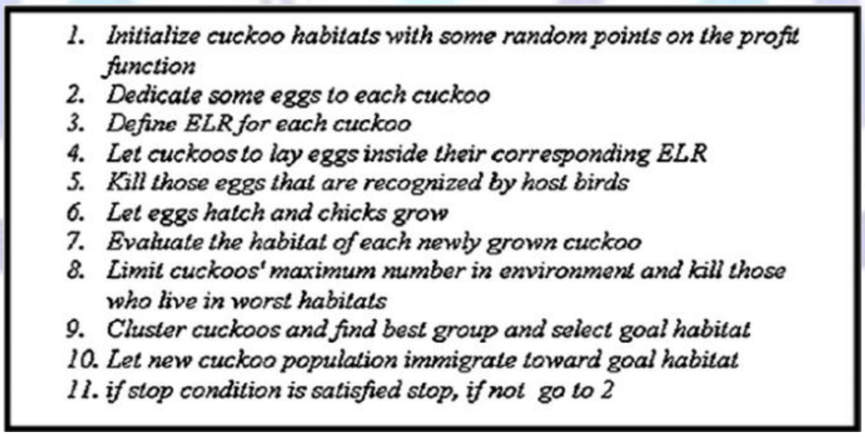
Relation (6) indicates practical migration formula in COA (Rajabioun, 2011):

$$X_{NextHabitat} = X_{CurrentHabitat} + F \times (X_{GoalPoint} - X_{CurrentHabitat}) \qquad (6)$$

F is a parameter that induced deviation.

$X_{GoalPoint}$ is optimal point for the cuckoo to migrate and $X_{CurrentHabitat}$ is current status of the cuckoo birds.

In order to better understand COA stages, in figure (8) semi- code of cuckoo algorithm is presented.



```
1.  Initialize cuckoo habitats with some random points on the profit
    function
2.  Dedicate some eggs to each cuckoo
3.  Define ELR for each cuckoo
4.  Let cuckoos to lay eggs inside their corresponding ELR
5.  Kill those eggs that are recognized by host birds
6.  Let eggs hatch and chicks grow
7.  Evaluate the habitat of each newly grown cuckoo
8.  Limit cuckoos' maximum number in environment and kill those
    who live in worst habitats
9.  Cluster cuckoos and find best group and select goal habitat
10. Let new cuckoo population immigrate toward goal habitat
11. if stop condition is satisfied stop, if not  go to 2
```

Fig 8: semi- code of cuckoo optimization algorithm (Rajabioun, 2011)

3.3 Optimization of fuzzy rules using cuckoo algorithm

In order to optimize the rules of table (1) that includes 49 rules, we used MATLAB tool and cuckoo algorithm. For this purpose, we conducted coding in MATLAB such that current values of rules of table (1) accepted as input and we optimized all its 49 values using cuckoo algorithm; in order that this table is usable in simulation, it is adjusted in the form of a three column file where first column indicates average percent of waiting time of packets in queue and second column represents percent of queue length at the time of control and third column indicates the percent of packets that must be forwarded from edge of router queue to central router. This file is adjusted in 10 steps and number of its inputs (rule lines) is 121. This number of mapping of 49 optimized values by cuckoo algorithm was obtained in interval 0 to 100 through MATLAB instructions. For example, two lines of this file are as follows:

Table2. An example of a three-column file output for use with MATLAB simulation

Waiting time	Queue length	forwarded packets
20	0	75
20	10	60

First line indicates that if average value of waiting time for packet in the queue of central router is 20% of total time and 0% of the queue of central queue is full (the queue is empty), 75% of packets in the queue of edge router must be transferred to the queue of central router. For example, the highest waiting time of a packet in central queue is 10 milliseconds and average waiting time at the time of control is 2 milliseconds and there are 20 packets in edge queue, 15 of its packets would be transferred to central queue in order to be sent to their destination. We expect the performance of optimized rules is better than primary rules indicated in table (1). It means that number of forwarded packets is more and number of dropped packets is less in the edge. For this purpose, we simulated performance of queue and its results confirm this.

4. Simulation and analysis of results

Implementation of guaranteed forwarded queuing based on fuzzy: in figure (9), the way the packets are processed to be forwarded in guarantied manner is indicated. In first stage, data packets are produced based on a certain rate by the source, and then they are classified in one of four classes in terms of their priority. This stage may be performed in the host machine of transmitter or in the first router's entrance to the network. In second stage, after putting the packets in priority classes in terms of the type of packet elimination, they are marked.

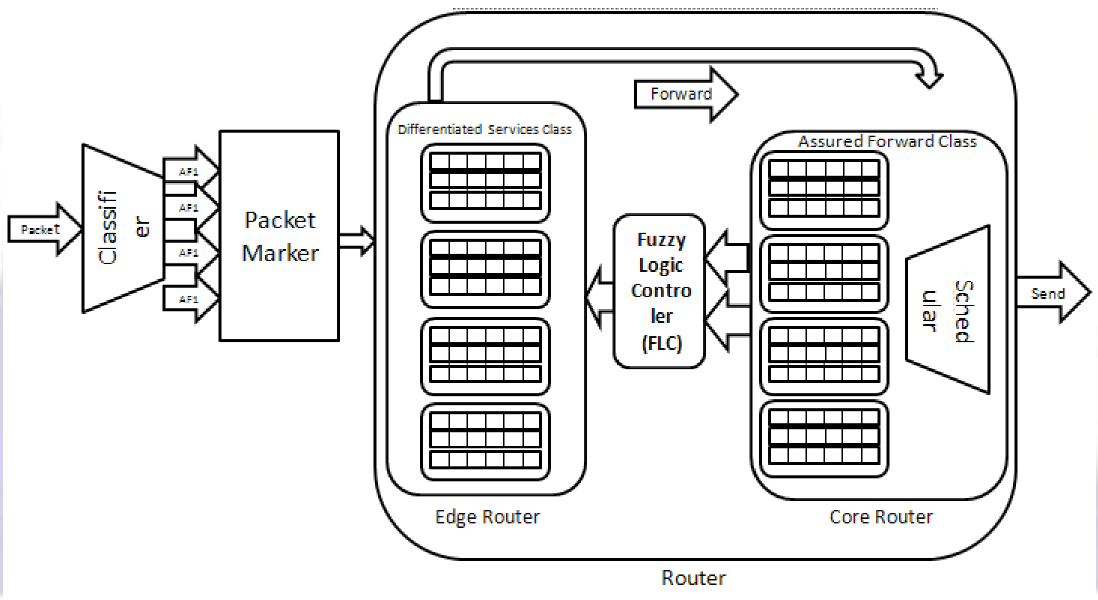


Fig 9: Packets classes based on guaranteed forwarding mechanism for input flow

Considering four classes and three elimination probabilities for each packet, totally 12 classes of differentiated services (DS) are developed.

In figure (9), packet classes based on guaranteed forwarding mechanism for an input flow after dividing the packets transmitter's side output flow, we transit to third stage which is router machine. In this stage, the buffer of router that is divided into core and edge is classified to 4 classes and to each class one physical queue is allocated. As it is evident in figure (9), classified packets enter to the edge of router with determined rate and a fuzzy controller determines the percentage of transferred packets from corresponding queues in the edge to the core that checks conditions of existing queues in this part during sending each packet from core of router by using two parameters of waiting time and current load of queue based on relations (7) & (8) and then the packets are forwarded in the core into physical and virtual queues and packets would be sent from the core of router in guaranteed manner(relation 9).

$$\text{Queue Load} = \text{PKts\_Num (Precl+Prec2+Prec3} \qquad (7)$$

$$\text{Waiting Time} = \text{PKts\_FTC-PKts\_AtE} \qquad (8)$$

Forward = FLC (QueueLoad, WaitingTime) (9)

In relation (7), PKts_Num represents total packets in each physical queue of router and Prec is number of packets in each virtual queue of router.

In relation (8), PKts_FTC represents the number of packets transmitted from the corresponding queue in the core sector and PKts_AtE is the number of packets transmitted from the corresponding queue in the edge sector.

In relation (9), Forward is the number of forwarded packets and FLC is controller of fuzzy logic.

Packets processing and servicing to different classes in the core of router is conducted in Round Robin manner. It means that after locating the packets in queue using a timer that conducts packets exit order on Round Robin basis, servicing or sending the packets to destination is done. In this case, proposed queue formed in router has following characteristics:

1- The packets enter the system using two CBR[12] traffic generators; one with 4Mb/s and the other with 6Mb/s speed on UDP protocol and with Poisson distribution rate.
2- Capacity of physical queues in the edge of router is 10 packets and total capacity of system is 80 packets.
3- Output band width of router for sending the packets set as 7Mb/s and input packets enter router queues through two separate links each with 10Mb band width.
4- Size of all packets is fixed as 1000 bytes and number of total packets produced by source is nearly 100,000 packets.
5- Each queue in the edge of router has a maximum threshold that is equal to 20 packets. However two states are present in this system:

A) If number of packets of a queue is less than maximum threshold, the packets enter related queue with any type of elimination priority.

B) If number of packets of a queue is more than maximum threshold, only LD[13] packets enter related queue until it exceeds capacity of queue[14]. In this case, congestion tail drop policy or Tail Drop.

12-Constant Bit Rate Traffic

13-Low Drop Precedence (LD)

14-Overflow

In order to assess the performance of optimized rules, stimulation scenario described in part two was implemented using optimized rules; for this purpose we used NS-2 stimulation software, version 2,34. This software is a computer network stimulator based on discrete and object- oriented events stimulation method. It is one of the useful tools for simulating local and extended computer networks. Network structure of figure (10) was implemented for comparing primitive rules performance to optimized rules performance:

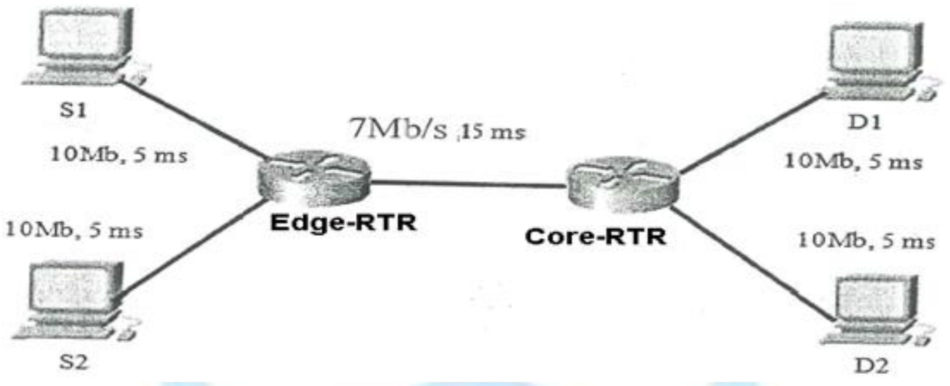


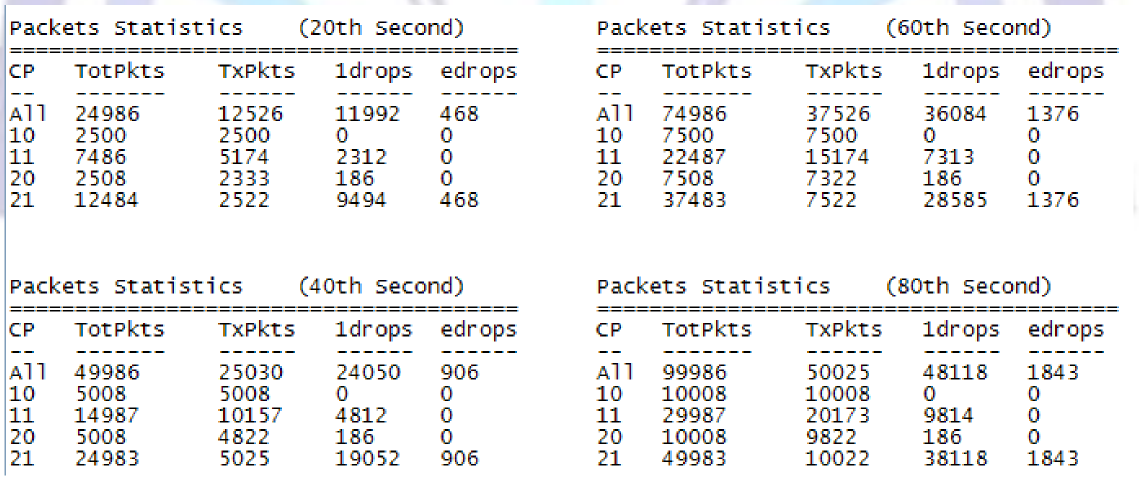
Fig 10: Experimental network implemented for testing queue congestion control with fuzzy system and optimized rules using cuckoo's algorithm.

According to previous scenario, we sampled the performance of queue in 20th, 40th, 60th and 80th seconds and total number of packets reached to queue and number of forwarded and eliminated packets were determined. In must be stated that titles of obtained report columns are as follows:

1) TotPKts; packets developed in system
2) EnPKts: packets forwarded to queues
3) TxPkts: sent packets
4) Idrops: eliminated packets

Moreover total produced packets were forwarded to one of four physical queues indicated by 1-4 numbers and to one of three virtual queues indicated by 0-2 numbers, which would arrive to destination if being not eliminated. The report was prepared in terms of virtual and real queues separation. For example, by queue 42 we mean physical queue No. 4 and virtual queue No.3. Duration on simulation is 85 seconds. Obtained results are presented in two states: using primitive rules and optimized rules using cuckoo algorithm. In addition, these results are also compared with the old rules (dsRED).

Table3. Control with dsRED rules (Old Method)



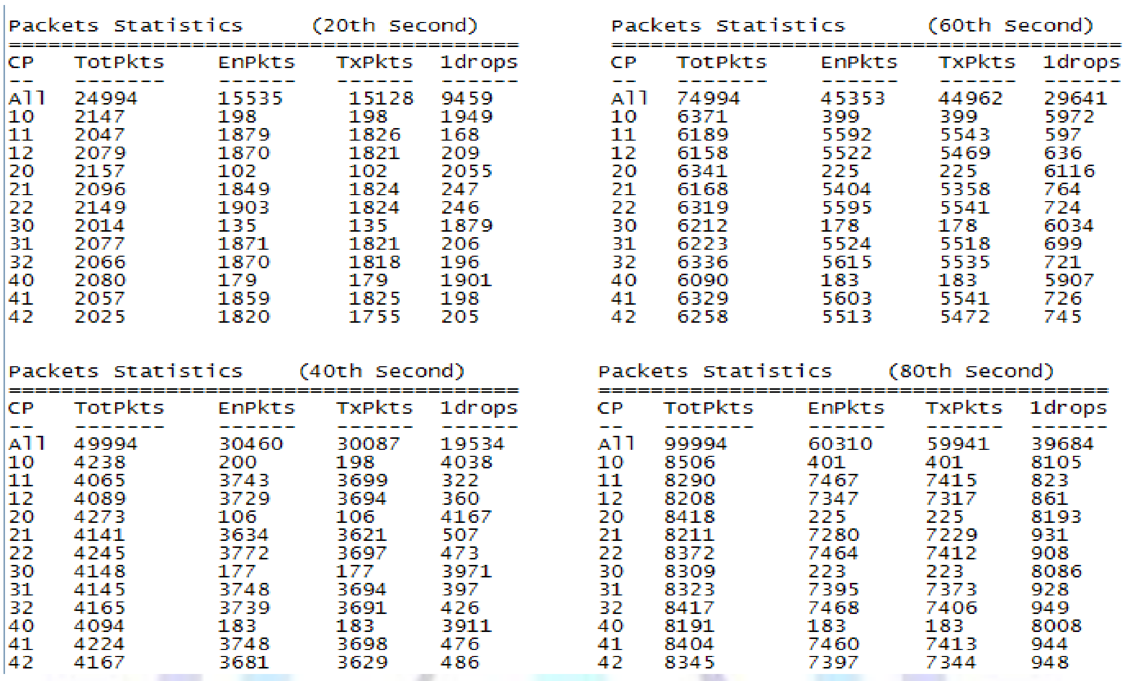
```
Packets Statistics     (20th Second)
=========================================
CP    TotPkts      TxPkts    ldrops   edrops
--    -------      ------    ------   ------
All   24986        12526     11992    468
10    2500         2500      0        0
11    7486         5174      2312     0
20    2508         2333      186      0
21    12484        2522      9494     468
```

```
Packets Statistics     (60th Second)
=========================================
CP    TotPkts      TxPkts    ldrops   edrops
--    -------      ------    ------   ------
All   74986        37526     36084    1376
10    7500         7500      0        0
11    22487        15174     7313     0
20    7508         7322      186      0
21    37483        7522      28585    1376
```

```
Packets Statistics     (40th Second)
=========================================
CP    TotPkts      TxPkts    ldrops   edrops
--    -------      ------    ------   ------
All   49986        25030     24050    906
10    5008         5008      0        0
11    14987        10157     4812     0
20    5008         4822      186      0
21    24983        5025      19052    906
```

```
Packets Statistics     (80th Second)
=========================================
CP    TotPkts      TxPkts    ldrops   edrops
--    -------      ------    ------   ------
All   99986        50025     48118    1843
10    10008        10008     0        0
11    29987        20173     9814     0
20    10008        9822      186      0
21    49983        10022     38118    1843
```

Table4. First state: control with primitive rules (dsFAFQ)



```
Packets Statistics      (20th Second)
==========================================
CP    TotPkts        EnPkts     TxPkts   1drops
--    -------        ------     ------   ------
All   24994          15535       15128   9459
10    2147           198         198     1949
11    2047           1879        1826    168
12    2079           1870        1821    209
20    2157           102         102     2055
21    2096           1849        1824    247
22    2149           1903        1824    246
30    2014           135         135     1879
31    2077           1871        1821    206
32    2066           1870        1818    196
40    2080           179         179     1901
41    2057           1859        1825    198
42    2025           1820        1755    205
```

```
Packets Statistics      (60th Second)
==========================================
CP    TotPkts        EnPkts     TxPkts   1drops
--    -------        ------     ------   ------
All   74994          45353      44962    29641
10    6371           399        399      5972
11    6189           5592       5543     597
12    6158           5522       5469     636
20    6341           225        225      6116
21    6168           5404       5358     764
22    6319           5595       5541     724
30    6212           178        178      6034
31    6223           5524       5518     699
32    6336           5615       5535     721
40    6090           183        183      5907
41    6329           5603       5541     726
42    6258           5513       5472     745
```

```
Packets Statistics     (40th Second)
==========================================
CP    TotPkts        EnPkts     TxPkts   1drops
--    -------        ------     ------   ------
All   49994          30460      30087    19534
10    4238           200        198      4038
11    4065           3743       3699     322
12    4089           3729       3694     360
20    4273           106        106      4167
21    4141           3634       3621     507
22    4245           3772       3697     473
30    4148           177        177      3971
31    4145           3748       3694     397
32    4165           3739       3691     426
40    4094           183        183      3911
41    4224           3748       3698     476
42    4167           3681       3629     486
```

```
Packets Statistics     (80th Second)
==========================================
CP    TotPkts        EnPkts     TxPkts   1drops
--    -------        ------     ------   ------
All   99994          60310      59941    39684
10    8506           401        401      8105
11    8290           7467       7415     823
12    8208           7347       7317     861
20    8418           225        225      8193
21    8211           7280       7229     931
22    8372           7464       7412     908
30    8309           223        223      8086
31    8323           7395       7373     928
32    8417           7468       7406     949
40    8191           183        183      8008
41    8404           7460       7413     944
42    8345           7397       7344     948
```

Table5. Second state: control with optimized rules in 20th seconds (dsFCRred)

```
FCR Packets Statistics

========================================

CP TotPkts   EnPkts   TxPkts   Idrops

--  -------   ------   ------   ------

All    24994    24994    17613     7381

  10     2015     2015     2014        1

11     2014     2014     1158      856

12     2095     2095     1227      868

  20     2088     2088     2088        0

21     2112     2112     1172      940

22     2142     2142     1144      998

  30     2143     2143     2143        0

31     2040     2040     1136      904

32     2064     2064     1127      937

  40     2125     2125     2122        3

41     2014     2014     1098      916

42     2142     2142     1184      958
```

Table6. Second state: control with optimized rules in 40th seconds (dsFCRred)

```
         FCR Packets Statistics
=========================================
  CP TotPkts   EnPkts   TxPkts   ldrops
  --  -------   ------   ------   ------
 All   49994    49994    35093    14901
  10    4113     4113     4112        1
  11    4148     4148     2321     1827
  12    4115     4115     2340     1775
  20    4166     4166     4166        0
  21    4156     4156     2322     1834
  22    4160     4160     2278     1882
  30    4274     4274     4272        2
  31    4134     4134     2275     1859
  32    4159     4159     2224     1935
  40    4253     4253     4246        7
  41    4068     4068     2216     1852
  42    4248     4248     2321     1927
```

Table7. Second state: control with optimized rules in 60th seconds (dsFCRred)

```
         FCR Packets Statistics
=========================================
  CP TotPkts   EnPkts   TxPkts   ldrops
  --  -------   ------   ------   ------
 All   74994    74994    52603    22391
  10    6263     6263     6261        2
  11    6190     6190     3407     2783
  12    6220     6220     3486     2734
  20    6233     6233     6231        2
  21    6244     6244     3465     2779
  22    6221     6221     3445     2776
  30    6308     6308     6305        3
  31    6214     6214     3455     2759
  32    6256     6256     3395     2861
  40    6357     6357     6349        8
  41    6126     6126     3300     2826
  42    6362     6362     3504     2858
```

Table8. Second state: control with optimized rules in 80th seconds (dsFCRred)

```
FCR Packets Statistics
=======================================
CP TotPkts   EnPkts   TxPkts   ldrops
--  -------   ------   ------   ------
All    99994    99994    70103    29891
  10     8372     8372     8370        2
11     8249     8249     4514     3735
12     8386     8386     4642     3744
  20     8310     8310     8307        3
21     8287     8287     4593     3694
22     8295     8295     4627     3668
  30     8427     8427     8424        3
31     8297     8297     4571     3726
32     8305     8305     4524     3781
  40     8429     8429     8421        8
41     8221     8221     4461     3760
 42  8416     8416       4649      3767
```

5. Conclusion

Results of current and previous simulation are presented in tables (2) and (3). For better understanding, the statistics are presented in two tables, so that comparison in both states is very easy. Tables consist of four columns with following titles and descriptions:

1) Time (Sec): duration of stimulation in terms of seconds.
2) TotPkts: total number of packets produced in system in each time period.
3) FGenR- TxPkts: packets sent with primitive rules.
4) FOpR-TxPKts: packets sent with optimized rules.
5) FGenR-ldrops: packets dropped with primitive rules.
6) FOpR-ldrops: packets dropped with optimized rules.
7) FdsRedR-TxPkts: Packets sent with old rules (dsRED).
8) FdsRedR-ldrops: Packets dropped with old rules (dsRED).

Table9. Total number of sent packets

Time(Sec)	FdsRed-TxPkts	FGenR-TxPkts	FOpR-TxPkts(New Method)
0	0	0	0
20	12526	15128	17613
40	25030	30087	35093
60	37526	44962	52603
80	50025	59941	70103

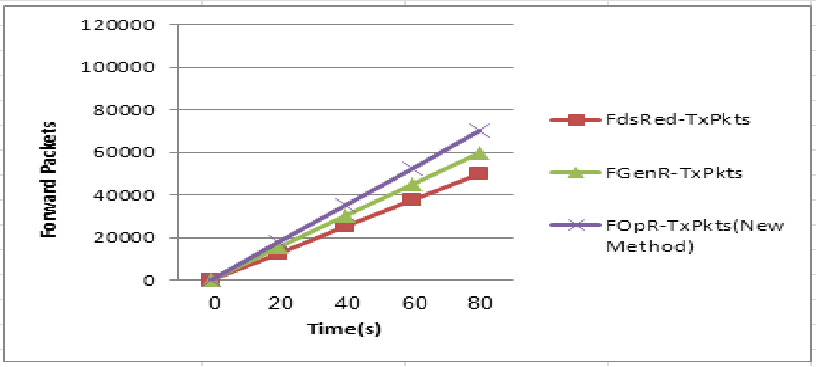


Fig 11: Diagram of total number of sent packets

In figure (11) we see that the line of sent packets in optimized state (dsFCRred) is above the line of sent packets in normal state (dsFAFQ) and (dsRED) methods and this means that number of sent packets in optimized state increased. Given results obtained from simulation we conclude that with optimization of rules of fuzzy controller system, performance of system becomes better, so that fuzzy controller system compared to the system with primitive rules (dsFAFQ) in sending the packets showed 10.2% increase and it increased 20.1% compared to older method dsRED.

Table10. Total number of dropped packets

Time(Sec)	FdsRedR-ldrops	FGenR-ldrops	FOpR-ldrops(New Method)
0	0	0	0
20	11992	9459	7381
40	24050	19534	14901
60	36084	29641	22391
80	48118	39684	29891

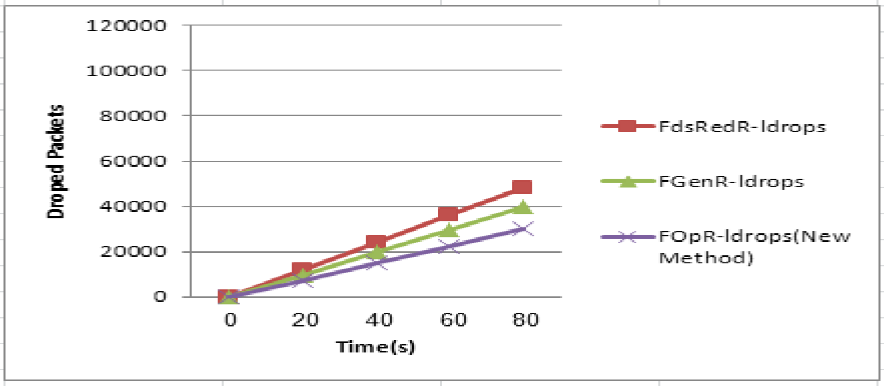


Fig 12: Diagram of total number of dropped packets

In figure (12) we see that the line of dropped packets in optimized state (dsFCRred) is lower than that of sent packets in normal state (dsFAFQ) and (dsRED) methods and this imply that number of dropped packets in optimized state compared to primitive rules (dsFAFQ) in dropping the packets are reduced by 9.8% and it reduced compared to older method (dsRED) by 18.2%.

6. Future propositions

Given significant improvement in system performance, studying the feasibility of practical implementation of this method may be conducted as a research work by other researchers and if it proves cost effective, it may be applied by producers of equipment. Also, by improving fuzzy rules by experts, efficacy of algorithm may be enhanced.

Applying neural networks for adjusting membership functions and fuzzy rules in knowledge base of fuzzy controller may be an alternative for congestion control.

References

1. A. Lotfi Zadeh, C. Berkly, 1995, Fuzzy Logic Toolbox For use with Mat lab, Copyright 1995-1999 by The Math Works, Inc.
2. A. Yousef Zadeh, 2011, Congestion Control in Computer Networks with an Intelligent Algorithm, M.Sc. Thesis, Islamic Azad University Science and Research Branch-Khozestan.
3. A. S. Tanenbaum, 2008, Interpreter: H. Pedram, Computer Networks, 13nd Ed.
4. A.M. Bidgoli, A. Yousef Zadeh Bahri, 2011**,** "Differentiated Services Fuzzy Assured Forward Queuing for Congestion Control in Intermediate Routers", WORLCOMP2011 Conference, COMP, July 18-21, Las Vegas, Nevada,USA, pp 145-152.
5. B. Braden, D. Clark, J. Crowcraft, B. Davie, S. Deering, S. Floyd, 1998, Recommendations on Queue Management and Congestion Avoidance in Internet, IETF RFC230.
6. B. Safaeezadeh, 2008, Congestion Control in Computer Networks with an Intelligent Algorithm, M.Sc. Thesis, Islamic Azad University Science and Research Branch-Tehran.
7. B. Safaiezadeh, A. Rahmani, E. Mahdipour, "A New Fuzzy Congestion Control Algorithm in Computer Networks, International Conference on Future Computer and Communication, 2009.
8. B. A. Forouzan, 2007, Data Communication and Networking, 4nd ed., McGrew-Hill Forouzan network Series Copyright by the McGrew-Hill Companies, Inc.
9. I. Tabash, M. Mamum & A. Nagi, "A Fuzzy Logic Based Network Congestion Control Using Active Queue Management Technique", Journal of Scientific Research, 2010.
10. Payne, B. Robert, the Cuckoos, Oxford University Press, 2005.
11. P. Peida, J. Ethridge, M. Baines F. Shallwani, 2000, A Network Simulator Differentiated Services Implementation, Open IP, Nortel Networks.
12. R. Ronald, P. Yager & Dimitar & R. Filev, 1994, Essentials of Fuzzy Modeling and Control, New York, NY: John Wiley.
13. R. Rajabioun, "Cuckoo Optimization Algorithm", In: Applied Soft Computing journal, Vol. 11, pp. 5508-5518, 2011.

14. S. Floyd, V. Jacobson, 1993, Random Early Detection Gateways for Congestion Avoidance, IEEE/ACM Transactions on Networking.
15. S. Rajasekaran, G. Vijayalksmi, 2003, Neural Network, Fuzzy Logic, and Genetic Algorithms: Synthesis and Application, Prentice-Hall of India Private Limited, New Delhi.
16. S. Cisco, Congestion Management Overview, Cisco IOS Quality of Service Solutions Configuration Guide, 2008.
17. T. Munakata, "Fundamentals of the New Artificial Intelligence, Neural, Evolutionary, Fuzzy and More”, Springer, 2008.
18. W. Zhang, B. Wu, W. Liu, "Anti-Congestion Fuzzy Algorithm for Traffic Control of a Class of Traffic Networks", In: IEEE International Conference on Granular Computing, 2007.
19. Yang, X.S. & Deb, S., "Cuckoo search via Lévy Flights", In: World Congress on Nature & Biologically Inspired Computing (NaBIC 2009). IEEE Publications, pp, 210-214, 2009.

Author’ biography with Photo

Fatemeh Moosakhah was born in Tehran, Iran February 13th 1986. She received her B.S. degree in Computer Engineering from the Islamic Azad University Tehran North Branch and the M.S. degree in Computer Engineering software from Islamic Azad University Tehran North Branch in Iran, in 2011 and 2014, respectively. Her research interests include computer networks and evolutionary algorithms.

Amir Massoud Bidgoli is a professor of Computer Engineering in the Department of computer engineering, Islamic Azad University, Tehran North Branch. He received his B.Sc. degree in Electronics Engineering from Lancaster University (UK) and the M.Sc. degree in Computer & Control Engineering from Salford University (UK), in 1987 and 1989, respectively. He received his Ph.D. in Computer Science from Salford & Manchester University (UK), in 1996. He is a member of IEEE and is indexed in IEEE Explorer. His research interests include computer networks & evolutionary algorithms, computer architecture, networking, distributed systems and image processing. He has over eighty publications in journals and ISI conferences.