JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 37, 1313-1326 (2021)
DOI: 10.6688/JISE.202111_37(6).0006

Automation Based Active Queue Management using Dynamic Genetic Algorithm in Real-Time Application

MAJID HAMID ALI+ AND SERKAN ÖZTÜRK
Computer Engineering Department
Erciyes University
Kayseri, 38039 Turkey
E-mail: hammajid80@gmail.com; majidalbayti80@yahoo.com; serkan@erciyes.edu.tr

Nowadays, there has been significant interest on designing coordinated automation systems for wide range of industrial applications. Automation brings the dynamic control in real time communication systems also. For real time communications, recently the TCP/IP gained the significant attentions in usage of best-effort networks. The focus is on assured the Quality of Service (QoS) while handling real-time communications through IP networks. QoS of such communications based various components of networking like the Active Queue Management (AQM). The AQM methods mainly designed to handle the network traffic efficiently so that no QoS degradations due to congestion in networks. In this research described novel AQM depend on Random Early Detection (RED). RED AQM method mainly designed to solve the network congestion problems in Internet routers. However RED does not support the automation according to traffic dynamics in network which may degrade the performance. This paper proposed automation-based RED using the Dynamic Genetic Algorithm (DGA) called DGARED to handle the congestion in TCP/IP networks. The DGARED based on dynamic tuning parameters of optimization technique GA which adjust the weight parameter dynamically enhance overall queue scale sensitivity at routers in order to update actual queue size dynamically in RED. Using the dynamic GA, we provided the technique to find the effective values for weight parameter, maximum threshold, and minimum threshold. The simulation results of DGARED are evaluated with existing RED and GARED algorithms in terms of throughput by considering the different network conditions. The results show that DGARED overcomes the problems of both methods.

Keywords: automation, dynamic tuning, genetic algorithm, active queue management, queue size, RED, throughput

1. INTRODUCTION

The fundamental territory of TCP application has been the wired network wherein the primary driver of parcel misfortune is client traffic customarily. In such networks, the blunders as a result of the transmission media are truly immaterial. Inevitably, such issues with wired communications further tended to through the optimized TCP conduct. Essentially, TCP is the association arranged and dependable start to finish data transmission protocol at the vehicle layer [1]. TCP is the piece of protocols layered progressive system that solely bolsters utilizations of multi-network. The primary point of standard TCP protocol is to control the parcel misfortune and give the dependable data transmission administrations by means of bundle retransmissions. In the present correspondence worldview, TCP is generally utilized. TCP gives correspondence administrations at a middle of the road level

Received September 10, 2020; revised September 23 & October 6&14, 2020; accepted October 14, 2020.
Communicated by Amine Mohammed El Abdelli.
+ Corresponding author.

between the application layer and the web protocol. Then again, on the off chance that the transmission medium is the remote, at that point such traditional TCP/IP protocol is wasteful, as the radio-incited bundle mistakes surpass clog as the prevailing wellspring of parcel misfortune [2].

The problems become challenging to control the congestion at internet routers by considering the real time communication applications. To address the congestion control in internet routers, the Active Queue Management (AQM) presented. The usefulness of AQM is to identify and deal with the clog to keep away from queue floods. There are commonly two different ways to deal with clog, for example, signal blockage to traffic sources expressly by setting Explicit Congestion Notification bits, or sign clog to traffic sources verifiably by dropping parcels [3]. AQM has three principle components: (1) blockage pointer; (2) control capacity; and (3) input system. The clog pointer identifies when the blockage happens or close to happen though control work chooses what must be done when the blockages has been shown, where the input instrument is the sign that will be sent to tell the sources about the clogs states so as decrease the sources sending rates [4]. Random Early Detection algorithm (RED) which is one of the most notable active queue management algorithms was proposed in (Floyd and Jacobson, 1993) in which an adaptation of REAL test system based on Columbia's Nest reenactment bundle with bunches of alterations was utilized to assess the proposed algorithm [5].

This research has been to describe issues of existing AQM schemes researcher implemented to enhance performance regards RED algorithm. The essential objective of this research is to achieve the automations in AQM regards real time applications using the dynamic optimization technique. The Dynamic parameter tuning based Genetic Algorithm introduced to enhance performance of RED algorithm called DGARED under this research. DGA proposed to optimize the values of weight parameter, minimum threshold, and maximum threshold values dynamically in RED routers to handle the congestions automatically. The proposed DGARED may drop the packets only during the congestion in network which helps to improve the QoS performance. In Section 2, the brief review of different AQM schemes proposed during last two decades. In Section 3, the proposed DGARED methodology and existing RED designs described. In Section 4, simulation results and discussions are introduced. In Section 5, conclusion and future work described.

2. THEORETICAL RSSI MODEL

In real time communications, the route through resource to destination has been calculated through various techniques of routers. The congestion control is mainly achieved by controlling the data packets under such routers. Congestion takes place under router at the time of total bandwidth regards arriving packets; end regards a certain output link, goes beyond links bandwidth. Several attempts made to handle the congestion situations in network and prevent the data loss since from 3-4 decades. The AQM schemes gained more researchers attentions for congestion control like RED. The RED scheme Assist manage overall size of quiz & hence it was helpful for traffic of delay sensitive real-time. This kind of traffic of real time generally not utilized TCP under transport layer. Despite evident has been observed, also RED describing weakness like lower throughput, sharing of dominated bandwidth, description of latency variable & network stability deterioration. There are

many techniques described to enhance RED performances recently under [6].

In [7], dynamic model of TCP conducts along with a straight input model of TCP/ RED proposed to examine and structure a component for RED boundary tuning in light of changing network conditions like traffic load, interface limit and full circle time. Further epic self-tuning RED algorithm acquainted in [8] all together with upgrade the exhibition of existing RED. Creators got the fitting *p*max is progressively as per history data of both *p*max and the normal queue size in a timeframe. Also, *w* is appropriately picked by a direct strength state of the normal queue length. The Weighted Random Early Detection (WT RED) proposed in [9] for the clog taking care of in TCP networks. WTRED gives a movable weight boundary to build the affectability of the normal queue size in RED entryways to the adjustments in the genuine queue size. In [10], creator previously examined the impacts of the weighted moving normal on parcel hanging tight time decrease for an AQM system, at that point proposed method for registering the normal queue length dependent on a distinction condition (a recursive condition). Contingent upon a specific optimality rule, legitimate boundaries of the changed weighted moving normal capacity can be picked. Another epic AQM procedure called Random Early Dynamic Detection (REDD) acquainted in [11] with address a few issues related with RED technique especially RED's dependence on the info boundaries. In [12], creator proposed stable tuning way for RED boundaries which can keep the balance point fixed confronted with busty traffic. The objective queue size was set alongside the deliberate harmony point progressively, and the alteration strategy for RED boundaries performed by tuning the slant of the parcel dropping likelihood. In [13], author attempted various methods to search w_q values using genetic algorithms which performed by adapting the values w_q dynamically according to the character of traffic. In [14], another recent RED based AQM proposed in which author combined the RED with threshold value (Max_{Th}, Min_{Th}). Their main was aim for detecting incipient congestion properly & for manage congestion notification to for finishing hosts under to improve the performance of throughput and packet delivery ratio. In [15], authors presented the effects of the constant RED parameters over the queuing network performance. They presented the mechanism and its evaluation for the parameters modifications for enhancing RED performance.

3. METHODOLOGY

In this section, we present the design of proposed DGARED algorithm for the AQM. Before presenting the methodology of proposed DGARED, first we discussed existing RED along with its limitations.

3.1 RED Methodology

Active Queue Management (AQM) is a way to deal with go before clog. RED is a particular AQM usage. It keeps up an Exponentially Weighted Moving Average (EWMA) and two predefined edges; the greatest edge (max_{th}) and least edge (min_{th}). RED screens the normal queue size as opposed to observing the genuine queue size. The RED entryway ascertains the normal queue size, utilizing a low-pass channel with an exponential weighted moving normal. The normal queue size is contrasted with two edges, a base limit min_{th},

and a most extreme edge $\max_{th}$. At the point when the normal queue size is not exactly the base limit, no parcels are stamped. At the point when the normal queue size is more prominent than the greatest edge, each showing up parcel is checked. Whenever checked parcels are in truth dropped, or if all source hubs are helpful, this guarantees the normal queue size doesn't essentially surpass the most extreme limit. At the point when the normal queue size is between the base and the greatest limit, each showing up bundle is set apart with likelihood p_a, where p_a is an element of the normal queue size *avg*. Each time that a parcel is denoted, the likelihood that a bundle is set apart from a specific association is generally corresponding to that a lot of the transfer speeds at the passage. The algorithm of RED fundamentally planned in two key advances, for example,

(1) *Calculate packet drop probability*: That kind of technique is basis on managing congestion before it was happened.
(2) *Calculate average queue length*: It was calculated through:

$$avg = (1 - W_q).avg + W_{q.q}. \tag{1}$$

Where, the W_q represents the weight parameter in range (0, 1) and q represents the current queue size. Formula regards packet dropped probability regards RED has been computed as:

$$p_b = \begin{cases} 0, & avg < \min_{th} \\ 1, & avg < \max_{th} \\ \frac{avg - \min_{th}}{\max_{th} - \min_{th}} * \max_p & \min_{th} \le avg \le \max_{th} \end{cases}. \tag{2}$$

Where $\max_p$ – largest packet drop probability, *avg* represents size of average queue, $\min_{th}$ represents minimum threshold value, $\max_{th}$ represents maximum threshold value, Formula 2 describing packet drop probability which was on value of average queue length. Fig. 1 describes drop function of RED.

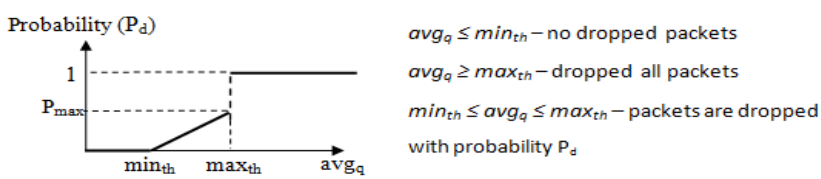

Fig. 1. Limitation of RED drop function.

The final packet marking probability p_a enhances normally calculation improves since last marked packet computed as:

$$p_a = \frac{p_b}{1 - count * p_b}. \tag{3}$$

Where, p_b represents immediately marking probability, p_a represents the accumulate drop probability, and *count* represents number of arrived packets since the last dropped one.

RED execution is delicate to the quantity of contending streams. It is exceptionally

delicate to its boundary settings. In RED, at any rate 4 boundaries, to be specific, a base limit, greatest edge, most extreme parcel dropping likelihood (max_p), and weighted queue (W_q), must be appropriately set. Execution of RED is delicate to the bundle size. The estimations of the base and most extreme edges are allocated relying upon the attractive genuine normal of the queue size. The base limit speaks to the greatest normal queue size that is permitted in the queue. Thus, higher greatest edges will build delays. Then again, a lower most extreme edge will diminish throughput. Likewise, higher least limits increment throughput and connection usage. Thus, dynamic tuning of these values according to communication requirements will helps to improve the RED performance. To address such challenges, we proposed the DGARED scheme in this paper.

3.2 DGARED

To overcome the problems of RED and its recent variants studied in literature we proposed the novel approach of searching the efficient values for the W_q, $\max_{th}$, and $\min_{th}$ using the dynamic genetic algorithm in order to select these values as per the current needs. There are two major parts in this system that is built:

(1) Genetic algorithm to get the weight value of EWMA (W_q), $\max_{th}$, and $\min_{th}$, and *avg*.
(2) The queue management mechanism using RED. The random early detection mechanism gets input values from the average queue size, moment queue size, and EWMA estimator weight (W_q).

We designed novel technique to obtain the queue size estimator (EWMA) weight value. The value will be continuously evaluated up to 20 iterations to get the most minimum MSE value. The W_q value generated from the initial generation process will be the input for the algorithm regards computing queue average size as per Eq. (1). In the system used, the parameters of the momentary queue size, average queue size and the average of averages queue in calculating the next queue average are obtained from the previous event. The $\max_{th}$, and $\min_{th}$ parameters are dynamically adjusting by calculating the average of average queue size (*avvg*) by dividing the summation of the average queue size (*avg*) by the number of attempts. Then the average percentage (*persAve*) of the increase in the average queue size is calculated by the difference between the initial average queue size and the averages of average queue size divided by *avvg* and multiply by 100. This average percentage is used to adjust the $\max_{th}$ and $\min_{th}$ by adding it to $\max_{th}$ and decreasing it from the $\min_{th}$ as follows:

$$avvg = \frac{avg\_sum}{attempts}. \tag{4}$$

The equation above is used to obtain the averages of average queue size *avvg* based on the summation of new average queue size *avg* and divide into 5.

$$persAve = \frac{avvg - initAve}{avvg} * 100. \tag{5}$$

The increase of *avg* can be calculated from the different of initial *avg* and the average of *avg*. Then increase the $\max_{th}$ and decrease $\min_{th}$ depend of the average percentage (*persAve*) of the increase in the average queue size:

$$\max_{th} = \max_{th} + persAve, \tag{6}$$

$$\min_{th} = \min_{th} - persAve. \tag{7}$$

Fig. 2 shows the architecture of DGARED method. The DGAED designed based on above dynamic parameters computations with DGA. It developed to find a solution in a domain of solutions that are random and have not been formulated empirically. Based on previous research there are several assumptions about getting weight values, $\max_{th}$, $\min_{th}$, and W_q, but all are still based on trial and error. Therefore, we proposed solution of automation system in which those values are generated based on the best individuals produced by genetic algorithms. The process generating those values is described above and using those values dynamically using the GA shown in Fig. 2. As shown in the figure, the proposed DGA based RED approach works in five steps of GA such as initialization, fitness function, selection, crossover, and mutation.

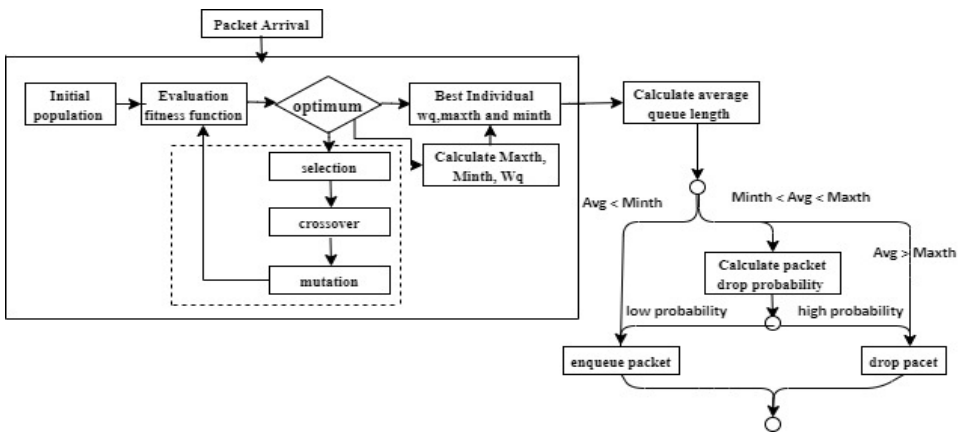

Fig. 2. Proposed DGARED algorithm.

Table 1. Initial population.

Initial Population	
Number of individuals	5
Number of Chromosome	13

- ***Initial Population*:** Table 1 shows the initialization parameters used in DGARED algorithm.
  The estimator weight domain is in the range 0.0020 to 0.5000, meaning that it requires the accuracy of 4 digits behind the comma. Accuracy of 4 digits behind the comma can be achieved with real numbers 1000 to be represented in binary numbers. While the results range in [0, 1], only 1000 is needed. This range can be achieved with binary numbers $2^9 < 1000 < 2^{10}$ or $512 < 1000 < 1024$. Based on the domain of results to be achieved, 10 gene bits are needed to form a chromosome.
- ***Fitness Function*:** The existing GA algorithm modified to get the dynamic fitness function values. The output generated by the fitness function is $\max_{th}$, $\min_{th}$, and W_q value which produces a minimum MSE. This value will be input for the selection process ba-

sed on the roulette wheel.

- ***Selection* (*Roulette Wheel*):** After obtaining a minimum MSE value, a minimal selection process will be carried out. Selection in this process is done by generating random numbers in the range 0 to 100, and then the value will be compared with the proportion of each MSE value to the total MSE value of all individuals who have been sorted first. Individual gets a proportion value from the sum of the previous individual fitness scores.
- ***Crossover*:** Individuals who have undergone a selection process to get the best parent then enter the process of crossing. In this system, a probability of 60% is used because it's the best value of crossover chosen as in Table 2, meaning that if the random numbers generated are more than the probability of crossing then it does not occur, whereas if it is below the value of 60%, a crossing process occurs. The resulting child chromosome will be a new individual to undergo the mutation process.

Table 2. Effect of the W_q, Max_{th} and Min_{th} rates of GA algorithm based on the crossover.

Parameters	Crossover Ratio %				
	20	40	60	70	90
Weighted Queue (W_q)	0.034	0.029	**0.017**	0.036	0.024
Maximum threshold (Max_{th})	7531	7534	7568	7529	7565
Minimum threshold (Min_{th})	2469	2465	2432	2470	2434

- ***Mutation*:** The process of mutation itself is very simple in representing binary numbers by changing the number 0 to 1 and vice versa. This change process occurs in individual with a randomly assigned chromosome. Individuals who have passed the mutation process even though they have not mutated will become new populations that will experience repetitions of the chromosome evolution process as much as the generations defined at the beginning. Based on several changes in the number of chromosomes on which the mutation will take place, we obtained the best mutation in 30% as in the table

Table 3. Effect of the W_q, Max_{th} and Min_{th} rates of GA algorithm based on the mutation.

Parameters	Mutation Rate %			
	10	30	50	70
Weighted Queue (w_q)	0.299458	**0.299336**	0.299254	0.299092
Maximum threshold (Max_{th})	7529	7529	7531	7531
Minimum threshold (Min_{th})	2470	2470	2469	2469

4. STUDY RESULTS AND DISCUSSION

The verification of the proposed DGARED algorithm is evaluated using the NS2 tool by measuring the key QoS parameter called throughput under different circumstances. The performance of DGARED is compared with existing RED and GA based RED called GARED to claim the efficiency of DGARED. The performance metrics are measured as:

- ***Average Throughput*:** This metrics calculates the total number of packets delivered per second *i.e.*, total number of messages which are delivered per second. The average throughput in Mbps is:

$$T = \left(\frac{R}{T^2 - T^1}\right). \tag{8}$$

Where R is complete received packets at all destination nodes, T^2 is simulation stop time and T^1 simulation start time.

4.1 Effect of Packet Size

(1) Throughput

Observations are carried out using a change scenario Pareto-distributed average value of packet size and assuming the average value of service time is fixed. Each scenario is simulated 5 times and then the results are averaged. The observations are demonstrated in Table 4.

Table 4. Throughput performance of varying packet size.

Packet Size (Bytes)	RED	GARED	DGARED
1000	**1231.79**	**1148.27**	1317.51
2000	1201.43	1037.3	**1431.84**
2500	1138.44	977.08	1424.23
5000	1117.9	768.828	1407.3
7500	1047.27	666.216	1374.2
10000	887.912	749.534	1295.29
15000	957.787	941.362	1286.4
20000	937.838	957.26	1314.38

From the observation Table 4 above it can be seen that the throughput value generated from the management of the DGARED queue has a greater value for all the average packet length as compared to RED and GARED methods. The throughput value generated using DGARED reaches a maximum of 1431.84 MBps at an average length of a 20 KB package. Whereas a GARED and RED reaches a maximum of 1148.27 MBps and 1231.79MBps at an average length of a 1 KB package respectively. In general, the resulting throughput increases stably and does not fluctuate. The trend of changing throughput compared to time can be seen in Fig. 3.

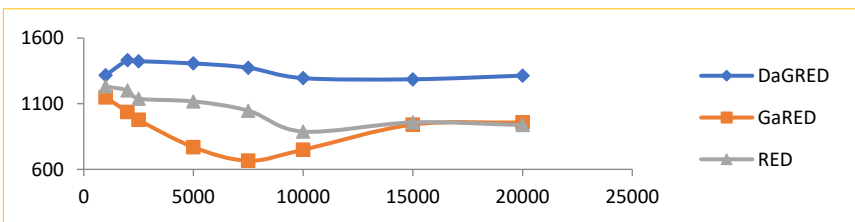

Fig. 3. Evaluations of throughput performance with varying packet size.

(2) Jitter

As observed in Table 5 and Fig. 4, it can be seen the use of the DGARED queue management reaching the minimum value if the packet size is determined by 2000 Bytes as compared to the RED and GARED methods.

At the packet size of 2 KB, the resulting jitter reaches 5.06 MBps which is two times lower than RED and GARED methods.

Table 5. Jitter performance of varying packet size.

Packet size	RED	GaRED	DGaRED
1000	**11.01**	**12.81**	7.6
2000	11.19	17.93	**5.06**
2500	13.29	20.98	5.32
5000	14.59	32.83	5.91
7500	19.05	42.26	7.16
10000	28.88	34.13	9.59
15000	24.24	21.01	10.35
20000	23.6	20.16	9.5

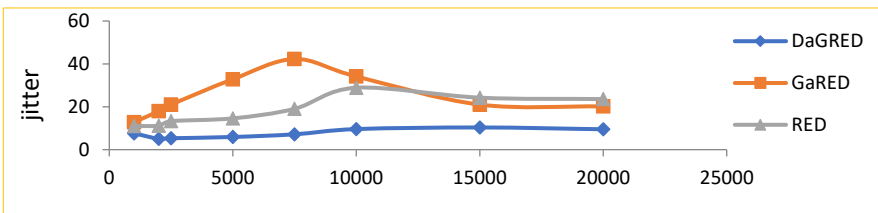

Fig. 4. Evaluations of Jitter performance with varying packet size.

(3) Delay

After the jitter investigations, next we investigated the effect of packet size over the delay of each investigated technique. The observations were made using a scenario of changes in the packet size to see the effect on delay generated by the DGARED queue management compared to the RED and GARED as in Table 6.

Table 6. Delay performance of varying packet size.

Packet size	RED	GaRED	DGaRED
1000	154.311	152.175	187.132
2000	155.046	151.883	190.112
2500	154.772	151.714	186.912
5000	155.585	150.339	184.678
7500	155.744	148.49	179.378
10000	155.067	**148.199**	**168.336**
15000	153.366	149.399	177.867
20000	**152.453**	149.457	180.694

Based on observations it was found that for all test points, the delay generated using DGARED queue management generally has a greater value than the RED and GARED. The lowest delay occurs when the packet size is 2000, while the highest value is produced when the packet size value is 10000. Whereas the GARED recorded the lowest value when the packet size is 10000 which is better than RED and DGARED. Fig. 5 shows the trends of delay with changing packet size.

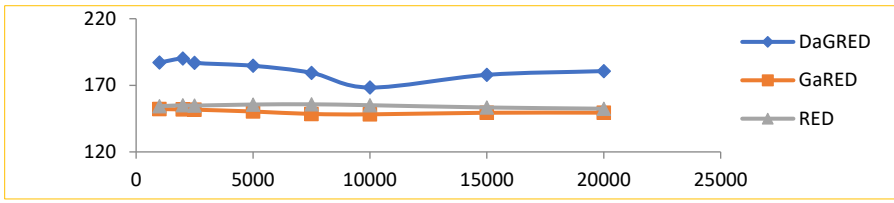

Fig. 5. Evaluations of delay performance with varying packet size.

4.2 Effect of Average Inter-arrival Time

(1) Throughput

After the packet size investigations, next we investigated the effect of average changing inter-arrival time over the throughput of each investigated technique. The observations were made using a scenario of changes in the value of the average inter-arrival time to see the effect on throughput generated by the DGARED queue management compared to the RED and GARED. By assuming the average length of a 10 KB package. The results of observations can be seen as follows in Table 7.

Table 7. Throughput performance of varying average inter-arrival time.

Average inter-arrival time (ms)	RED	GARED	DGARED
0.05	752.549	459.632	1102.06
0.10	697.77	386.086	874.564
0.15	735.114	529.035	947.71
0.20	706.224	594.531	1479.62
0.25	927.328	773.936	1438.36
0.30	995.128	747.355	1383.27
0.35	1012.25	842.542	1318.59
0.40	1087.92	963.602	1472.61
0.45	1149.76	1056.04	**1571.74**
0.50	**1229.3**	**1128.99**	1560.67

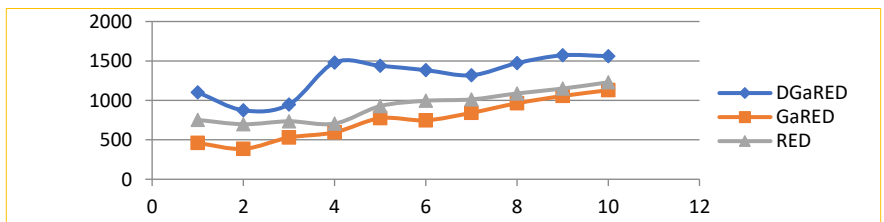

Fig. 6. Evaluations of throughput performance with average inter-arrival time.

Based on observations it was found that for all test points, the throughput generated using DGARED queue management generally has a greater value than the RED and GARED. The lowest throughput occurs when the Average Inter-arrival Time is 0.10 ms, while the highest value is produced when the average arrival time value is 0.45 ms Fig. 4 shows the trends of throughput with changing average inter-arrival time. Based on the results of observations in Table 3 and Fig. 4 above, the throughput generated by the DGARED queue management has a greater value than the RED and GARED for the average value between the arrival times between 0.2-0.3 ms.

(2) Jitter

As observed in Table 8 and Fig. 7, it can be seen that the use of the DGARED queue management reaching the maximum value if the average changing inter-arrival time 0.40 ms as compared to the RED and GARED methods. At the average changing inter-arrival time 0.40 ms, the resulting jitter reaches 7.8 which is better than RED and GARED methods.

Table 8. Jitter performance of varying average inter-arrival time.

Arrival time	RED	GaRED	DGaRED
0.05	**37.97**	75.1	17.25
0.10	47.38	94.21	35.64
0.15	48.01	80.25	30.89
0.20	52.79	88.06	21.62
0.25	50.54	**63.94**	14.65
0.30	76.07	70.23	20.85
0.35	86.65	85.37	14.42
0.40	98.12	97.67	**7.8**
0.45	95.2	85.05	47.46
0.50	111.64	76.36	44.41

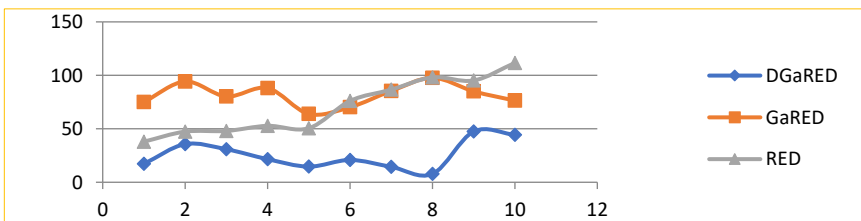

Fig. 7. Evaluations of Jitter performance with average inter-arrival time.

(3) Delay

As observed in Table 9 and Fig. 8, the GARED queue management reaching has the minimum value if the average changing inter-arrival time 0.50 ms as compared to the RED and DGARED methods. At the average changing inter-arrival time 0.50 ms, the resulting delay reaches 132.733. At the same time the DGARED method reaches 232.524, and the RED method reaches 151.02.

Table 9. Delay performance of varying average inter-arrival time.

Arrival time	RED	GaRED	DGaRED
0.05	168.16	143.51	174.639
0.10	178.617	139.691	181.127
0.15	184.608	143.29	187.736
0.20	195.626	143.921	215.646
0.25	180.971	146.101	195.624
0.30	170.904	144.872	205.131
0.35	169.336	146.948	213.114
0.40	160.218	146.435	224.018
0.45	160.845	139.849	223.45
0.50	151.02	**132.733**	**232.524**

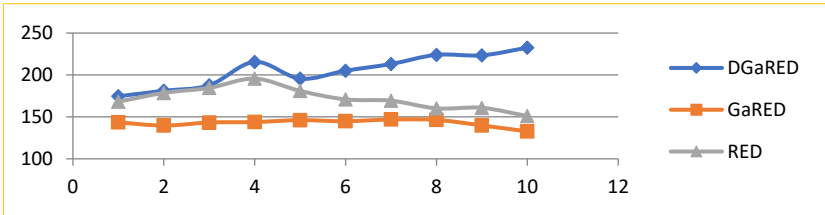

Fig. 8. Evaluations of delay performance with average inter-arrival time.

5. CONCLUSIONS

In this paper we proposed the novel AQM method for coordinated automation systems and their industrial applications related the network communications. We exploited the genetic algorithm to design the adaptive AQM in this paper which is based on the searching the key parameters values dynamically in order to adjust the queue size as per the actual requirements. The proposed DGARED method dynamically selects the values of weigh parameter, maximum threshold, minimum threshold and average parameters to suppress the limitations of previous RED versions and RED method. The simulation results claim the significant improvement over the RED and GARED techniques by considering the different network conditions. For future work, we suggest to investigate the other key performance metrics such as jitter, delay, packet delivery ratio *etc.*

REFERENCES

1. A. Kudeshia and A. K. Jagannatham, "Optimal viterbi based total variation sequence detection TVSD for robust image/video decoding in wireless sensor networks," *IEEE Signal Processing Letters*, Vol. 21, 2014, pp. 722-7261.
2. M. Allman, V. Paxson, and W. Stevens, "TCP congestion control," RFC2581, April 1999.
3. B. Francis, V. Narasimhan, A. Nayak, and I. Stojmenovic, "Techniques for enhancing TCP performance in wireless networks," in *Proceedings of IEEE International Conference on Distributed Computing Systems Workshops*, 2012, pp. 222-230.
4. D. Augustyn, A. Doma´nski, and J. Doma´nska, "Active queue management with non linear packet dropping function," in *Proceedings of the 6th International Conference on Performance Modelling and Evaluation of Heterogeneous Networks HET-NETs*, 2010, pp. 133-142.
5. D. Augustyn, A. Doma´nski, and J. and Doma´nska, "A choice of optimal packet dropping function for active queue management," in *Proceedings of International Conference on Computer Networks*, Vol. 79, 2010. pp. 199-206,
6. K. Grinnemo, J. Garcia, and A. Brunstrom, "Taxonomy and survey of retransmission-based partially reliable transport protocols," *Computer Communications*, Vol. 27, 2004, pp. 1441-1452.
7. X. Wang, "Active queue management for real-time IP traffic," Ph.D. Thesis, Department of Electronic Engineering, University of London, 2006.
8. W. Chen and S.-H. Yang, "The mechanism of adapting RED parameters to TCP tra-

ffic," *Computer Communications*, Vol. 32, 2009, pp. 1525-1530.
9. J. G. Chen, C. Hu, and Z. Ji, "Self-tuning random early detection algorithm to improve performance of network transmission," Hindawi Publishing Corporation, Mathematical Problems in Engineering, Vol. 2011, Article ID 872347, 17 pages.
10. N. Hamadneh, D. Murray, M. Dixon, and P. Cole, "Dynamic weight parameter for the random early detection (RED) in TCP networks," *International Journal on New Computer Architectures and Their Applications*, Vol. 2, 2012, pp. 342-352.
11. J. D. Ska, A. D. Ski, D. R. Augustyn, and J. Klamka, "A RED modified weighted moving average for soft real-time application," *International Journal of Applied Mathematics and Computer Science*, Vol. 24, 2014, pp. 697-707.
12. H. Abdel-Jaber, F. Thabtah, M. Woodward, A. Jaffar, and H. Al-Bazar, "Random early dynamic detection approach for congestion control," *Baltic Journal of Modern Computing*, Vol. 2, 2014, pp. 16-31.
13. L. Yang, J. Zhu, W. Xie, and X. Tan, "Stable tuning for random early detection algorithm," in *Proceedings of the 33rd Chinese Control Conference*, 2014, pp. 5470-5475.
14. P. H. Hendrawan, "Random early detection utilizing genetics algorithm," in *Proceedings of the 8th International Conference on Telecommunication Systems Services and Applications*, 2014, pp. 1-7.
15. K. Pandey, C. Agrawal, and Y. K. Rana, "ERED: Enhanced RED using dynamic threshold an efficient congestion control algorithm," *International Journal of Engineering Technology and Applied Science*, Vol. 1, 2015, No. 13018.
16. M. H. Malik and A. Majeed, "A parametric study for congestion control in qeuing networks," in *Proceedings of International Conference on Future Networks and Distributed Systems*, 2017, Article 47.
17. H. Xua, K. Huan, and H. Wang, "Adaptive FEC coding and cooperative relayed wireless image, transmission," *Digital Communications and Networks*, Vol. 1, 2015, pp. 213-221.
18. H. Yang, K. W. Wong, X. Liao, W. Zhang, and P. Wei, "A fast image encryption and authentication scheme based on chaotic maps," *Communications in Nonlinear Science and Numerical Simulation*, Vol. 15, 2010, pp. 3507-3517.
19. J. M. Shapiro, "Embedded image coding using zero trees of wavelet coefficients," *IEEE Transactions on Signal Processing*, Vol. 41, 1993, pp. 3445-3462.
20. J. N. Laneman, D. N. C. Tse, and G. W. Wornell, "Cooperative diversity in wireless networks: efficient protocols and outage behavior," *IEEE Transactions on Information Theory*, Vol. 50, 2004, pp. 3062-3080.

Majid Hamid Ali received the M.S. degree in Computer Science \ Information Technology from Utara University Malaysia. He currently pursuits in his Ph.D. degree in the Department of Computer Engineering, Erciyes University, Turkey.

Serkan Öztürk received his Post Doctorate Ryerson University, Faculty of Science, Computer Science, Canada, 2002-2009, and Doctorate, Erciyes University, Institute of Science and Technology, Electronics Engineering, Turkey, 1999-2002 Post Graduate, Erciyes University, Institute of Science and Technology, Electronics Engineering, Turkey.