

# Random Early Dynamic Detection Approach for Congestion Control

Hussein ABDEL-JABER<sup>1</sup>, Fadi THABTAH<sup>2</sup>, Mike WOODWARD<sup>3</sup>,  
Ahmad JAFFAR<sup>2</sup>, Hussein AL BAZAR<sup>1</sup>

<sup>1</sup>Faculty of Computer Studies, Department of Information Technology and Computing,  
Arab Open University, Dammam, Kingdom of Saudi Arabia

<sup>2</sup>Ebusiness Department, Canadian University of Dubai, 1st Interchange,  
Sheikh Zayed Road, UAE

<sup>3</sup>Department of Computing, University of Bradford, BD71DP, UK

habeljaber@arabou.edu.sa, fadi@tud.ac.ae,  
m.e.woodward@bradford.ac.uk, ahmad@tud.ac.ae,  
halbazar@arabou.edu.sa

**Abstract.** Congestion is one of the key issues that may occur when the amount of available resources at a router buffer cannot accommodate all arriving packets. In order to detect and control congestion, Active Queue Management (AQM) methods such as Random Early Detection (RED) are used before the buffer overflows. However, most of these methods including RED suffer from issues such as their reliance on the setting parameters and require affixing input parameters to appropriate values in order to gain a satisfactory performance. Furthermore, RED like methods are not able to stabilise the average queue length between the minimum and maximum threshold positions when the number of sources increases and as such a heavy congestion occurs. To enhance RED's performance in regards to mean queue length and packets waiting time a new method called Random Early Dynamic Detection (REDD) is proposed in this paper. REDD has been compared with popular congestion control methods, i.e. Drop-tail, RED and Adaptive Maximum Threshold (AMT). This comparison is based different known performance measure such as throughput, packets waiting time, packet loss probability, etc. The experimental results indicate that REDD has a better performance results than its predecessor for the majority of the measures utilised and using Bernoulli and 2-MMBP processes.

**Keywords:** Analytical Models, Congestion Control, Network Management, Simulation

## 1. Introduction

Internet has tuned into one of the fastest developing technologies due to the different applications and the variety of data traffic (Stallings, 2009). Internet assures Quality of Service (QoS) (Zhang et al., 2011; Zhou et al., 2011; Zhong et al., 2011) by fulfilling services with different requirements that allow packets to traverse through it successfully. One of the internet services is the "best effort" (Chrysostomou et al., 2003) where packets are delivered fairly to their destinations. Briefly, there is no distinction among the "best effort" packets in a network. The "best effort" cannot be used with different requirements of data traffic since it processes all services in the same degree. In order to serve different data traffic services with different requirements, both assured

and/or expedited data traffic can be used. The assured and expedited services are designed for providing a satisfactory QoS for different service classes.

Non-real-time data traffic can be used in many internet applications such as electronic mail (e-mail) and file transfer. In these applications, data traffic is sensitive to packet loss, packet delay and jitter. However, real-time data traffic is employed in several internet applications, e.g. voice over internet protocol (VoIP), video conferencing, live voice and live video. In real-time applications, the data traffic is sensitive to packet delay and insensitive to packet loss. Some real and non-real-time internet applications along with their sensitivity to QoS requirements, i.e. bandwidth, delay, jitter and packet loss, are presented in Tables 1 and 2 respectively.

Delivering data across a network necessitates high speed router buffers. Hence router buffers work on routing data from one network to another on the path from the source application to the destination (Tanenbaum, 2002). Congestion is one of the primary issues that may occur at the router buffers due to arriving packets which are not absorbed by the available network resources (Braden et al., 1998), i.e. buffer rooms and bandwidth. A network performance often deteriorates due to congestion, in which 1) the throughput is reduced, 2) many arriving packets are lost, 3) the mean waiting time of packets in the network is increased, 4) instability of queue length at the router buffers and 5) unfair share of network resources among sources.

To manage congestion, many researchers have proposed control techniques such as Drop-tail (Brandauer et al., 2001), Active Queue Management (AQM) (Ahammed and Banu, 2011; Ali et al., 2007; Feng et al., 2002; Athuraliya et al., 2001; Aweya et al., 2001; Floyd et al., 2001; Floyd, 2000; Floyd and Jacobson, 1993), which are based on simulation and analytical models (Ababneh et al., 2011; Thabtah et al., 2009; Abdeljaber

Table 1: Some real-time applications with their sensitivity to QoS (Tanenbaum, 2002).

Real-Time Applications	Sensitivity			
	Bandwidth	Delay	Jitter	Packet Loss
Telephony	Low	High	High	Low
Live video	High	High	High	Low
Video conferencing	High	High	High	Low
Confidential video conferencing	High	High	High	Low
Data conferencing	High	High	High	Low
VoIP	High	High	High	Low
Video on demand (VOD)	High	Low	High	Low
Audio on demand (AOD)	High	Low	High	Medium

Table 2: Some non-real-time applications with their sensitivity to QoS (Tanenbaum, 2002).

Non-Real-Time Applications	Sensitivity			
	Bandwidth	Delay	Jitter	Packet Loss
File transfer	Low Medium High	Low	Low	High
Email	Low	Low	Low	High
Confidential email	Low	Low	Low	High
Money transactions	Low	Low	Low	High

et al., 2008a; Abdeljaber et al., 2008b; Zhou and Wang, 2008; Lieu and He, 2007), i.e. discrete-time queues or continuous-time queues. In this paper, we focus on AQM methods which are implemented using simulation. Examples of common AQM methods are Random Early Detection (RED) (Floyd, and Jacobson, 1993), Adaptive RED (Floyd et al., 2001), Gentle RED (Floyd, 2000), Random Exponential marking (REM) (Athuraliya et al., 2001), Dynamic Random Early Drop (DRED) (Aweya et al., 2001), Generalised Random Early Evasion Network (GREEN) (Feng et al., 2002b), an Adaptive Maximum Threshold (AMT) (Ali et al., 2007) and others. Each AQM method is designed to give a high throughput performance or a low average packet waiting time or a small queue length in order to prevent quickly building up the content of router buffers. The majority of AQM methods have been implemented using simulation because of the availability of customised software packages and the build-in-methods in certain Object Oriented programming languages.

Some AQM methods including RED suffer from issues such as the reliance on setting the input parameters (minimum threshold, maximum threshold, queue weight and maximum dropping probability) to particular values. For instance, RED must tune its input parameters to appropriate values in order to obtain a satisfactory performance. In addition, it suffers from maintaining the average queue length between the minimum and the maximum threshold positions when the number of sources increases (heavy congestion). To enhance RED's performance, a method called Dynamic Random Early Detection (REDD) is proposed in this paper. Unlike RED which uses a constant parameter as a congestion detector, REDD utilises an adaptive parameter for adjusting its value for every arriving packet during congestion. Moreover, REDD do not depend on the input parameters as RED since it employs a variable maximum parameter instead of a fixed one.

To measure the success and failure of our method we use different performance measures such as mean queue length ( $mql$ ), the waiting time for packets ( $D$ ), the overflow packet loss probability ( $P_L$ ), etc., to derive the experimental results. We also contrast REDD with other methods to answer the question "which of the compared methods in this article can offer more satisfactory performance using both Bernoulli and 2-MMBP?"

This paper is structured as follows: Sections 2 and 3 present congestion control methods and the new developed method using. Section 4 highlights the comparison results of REDD and three other AQM methods (Drop-tail, RED, and AMT). Finally, conclusions and future work are given in Section 5.

## 2. Literature Review

### 2.1 Late Congestion Control Approach

Methods that control congestion in a late stage like Drop-tail (Brandauer et al., 2001) lose packets after the router buffer overflows and when the congestion appears. Drop-tail sets its router buffers to a maximum length in order to achieve high throughput. Nevertheless, this may degrade the network performance because of the high packets loss and waiting time. When Drop-tail method sets the router buffer to a small length,

the packets loss and waiting time performance are enhance but the throughput performance deteriorates.

## 2.2 Early Congestion Control Approach

In early congestion control methods, the router buffer of a network will be early identified and controlled before it overflows (Feng et al., 2002a; Floyd et al., 2001; Floyd, 2000). Early congestion control methods may avoid full or almost full router buffers for a long period of time. Though, controlling congestion early may alleviate the lockout phenomenon by leaving buffer spaces. The lockout phenomenon causes unfair share of network resources among flows. These buffer spaces are accommodated by idle flow packets and as a result, other flow packets can be utilised from the buffer spaces.

An example of early congestion control models is RED (Floyd and Jacobson, 1993). Initially, RED’s performance was satisfactory and latterly it has deteriorated due to data traffic types (Thabtah et al., 2009; Woodward, 1993), i.e. voice and video. To deal with this issue, other AQM methods have been proposed to enhance the RED’s performance such as GRED, ARED, REM, DRED, SRED, AMT and others. We focus in this section on two methods, i.e. RED and ATM, since we have utilised them in the experimental section for comparison purposes besides being a known methods in the literature.

### 2.2.1 RED

RED has been proposed to early control congestion and it has been approved by the Internet Engineering Task Force (IETF) in RFC 2309 (Braden et al., 1998). RED depends on a congestion measure called the average queue length (*aql*), in which its router buffer calculates the *aql* for every arriving packet based on equation (1).

$$aql = aql \times (1 - qw) + qw \times q \dots\dots\dots(1)$$

where *qw* is the queue weight and it was set to 0.002 and *q* is the current queue length.

The *aql* is normally computed using a low pass filter with an exponential weighted moving average. After the *aql* for a router buffer is computed, it will be compared to two queue positions at the router buffer. These two positions are called minimum and maximum thresholds. The maximum threshold has been set to at least double of the minimum threshold to maintain the throughput performance. If the *aql* value is smaller than the minimum threshold, the router buffer will save the arriving packets and then forwards them to their destinations. Therefore, the packet dropping value becomes zero since no packets are dropped. In cases when the *aql* is between the minimum and maximum thresholds, the router buffer will drop the arriving packets probabilistically as

$$pd_{max} \times \frac{(aql - \text{min threshold})}{(\text{max threshold} - \text{min threshold})}$$

where *pd<sub>max</sub>* is the maximum dropping value for packets and can be set to 0.1 as in RED. This dropping probability of packets is used in order to manage congestion. Finally, if the *aql* is equal to or larger than the maximum threshold value, the router buffer will drop arriving packet with a probability equal to 1.

**2.2.2 AMT**

AMT employs *aql* to control congestion similar to RED. However, AMT uses an adaptive threshold position to maintain the *aql* at a target value. This value often located in the middle between the minimum and the maximum threshold positions. The stability of *aql* at a target value prevents the router buffer from building up and as such the number of dropped packets will decrease. In (Ali et al., 2007), there are two packet dropping probabilities (PDP) that are employed to maintain *aql* at a target value. These are calculated based on equations (2-3).

$$PDP(1) = pd_{max} \times \frac{(aql - min\_threshold)}{(max\_threshold(1) - min\_threshold)} \dots\dots\dots (2)$$

$$PDP(1) = pd_{max} \times \frac{(aql - min\_threshold)}{(max\_threshold(2) - min\_threshold)} \dots\dots\dots (3)$$

The values of *maxthreshold(1)* and *maxthreshold(2)* represent the adaptive of maximum threshold position, and they depend on the *aql* position as in Fig. 1.

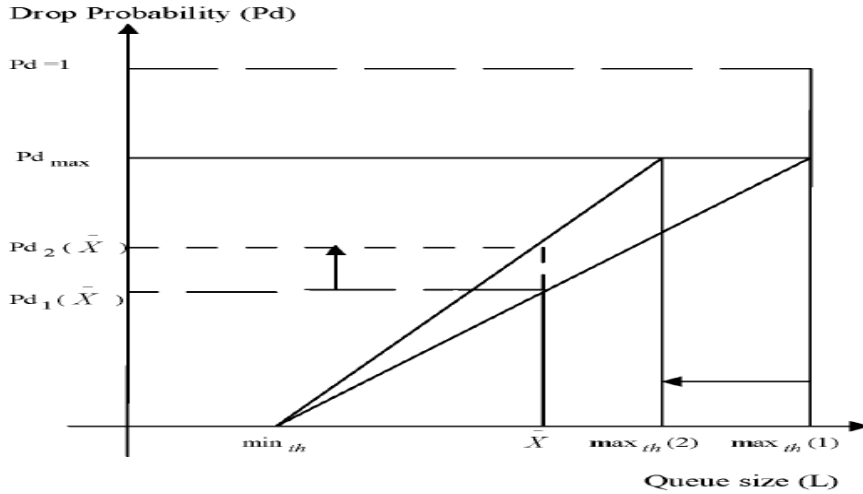


Fig. 1: The two packet dropping probabilities of AMT method (Ali et al., 2007).

**3. The Proposed Method**

Due to the variety of data traffic, RED’s performance becomes deteriorated because *aql* increases when the number of sources increases. This may lead to the *aql* above the maximum threshold. Therefore, every arriving packet will be dropped. In order to solve the above issue, is proposed REDD, which uses an adaptive maximum threshold position in order to identify and control congestion. REDD computes the *aql* similar to RED and compares the *aql* with the minimum and the maximum threshold positions. If the *aql* is between the minimum threshold and its value and the value of the maximum threshold is, at least, double of the minimum threshold, REDD reduces the maximum threshold value by 2. This will move the *aql* towards its target value.

On the other hand, if the  $aql$  is between the target value and the maximum threshold and also the maximum threshold is smaller than or equal to the difference of the router buffer capacity and the minimum threshold, the maximum threshold value will increase by 2. Therefore, the  $aql$  will stabilize at the target value and the throughput result may increase by giving 2 more rooms for the maximum threshold. Choosing “2” as the decreasing or the increasing amount of the maximum threshold can help in determining congestion before the current queue length of the router buffer overflows. The source code of REDD is shown in Fig. 2.

```

For every packet arriving at the router buffer
If ( $aql < \text{target } aql$  and maximum threshold  $\geq 2 \times$  minimum threshold)
{
    //Decreasing the maximum threshold value by 2 as follows:
    maximum threshold = maximum threshold - 2;
}
If ( $aql > \text{target } aql$  and maximum threshold  $\leq$  (router buffer capacity - minimum threshold))
{
    //Increasing the  $\text{max threshold}$  by 2 as follows:
    maximum threshold = maximum threshold + 2;
}

```

Fig. 2: The source code of REDD.

### 3.1 REDD Parameters

This subsection discusses the proposed method main parameters setting, i.e.  $pdmax$ ,  $qw$ , etc. In REDD,  $pdmax$  and  $qw$  are set to 0.1 and 0.002 respectively and the target  $aql$  is set to the middle value between the minimum and the maximum thresholds. The minimum threshold is set to the same value as in RED. Initially, the maximum threshold is set to the value of ( $3 \times$  minimum threshold). The maximum threshold value of the REDD depends on the value of  $aql$  in which if the  $aql$  value is below the minimum threshold, the maximum threshold will be set to ( $3 \times$  minimum threshold). Whereas if the  $aql$  is larger than or equal to (router buffer capacity - minimum threshold), the maximum threshold will be set to (router buffer capacity - minimum threshold). In the case that  $aql$  is between the minimum threshold and target  $aql$  the maximum threshold will be decreased to (maximum threshold - 2). On the other hand, if  $aql$  is between the target  $aql$  and the maximum threshold, the maximum threshold will be decreased to the value of (maximum threshold + 2).

The highest value of the maximum threshold equals to (router buffer capacity - minimum threshold) since congestion is required to be identified and controlled early and before the router buffer gets overflowed (Wang et al., 2004). Moreover, the highest mean waiting time for packets can be calculated based on the highest maximum threshold. The lowest value of the maximum threshold has been chosen as ( $2 \times$  minimum threshold) in order to maintain the throughput performance. Lastly, the maximum threshold decreases or increases by 2 when congestion occurs to adjust the  $aql$  in a slow manner, thus, arriving packets will overflow.

## 4. Experimental Results

This section presents the simulation details of the following methods: REDD, Drop-tail, RED and AMT and the results obtained using the following performance measure:  $mql$ ,  $T$ ,  $D$ ,  $P_L$  and  $D_p$ . We would like to identify the AQM method with better performance

results. The performance measure results are obtained based on setting the packet arrival probability parameter to variable values.

### 4.1 Settings

Information like arrival and departure processes, the number of router buffers and scheduling manner for arriving packets are discussed in this sub-section. Packet inter-arrival times and service times are geometrically distributed by means  $1/\alpha$  and  $1/\beta$ , where  $\alpha$  and  $\beta$  are probabilities of packets arrival and departure respectively. The arrival process used to model the arriving packets is a Bernoulli process. A geometrically distribution is used for the departing packets. Moreover, Markov-Modulated Bernoulli Process (MMBP) (Ng et al., 1999) is also used to model the arrival process in order to absorb the properties of burstiness and correlation. Two states such as 0 and 1 of MMBP (2-MMBP) have been employed for simplifying the analysis. The packets are generated by two traffics (traffic 1 and traffic 2), where the source of traffic 1 is state 0 and for traffic 2 is state 1.

The arrival process may stay in state 0 at slot  $n$ , and next time at  $(n+1)$  with probability  $X$ , and can generate arrival packets with probability  $\alpha_0$ . On the other hand, when the arrival process is at state 1 at slot  $n$ , the probability that the arrival process may stay at the same state at slot  $(n+1)$  is  $Y$ , and the arrival process can generate arrival packets with probability  $\alpha_1$ . The transition probability from state 0 to state 1 is  $(1-x)$ , and from state 1 to state 0 is  $(1-y)$ . The 2-MMBP can be summarised in two matrices: The state matrix of probability transition (see equation 4) and probabilities matrix of packet arrival (see equation 5). Fig. 3 shows the 2-MMBP with both transition probabilities of states and arrival probabilities of packets.

$$T = \begin{bmatrix} x & 1-x \\ 1-y & y \end{bmatrix} \dots\dots\dots (4)$$

$$\alpha = \begin{bmatrix} \alpha_0 & 0 \\ 0 & \alpha_1 \end{bmatrix} \dots\dots\dots (5)$$

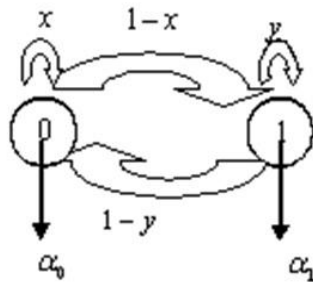


Fig. 3:2-MMBP source model

The balance equations of the 2-MMBP are:

$$p_0 = xp_0 + (1-y)p_1 \dots\dots\dots (6)$$

$$p_1 = (1-x)p_0 + yp_1 \dots\dots\dots (7)$$

In equations 6 and 7,  $p_0$  and  $p_1$  are the steady state probabilities of states 0 and 1, where  $p_0$  is the steady state probability that state 0 contains MMBP and  $p_1$  is the steady state probability that state 1 contains MMBP.

The simulations of the compared methods are performed using a discrete-time queue approach (Xylomenos et al., 2011). In a discrete-time queue, a time unit called slot is utilised. For every slot, packet arrival and/or departure may occur. Moreover, neither packet arrival nor departure may exist. It is considered that a packet may arrive to and/or depart from the router buffer. An assumption that the number of router buffers for every considered method is 1, and the scheduling manner for arriving packets is First Come First Served (FCFS). The compared methods have been implemented using Java on Core 2 Duo Centrino processor with 1.66 GHz and on 1 GB RAM. The simulated router buffers are shown in Fig. 4-6. In which Fig. 4 and 5 illustrate the router buffers of Drop-tail and RED methods respectively. Also the router buffer method of the AMT and REDD is depicted in Fig. 6.

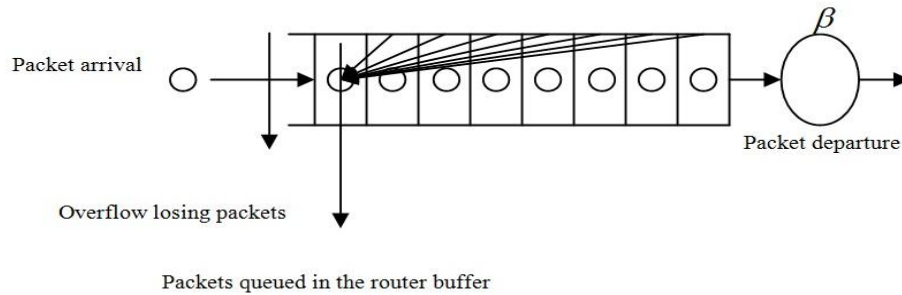


Fig. 4: The single router buffer for Drop-tail.

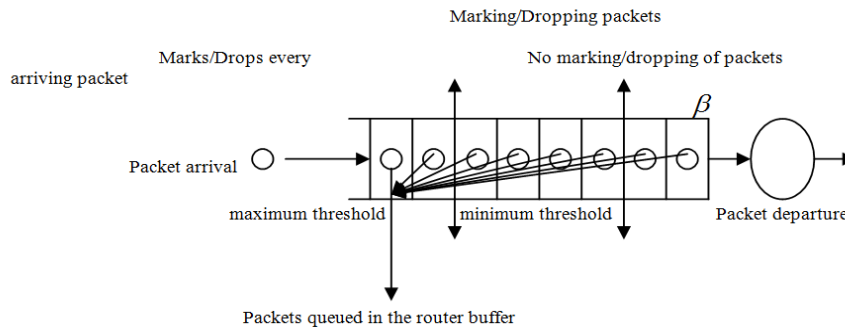


Fig. 5: The single router buffer for either AMT/REDD



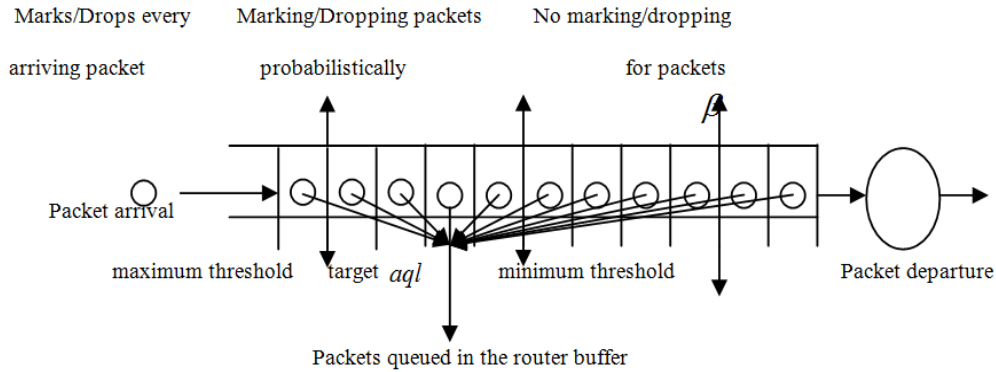


Fig. 6: The single router buffer for RED

## 4.2 Performance Evaluation

The performance measure results, i.e.  $mql$ ,  $T$ ,  $D$ ,  $P_L$  and  $D_p$  of Drop-tail, RED, AMT and REDD are conducted in order to identify the method with better performance. We used Bernoulli and 2-MMBP processes as described later in this section. Table 3 shows the parameter setting values for Drop-tail, RED, AMT and REDD. The parameters of  $qw$  and  $pdmax$  have been set to 0.002 and 0.1 respectively as in RED. Packet departure probability is set to 0.5 and the packet arrival probability is set to different values, i.e. (0.15-0.9) in order to evaluate the performance measure results before and after congestion. The target  $aql$  has been tuned to  $(\text{minimum threshold} + \text{maximum threshold})/2$ . The minimum threshold value has been set to 3 since the router buffer size,

Table 3: The setting parameter values of the considered methods

Parameters	Drop-tail	RED	AMT	REDD
Probability of packet arrival 1	0.15-0.9	0.15-0.9	0.15-0.9	0.15-0.9
Probability of packet arrival 2	0.5	0.5	0.5	0.5
Probability of packet departure	0.5	0.5	0.5	0.5
Router buffer capacity	20	20	20	20
Minimum threshold	Not used	3	3	3
Maximum threshold or Maximum threshold (2)	Not used	9	9	9
Maximum threshold(1)	Not used	Not used	20	Not used
$qw$	Not used	0.002	0.002	0.002
$pdmax$	Not used	0.1	0.1	0.1
Target $aql$	Not used	Not used	(minimum threshold + maximum threshold) / 2	(minimum threshold + maximum threshold) / 2
X	0.9	0.9	0.9	0.9
Y	0.9	0.9	0.9	0.9
Number of slots	2000000	2000000	2000000	2000000

i.e. 20, is small, and the maximum threshold value has been set as in RED. Moreover, the maximum threshold (1) and maximum threshold (2) are set as in (Ali et al., 2007).  $X$  and  $Y$  parameters are set to 0.9 for enabling the arrival process to stay in the same state with larger probability than that when they move to another state. The number of slots that has been used in the simulations is large, i.e. 2000000 to provide more accurate results.

**4.2.1 The Performance Measure Results Using Bernoulli Process**

The measures used in evaluating the performance of the methods in this section are:

**1. Mean Queue Length ( $mql$ )**

Obtaining a small result for  $mql$  can avoid building up the queue size of the router buffer, and thus, the congestion probability will be reduced. Further, the  $mql$  participates in calculating the packets mean waiting time ( $D$ ). The  $mql$  can be computed as follows:

$$mql = \sum_{i=0}^K i \times p_i \dots\dots\dots (8)$$

Where  $K$  is the router buffer capacity and  $p_i$  are the steady state probabilities ( $p_0 - p_K$ )

**2. Throughput ( $T$ )**

$T$  denotes the amount of packets that passed through the router buffer successfully per unit time.  $T$  assists in evaluating  $D$ . The  $T$  can be calculated as:

$$T = \beta \sum_{i=1}^K p_i = \beta (1 - p_0) \dots\dots\dots (9)$$

where  $\beta$  is the probability of packet departure

**3. Packets Mean Waiting Time ( $D$ )**

$D$  can be computed using a Little's formula (Xylomenos et al., 2011) as below:

$$D = \frac{mql}{T} \text{ slots} \dots\dots\dots (10)$$

**4. Overflow Packet Loss Probability ( $P_L$ ):** It is the packet loss probability due to an overflow. It is computed as:

$$P_L = \frac{\text{number\_of\_lost\_packets}}{\text{number\_of\_arrived\_packets}} \dots\dots\dots (11)$$

**5. Packet Dropping Probability ( $D_p$ ):** It is the packet loss probability before the router buffer has a full size.  $D_p$  can be computed as:

$$D_p = \frac{\text{number\_of\_dropped\_packets}}{\text{number\_of\_arrived\_packets}} \dots\dots\dots (12)$$

The performance measures of the considered methods are computed according to variable values of the packet arrival probability. After the detection of a steady state of the system, the performance results can be achieved. Every compared simulation method has a ten runs for each packet arrival probability value. In each run, a different seed is used for a random number generator to minimise bias results and to provide confidence

intervals for the performance measures. The result for every measure is the mean of the ten runs at a value of packet arrival probability. The performance results versus the packet arrival probability values are given in Fig. 7-11, where the results of  $mql$ ,  $T$  and  $D$  are shown in Fig. 7-9 respectively and the results of  $P_L$  and  $D_p$  are respectively illustrated in Fig. 10-11.

From the figures, and when no congestion or light congestion happens, i.e. packet arrival probability  $\leq$  packet departure probability, all considered methods offer similar  $mql$ ,  $D$ ,  $P_L$  and  $D_p$  results. This is since all of them build their router buffer at a similar time and may begin dropping packets with the exception of Drop-tail when packet arrival probability = 0.45. Moreover, Fig. 8 shows that the results of  $T$  for all compared methods are similar in any situation.

Fig. 7 and 9 indicate that REDD generates smaller  $mql$  and  $D$  numbers than Drop-tail, RED and AMT in congestion scenarios. This is because REDD starts dropping packets earlier than the rest of the methods (see Fig. 11). Moreover, Drop-tail provides the highest  $mql$  and  $D$  results when congestion is triggered since its router buffer starts dropping packets late. Furthermore, when congestion occurs the Drop-tail method loses more packets since its buffer gets overloaded more frequently than the other methods. In addition, when the value of packet arrival probability is larger than 0.6, AMT loses fewer packets than most of the considered methods since it gets overloaded early. Finally, both RED and REDD provide similar  $P_L$  results when the value of packet arrival probability is less than 0.9; whereas when the value reaches 0.9, the  $P_L$  results of REDD are less than that of RED.

Fig. 11 shows that Drop-tail does not drop packets early and thus congestion can be discovered late. Accordingly, the  $D_p$  results for Drop-tail are equal to zero. The figure reveals that RED and REDD offer greater  $D_p$  results than AMT when the value of packet arrival is equal 0.6. Whereas AMT provides higher  $D_p$  than the two methods when the value of packet arrival probability is larger than or equal to 0.75.

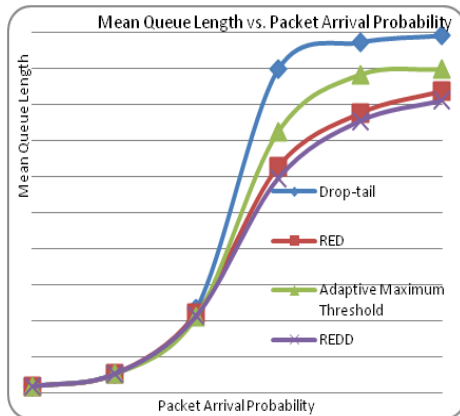


Fig. 7:  $mql$  vs. probability of packet arrival.

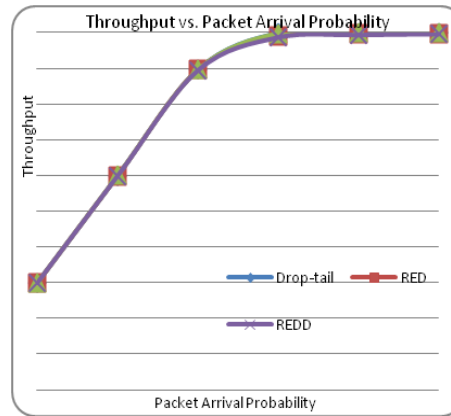


Fig. 8:  $T$  vs. probability of packet arrival.

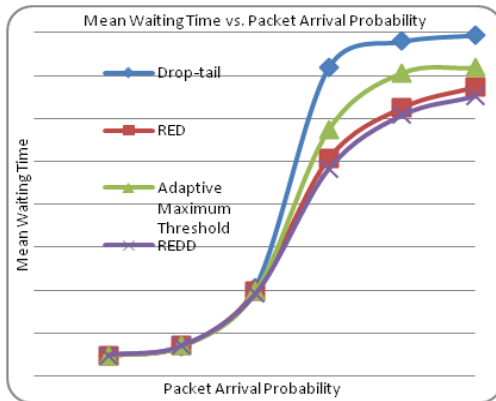


Fig. 9:  $D$  vs. probability of packet arrival1.

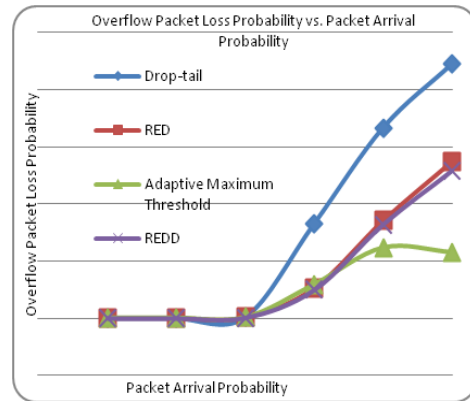


Fig. 10:  $P_L$  vs. probability of packet arrival1.

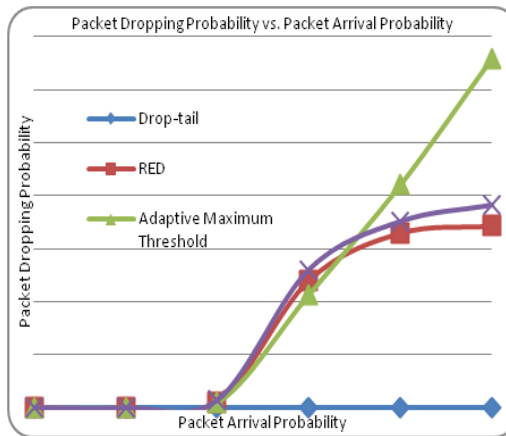


Fig. 11:  $D_p$  vs. probability of packet arrival1.

#### 4.2.2 The Performance Measure Results Using 2-MMBP Process

In this subsection, 2-MMBP is used as the arrival process for modelling the source model. 2-MMBP is applied in order to deal with the properties of burstiness and correlations within packets. Therefore, the performance measure results will take into account burstiness and correlations properties. The judgment of which method offers more satisfactory performance is only provided based on the values of packet arrival probability, and after the system reaches a steady state. Same as the previous section, each method has ten runs per packet arrival probability value, and different seeds for a random number generator are utilised.

The results of performance measures versus the values of packet arrival probability are illustrated in Tables 4-6 and Fig. 12-13. Tables 4-6 present the results of  $mql$ ,  $T$  and  $D$ , respectively, and  $P_L$  and  $D_p$  results are introduced in Fig. 12 and 13, respectively. It is noted from Table 5 that all methods offer similar  $T$  results in any situation.

Tables 4, 6 and Fig. 7, 8 demonstrate that the compared methods provide similar performance results with reference to  $mql$ ,  $D$ ,  $P_L$  and  $D_p$  when the probability of packet

arrival is less than or equal 0.3. When the value of packet arrival probability increases to be greater than 0.3, both RED and REDD present more satisfactory performance than Drop-tail and AMT with reference to  $mql$  and  $D$  results. Also,  $mql$  and  $D$  results of REDD are slightly better than those results of RED.

In Fig. 12, when the value of packet arrival is greater than 0.3, the Drop-tail offers the largest  $P_L$  results among the other methods since it loses more packets and it overflows earlier than the other methods. Furthermore, AMT loses more packets than RED and REDD when the value of packet arrival probability is greater than 0.45 and smaller than 0.9, and both RED and REDD offer similar  $P_L$  and  $D_p$  results (see Fig. 13) since they lose and drop similar number of packets. For the  $P_L$ , and when the packet arrival probability value is equal to 0.45 or at least 0.9, AMT, RED and REDD provide similar  $P_L$  results.

Table 4:  $mql$  results of the compared methods based on 2-MMBP process.

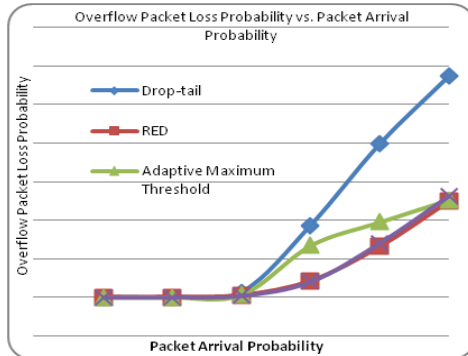
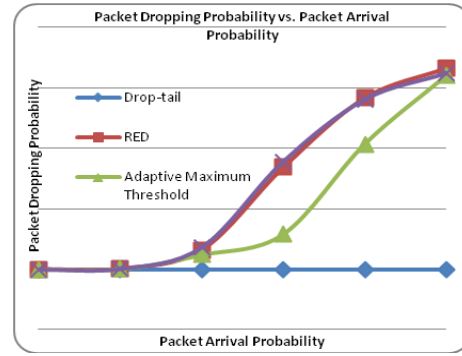
$mql$ results based on 2-MMBP				
Probability of packet arrival	Drop-tail	AMT	RED	REDD
0.15	1.610	1.610	1.610	1.610
0.3	2.596	2.585	2.580	2.579
0.45	6.5775	5.9129	5.6876	5.5156
0.6	15.9281	13.8848	10.1024	9.7090
0.75	18.2127	14.8296	12.6857	12.5850
0.9	18.8995	16.3875	14.1665	14.1155

Table 5:  $T$  results of the compared methods based on 2-MMBP process.

$T$ results based on 2-MMBP				
Probability of packet arrival	Drop-tail	AMT	RED	REDD
0.15	0.3264	0.3264	0.3264	0.3264
0.3	0.400	0.400	0.400	0.400
0.45	0.4692	0.4662	0.4652	0.4640
0.6	0.4993	0.4964	0.4911	0.4886
0.75	0.4999	0.4983	0.4949	0.4927
0.9	0.5	0.4996	0.4958	0.4941

Table 6:  $D$  results of the compared methods based on 2-MMBP process.

$D$ results based on 2-MMBP				
Probability of packet arrival	Drop-tail	AMT	RED	REDD
0.15	4.932	4.932	4.932	4.932
0.3	6.4787	6.4530	6.4401	6.4390
0.45	14.0129	12.6807	12.2234	11.8852
0.6	31.8974	27.9677	20.5710	19.8695
0.75	36.4273	29.7582	25.6278	25.5381
0.9	37.7991	32.8000	28.5702	28.5640

Fig. 12:  $P_L$  vs. probability of packet arrival1.Fig. 13:  $D_p$  vs. probability of packet arrival1.

#### 4. Conclusions and Future Work

A new AQM method named Random Early Dynamic Detection (REDD) has been proposed in this paper in order to overcome some problems associated with RED method particularly RED's reliance on the input parameters. Difference performance measures and a number of known AQM methods have been utilized to show the pros and cons of REDD in the Bernoulli and 2-MMBP processes. In particular, REDD has been compared with Drop-tail, RED and AMT with reference to  $mql$ ,  $T$ ,  $D$ ,  $P_L$  and  $D_p$  measures in order to discover the one with better performance. To obtain accurate results, for a method, each packet arrival probability value has ten times by changing the seed value for a random number generator. All the considered methods have been implemented using Java. The experimental results can be summarized as follows:

Using Bernoulli process

- All compared methods offered similar  $T$  results in any situation. This is also true when using 2-MMBP process.
- All methods provided similar  $mql$ ,  $D$ ,  $P_L$  and  $D_p$  results when no congestion or light congestion occurs.
- REDD generated smaller  $mql$  and  $D$  results than Drop-tail, RED and AMT in congestion scenarios.
- Drop-tail lost the largest number of packets due to early buffers overflow when congestion happens. In addition, both RED and REDD provided greater  $D_p$  results than AMT when the value of packet arrival is equal 0.6.

Using 2-MMBP process

- The compared methods offered similar  $mql$ ,  $D$ ,  $P_L$  and  $D_p$  results when the probability of packet arrival is less than or equal 0.3.
- RED and REDD offered better  $mql$  and  $D$  results than Drop-tail and AMT when the value of packet arrival probability is greater than 0.3.
- Drop-tail provided largest  $P_L$  results among other methods when the value of packet arrival is greater than 0.3.
- AMT, RED and REDD produced similar  $P_L$  and  $D_p$  results when the packet arrival probability value is equal 0.45 or at least 0.9.

This study pursues the following suggestions for future work:

- Apply REDD at the router buffers of internet as congestion control method.
- Use batch of arrivals and/or batch of departures with the proposed method instead of single arrival and departure.

## References

- Ababneh J., Abdeljaber H., Thabtah F., Hadi W., and Badarneh E. (2011) Derivation of Three Queue Nodes Discrete-Time Analytical Model Based on DRED Algorithm. The Seventh IEEE International Conference on Information Technology: New Generations (ITNG 2011). IEEE Computer Society, pp. 885-890, April 2011, Las Vegas, USA.
- Abdeljaber H., Thabtah F., Woodward M. (2008a) Traffic Management for the Gentle Random Early Detection using Discrete-time Queueing. Proceedings of the International Business Information Management Conference (9th IBIMA), pp. 289-298, Marrakech, Morocco, January 2008.
- Abdeljaber H., Woodward M., Thabtah F. (2008b) Performance evaluation for DRED discrete-time queueing network analytical model. Proceedings of the Journal of Network and Computer Applications, Volume 31, Issue 4, pp. 750-770, Elsevier, November 2008.
- Ahamed G.F., and Banu, R. (2011) Analyzing the Performance of Active Queue Management Algorithms. International Journal of Computer Networks (IJCN), volume 2, pp. 1-19. no. 2, March 2011.
- Ali A.G., Woodward M.E., and Etbega M. (2007) Two Different Approaches of Active Queue Management. Proceedings of the 2007 IEEE International Conference on Networking, Sensing and Control, pp. 579-583, London, U.K., April 2007.
- Athuraliya S., Li V.H., Low S.H., and Yin, Q. (2001) REM: Active Queue Management. IEEE Network, 15(3), pp. 48-53, May 2001.
- Aweya J., Ouellette M., and Montuno D.Y. (2001) A Control Theoretic Approach to Active Queue Management. Computer Net., volume 36, issue 2-3, pp. 203-35, July 2001.
- Braden R., Clark D., Crowcroft J., Davie B., Deering S., Estrin D., Floyd S., Jacobson V., Minshall G., Partridge C., Peterson L., Ramakrishnan K., Shenker S., Wroclawski J., and Zhang L (1998) Recommendations on Queue Management and Congestion Avoidance in the Internet. RFC 2309, April 1998.
- Brandauer C., Iannaccone G., Diot C., Ziegler T., Fdida S., and May M. (2001) Comparison of Tail Drop and Active Queue Management Performance for bulk-data and Web-like Internet Traffic. In Proceeding of ISCC, pp. 122-129, IEEE, July 2001.
- Chrysostomou, C., Pitsillides, A., Hadjipollas, G., Sekercioglu, A., and Polycarpou, M. (2003) Fuzzy Explicit Marking for Congestion Control in Differentiated Services Networks. Proceedings of the Eight IEEE International Symposium on Computers and Communication (ISCC'03), volume 1, pp.312-319, 2003.
- Feng W., Kapadia A., and Thulasidasan S. (2002) GREEN: Proactive Queue Management over a Best-Effort Network. The Proceeding of IEEE Global Telecommunications Conference, GLOBECOM '02, volume 2, pp. 1774-1778, Taipei, Taiwan, LA-UR 02-5524, November 2002.
- Feng W., Shin K.G., and Kandlur D. (2002) The Blue Active Queue Management Algorithms. IEEE/ACM Transactions on Networking, volume 10, issue 4, pp. 513-528, August 2002.
- Floyd S., and Jacobson V. (1993) Random Early Detection Gateways for Congestion Avoidance. IEEE/ACM Transactions on Networking, volume 1, issue 4, 1(4): pp. 397-413, August 1993.
- Floyd S., Ramakrishnan G., and Shenker S. (2001) Adaptive RED: An Algorithm for Increasing the Robustness of RED's Active Queue Management. Technical report, ICSI, August 1, 2001.

- Floyd S. (2000) Recommendations on using the gentle variant of RED. May 2000, Available at <http://www.aciri.org/floyd/red/gentle.html>.
- Lieu X., and He W. (2007) Active Queue Management Design Using Discrete-Event Control. Decision and Control, 2007 46th IEEE Conference, New Orleans, LA, December 2007.
- Ng, C., C., Yuan, L., Fu, W., and Zhang, L., (1999) Methodology for traffic modeling using two-state Markov-modulated Bernoulli process. *Computer Communications* (13) (1999) 1266–1273.
- Stallings, W. (2009) *Data and Computer Communications*. Eighth edition. Pearson Prentice Hall, Pearson international edition, 2009.
- Tanenbaum A.S. (2002) *Computer Networks*, Fourth Edition, Prentice Hall PTR, 2002.
- Thabtah F., Woodward M., Abdeljaber H., Mahafzeh M., Magld K., Aldiabat M. (2009) A Traffic Load Comparison of RED and DRED Analytical Models. Third International Conference on Internet Technologies and Applications, Wrexham North Wales, U.K. September 2009.
- Wang C., Li B., Hou T., Soharaby K., and Lin Y. (2004) LRED: A Robust Active Queue Management Scheme Based on Packet Loss Ration. INFOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies. Volume 1, pp. 1-12, March 2004.
- Xylomenos, George; Katsaros, Konstantinos; Tsakanikas, Vasilis (2011) Support of multiple content variants in the multimedia broadcast/multicast service. *International Journal of Communication Systems*, Volume 24 Issue 6, pp. 691-708, June 2011.
- Zhang, Wei-jiong; Guan, Zhang-jun; Zhou, Xi-lang (2011) Analysis of SER of MIMO-MRC systems with imperfect channel estimation in the presence of non-Rayleigh CCIs. *International Journal of Communication Systems*, Volume 24 Issue 7, pp. 889-901, July 2011.
- Zhong, Sheng; Chen, Tingting (2011) An efficient identity-based protocol for private matching. *International Journal of Communication Systems*, Volume 24, Issue 4, pages 543–552, April 2011.
- Zhou, LiFeng; Chen, Lei; Pung, Hung Keng; Ngoh, Lek Heng (2011) Identifying QoS violations through statistical end-to-end analysis. *International Journal of Communication Systems*, Volume 24, Issue 10, pages 1388–1406, October 2011.
- Zhou W.H., and Wang A.H. (2008) Discrete-time queue with Bernoulli bursty source arrival and generally distributed service times. *The Journal of Applied Mathematical Modelling*, volume 32, pp. 2233-2240, November 2008.

Received November 7, 2013, revised February 13, 2014, accepted February 17, 2014