King Saud University

Journal of King Saud University – Computer and Information Sciences

www.ksu.edu.sa
www.sciencedirect.com

Performance study of Active Queue Management methods: Adaptive GRED, REDD, and GRED-Linear analytical model

Hussein Abdel-jaber

Faculty of Computer Studies, Department of Information Technology and Computing, Arab Open University, Saudi Arabia

Received 23 July 2014; revised 4 December 2014; accepted 26 January 2015
Available online 11 September 2015

KEYWORDS

Adaptive GRED;
Discrete-time queues;
GRED Linear analytical model;
Performance measures;
REDD

Abstract Congestion control is one of the hot research topics that helps maintain the performance of computer networks. This paper compares three Active Queue Management (AQM) methods, namely, Adaptive Gentle Random Early Detection (Adaptive GRED), Random Early Dynamic Detection (REDD), and GRED Linear analytical model with respect to different performance measures. Adaptive GRED and REDD are implemented based on simulation, whereas GRED Linear is implemented as a discrete-time analytical model. Several performance measures are used to evaluate the effectiveness of the compared methods mainly mean queue length, throughput, average queueing delay, overflow packet loss probability, and packet dropping probability. The ultimate aim is to identify the method that offers the highest satisfactory performance in non-congestion or congestion scenarios. The first comparison results that are based on different packet arrival probability values show that GRED Linear provides better mean queue length; average queueing delay and packet overflow probability than Adaptive GRED and REDD methods in the presence of congestion. Further and using the same evaluation measures, Adaptive GRED offers a more satisfactory performance than REDD when heavy congestion is present. When the finite capacity of queue values varies the GRED Linear model provides the highest satisfactory performance with reference to mean queue length and average queueing delay and all the compared methods provide similar throughput performance. However, when the finite capacity value is large, the compared methods have similar results in regard to probabilities of both packet overflowing and packet dropping.

1. Introduction

With the emergence of modern communications and computer networks, network sources require sufficient resources to deliver their data to their destinations (Tanenbaum, 2002). Insufficient resources lead to congestion, which happens when the amount of incoming packets exceeds the available network

E-mail address: habdeljaber@arabou.edu.sa

Peer review under responsibility of King Saud University.

ELSEVIER Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.jksuci.2015.01.003

resources (Welzl, 2005). Congestion causes several forms of degradation in network performance such as: (1) instability in average queue length, so many arriving packets will be dropped because of the congested contents of router buffers, (2) high loss and average queueing delay of packets, (3) low throughput, and (4) unbalanced share of network resources among network sources.

Congestion control is important to improve network performance and can enable efficient use of networks by users (Aweya et al., 2001; Welzl, 2005). Congestion control is a challenge because of the sensitivity of network traffic (Welzl, 2005). Congestion control has become an active research field because of the advancement of real-time or demand traffic. At present, several media traffic applications, such as video conferencing, voice over IP (VoIP), video on demand (VoD), video streaming, and so on, can be found online. The amount of users who use these media traffic applications are increasing rapidly, which can cause congestion. When congestion is present in network applications based on TCP, TCP flows will decrease their transmitting rate to manage congestion. Therefore, a fair share of network flows is generated. By contrast, when congestion occurs within non-TCP network applications, non-TCP flows will keep transmitting in their original rate, which leads to an unfair share of network resources.

Several researchers have proposed congestion control mechanisms such as Drop-tail (Brandauer et al., 2001) and Active Queue Management (AQM) Abdeljaber et al., 2011; Aweya et al., 2001; Feng et al., 1999; Floyd and Jacobson, 1993; Floyd et al., 2001; Floyd, 2000. Drop-tail determines congestion when router buffers are overflowed (Braden et al., 1998; Brandauer et al., 2001). Therefore, congestion is identified after the router buffers become full, which means that congestion in Drop-tail is found at a late stage. AQM mechanisms identify congestion before the router buffers become full. In other words, AQM mechanisms discover congestion at an early stage. Random Early Detection (RED) (Floyd and Jacobson, 1993) and its variants (Floyd et al., 2001; Floyd, 2000) were proposed to control congestion as AQM mechanisms and were built based on simulation. The variants of RED have been proposed to deal with the deficiencies of RED, such as the following: (1) The congestion measure of RED (average queue length (*aql*)) varies with the congestion level such that when light congestion exists, the *aql* value is near the minimum threshold position (min *threshold*) on the RED router buffer. On the contrary, when heavy congestion occurs, the *aql* value is near the maximum threshold position (max *threshold*) on the RED router buffer. (2) RED is sensitive to its parameters (min *threshold*, max *threshold*, maximum value of packet dropping probability (*pd* max), and queue weight (*qw*)). (3) The *aql* value relies on the number of TCP flows. Thus, the *aql* value may exceed the max *threshold* position (heavy congestion) when the number of TCP flows is high, which means that every arriving packet will be dropped. (4) RED cannot maintain the *aql* value between min *threshold* and max *threshold* positions when busy traffic is present. Therefore, the *aql* value may exceed the max *threshold* position, which leads to heavy congestion.

The current paper aims to evaluate the performance of three AQM methods under a single queue node. Two of the AQM methods, which are Adaptive GRED and REDD, are implemented based on simulation, whereas the last AQM method was built as a discrete-time queue analytical model, and is called the GRED Linear analytical model.

This paper critically compares congestion control methods primarily Adaptive GRED, REDD, and GRED Linear using different evaluation measures (*mql*, T, D, P_L, and D_P). This paper aims to identify the method that gives the best performance after deeply analyzing the packet dropping probability which affects mean queue length; average queueing delay and packet overflow. The results of performance measure results can help in choosing the method that can be applied as a congestion control method in computer networks such as the Internet.

The paper is structured as follows: Related works are reviewed in Section 2. Sections 3–5 present the Adaptive GRED method, GRED Linear analytical model, and REDD method, respectively. A previous comparison between the GRED and Adaptive GRED methods is presented in Section 6. The simulation details of Adaptive GRED and REDD are demonstrated in Section 7. Section 8 presents different analytical modeling approaches besides their verification and validation using Simulation. Finally, Sections 9 and 10 show the results of the performance evaluation of Adaptive GRED, REDD, and GRED Linear based on varying values of packet arrival probability and varying values of finite capacity of queues. The conclusions and suggestions for future work are presented in Section 11.

2. Related work

Other examples of AQM mechanisms are analytical models, which were proposed based on discrete-time queue approach (Abdeljaber et al., 2008, 2008; Al-Diabat et al., 2012) and different AQM mechanisms based on simulation. Woodward wrote a book about building discrete-time queue analytical models, in which these analytical models were built by modeling and analyzing the performance of queuing systems, such as computer communications and networks (Woodward, 1993). Abdeljaber et al. (2008, 2008) and Al-Diabat et al. (2012) were built as discrete-time queue analytical models based on DRED, GRED, and BLUE mechanisms. Another analytical model based on RED was presented in Bonald et al. (2000). These analytical models controlled congestion by decreasing packet arrival probability either from fixed value to another or linearly. (Lim et al., 2011) was proposed as an analytical model and uses aggregated internet traffic as the sources of different classes of traffic. This analytical model works as a queue management mechanism for sustaining queueing delay at a specific level on the router buffer (Lim et al., 2011). This developed model utilizes the control method of closed loop feedback (Lim et al., 2011) to limit average queuing delay by implicitly informing the arrival rate and by moving the queuing threshold (Lim et al., 2011). The proposed model derived the relationship between the queueing thresholds and average queueing delay based on the traffic model, which represents aggregated internet traffic using the superposition of the arrival processes of N Markov Modulated Bernoulli Process-2 (MMBP-2). The value of the queueing threshold is revised based on the derived relationship between the queueing thresholds and average queueing delay, as well as the evaluation of average queueing delay feedback. Packets can be dropped dynamically in terms of the revisions of queueing threshold values and events of packet loss (Lim et al., 2011).

Wang et al. (2011) was introduced as a modeling system and as an analysis method for a single buffer that estimates queueing delay and controls the average queueing delay to a necessitated value within various traffic network flows. The sensitive delay of the application can be improved in terms of its Quality of Service (QoS) performance using this method (Wang et al., 2011). In Wang et al. (2011), an analytical model based on discrete-time queues was proposed to derive a relationship between average queueing delay and queueing thresholds. This analytical model used various traffic flows that have been modeled using a binomial distribution as an arrival process (Wang et al., 2011). Queue thresholds are revised as a controlling method for the average queueing delay. Packets are dropped as congestion notifications to the arrival process to update their rate.

Al-Bahadili et al. (2011) proposed an analytical model for a network that contains multi-queue nodes. This model derived two performance measures, namely, system utilization and queue node. Al-Bahadili et al. (2011) presented two scenarios that demonstrated the variation of system utilization and queue node versus the dropping probabilities of queue nodes under several routing probabilities and system sizes.

Kamoun (2006) presented a precise transient and equilibrium analysis of the discrete-time queuing of a statistical multiplexer using a correlated arrival process with a limited number of input links. This model also used a constant number of constant packet lengths. This analytical model helped obtain the generating function with reference to transient probability of the buffer capacity (Kamoun, 2006).

Andrzej and Chrost (2011) studied a probability based on queue length and the time when job arrivals are blocked then lost. AQM algorithms were used in this study. The outputs of this study include analytical solutions, queue length distribution, packet loss, and throughput. These analytical solutions also contain blocking probabilities such as dropping functions.

3. Adaptive GRED

Adaptive GRED was proposed to enhance the performance measures of GRED in terms of *mql*, *T*, and P_L (Abdeljaber et al., 2011). Similar to GRED, Adaptive GRED uses *aql* as a congestion measure (Abdeljaber et al., 2011). Adaptive GRED was also developed as a congestion control approach that identifies early congestion within networks (Abdeljaber et al., 2011). In addition, Adaptive GRED aimed to improve the parameter tunings of max *threshold* (a position at the Adaptive GRED router buffer) and *pd* max (the maximum value of dropping probability) of GRED (Floyd, 2000). In Adaptive GRED, when a packet arrives at the router buffer and the router buffer calculates, the *aql* and compares it with three threshold positions at the Adaptive GRED router buffer. These threshold positions are min *threshold*, max *threshold*, and *double* max *threshold*. The max *threshold* is set to at least two times of max *threshold* (Floyd and Jacobson, 1993), whereas *double* max *threshold* is tuned to double the max *threshold* (Floyd, 2000). Congestion is not present if the *aql* value is less than the min *threshold* value and no packets are dropped. Congestion occurs and the router buffer drops packets probabilistically when the *aql* value is equal to or larger than the min *threshold* value and smaller than the *double* max *threshold* value. For instance, when max $threshold \leqslant aql < double$max *threshold*, the router buffer drops packets with probability $D_{\max} + \frac{\frac{(1-D_{\max})}{2} \times (aql - \max\ threshold)}{\max\ threshold}$ (Abdeljaber et al., 2011).

The packet dropping probability value varies from *pd* max to 0.5 because the *aql* value varies from max *threshold* to *double* max *threshold*. Thus, Adaptive GRED enhances GRED (Floyd, 2000) when configuring the parameters of max *threshold* and *pd* max.

Finally, heavy congestion occurs and the router buffer drops every arriving packet if the *aql* value is equal to or greater than the *double* max *threshold* value.

4. GRED Linear analytical model

The GRED Linear model is a discrete-time queue analytical model that was proposed to model and analyze the performance of a single queue node (see Fig. 1) (Abdeljaber et al., 2008). The results of the performance measure of the GRED Linear model can be used as the performance results of a congestion control method. Therefore, the GRED Linear model can be used as a congestion control method. In the GRED Linear model, if the current queue length (*ql*) is less than the min *threshold* position, congestion will not occur and no packets can be dropped ($D_p = 0.0$). Furthermore, the packet arrival probability to the GRED Linear router buffer is $\alpha 1$. Congestion is present if the current *ql* value is equal to or larger than the value of min *threshold* and less than the value of max *threshold*. To control congestion, the packet arrival probability value decreases linearly from $\alpha 1$ to α_i, where *i* is the queue state and $\alpha_i = \alpha 1 - (1 + i - \min\ threshold)\frac{(\alpha 1 - \alpha 2)}{(1 + double\max\ threshold - \min\ threshold)}$, if min $threshold \leqslant i < double$max *threshold* and the α_i relies upon the state transition diagram (see Fig. 2), this partial suggestion method implies that the packet arrival probability value decreases based on the queue state (*i*), and it is decreased linearly when *i* is greater than or equal to min *threshold* and less than *double* max *threshold* positions at the router buffer. The value of D_p increases linearly. Thus, $\left(\frac{\alpha 1 - \alpha_i}{\alpha 1}\right)$, where min $threshold \leqslant i < double$ max *threshold*, this indicates that congestion packet dropping probability increases linearly on one side and the packet arrival probability decreases linearly to manage the congestion on the other side. Lastly, when queue state *i* increases to be equal to or greater than *double* max *threshold* a heavy congestion occurs. To manage this congestion, the value of packet arrival probability is then decreased by changing it from α_i to $\alpha 2$, and D_p value is then increased by changing it from $\left(\frac{\alpha 1 - \alpha_i}{\alpha 1}\right)$ to $\left(\frac{\alpha 1 - \alpha 2}{\alpha 1}\right)$.

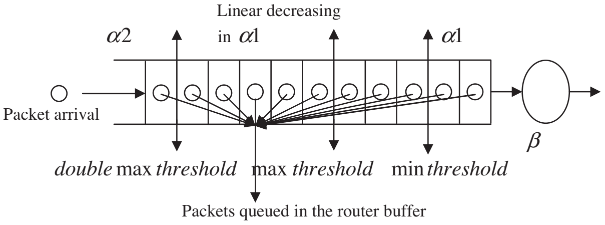

Figure 1 The single queue node system for the GRED Linear model.

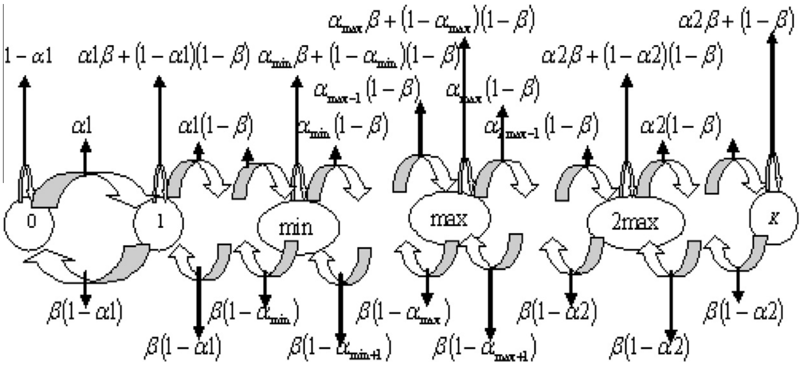
$1-\alpha 1$
$\alpha 1\beta+(1-\alpha 1)(1-\beta)$
$\alpha_{\min}\beta+(1-\alpha_{\min})(1-\beta)$
$\alpha_{\max}\beta+(1-\alpha_{\max})(1-\beta)$
$\alpha 2\beta+(1-\alpha 2)(1-\beta)$
$\alpha 2\beta+(1-\beta)$
$\alpha 1$
$\alpha 1(1-\beta)$
$\alpha_{\min}(1-\beta)$
$\alpha_{\max-1}(1-\beta)$
$\alpha_{\max}(1-\beta)$
$\alpha_{2\max-1}(1-\beta)$
$\alpha 2(1-\beta)$
0
1
min
max
2max
K
$\beta(1-\alpha 1)$
$\beta(1-\alpha 1)$
$\beta(1-\alpha_{\min})$
$\beta(1-\alpha_{\min+1})$
$\beta(1-\alpha_{\max})$
$\beta(1-\alpha_{\max+1})$
$\beta(1-\alpha 2)$
$\beta(1-\alpha 2)$
$\beta(1-\alpha 2)$

Figure 2 The state transition diagram for the GRED-Linear analytical model.

The probabilities of packet arrival to the GRED Linear router buffer in a fixed time are $\alpha 1$, α_i, and $\alpha 2$. When the current *ql* result is smaller than the min *threshold*, *ql* is larger than or equal to min *threshold* and less than the *double* max *threshold*, and *ql* is equal to or greater than the *double* max *threshold*, respectively. β is the probability of packet departure from the router buffer in a slot. Similarly, identical independently distributed (i.i.d) Bernoulli process (Abdeljaber et al., 2008) is used to model the packet arrivals, $a_n \in \{0, 1\}$, $n = 0, 1, 2, \ldots$, where a_n represents the number of packet arrivals in slot n.

The packet departures are modeled using geometrical distribution. The inter-arrival times of packets are geometrically distributed using the following equation: $\frac{1}{\alpha 1}$, $\frac{1}{\alpha_i}$, and $\frac{1}{\alpha 2}$. Packet departure times are geometrically distributed with a mean of $\frac{1}{\beta}$. K refer to the finite router buffer capacity of packets that contains the packet is currently in service. The remaining parameters of the GRED Linear model (min *threshold*, max *threshold*, *double* max *threshold*) are similar to those for the Adaptive GRED method. The queueing system of the GRED Linear model is First Come First Served (FCFS). The single queue node system is considered to be in equilibrium. The queue length process is a Markov chain with finite state space, such as {0, 1, 2, 3, …, min *threshold*, max *threshold*, *double* max *threshold*, …, $K - 1$, K}. Assuming that half of the $\alpha 1$ values are larger than the β value and the other half are smaller than the β value, such that $\alpha 1 > \alpha_i$, $\alpha_i > \alpha 2$. Fig. 2 shows the state transition diagram of the GRED Linear model.

The balance equations for the GRED Linear model are evaluated in Eqs. (1)–(10) using Fig. 2.

$$p_0 = (1-\alpha 1)p_0 + [\beta(1-\alpha 1)]p_1 \tag{1}$$

$$p_1 = \alpha 1 p_0 + [\alpha 1\beta + (1-\alpha 1)(1-\beta)]p_1 + [\beta(1-\alpha 1)]p_2 \tag{2}$$

In general:

$$p_i = [\alpha 1(1-\beta)]p_{i-1} + [\alpha 1\beta + (1-\alpha 1)(1-\beta)]p_i + [\beta(1-\alpha 1)]p_{i+1}, \quad \text{where } i = 2,3,4,\ldots,\min \textit{threshold} - 2 \tag{3}$$

$$\begin{aligned} p_{\min\ \textit{threshold}-1} = {} & [\alpha 1(1-\beta)]p_{\min\ \textit{threshold}-2} \\ & + [\alpha 1\beta + (1-\alpha 1)(1-\beta)]p_{\min\ \textit{threshold}-1} \\ & + [\beta(1-\alpha_{\min\ \textit{threshold}})]p_{\min\ \textit{threshold}} \end{aligned} \tag{4}$$

$$\begin{aligned} p_{\min\ \textit{threshold}} = {} & [\alpha 1(1-\beta)]p_{\min\ \textit{threshold}-1} + [\alpha_{\min\ \textit{threshold}}\beta \\ & + (1-\alpha_{\min\ \textit{threshold}})(1-\beta)]p_{\min\ \textit{threshold}} \\ & + [\beta(1-\alpha_{\min\ \textit{threshold}+1})]p_{\min\ \textit{threshold}+1} \end{aligned} \tag{5}$$

$$p_i = [\alpha_{i-1}(1-\beta)]p_{i-1} + [\alpha_i\beta + (1-\alpha_i)(1-\beta)]p_i + [\beta(1-\alpha_{i+1})]p_{i+1}, \tag{6}$$

where i = min *threshold* + 1, min *threshold* + 2, min *threshold* + 3, …, *double* max *threshold* − 2

$$\begin{aligned} p_{\textit{double}\ \max\ \textit{threshold}-1} = {} & [\alpha_{\textit{double}\ \max\ \textit{threshold}-2}(1-\beta)]p_{\textit{double}\ \max\ \textit{threshold}-2} \\ & + \left[\begin{array}{c} \alpha_{\textit{double}\ \max\ \textit{threshold}-1}\beta \\ +(1-\alpha_{\textit{double}\ \max\ \textit{threshold}-1})(1-\beta) \end{array}\right] \\ & \times p_{\textit{double}\ \max\ \textit{threshold}-1} + [\beta(1-\alpha 2)]p_{\textit{double}\ \max\ \textit{threshold}} \end{aligned} \tag{7}$$

$$\begin{aligned} p_{\textit{double}\ \max\ \textit{threshold}} = {} & [\alpha_{\textit{double}\ \max\ \textit{threshold}-1}(1-\beta)]p_{\textit{double}\ \max\ \textit{threshold}-1} \\ & + [\alpha 2\beta + (1-\alpha 2)(1-\beta)]p_{\textit{double}\ \max\ \textit{threshold}} \\ & + [\beta(1-\alpha 2)]p_{\textit{double}\ \max\ \textit{threshold}+1} \end{aligned} \tag{8}$$

$$\begin{aligned} & p_i = [\alpha 2(1-\beta)]p_{i-1} + [\alpha 2\beta + (1-\alpha 2)(1-\beta)]p_i + [\beta(1-\alpha 2)]p_{i+1}, \\ & \text{where } i = \textit{double} \max \textit{threshold} + 1,\ \textit{double} \max \textit{threshold} \\ & +2,\ldots,K-1 \end{aligned} \tag{9}$$

Finally,

$$p_K = [\alpha 2(1-\beta)]p_{K-1} + [\alpha 2\beta + (1-\beta)]p_K \tag{10}$$

where K = *double* max *threshold* + X, *double* max *threshold* = max *threshold* + J and max *threshold* = min *threshold* + I. Therefore, K can also be given as:

$$K = \min \textit{threshold} + I + J + X \tag{11}$$

$$\text{Let } \gamma i = \frac{\alpha i(1-\beta)}{\beta(1-\alpha i)}, \quad i = 1,2. \tag{12}$$

$$\text{and } \gamma_i = \frac{\alpha_i(1-\beta)}{\beta(1-\alpha_i)}, \quad \text{where } \min \textit{threshold} \leqslant i < \textit{double} \max \textit{threshold} \tag{13}$$

Eq. (13) is used to simplify obtaining the equilibrium probabilities (p_i, i = min *threshold*, min *threshold* + 1, min *threshold* + 2, …, *double* max *threshold* − 1), and these probabilities are contributed in providing the performance measures of the GRED Linear analytical model. The equation contents represent at each queue state i, the packet arrival probability at this queue state $i(\alpha_i)$ multiplied by one minus the probability of packet departure $(1 - \beta)$. Then the result is divided by the multiply of the probability of packet departure (β) by one minus the packet arrival probability at this queue state $i(1 - \alpha_i)$.

The equilibrium probabilities of the GRED Linear model are computed by solving the balance equations recursively and by applying Eqs. (12) and (13). The equilibrium probabilities are shown in Eqs. (14)–(16).

$$\sum_{i=0}^{K} p_i = 1 \tag{17}$$

$$p_0 = \left[\frac{1-\gamma 1^{\min\ threshold} - \beta(1-\gamma 1)}{(1-\beta)(1-\gamma 1)} + \frac{\gamma 1^{\min\ threshold}(1-\alpha 1)}{(1-\beta)} \sum_{i=\min\ threshold}^{double\ \max\ threshold-1} \prod_{j=\min\ threshold}^{i-1} (\gamma_j).\frac{1}{(1-\alpha_i)} + \frac{\gamma 1^{\min\ threshold}(1-\alpha 1)(1-\gamma 2^{X+1})}{(1-\beta)(1-\alpha 2)(1-\gamma 2)} \prod_{l=\min\ threshold}^{double\ \max\ threshold-1} (\gamma_l) \right]^{-1} \tag{18}$$

In general,

$$p_i = \frac{\alpha 1^i (1-\beta)^{i-1}}{\beta^i (1-\alpha 1)^i} p_0 = \frac{\gamma 1^i}{(1-\beta)} p_0,$$
$$\text{where } i = 1, 2, 3, \ldots, \min\ threshold - 1 \tag{14}$$

Eq. (14) represents the equilibrium probabilities ($p_0, p_1, \ldots, p_{\min\ threshold-1}$) of the GRED Linear analytical model, and again these probabilities are contributed in providing the performance measures of GRED Linear analytical model.

The performance measures of the GRED Linear model are calculated as follows. First, *mql*, which is the mean queue length, should be small as possible to avoid building up the contents of the router buffer. The *mql* is computed using the generating function $P(z)$, which is illustrated in Eq. (19). The *mql* is equal to the first derivative of $P(z)$ at $z = 1$, as in Eq. (20).

$$P(z) = \sum_{i=0}^{K} z^i p_i \tag{19}$$

$$mql = P^{(1)}(1) = \frac{p_0}{(1-\beta)} \left[\begin{array}{l} \frac{\gamma 1 - \gamma 1^{\min\ threshold}[\gamma 1 + \min\ threshold(1-\gamma 1)]}{(1-\gamma 1)^2} + \gamma 1^{\min\ threshold}(1-\alpha 1) \\ \left(\sum_{i=\min\ threshold}^{double \max\ threshold-1} \prod_{j=\min\ threshold}^{i-1} (\gamma_j).\frac{i}{(1-\alpha_i)} + \frac{double\ \max\ threshold(1-\gamma 2)(1-\gamma 2^{X+1}) + \gamma 2 - \gamma 2^{X+1}[1+X(1-\gamma 2)]}{(1-\alpha 2)(1-\gamma 2)^2} \prod_{l=\min\ threshold}^{double\ \max\ threshold-1} (\gamma_l) \right) \end{array} \right] \tag{20}$$

$$p_i = \frac{\alpha 1^{\min\ threshold}(1-\beta)^{i-1} \prod_{j=\min\ threshold}^{i-1} \alpha_j}{\beta^i (1-\alpha 1)^{\min\ threshold-1} \prod_{j=\min\ threshold}^{i} (1-\alpha_j)} p_0$$
$$= \frac{\gamma 1^{\min\ threshold}(1-\alpha 1) \prod_{j=\min\ threshold}^{i-1} \gamma_j}{(1-\alpha_i)(1-\beta)} p_0. \tag{15}$$

where i = min *threshold*, min *threshold* + 1, min *threshold* + 2, …, *double* max *threshold* – 1 and $\alpha_j = 1$, if $j <$ min *threshold*.

Second, T is the throughput, which can be defined as the number of packets that successfully passed through the router buffer. T is obtained in Eq. (21).

$$T = \beta \sum_{i=1}^{K} \prod_{i} = \beta(1-p_0)\text{packets/slot} \tag{21}$$

Third, D is the average queueing delay for packets and can be calculated using Little's law $\left(\frac{mql}{T}\right)$ (Woodward, 1993). D is found in Eq. (22).

$$p_{double\ \max\ threshold+i} = \frac{\alpha 1^{\min\ threshold} \alpha 2^i (1-\beta)^{double\ \max\ threshold+i-1} \prod_{j=\min\ threshold}^{double\ \max\ threshold-1} \alpha_j}{\beta^{double\ \max\ threshold+i}(1-\alpha 1)^{\min\ threshold-1}(1-\alpha 2)^{i+1} \prod_{j=\min\ threshold}^{double\ \max\ threshold-1} (1-\alpha_j)} p_0$$
$$= \frac{\gamma 1^{\min\ threshold} \gamma 2^i (1-\alpha 1) \prod_{j=\min\ threshold}^{double\ \max\ threshold-1} \gamma_j}{(1-\alpha 2)(1-\beta)} p_0$$
$$\text{where } i = 0, 1, 2, \ldots, X \tag{16}$$

By applying the equilibrium probabilities in the normalizing equation displayed in Eq. (17), the probability of no packets in the single node system (p_0) is calculated and p_0 is obtained in Eq. (18).

$$D = \frac{mql}{T} slots = \frac{P^{(1)}(1)}{T} slots = \frac{\sum_{i=0}^{K} i \times p_i}{T} slots \tag{22}$$

Fourth, P_L is the probability of packet loss caused by the overflowing of the router buffer. Eq. (23) demonstrates the P_L result.

$$P_L = (1-\beta)p_K, \tag{23}$$

where p_K is the probability of the single queue node system is full, and K is the finite capacity of the single queue node.

Finally, the probability of dropped packets before the router buffer is full is represented by D_P. D_P is calculated in Eq. (24).

$$D_p = \sum_{i=\min\ threshold}^{double\ \max\ threshold-1} \left(\frac{\alpha 1-\alpha_i}{\alpha 1}\right) \times p_i + \left(\frac{\alpha 1-\alpha 2}{\alpha 1}\right) \times \sum_{i=double\ \max\ threshold}^{K-1} p_i \tag{24}$$

where $\sum_{i=double\ \max\ threshold}^{K-1} p_i$ represents the sum of equilibrium probabilities ($p_{double\ \max\ threshold}$, $p_{double\ \max\ threshold+1}$, $p_{double\ \max\ threshold+2,\ldots,}$ p_{K-1}) multiplied by $\left(\frac{\alpha 1-\alpha 2}{\alpha 1}\right)$, the result of this represents the packet dropping probability when queue state i is greater than or equal to *double* max *threshold* and less than K.

5. REDD

REDD (Abdeljaber et al., 2014) was proposed to overcome the RED's problems that are (1) the congestion measure (average queue length (*aql)*) of RED varies with the congestion level such that when light congestion exists, the *aql* value is near the minimum threshold position (min *threshold*) on the RED router buffer. On the contrary, when heavy congestion occurs, the *aql* value is near the maximum threshold position (max *threshold*) on the RED router buffer, and (2) the probability of exceeding the position of the max *threshold* when the number of sources increases with the variety of data traffic, which indicates heavy congestion. Therefore, every arriving packet will be dropped. REDD overcomes the earlier problems using an adaptive max *threshold* position. REDD calculates an *aql* value similar to that of RED for every arriving packet at the router buffer of REDD. REDD decreases the max *threshold* value by 2 if the value of *aql* is between the min *threshold* value and the target *aql* value and the value of the adjusted max *threshold* at least twice the min *threshold* value. This decrease will move the *aql* toward its target value. By contrast, if the *aql* value is greater than the Target *aql* value and the value of the max *threshold* is less than or equal to the difference between finite buffer capacity (*K*) and the min *threshold* value, then the value of the max *threshold* will be increased by 2, which will move the *aql* value toward the target *aql* value to stabilize it on target *aql*. This change also offers two more rooms for the max *threshold* value to increase the throughput performance. Selecting "2" for decreasing and increasing the max *threshold* value can help specify congestion and precede the current number of packets that reach the finite capacity of the router buffer. Fig. 3 shows the pseudo-code of the REDD method.

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

Figure 3 The pseudo-code of REDD.

The parameters of REDD, such as *pdmax*, *qw*, and others, are set as follows (Abdeljaber et al., 2014): *pdmax*, *qw*, and min *threshold* are set to the same values in RED (Abdeljaber et al., 2014). The value of *aql* is set as the middle value between the min *threshold* and max *threshold*. The initial value of max *threshold* is three times the min *threshold* value. The value of max *threshold* value relies on the value of *aql*. If the value of *aql* is less than the min *threshold*, then the max *threshold* is set to three times the min *threshold*. If the value of *aql* is equal to or greater than the value of (*K*-min *threshold*), then the value of max *threshold* will be set to the value of (*K*-min *threshold*). If the *aql* value is between the min *threshold* and the Target *aql* values, then the max *threshold* value will be reduced to (max *threshold* − 2). If the *aql* value is between the target *aql* and the max *threshold* values, then the max *threshold* value will be increased to (max *threshold* + 2).

The max *threshold* was set to (*K*-min *threshold*) as the highest value to identify and control congestion in an early stage and before the router buffer has overflowed (Floyd and Jacobson, 1993). The max *threshold* value was set to (2 xmin *threshold*) to maintain throughput performance. The value of "2" was chosen to decrease or increase the max *threshold* value when congestion is present is to adjust the *aql* value slowly, which can provide more possibility in identifying congestion when heavy traffic occurs.

6. Previous comparison between Adaptive GRED and GRED methods

The Adaptive GRED method was compared with the GRED method to identify the method that offers better performance measures (Abdeljaber et al., 2011). Abdeljaber et al. (2011) showed that the Adaptive GRED and GRED methods provided similar *mql* and *D* results when the values of packet arrival probability are either 0.18 or 0.33. When packet arrival probability values are increased such that ⩾ 0.48, Adaptive GRED outperformed GRED with reference to the results of *mql* and *D*. Adaptive GRED and GRED generated similar *T*

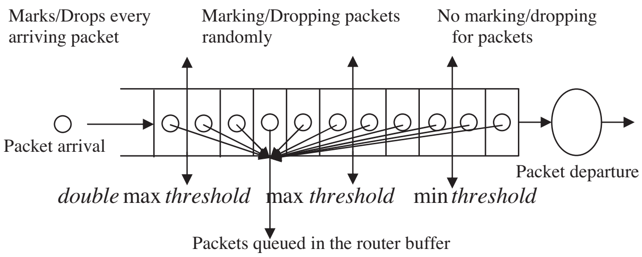
Marks/Drops every arriving packet
Marking/Dropping packets randomly
No marking/dropping for packets
Packet arrival
Packet departure
double max *threshold*
max *threshold*
min *threshold*
Packets queued in the router buffer

Figure 4 The single router buffer for Adaptive GRED.

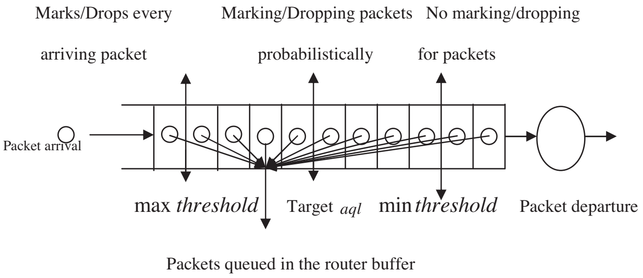

Figure 5 The single router buffer for REDD.

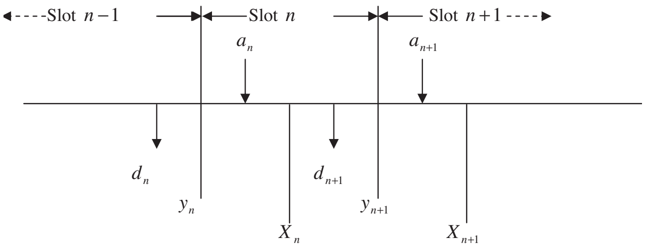

Figure 6 The state of the discrete-time queue with arrivals and departures in each slot (Woodward, 1993).

results regardless of the value of packet arrival probability. Adaptive GRED and GRED offered similar P_L and D_P results when the values of packet arrival probability are $\leqslant 0.63$. However, Adaptive GRED presented better P_L results than GRED when the values of packet arrival probability are > 0.63, whereas GRED outperformed Adaptive GRED in terms of D_P results.

7. Simulation details of Adaptive GRED and REDD

The probability of packet arrival in a fixed time unit named slot (Woodward, 1993) to the Adaptive GRED/REDD router buffer is $\alpha 1$. β is the probability of packet departure from the router buffer of Adaptive GRED/REDD in a slot. The Bernoulli process (Woodward, 1993) is used to model the packet arrivals, whereas packet departures are modeled using geometrical distribution. Packet inter-arrival times are geometrically distributed with mean$\frac{1}{\alpha 1}$. Packet departure times are geometrically distributed with mean $\frac{1}{\beta}$.

Packet arrival or departure probability occurs in a slot. Every slot can hold packet arrival and/or departure event(s) or none of these events (Woodward, 1993). The Adaptive GRED and REDD simulate the network of a single queue node, which is shown in Figs. 4 and 5, respectively. Packets arrived to and departed from a single router in a single manner. The queueing discipline for the network is FCFS.

8. Different analytical modeling approaches

An analytical model is built by modeling and analyzing queueing network systems. The results of an analytical model can be the following: (1) The balance equations, (2) the steady state probabilities, and the performance measures such as mean queue length, throughput, etc. Generally, every queueing system can be described by Kendall's notation by five components (Woodward, 1993) as follows:

1. The arrival process: A stochastic process that shows how customers (packets) arrive to the queueing system. The arrival process is denoted A.
2. The service process: A stochastic process that illustrates the amount of time spent by a customer (packet) in the server. The service process is denoted B.
3. The number of servers (C).
4. The system capacity: It represents the maximum number of customers (packets) inside the system including packets currently in the service. The system capacity is denoted K.
5. The customer population: It represents the limit on the total number of customers who participate in the arrival process. The customer population is denoted P.

It should be noted that there is another factor related to Kendall's components called the queueing service discipline, which can be defined as the set of laws that make a decision of which customer in the queueing system should be served, i.e. first come first served (FCFS), last come first served (LCFS) and processor sharing (PS) are examples of this.

The K and P components can be removed where in this case these components are considered to be infinite values. The C, K and P components are positive integers, and A and B components are selected based on the set of descriptors.

This section presents two different approaches to build analytical models depending on modeling and analyzing the queueing systems, these are continuous-time queues (Ross, 2010) and discrete-time queues (Woodward, 1993).

8.1. Discrete-time queue approach

Discrete-time queue is an approach to model and analyze the performance of queueing systems in communication and computer networks (Woodward, 1993). In discrete-time queues, the interarrival and service times are "geometrically" distributed, and the exponential distribution is possible in cases of multiple arrivals and departures. A basic time unit called a slot is used, where in each slot single or multiple events may occur. An example of a single event is the occurrence of a packet arrival or departure, whereas both packet arrivals and departures may occur in multiple events. Often, packet arrivals take place after the start of a slot, and packet departures happen before the end of the slot. The number of arrivals and departures in a slot n are defined by $\{a_n, n = 0, 1, 2, \ldots\}$ and $\{d_{n+1}, n = 0, 1, 2, \ldots\}$, respectively, where $\{a_n\}$ denotes the sequence of identical and independently distributed (i.i.d) random variables with a specific distribution. $d_0 = 0$ since no packets can depart before they have arrived. The state of a discrete-time queue with arrivals and departures in each slot is depicted in Fig. 6. The process of queue length at boundaries of slots is represented by $\{y_n, n = 0, 1, 2, 3, \ldots\}$ with y_0 is arbitrary. Therefore, Eq. (25) is presented.

$$y_{n+1} = y_n + a_n - d_{n+1} \tag{25}$$

Eq. (26) denotes the process of the queue length after the arrivals happen $\{X_n = n = 0, 1, 2, \ldots\}$.

$$X_{n+1} = X_n - d_{n+1} + a_{n+1} \tag{26}$$

This latter convention (Eq. (26)), sometimes called "departure first", will be used in the models in this paper.

One of the stochastic processes is called the Markov process, and one of its particular kinds is Markov Chain (Woodward, 1993). The Markov process is a stochastic process that is specialized by a Markov property called Memoryless property that can be existed to be a property in exponential distributions as well as in the interarrival times of a Poisson process. Thus, the Poisson process is a special case of Markov process.

Assume $x = \{x_n = x; n = 0, 1, 2, \ldots\}$, $x \in X$ is a Markov chain, where x_n is the state at time n, n is the time index that provided with discretized the time, n obtains successive nonnegative integer values $\{0, 1, 2, \ldots\}$. x is the states that can be have numbers with nonnegative integer values in the set of $\{0, 1, 2, \ldots\}$, and X is the state space $\{0, 1, 2, \ldots\}$ (Woodward, 1993).

The expression of Markov property can be given as follows:

$$P(x_{n+1} = j | x_0, x_1, x_2, \ldots, x_n) = (x_{n+1} = j | x_n) j \in X, \quad n = 0, 1, 2, \ldots$$

This means in a given state at time n, the state at time $n + 1$ is independent with all past states at times 0, 1, 2, ..., $n - 1$. The evolution of Markov chain is explained by its one step transition probabilities $P_{ij}(n)$, that given a state i at time n, the chain will move to state j at time $n + 1$.

$$P_{ij}(n) = P(X_{n+1} = j | X_n = i) \quad i, j \in X, \quad n = 0, 1, 2, \ldots$$

An example on a Markov chain that one step transition probability does not rely on the time index n is called time-homogeneous (Woodward, 1993), and this is provided as follows:

$$p_{i,j}(n) = p_{ij}, \quad i, j \in X, \quad n = 0, 1, 2, \ldots \tag{27}$$

The following are examples of discrete-time queues that can be modeled and analyzed to build analytical models:

M/D/1 queueing system might be used in modeling a buffer for a user in a network of computer communication (Woodward, 1993). M in this queueing system represents that this system has geometrically distributed interarrival times with zero or one packet is allowed to arrive in a slot. Also, D represents that this system has constant service times with zero or one packet is allowed to serve in a slot. The number of servers is one, the system capacity and customer population are infinite, and the queueing discipline is FCFS.

The second example for discrete-time queueing system is M/M/1, the first M denotes that the queueing system has geometrically distributed interarrival times with one or zero packet is allowed to arrive in a slot. The second M denotes that the system has geometrically distributed service times with zero or one departure is allowed in a slot.

8.2. Continuous-time queue approach

Continuous-time queues are utilized in modeling and analyzing the performance queueing systems in communication and computer networks (Woodward, 1993). Continuous-time queues use Kendall's notation as discrete-time queues, but A and B are chosen when the set of descriptors are Markovian (M), the interarrival and service time distributions are exponential in continuous-time queues. For instance, the interarrival time is a Poisson process and the service time is "exponentially" distributed. The continuous-time Marcov chain is a process $\{x(t), t \geqslant 0\}$ with nonnegative values in $\{0, 1, 2, 3, \ldots\}$, for example

Provided the past, present and future are independent", such that, for all y, $t \geqslant 0$ and all states i, j, $x(u)$,

$$\begin{aligned} P(X(t+y) = j | X(y) = i, \ X(u) = x(u), 0 \leqslant u < y) \\ = P(X(t+y) = j | X(y) = i). \end{aligned} \tag{28}$$

The function of stationary transition probability is as follows:

$$Pij(t) = P(X(t+y) = j | X(y) = i). \tag{29}$$

Examples of continuous-time queues is M/M/1 queueing system where in this system the first M represents that the system has exponentially distributed interarrival times with permitted at most one packet to be arrived at a specific time. The second M represents that the system has exponential service times distribution with permitted at most one packet to be departed at a specific time. The number of servers is one. The values for the system capacity and the customer population are infinite. The queueing discipline is FCFS.

8.3. Verification and validation of analytical models using simulation

In the previous subsections we present the results of analytical models and here the verification of analytical models using computer simulation is discussed. Validation of an analytical model means discovering that the built model is an accurate representation for actual system that has been analyzed (Ignall et al., 1978; Pace, 2004). In many occasions, the solutions of analytical models considered circumstances that do not hold in the real world, this makes the analysts unsure of using them. One way to overcome this issue is validating the analytical models using simulation. The simulation models can be employed in giving an approximate or to achieve behavior description of a system. Once the results of the simulation are validated then the analytical model can be used in a safe way in describing the behavior of a system.

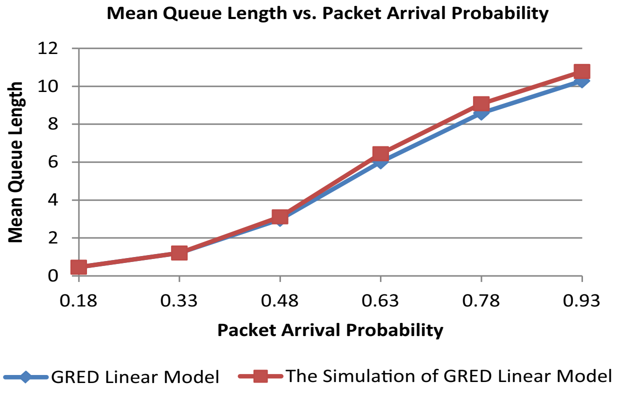

Figure 7 *mql* vs. α1.

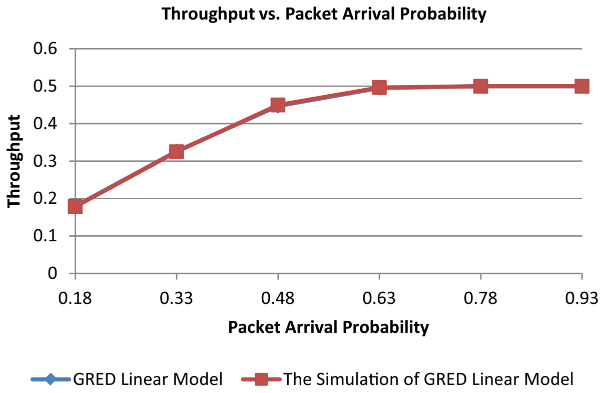

Figure 8 T vs. $\alpha 1$.

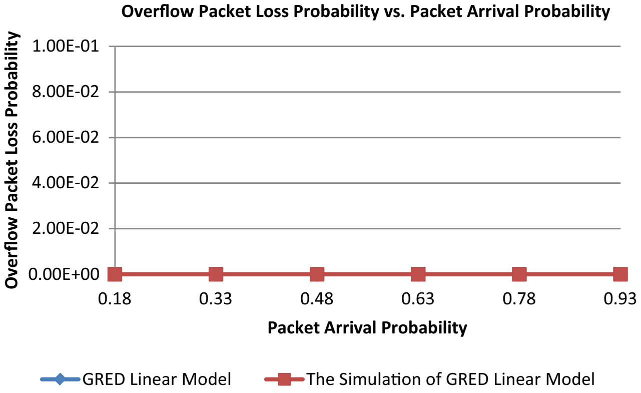

Figure 10 P_L vs. $\alpha 1$.

8.3.1. Verification and validation of GRED-Linear analytical model

This subsection aims to verify and validate the GRED-Linear model by developing a computer simulation program which it simulates the single queue node introduced in Fig. 1. In simulation program the congestion measure is the same as in the GRED-Linear model, which is the queue length. The validation of the GRED-Linear model is presented by offering the performance measure results (*mql*, T, D, P_L, and D_P) of the simulation program that almost match those results of the GRED-Linear model.

The performance measure results of GRED-Linear and the simulation program versus the packet arrival probability are illustrated in Figs. 7–11. In particular, Figs. 7–9 show the results of *mql*, T and D versus $\alpha 1$, respectively, and the results of P_L and D_P versus $\alpha 1$ appear in Figs. 10 and 11, respectively.

The parameters of GRED-Linear and the simulation program are set as follows: $\alpha 1$, $\alpha 2$, and β, which are set to [0.18–0.93], 0.1, and 0.5, respectively, to create situations of congestion (i.e., $\alpha 1 > \beta$) and non-congestion (i.e., $\alpha 1 < \beta$). The finite capacity (K) of the router buffer is set to 20 packets to measure the performance with small buffer sizes. The values of min *threshold*, max *threshold* are set to 3 and 9, respectively, similar to those of the GRED method (Floyd, 2000). The *double* max *threshold* is set to 18, similar to the GRED method (Floyd, 2000). Slots are the time unit used in discrete-time queues approach (Woodward, 1993) to produce a warming up period that expires when the system reaches a steady state. The length of slots is set to 2,000,000. The GRED-Linear model and the simulation program of GRED-Linear are implemented using a Java environment in a computer with the following specifications: i7 processor 1.66 GHz and 4 GB RAM.

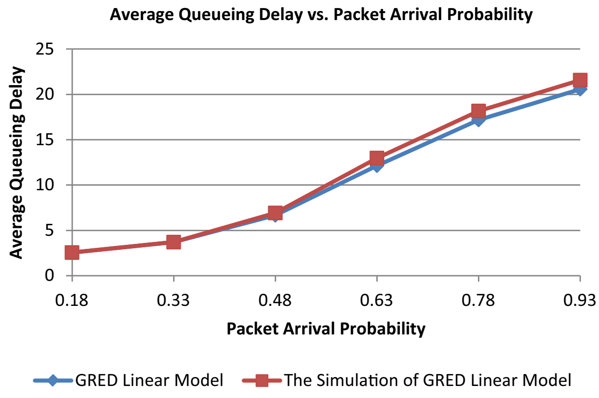

Figure 9 D vs. $\alpha 1$.

It is noted in Figs. 7–11 that the performance measure results of the program simulation are almost similar to their corresponded performance results for GRED-Linear. This is a proof that the GRED-Linear model is verified and validated by almost matching the performance measure results of the simulation program. This leads to that the GRED-Linear model is a right model and operating correctly, and it can be used as an approximation model for the simulation program model in describing the behavior of the single queue node system given in Fig. 1. Furthermore, the performance measure results of the GRED-Linear model can be used to describe the performance evaluation of the single queue node system that shown in Fig. 1.

9. Performance evaluation of Adaptive GRED, REDD, and GRED Linear based on variation of $\alpha 1$ parameter

This section compares the Adaptive GRED, REDD, and GRED Linear analytical models to identify the method that offers the best performance measure results. The performance measures obtained are: *mql*, T, D, P_L, and D_P. The parameters of Adaptive GRED, REDD, and GRED Linear are set as follows: $\alpha 1$, $\alpha 2$, β, and K which are set to values as those given in Subsection 8.3.1. As mentioned earlier, the setting values of $\alpha 1$ are aiming to create situations of congestion (i.e., $\alpha 1 > \beta$) and non-congestion (i.e., $\alpha 1 < \beta$). Therefore, the current study can evaluate the performances of the Adaptive GRED, REDD and GRED Linear models with and without congestion, and setting K to 20 is for the same reason mentioned before that is to measure the performance with small buffer sizes. The values of min *threshold*, max *threshold*, *qw*, and *pd* max are set to 3, 9, 0.02, and 0.1, respectively, similar to those of the RED method (Floyd and Jacobson, 1993). The *double* max *threshold* is set to 18, similar to the GRED method (Floyd, 2000). Slots are the time unit used in discrete-time queue approach (Woodward, 1993) to produce a warming up period that expires when the system reaches a steady state. Adaptive

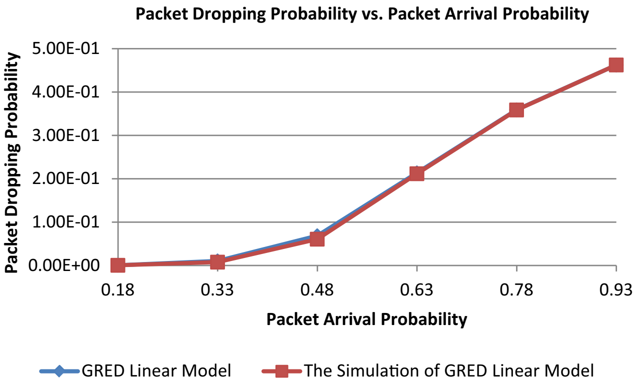

Figure 11 D_P vs. α1.

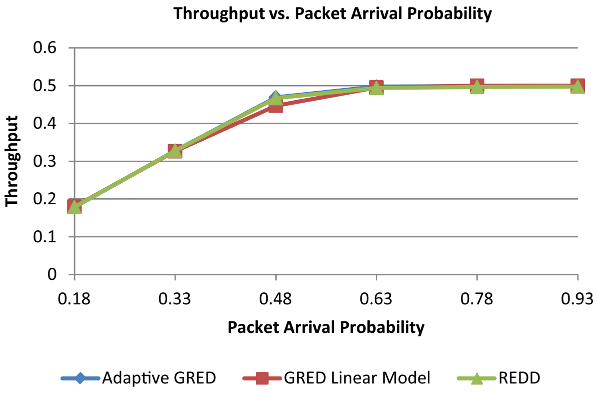

Figure 13 T vs. α1.

GRED, REDD, and GRED Linear are implemented using Java environment with i7 processor 1.66 GHz and 4 GB RAM and the length of slots is set to 2,000,000.

Deciding whether Adaptive GRED, REDD, or GRED Linear gives better performance results than others is solely based on the values of α1. The performance measure results are obtained after the system reaches the steady state. Each value for α1 is run 10 times. The seed number for the random value generator is changed in every run to remove bias in the performance measure results. For each α1 value, the performance measure result is the mean of 10 ten run times for that of α1 value.

Figs. 12–16 show the performance measure results compared with the α1 values. Figs. 12–14 show *mql*, T, and D versus α1, respectively. Figs. 15 and 16 show the performance measure results with regard to P_L and D_P versus α1, respectively.

Figs. 12 and 14 show that the Adaptive GRED, REDD, and GRED models offer similar *mql* and D results when $\alpha 1 \leqslant 0.33$. However, congestion occurs when α1 is increased (i.e., $\alpha 1 \geqslant 0.48$). In this situation, the GRED Linear model provides smaller *mql* and D results than either Adaptive GRED or REDD. This result is attributed to the fact that the GRED Linear model drops more packets than either Adaptive GRED or REDD when congestion is present (see Fig. 16). Adaptive GRED and REDD offer comparable *mql* and D results when α1 is increased up to 0.63 because both of these methods offer similar D_P results (see Fig. 16). Adaptive GRED provides less *mql* and D results than those of REDD when $\alpha 1 > 0.63$ because Adaptive GRED drops further packets than REDD.

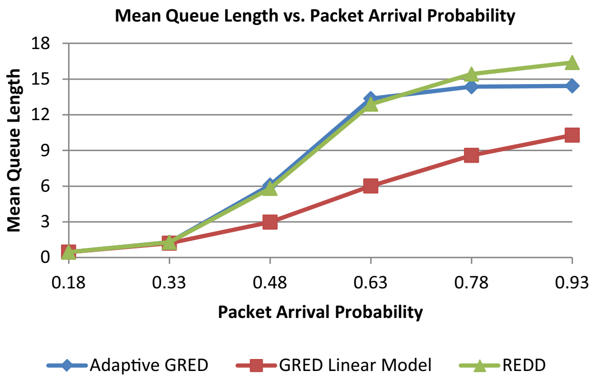

Figure 12 *mql* vs. α1.

Fig. 13 shows that the Adaptive GRED, REDD, and GRED Linear models generate similar T results at all values of α1, except 0.48. Hence, the Adaptive GRED and REDD offer similar T results when $\alpha 1 = 0.48$, which are marginally higher than those of the GRED Linear model. This result is attributed to the fact that the GRED Linear model serves less packets than Adaptive GRED and REDD.

An analysis of Fig. 15 shows that the compared methods offer comparable P_L results when $\alpha 1 \leqslant 0.48$ because the router buffers of these methods overflow in a similar number of times. GRED Linear provides less P_L results than both of Adaptive GRED and REDD when $\alpha 1 \geqslant 0.63$ because the number of times of router buffer overflow in the GRED Linear model is at the lowest. Moreover, Adaptive GRED and REDD provide similar P_L results when $\alpha 1 = 0.63$. However, Adaptive GRED loses packets because of fewer overflow than REDD when heavy congestion occurs ($\alpha 1 > 0.63$) because REDD drops fewer packets before the router buffer is full compared with that of Adaptive GRED (see Fig. 16).

Fig. 16 shows that the compared methods drop similar numbers of packets before the router buffers are full when $\alpha 1 \leqslant 0.33$. The GRED Linear model drops more packets than Adaptive GRED or REDD when $\alpha 1 > 0.33$ because the GRED Linear model offers less *mql* results than those of Adaptive GRED or REDD. Adaptive GRED and REDD offer similar D_p results when α1 is up to 0.63. REDD drops fewer packets than Adaptive GRED when $\alpha 1 \geqslant 0.63$ because the *mql* results of REDD are higher than those of Adaptive GRED.

10. Performance evaluation of Adaptive GRED, REDD, and GRED Linear based on variation of *K* parameter

Another comparison between Adaptive GRED, REDD, and GRED Linear analytical models are conducted in this section, which aims to find out which method provides the most satisfactory performance measures based on different finite capacities of queues and occurrence of congestion. The parameters of

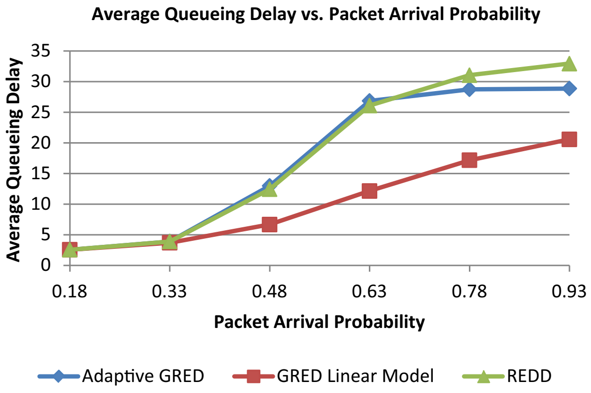

Figure 14 D vs. $\alpha 1$.

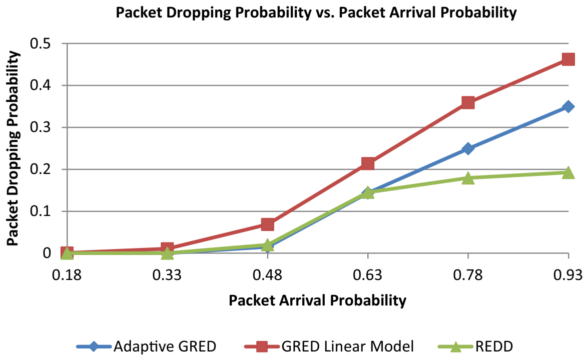

Figure 16 D_P vs. $\alpha 1$.

the compared methods are the same as those given in Subsection 8.3.1 except the following: the values of qw, and pd max are set as those given in Section 10, and setting $\alpha 1$ value to 0.93 in order to use high traffic load that ensures creating congestion situation, and setting to K to several values; these are 20, 40, 60, 80 and 100. This section produces the performance measure results of Adaptive GRED, REDD, and GRED Linear analytical models based on varying the K parameter and in existence of congestion situation. In this comparison, also the compared methods are implemented using the same environment (Java with its features) given in Subsection 8.3.1. The decision which method offers more satisfactory performance measures than others is only made based on the values of K. The performance measure results are accomplished after the system reaches steady state. Every value of K is run 10 times, and the seed number of the random generator value is changed in each run to delete bias in the performance measure results.

The performance measure results versus K values are illustrated in Figs. 17–21. Figs. 17–19 demonstrate mql, T, and D versus K, respectively. Figs. 20 and 21 demonstrate the performance measure results with reference to P_L and D_P versus K, respectively.

It is shown in Figs. 17 and 19 that the GRED Linear model offers the smallest mql and D results since these results remain similar and not affected with changing the K value. Moreover, Adaptive GRED provides less mql and D results than those of REDD in case the K value is 20, and this is because Adaptive GRED drops more packets than those of REDD. On another hand, when K value is greater than 20, REDD provides less mql and D results than those of Adaptive GRED due to the number of packets queued at REDD's router buffer is fewer than the number of packets queued at the router buffer of Adaptive GRED.

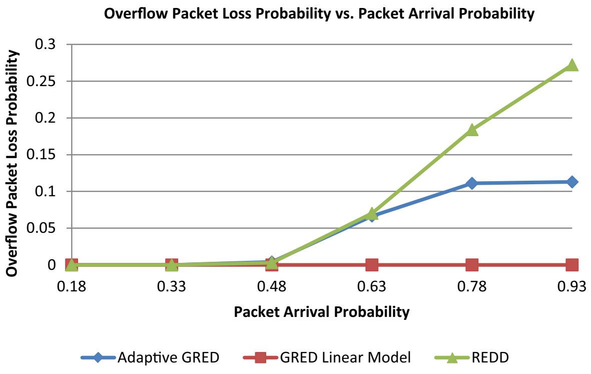

Figure 15 P_L vs. $\alpha 1$.

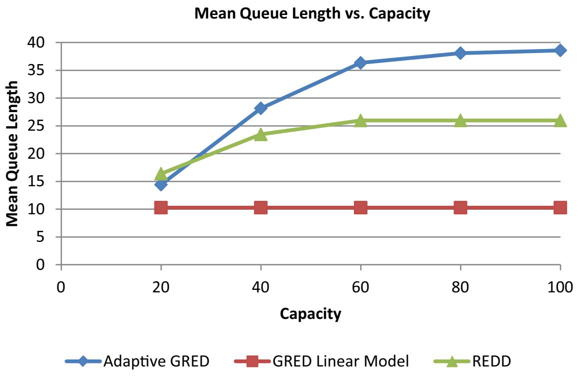

Figure 17 mql vs. K.

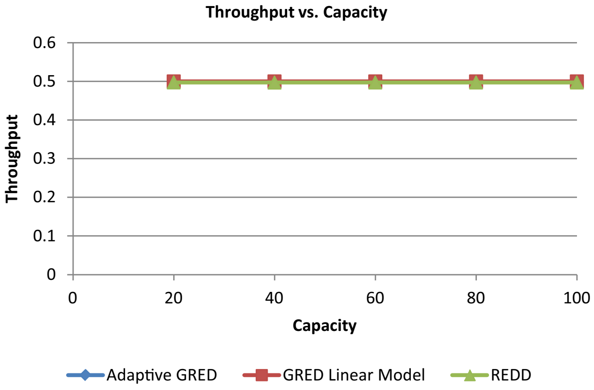

Figure 18 T vs. K.

Fig. 18 shows that all compared methods generate similar T results in all tested values of K parameter, and T results remain similar and not influenced with changing the K values.

Fig. 20 demonstrates that the GRED Linear analytical model loses fewer packets than either Adaptive GRED or REDD due to buffer overflow when K value is less than 60

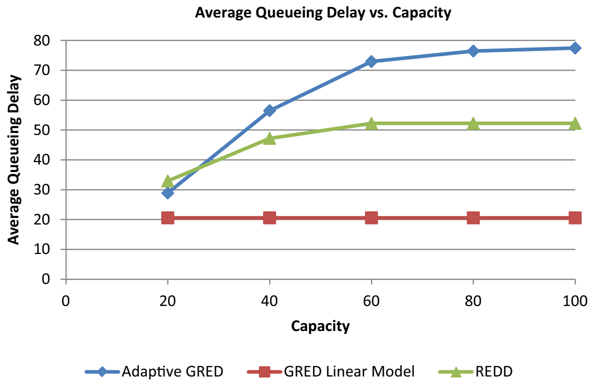

Figure 19 *D* vs. *K*.

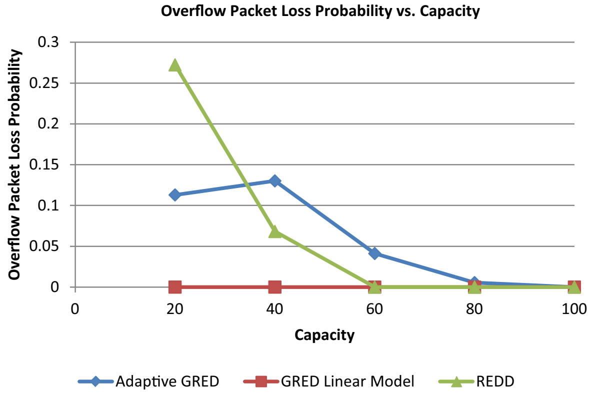

Figure 20 P_L vs. *K*.

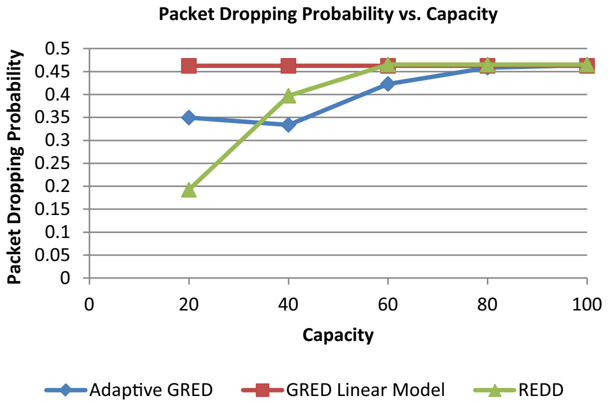

Figure 21 D_P vs. *K*.

since the router buffer of the GRED Linear model overflows on less number of times than those of Adaptive GRED or REDD. Moreover, the router buffer of Adaptive GRED overflows on less number of times than that of REDD when *K* value is 20, however, REDD's router buffer is overflowed on a number of times less than that of Adaptive GRED when *K* value is greater than or equal to 40 and less than 80. Both the GRED Linear model and REDD provide similar P_L results when *K* value is greater than or equal to 60, Adaptive GRED presents similar P_L results as those of GRED Linear and REDD when *K* value is greater than or equal to 80.

Fig. 21 exhibits that the GRED Linear model drops more number of packets than those of Adaptive GRED or REDD before their router buffers become full and when *K* value is less than 60, and this is because the router buffer of the GRED Linear model becomes overflow on less number of times than those of Adaptive GRED or REDD. Furthermore, REDD's router buffer drops fewer number of packets than that of Adaptive GRED when *K* value is 20 due to that the router buffer of REDD becomes full on further times than that of Adaptive GRED. On the contrary, Adaptive GRED's router buffer drops less number of packets than that of REDD when *K* value is greater than or equal to 40 and less than 80 and this is because the router buffer of Adaptive GRED is become full on further times than that of REDD. Both the GRED Linear model and REDD offer similar D_p results when *K* value is greater than or equal to 60, and Adaptive GRED produces similar D_p results to those of the GRED Linear model and REDD when *K* value is greater than or equal to 80.

11. Conclusions and suggestions for future work

This paper compared the following AQM methods: Adaptive GRED, REDD, and GRED Linear analytical models, with reference to different performance measures, such as *mql*, *T*, *D*, P_L, and D_P. The comparison is conducted in three parts, the first comparison was conducted between the GRED-Linear and the computer simulation program of this model aiming to verify and validate the GRED-Linear model. The results of this comparison showed that the performance measure results for both the GRED-Linear model and it's simulation program model are almost similar, and this is a proof that the GRED-Linear model is verified and validated based on its simulation program model. The second comparison is based on different values of parameter $\alpha1$ and the third comparison is using varying values of parameter *K*. The second and third comparison parts identified the method that offers the most satisfactory performance. The comparison results based on varying values of parameter $\alpha1$ can be summarized as follows:

- The compared methods offered comparable *mql* and *D* results when $\alpha1 \leqslant 0.33$. Nevertheless, as $\alpha1$ is increased (i.e., $\alpha1 \geqslant 0.48$), the GRED Linear model provided less *mql* and *D* results than either Adaptive GRED or REDD. Adaptive GRED and REDD still presented similar *mql* and *D* results when $\alpha1$ is increased up to 0.63. Adaptive GRED generated less *mql* and *D* results than those of REDD when $\alpha1 > 0.63$.
- The compared methods offered similar *T* results at all values of $\alpha1$, except 0.48. Hence, Adaptive GRED and REDD offer similar *T* results that are marginally higher than those of the GRED Linear model when $\alpha1 = 0.48$.
- The compared methods provided similar P_L results when $\alpha1 \leqslant 0.48$ because the router buffers of these methods overflowed a similar number of times. The GRED Linear model provided less P_L results than both Adaptive GRED and REDD when $\alpha1 \geqslant 0.63$. Moreover, Adaptive GRED and

REDD still provided similar P_L results when $\alpha 1 = 0.63$. However, Adaptive GRED loses fewer packets than REDD due to overflow when heavy congestion occurs ($\alpha 1 > 0.63$).

- The compared methods drop a similar number of packets before the router buffers are full when $\alpha 1 \leqslant 0.33$. The GRED Linear model drops more packets than Adaptive GRED or REDD when $\alpha 1 > 0.33$. Adaptive GRED and REDD offer similar D_p results when $\alpha 1$ is up to 0.63. REDD outperformed Adaptive GRED with reference to D_P when $\alpha 1 > 0.63$.

The comparison part results based on different values of parameter K can be summarized as follows:

- The GRED Linear model offered the least *mql* and D since these results remain similar and not affected with changing the K value. Moreover, Adaptive GRED provided less *mql* and D results than those of REDD in the case K is 20. On the other hand, when K is greater than 20, REDD provided less *mql* and D than those of Adaptive GRED.
- All compared methods generated similar to T results in all tested values of K parameter, and T results remain similar and not influenced with changing the K values.
- The GRED Linear analytical model offered less P_L results than both Adaptive GRED or REDD when K is less than 60. Moreover, Adaptive GRED provided P_L results less than those of REDD when K is 20, however, REDD outperformed Adaptive GRED with reference to P_L results when K is greater than or equal to 40 and less than 80. Both the GRED Linear model and REDD provided similar P_L results when K is greater than or equal to 60, Adaptive GRED presented similar P_L results as those of GRED Linear and REDD when K is greater than or equal to 80.
- The GRED Linear model offered higher D_P results than those of Adaptive GRED or REDD when K is less than 60.
- Furthermore, REDD outperformed Adaptive GRED regarding D_P results when K is 20. On the contrary, Adaptive GRED outperformed REDD regarding D_P results when K is greater than or equal to 40 and less than 80. Both the GRED Linear model and REDD offered similar D_p results when K value is greater than or equal to 60, and Adaptive GRED produced similar D_p results to those of the GRED Linear model and REDD when K value is greater than or equal to 80.

Future studies are recommended to build an analytical model for controlling congested networks based on the GRED method that exponentially decreases its $\alpha 1$ when congestion occurs in the router buffer. Further studies are also recommended to compare the performance of the GRED Exponential model with that of the existing GRED Linear model to evaluate the performance of the GRED Exponential model.

References

Abdeljaber, H., Ababneh, J., Thabtah, F., Daoud, A.M., Baklizi, M., 2011. Performance analysis of the proposed adaptive gentle random early detection method under non congestion and congestion situations. The International Conference on Digital Enterprise and Information Systems (DEIS). Springer-Verlag, Berlin Heidelberg, London, U.K. (The paper will be published in the "Communications in Computer and Information Science" (CCIS), pp. 592–603).

Abdeljaber, H., Thabtah, F., Woodward, M., 2008. Traffic management for the gentle random early detection using discrete-time queueing. International Business Information Management Association (9th IBIMA) Conference. The Conference Proceedings. ISBN: 0-9753393-8-9. Marrakech, Morocco. pp. 289–298.

Abdeljaber, H., Thabtah, F., Woodward, M., Jaffar, A., Al bazaar, H., 2014. Random early dynamic detection approach for congestion control. Baltic J. Mod. Comput. 2 (1), 16–31.

Abdeljaber, H., Woodward, M., Thabtah, F., 2008. Performance evaluation for DRED discrete-time queueing network analytical model. In: Proceedings of the Journal of Network and Computer Applications, vol. 31. Elsevier (Issue 4, pp. 750–770).

Al-Bahadili, H., Ababneh, J., Thabtah, F., 2011. Analytical modeling of a multi-queue nodes network router. Int. J. Automat. Comput. 8 (4), 459–464.

Al-Diabat, M., Abdeljaber, H., Thabtah, F., Abou-rabia, O., Kishta, M., 2012. Analytical models based discrete-time queueing for the congested network. Int. J. Model. Simul. Sci. Comput. (IJMSSC) 3 (01) (22 pages, Published 8 December 2011).

Andrzej, C., Chrost, L., 2011. Analysis of AQM queues with queue size based packet dropping. Proc. Int. J. Appl. Math. Comput. Sci. 21 (3), 567–577.

Aweya, J., Ouellette, M., Montuno, D.Y., 2001. A control theoretic approach to active queue management. Comput. Net. 36 (2–3), 203–235.

Bonald, T., May, M., Bolot, J., 2000. Analytic evaluation of RED performance, INFOCOM 2000. The Proceeding of Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 3. pp. 1415–1424.

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., Zhang, L., April 1998. Recommendations on Queue Management and Congestion Avoidance in the Internet. RFC 2309.

Brandauer, C., Iannaccone, G., Diot, C., Ziegler, T., Fdida, S., May, M., 2001. Comparison of tail drop and active queue management performance for bulk-data and web-like internet traffic. In: Proceeding of ISCC. IEEE, pp. 122–129.

Feng, W., kandlur, D., Saha, D., and Shin, K.G., April 1999. Blue: a new class of active queue management algorithms. University of Michigan, Ann Arbor, MI. Technical Report, UM CSE-TR-387-99.

Floyd, S., Jacobson, V., 1993. Random early detection gateways for congestion avoidance. IEEE/ACM Trans. Netw. 1 (4), 397–413.

Floyd, S., Ramakrishnan, G., Shenker, S., August 1, 2001. Adaptive RED: an algorithm for increasing the robustness of RED's active queue management. Technical report, ICSI.

Floyd, S., May 2000. Recommendations on using the gentle variant of RED. Available at <http://www.aciri.org/floyd/red/gentle.html>.

Ignall, E.J., Kolesar, P., Walker, W.E., 1978. Using simulation to develop and validate analytic models: some case studies. Proc. Oper. Res. 26 (2), 237–253.

Kamoun, F., 2006. Performance analysis of a discrete-time queuing system with a correlated train arrival process. J. Perform. Eval. 63 (4), 315–340 (Elsevier Science Publishers B.V. Amsterdam, The Netherlands).

Lim, L.B, Guan, L., Grigg, A., Philips, I.W., Wang, X.G., Chi, X., Awan, I.U., 2011. Controlling mean queuing delay under multi-class bursty and correlated traffic. J. Comput. Syst. Sci. 77 (5), 898–916 (Elsevier).

Pace, D.K., 2004. Modeling and simulation verification and validation challenges. Johns Hopkins APL Tech. Digest 25 (2), 163–172.

Ross, S.M., 2010. Introduction to Probability Models, tenth ed. Academic Press.

Tanenbaum, A.S., 2002. Computer Networks, fourth ed. Prentice Hall PTR.

Wang, J., Guan, L., Lim, L.B., Wang, X.G., Grigg, A., Awan, I., Philips, I., Chi, X., 2011. QoS enhancements and performance analysis for delay sensitive applications. J. Comput. Syst. Sci. 77 (4), 665–676 (Elsevier).

Welzl, M., 2005. Network Congestion Control: Managing Internet Traffic. p. 282.

Woodward, M.E., 1993. Communication and Computer Networks: Modelling with Discrete-Time Queues. Pentech Press, London.