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

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
Congestion control is important to improve network performance and 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 is 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; Peng 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, Pd, and Dp). 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 queueing delay by implicitly informing the arrival rate and by moving the queueing 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 queueing 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_t (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 ≤ aql < double max threshold, the router buffer drops packets with probability

\[
D_{\text{max}} + \frac{(\min \text{ threshold} - aql)}{(\max \text{ threshold} - aql)}
\]

(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 x1. 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 x1 to x2, where i is the queue state and x_i = x_1 - (1 + i - min threshold) \left(\frac{(\max \text{ threshold} - min \text{ threshold})}{(x_1 - x_2)}\right), where min threshold ≤ i < double max threshold and the x_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, \(\frac{(x_i - x_2)}{x_1 - x_2}\), where min threshold ≤ 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 x_2 to x_2, and D_p value is then increased by changing it from \(\frac{(x_2 - x_2)}{x_1 - x_2}\) to \(\frac{(x_1 - x_2)}{x_1 - x_2}\).

![Figure 1](image-url) The single queue node system for the GRED Linear model.
The probabilities of packet arrival to the GRED Linear router buffer in a fixed time are \(a_1\), \(x_i\), and \(a_2\). When the current \(q_t\) result is smaller than the min threshold, \(q_t\) is larger than or equal to min threshold and less than the double max threshold, and \(q_t\) is equal to or greater than the double max threshold, respectively. \(\beta\) 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, \(a_t \in \{0,1\}, n = 0, 1, 2, \ldots\), where \(a_t\) 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: \(1/\lambda_i\) and \(1/\lambda\). Packet departure times are geometrically distributed with a mean of \(1/\lambda\). \(K\) 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 \([0, 1, 2, 3, \ldots, \text{min threshold}, \text{max threshold, double max threshold}, \ldots, K-1, K]\). Assuming that half of the \(x_i\) values are larger than the \(\beta\) value and the other half are smaller than the \(\beta\) value, such that \(x_1 > \beta\), \(x_i > \beta\), and \(x_2 < \beta\).

Figure 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.

\[
p_0 = (1 - x_1)p_0 + [\beta(1 - x_1)]p_1
\]

\[
p_1 = x_1p_0 + [x_1\beta + (1 - x_1)(1 - \beta)]p_0 + [\beta(1 - x_1)]p_2
\]

In general:

\[
p_i = [x_1(1 - \beta)]p_{i-1} + [x_1\beta + (1 - x_1)(1 - \beta)]p_i + [\beta(1 - x_1)]p_{i+1}, \quad \text{where} \quad i = 2, 3, 4, \ldots, \text{min threshold} - 2
\]

\[
P_{\text{min threshold} - 1} = [x_1(1 - \beta)]p_{\text{min threshold} - 2} + [x_1\beta + (1 - x_1)(1 - \beta)]p_{\text{min threshold} - 1} + [\beta(1 - x_1)]p_{\text{min threshold}}
\]

\[
P_{\text{min threshold}} = [x_1(1 - \beta)]p_{\text{min threshold} - 1} + [x_1\beta + (1 - x_1)(1 - \beta)]p_{\text{min threshold}} + [\beta(1 - x_1)]p_{\text{min threshold} - 1}
\]

where \(i = \text{min threshold} + 1, \text{min threshold} + 2, \text{min threshold} + 3, \ldots, \text{double max threshold} - 2\)

\[
P_{\text{double max threshold} - 1} = [\beta\text{double max threshold} - (1 - \beta)]p_{\text{double max threshold} - 2}
\]

\[
P_{\text{double max threshold}} = [\beta\text{double max threshold} - (1 - \beta)]p_{\text{double max threshold} - 1} + [\beta(1 - x_2)]p_{\text{double max threshold} + 1}
\]

where \(i = \text{double max threshold} + 1, \text{double max threshold} + 2, \ldots, K - 1\)

Equation (13) is used to simplify obtaining the equilibrium probabilities \((p_n, i = \text{min threshold}, \text{min threshold} + 1, \text{min threshold} + 2, \ldots, \text{double max threshold} - 1, \ldots, \text{double max threshold})\), 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 \(x_i\) 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 \(\beta\) by one minus the packet arrival probability at this queue state \(i(1 - x_i)\).
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).

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

(17)

$$P_0 = \left[ \frac{1 - \beta^{\text{double max threshold}-1}}{(1 - \beta)} + \frac{\beta^{\text{min threshold}}}{(1 - \beta)} \right] \sum_{l=\text{min threshold}}^{\text{double max threshold}-1} \frac{1}{l} \prod_{j=\text{min threshold}}^{l-1} (\gamma_j), \frac{1}{1 - \gamma_j} + \frac{\beta^{\text{max threshold}(1-z)+1}}{(1 - \beta)(1 - z)} \prod_{l=\text{min threshold}}^{\text{double max threshold}-1} (\gamma_j) \right]^{-1}$$

(18)

In general,

$$P_i = \frac{\beta^i(1 - \beta)^{i-1}}{(1 - \beta)} P_0 = \frac{\gamma^i(1 - \beta)^{i-1}}{(1 - \beta)} P_0,$$

where $$i = 1, 2, 3, \ldots, \text{min threshold} - 1$$

(14)

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

$$\text{mql} = P^{(1)}(1) = \frac{P_0}{(1 - \beta)} \left[ \frac{1 - \beta^{\text{double max threshold}-1}}{(1 - \beta)} + \frac{\beta^{\text{min threshold}}}{(1 - \beta)} \right] \prod_{j=\text{min threshold}}^{\text{double max threshold}-1} (\gamma_j), \frac{1}{1 - \gamma_j} + \frac{\beta^{\text{max threshold}(1-z)+1}}{(1 - \beta)(1 - z)} \prod_{l=\text{min threshold}}^{\text{double max threshold}-1} (\gamma_j) \right]^{-1}$$

(20)

where $$i = \text{min threshold}, \text{min threshold} + 1, \text{min threshold} + 2, \ldots, \text{double max threshold} - 1$$ and $$z_j = 1, \text{if } j < \text{min threshold}.$$ 

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).

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

(21)

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

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

(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.
\[ P_k = (1 - \beta) P_{k-1} \]  

(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).

\[
D_P = \sum_{i=\text{min threshold}}^{\text{double max threshold} - 1} \left( \frac{x_1 - x_i}{x_1 - x_1} \right) \times p_i + \left( \frac{x_1 - x_2}{x_1 - x_1} \right) \times \sum_{i=\text{double max threshold}}^{\text{max threshold} + 2} p_i \]  

(24) 

where \( \sum_{i=\text{min threshold}}^{\text{double max threshold}} p_i \) represents the sum of equilibrium probabilities \( p_{\text{double max threshold}} = p_{\text{double max threshold} - 1} \) multiplied by \( \left( \frac{x_1 - x_2}{x_1 - x_1} \right) \), the result of this represents the packet dropping probability when queue state \( i \) is greater than or equal to \( \text{double max threshold} \) 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.

The parameters of REDD, such as pmult, qw, and others, are set as follows (Abdeljaber et al., 2014): pmult, 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 \((K\text{-min threshold})\), then the value of max threshold will be set to the value of \((K\text{-min threshold})\). If the aql value is between the min threshold and the Target aql values, then the max threshold value will be reduced to \((\text{max threshold} - 2)\). If the aql value is between the target aql and the max threshold values, then the max threshold value will be increased to \((\text{max threshold} + 2)\).

The max threshold was set to \((K\text{-min threshold})\) 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 \((2 \times \text{min threshold})\) to maintain throughout 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 \( \geq 0.48 \), Adaptive GRED outperformed GRED with reference to the results of mql and D. Adaptive GRED and GRED generated similar T

For every packet arriving at the router buffer

If (aql < Target aql and max threshold \( \geq 2 \times \text{min threshold} \))

{ //Decreasing the max threshold value by 2 as follows:
max threshold = max threshold - 2;
}

If (aql > Target aql and max threshold \( \leq (K - \text{min threshold}) \))

{ //Increasing the max threshold 2 as follows:
max threshold = max threshold + 2;
}

Figure 3 The pseudo-code of REDD.

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

Figure 4 The single router buffer for Adaptive GRED.
results regardless of the value of packet arrival probability. Adaptive GRED and GRED offered similar \( P_t \) and \( D_t \) results when the values of packet arrival probability are \( \leq 0.63 \). However, Adaptive GRED presented better \( P_t \) results than GRED when the values of packet arrival probability are \( > 0.63 \), whereas GRED outperformed Adaptive GRED in terms of \( D_t \) 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 \( Z_1 \). \( \beta \) 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 \( \frac{1}{\lambda} \). Packet departure times are geometrically distributed with mean \( \frac{1}{\mu} \).

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 interarrivals 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 \( \{y_n, n = 0,1,2,\ldots\} \) 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 \( \{0, 1, 2, \ldots \} \) is the states that can be have numbers with nonnegative integer values in the set \( \{0, 1, 2, \ldots \} \), and \( X \) is the state space \( \{0, 1, 2, \ldots \} \) (Woodward, 1993).

The expression of Markov property can be given as follows:

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

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

\[
P_j(n) = P(X_{n+1} = j|X_n = i) \quad i, j \in X, \quad n = 0, 1, 2, \ldots
\]  

(27)

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:

\[
p_{ij}(n) = p_{ij}, \quad i, j \in X, \quad n = 0, 1, 2, \ldots
\]

(28)

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 Markov chain is a process \( \{ x(t), t \geq 0 \} \) with nonnegative values in \( \{0, 1, 2, 3, \ldots \} \), for example

\[
P(X(t + y) = j|X(y) = i) = P(X(t + y) = j|X(y) = i).
\]

(29)

The function of stationary transition probability is as follows:

\[
P_j(t) = P(X(t + y) = j|X(y) = i).
\]

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.
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 $z_1$, respectively, and the results of $P_L$ and $D_P$ versus $z_1$ appear in Figs. 10 and 11, respectively.

The parameters of GRED-Linear and the simulation program are set as follows: $z_1, z_2$, and $\beta$, which are set to $[0.18–0.93], 0.1, and 0.5$, respectively, to create situations of congestion (i.e., $z_1 > \beta$) and non-congestion (i.e., $z_1 < \beta$). The finite capacity ($K$) of the router buffer is set to 20 packets to measure the performance with small buffer sizes. The values of $\text{min threshold}$, $\text{max threshold}$ are set to 3 and 9, respectively, similar to those of the GRED method (Floyd, 2000). The $\text{double max threshold}$ 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.

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 $z_1$ 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: $z_1, z_2$, $\beta$, and $K$ which are set to values as those given in Subsection 8.3.1. As mentioned earlier, the setting values of $z_1$ are aiming to create situations of congestion (i.e., $z_1 > \beta$) and non-congestion (i.e., $z_1 < \beta$). 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 $\text{min threshold}$, $\text{max threshold}$, $qw$, 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 $\text{double max threshold}$ 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
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 \( z_1 \). The performance measure results are obtained after the system reaches the steady state. Each value for \( z_1 \) 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 \( z_1 \) value, the performance measure result is the mean of 10 ten run times for that of \( z_1 \) value.

Figs. 12–16 show the performance measure results compared with the \( z_1 \) values. Figs. 12–14 show \( mql \), \( T \), and \( D \) versus \( z_1 \), respectively. Figs. 15 and 16 show the performance measure results with regard to \( P_L \) and \( D_P \) versus \( z_1 \), respectively.

Figs. 12 and 14 show that the Adaptive GRED, REDD, and GRED models offer similar \( mql \) and \( D \) results when \( z_1 \leq 0.33 \). However, congestion occurs when \( z_1 \) is increased (i.e., \( z_1 \geq 0.48 \)). 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 \( z_1 \) 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 \( z_1 > 0.63 \) because Adaptive GRED drops further packets than REDD.

Fig. 13 shows that the Adaptive GRED, REDD, and GRED Linear models generate similar \( T \) results at all values of \( z_1 \), except 0.48. Hence, the Adaptive GRED and REDD offer similar \( T \) results when \( z_1 = 0.48 \), 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 \( z_1 \leq 0.48 \) 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 \( z_1 \geq 0.63 \) 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 \( z_1 = 0.63 \). However, Adaptive GRED loses packets because of fewer overflow than REDD when heavy congestion occurs (\( z_1 > 0.63 \)) 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 \( z_1 \leq 0.33 \). The GRED Linear model drops more packets than Adaptive GRED or REDD when \( z_1 > 0.33 \) 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 \( z_1 \) is up to 0.63. REDD drops fewer packets than Adaptive GRED when \( z_1 \geq 0.63 \) 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
the compared methods are the same as those given in Subsection 8.3.1 except the following: the values of $q_w$ and $p_{d_{max}}$ are set as those given in Section 10, and setting $z_1$ 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.

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
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 its 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 four different values of parameter $z1$ 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 $z1$ can be summarized as follows:

- The compared methods offered comparable $mql$ and $D$ results when $z1 \leq 0.33$. Nevertheless, as $z1$ is increased (i.e., $z1 \geq 0.48$), 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 $z1$ is increased up to 0.63. Adaptive GRED generated less $mql$ and $D$ results than those of REDD when $z1 > 0.63$.
- The compared methods offered similar $T$ results at all values of $z1$, except 0.48. Hence, Adaptive GRED and REDD offer similar $T$ results that are marginally higher than those of the GRED Linear model when $z1 = 0.48$.
- The compared methods provided similar $P_L$ results when $z1 \leq 0.48$ 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 $z1 \geq 0.63$. Moreover, Adaptive GRED and
REDD still provided similar $P_L$ results when $z_1 = 0.63$. However, Adaptive GRED loses fewer packets than REDD due to overflow when heavy congestion occurs ($z_1 > 0.63$).

- The compared methods drop a similar number of packets before the router buffers are full when $z_1 \leq 0.33$. The GRED Linear model drops more packets than Adaptive GRED or REDD when $z_1 > 0.33$. Adaptive GRED and REDD offer similar $D_p$ results when $z_1$ is up to 0.63. REDD outperformed Adaptive GRED with reference to $D_p$ when $z_1 > 0.63$.

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 $K = 20$. 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 $K = 20$, 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 $K = 20$. 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 $x_1$ 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


