Chapter One

Introduction

1.1 Background and Motivation

Since the massive developments of internet technology in several applications including, audio and video data traffic, the internet has become one of the fastest developing technologies in recent years [10, 26, 117]. These internet applications necessitate high speed router buffers to deliver data transmitted as quickly as possible to their prospective receivers. One key issue related to internet application performance in computer networks is congestion. The network sources such as users and connections compete for the available network resources, such as bandwidth and buffer space. Congestion occurs at a particular router buffer when the demand on network resources exceeds the available bandwidth capacity [10, 20], which consequently causes a deterioration in internet performance. Congestion can also be defined as an incident that occurs in networks when the network resources cannot accommodate the number of injected packets [12, 51, 117].

Congestion degrades the network performance by reducing the throughput and increasing both packet loss rate and queueing delay [117]. Congestion may also trigger an unbalanced share among the network sources. In order to control the congestion, Drop-Tail method (DT) [20, 22] (discussed in Subsection 2.3.1) is normally used. The DT depends on setting the router buffer length in the internet to a maximum size in order to achieve a high throughput. However, this also leads to high packet queueing delay as well as high packet loss rate since the DT routers are full, which is not desirable. On the other hand, if the router buffer is set to a small length, low queueing delay will be achieved but the throughput performance will deteriorate.

The DT method has poor performance within Transport Control Protocol (TCP) networks since the TCP sources update their transmitting rate after the router buffers overflow. Therefore the TCP sources become aware of the congestion at the router after the router overflows. Also, the DT method has other drawbacks such as the lockout phenomenon, full queues, the bias versus bursty traffic and global synchronisation [20, 45], which all degrade the network performance. The DT drawbacks are explained as follows; first, the lock-out phenomenon occurs when a few connections monopolise the router buffer spaces for long time in order to send their packets, whereas the rest of the connections are idle waiting for the monopolised connections to release the buffer spaces to be able to send their packets. Second, full queues [20] happen when the DT router buffers are full for some time, thus both the queueing delay and the packet loss rate increase since every arriving packet at the DT router is dropped.

Third, global synchronisation impacts the throughput performance when congestion occurs in the router buffers. The DT router buffers notify all the connections about the congestion in order to reduce their transmitting rates. However, there could be other connections that do not contribute to the occurrence of the congestion, therefore reducing their transmitting rates is unnecessary, and may reduce their throughput performance. Fourth, the "bias versus bursty" occurs when the packet sending rates vary (low or high) or change suddenly, since when the packet sending rate is high the buffer quickly becomes full. Thus, the DT router buffer drops the arriving packets because of the high congestion. Moreover, the DT routers prefer traffic flows with short round trip time (RTT) to achieve high throughput. The traffic flows with short RTT increase their transmission window quicker than those with large RTT, and as a result flows with short RTT increase their share of the network resources quicker than those with large RTT, which may lead to an unfair share among the network flows. Moreover, flows with short RTT contribute in building up the router buffers more than that of large RTT, and when the DT router buffers become full, they do not allow for bursty traffic to queue in their waiting rooms since no room is available, and thus they drop the packets.

As a result of the above DT drawbacks, many researchers [9, 11, 14, 39, 46, 49, 50, 73, 90] have developed Active Queue Management (AQM) techniques to achieve the following goals:

- Control the congested network in an early stage. Early in this context means to identify a congestion incident at an early stage (at a certain level in router buffers) and before the router buffers overflow.

- Obtain a satisfactory performance such as high throughput, low average queueing delay, low packet loss and dropping rates.
- Maintain a mean queue length as small as possible to prevent filling up the router buffer.
- Distribute a fair share of network resources among the network sources.

One of the best known AQM methods is Random Early Detection (RED) [46]. At first, RED performance was satisfactory [46], but subsequently and since the diversity of traffic types, such as video, images and voice web transfer, RED has become inadequate in that it is unable to configure its parameters ($\min threshold$, $\max threshold$, the maximum dropping probability value ($D_{\max}$) and queue weight (qw) [46, 83, 84, 90]). Thus, one must tune the RED's parameters to particular values to ensure a satisfactory performance. Also, the congestion measure (average queue length(aql)) in RED varies according to the congestion status, hence, if the congestion status is light, the aql becomes near the $\min threshold$. Whereas if the congestion status is severe, the aql becomes near the $\max threshold$. Another drawback is that the computed aql in RED relies on the traffic load (number of connections) and if the traffic load is high, the aql might be above the $\max threshold$. Therefore the RED router buffer will drop every arriving packet. Whereas, if the traffic load is low, the aql will normally be less than the $\min threshold$, and consequently, no arriving packet is dropped. Finally, RED cannot stabilise it's aql

between $\min threshold$ and $\max threshold$ positions at router buffers when the traffic load changes suddenly [11, 39, 43, 49, 90], i.e. bursty traffic.

In RED, if the parameters (threshold settings and weight factor) are set up to handle a specific type of service this may not be effective for another different type of service; that is it will treat all packets the same irrespective of the type of service class. Clearly, the settings of parameters depend on the service requirements, for example, giving the minimum threshold a small value will result in dropping packets early and this will degrade the throughput performance. The minimum threshold is sensitive to the packet loss requirement. Furthermore, setting the maximum threshold at a low value will lend itself to any system with low delay requirements and vice versa. This indicates that the maximum threshold is sensitive to the delay requirement. Also, the choice of the weight parameter (wq) in the exponentially weighted moving average would depend on the traffic profile. Making wq small gives a low weighting to the current queue value and so would tend to smooth and burstiness in the traffic. Making wq relatively large would give a more sensitive response to a bursty profile.

To enhance RED performance, different variations of RED have been proposed, i.e. Adaptive Random Early Detection (ARED) [49], Gentle Random Early Detection (GRED) [50], Random Exponential Marking (REM) [9, 73], Stabilised Random Early Drop (SRED) [90], BLUE [39, 43] and Dynamic Random Early Drop (DRED) [11]. These AQM methods have shown competitive results with regard to managing congestion in networks if compared to the DT classic methods [20, 22, 107].

The motivations and contributions of this thesis are presented as follows: 1) RED cannot stabilise its *aql* between $\min$*threshold* and $\max$*threshold* positions at router buffers when the traffic load is high. Therefore, RED's router buffer becomes full and overflows, and consequently, packets will be lost. 2) RED relies on configuring its parameters to obtain a satisfactory performance. To overcome the above RED's problems, a new AQM method based on RED, called Dynamic RED (REDD) is proposed. REDD aims to stabilise *aql* between $\min$*threshold* and $\max$*threshold* positions at a specific level by updating the $\max$*threshold* . Consequently, the REDD algorithm avoids overflow of the router buffer better than RED, and decreases RED dependency on its parameters by using a dynamic $\max$*threshold* rather than a fixed one. Furthermore, new AQM methods based on both RED and fuzzy logic (FL) are proposed in order to reduce the dependency of RED on its $\min$*threshold* and $\max$*threshold* parameters as congestion boundaries. This is obtained by employing a fuzzy inference process (FIP) as a congestion detector and controller. The proposed FL algorithms are called REDD1, REDD2 and REDD3.

Lastly, since there are relatively few congestion control research works done on building analytical models based on discrete-time queues, an important aim of this thesis is to design and develop discrete-time analytical models based on different known AQM methods to manage and alleviate the congestion problem in wired networks. In addition, it is desired to obtain a satisfactory performance in cases of a high congestion using the proposed analytical models. The proposed analytical models are named DRED-Alpha, DRED-Linear, BLUE-Alpha, BLUE-Linear, RED-Alpha, RED-Linear, GRED-Alpha and

GRED-Linear. It is essential to enhance the performances of current AQM methods such as DRED, RED and GRED with reference to mean queue length (mql) and average queueing delay (D) in order to decrease their buffer overflow. This decreasing will reduce packet loss probability (P_L). The mql and D performances of the DRED, RED and GRED algorithms are enhanced using the following proposed analytical models: DRED-Alpha, RED and GRED analytical models, where these models maintain their mql and D at values smaller than those of their original algorithms. As a result, the proposed models therefore lose fewer packets than the corresponding classic AQM algorithms. It should be noted that in these analytical models, the basic time unit used is called a slot rather than the use RTT, where slots are assumed to be of fixed length and sufficiently small to capture the performance measures of interest.

1.2 Quality of Service (QoS)

Quality of Service (QoS) can be defined as requirements related to a service, which every network needs to meet while packets are travelling across the network [34]. For instance, in the internet, the "best effort" service is used in which only the best effort traffic is processed with no guarantee that the best effort packets are delivered [10, 31]. "Best effort" networks do not distinguish among the different types of service, thus they process packets of different services fairly. In cases where more than one traffic type with several

service requirements need to be processed, the "best effort" service will be inactive since it treats all traffic types in the same way and cannot meet the diversity of service requirements [111].

Non real time applications such as file transfer and electronic mail (email), have high sensitivity for packet loss and low sensitivity for packet delay and jitter. On the other hand, real time applications like Voice over IP (VoIP), live video and video conferencing, necessitate high bandwidth, low packet delay and jitter and have low sensitivity for packet loss. Tables 1.1 and 1.2 show some real time and non real time applications with their sensitivity to QoS requirements (bandwidth, delay, jitter and packet loss), respectively.

Table 1.1: Some real time applications with their sensitivity to QoS requirements [111].

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

Table 1.2: Some non real time applications with their sensitivity to QoS requirements [111].

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

The QoS is offered by the network service suppliers to the network users with regard to service level agreement (SLA) between them [111]. In other words, the SLA is a contract between the network service suppliers and the users of what QoS the service suppliers are offering to the users. This contract has two aspects according to [111]: 1) Type of traffic offered to the users and 2) Types of service which the network service suppliers and users deal with. The first aspect gives details about traffic types that the network supplies to the users and the second aspect determines the QoS required by the users and accepted by the suppliers.

1.3 Discrete-time Queues Approach

Discrete-time queues are an approach to model and analyse the performance of queueing systems in communication and computer networks [118]. Generally, every queueing system can be described by Kendall's notation by five components [118] 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, such as deterministic (D). In D, the interarrival and/or service time distribution is constant. If the descriptor is 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. On the other hand, in discrete-time queues, the interarrival and service times are "geometrically" distributed denoted Geo, and also using the exponential distribution is possible in case the arrivals and departures are multiple [118]. Lastly, if the descriptor is "generally" distributed (G), then no constraints are enforced on the type of the distribution. In the discrete-time queues [118], a basic time unit called a slot is used, where in each slot single or multiple events can 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.

In discrete-time queueing systems, 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 $\{a_n, n = 0,1,2,...\}$ and $\{d_{n+1}, n = 0,1,2,...\}$, 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 Figure 1.1.

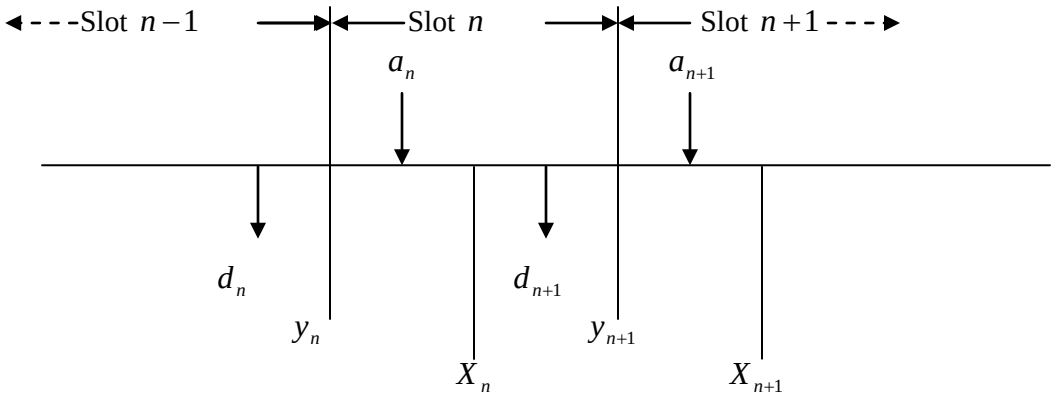

Figure 1.1: The state of the discrete-time queue with arrivals and departures in each slot [118].

$$y_{n+1} = y_n + a_n - d_{n+1} \quad \text{..... (1.1)}$$

Equation (1.2) denotes the process of the queue length after the arrivals happen $\{X_n = n = 0,1,2,...\}$.

$$X_{n+1} = X_n - d_{n+1} + a_{n+1} \quad \text{..... (1.2)}$$

This latter convention (equation 1.2), sometimes called “departure first”, will be used in the models in this thesis. A number of research works have been implemented on modelling queueing networks using discrete-time queues or continuous-time queues, i.e. [2, 5, 19, 59, 60, 83, 118]. For instance, [19, 83] have been constructed analytical models based on RED to omit the bias versus bursty flows and to manage average queueing delay. Furthermore, some researchers proposed discrete-time queueing network analytical models based on multiple arrivals and departures of packets, i.e. [59, 60, 118]. In [59, 60], the authors proposed a product form result for a particular class of queueing networks. Furthermore, the authors were assumed that the network state acts like an irreducible Markov chain. Moreover, Woodward [118] presented a book related in modelling and analysing the performance of the queueing systems in computer networks and communications. This includes information about the discrete-time queue mechanism and how this mechanism can analyse the performance of queueing network systems in computer networks and communications.

1.4 Performance Measures

The performance of every fixed or wireless network [70] needs to be measured by metrics called performance measures in order to determine its performance and effectiveness. Different performance measures are considered in this thesis and they are presented as follows:

1.4.1 Mean Queue Length (mql)

Keeping the mql as low as possible prevents the router queue from building up its size, and thereby the router queue avoids congestion. In addition, the mql metric helps in computing the average queueing delay (D), which will be discussed in Subsection 1.4.3.

1.4.2 Throughput (T)

This can be defined as the number of packets that successfully passed through the queue per unit time. It should be noted that the T metric contributes to the calculation of D, which is discussed in Subsection 1.4.3.

1.4.3 Average Queueing Delay (D)

The average queueing delay (D) is one of the performance measures that calculates the average waiting time for packets at the router queue. D can be evaluated using Little's law as (mql / T) [118].

1.4.4 Packet Loss Probability Due to Overflow (P_L)

Packet loss probability (P_L) can be defined as the probability of losing packets due to buffer overflow.

1.4.5 Packet Dropping Probability (D_p)

Packet dropping probability (D_p) can be defined as the probability of losing packets due to probabilistically dropping them before the buffer is full.

1.4.6 Total of Packet Loss and Dropping Probabilities (P_{Loss})

The P_{Loss} is the total of loss and dropping probabilities for packets at a router buffer.

1.5 Thesis Aims

The main aim of this thesis is to propose discrete-time queueing network analytical models based on different AQM approaches, such as RED, GRED, DRED and BLUE in order to identify how the performance of these methods might be improved.

In particular, the thesis aims to achieve satisfactory performance measure results ($mql, T, D, P_L, D_p, P_{Loss}$) using the proposed analytical models when a high congestion is present. Finally, the thesis aims to develop AQM methods based on the RED algorithm and based on both RED and fuzzy logic (FL) to achieve the following sub-goals:

- Deriving better mql and D performance results than that of RED, particularly when congestion occurs.
- Losing fewer packets due to overflow a router buffer than either RED or ARED when congestion is present.
- Improve stability of the aql in order to reduce buffer overflow.
- Reduce dependency on parameters, i.e. $\max threshold$ setting.
- Avoid reliance on $\min threshold$ and $\max threshold$ as congestion boundaries.

1.6 Thesis Contributions

The thesis contributions are summarised as follows:

1. Extensive Literature Review on TCP and AQM Approaches

A review on congestion control methods such as TCP and AQM is presented in Chapter 2, where many researchers in the area of computer networks may profit from it. Examples of

TCP congestion control methods are slow start (SS), congestion avoidance (CA), fast retransmission (FR), and fast recovery (FRec). Whereas the following are examples on AQM methods: RED, GRED, ARED, DRED, and BLUE. A focus on congestion measures (aql, instantaneous queue length, price), the way the D_p function is calculated, and the packet dropping policies (randomly, periodically, etc) for each method, are addressed. For further details, see Chapter 2.

2. Some Simulations of AQM Approaches

The following AQM approaches: RED, GRED, ARED, DRED and BLUE are experimentally compared with respect to several different performance measures, including mql, T, D, P_L, D_p, and P_{Loss}, in order to identify which of them offers better performance. In addition, statistical analysis results for the considered AQM methods are presented, which aim to analyse ten run results for each performance measure. The statistical analysis is performed using variance (σ^2), standard deviation (σ), 95% confidence interval (CI), lower limit for 95% CI) [32], and upper limit for 95% CI) [32]. These measurements describe ten run results for every performance measure to indicate the amount by which results for each performance measure are dispersed from their mean. For more information, see Chapter 3.

3. Dynamic RED (REDD) algorithm uses an adaptive maximum threshold

The RED algorithm suffers from problems such as stabilising it's computed aql between $\min threshold$ and $\max threshold$ positions when the traffic load changes suddenly, i.e. from low to high. This may lead to degrading the network performance [46, 83, 84, 90]. Furthermore, when the congestion level increases, the aql increases as well [46, 83, 84, 90]. As a result, if the congestion level is high, the computed aql might exceed the $\max threshold$ position at the router buffer, and thus every arriving packet is dropped since the router buffers are congested. In order to overcome the above problem, the Adaptive RED algorithm was proposed in [49], which aims to stabilise aql at a certain level (T_{aql}) (between $\min threshold$ and $\max threshold$ positions) by updating the $D_{\max}$. This solves the problem of tuning the $D_{\max}$ parameter.

This thesis proposes an algorithm based on RED called REDD to handle the problem described above (Chapter 4 gives further details) using an adaptive $\max threshold$ position. The REDD algorithm stabilises the aql at T_{aql} between $\min threshold$ and $\max threshold$ by modifying the $\max threshold$ position. Implementing the REDD algorithm reduces the dependency of RED on the $\max threshold$ as a fixed value, so it uses an adaptive $\max threshold$. In Section 4.4, the performance measures ($mql, T, D, P_L, D_p, P_{Loss}$) of the REDD algorithm reveal that the proposed REDD method outperforms both RED and ARED with reference to the mql and D results when the packet arrival probability values

are larger than packet departure probability (congestion). Also, at these packet arrival probability values, the proposed REDD loses fewer packets than RED or ARED due to router buffer overflows. The statistical results (σ^2, σ and 95% CI, lower limit for 95% CI, upper limit for 95% CI) of the REDD algorithm are shown to be extremely small. Thus, the results for the performance measure are not largely dispersed from their mean. For more information, go to Chapter 4.

4. Fuzzy logic controller algorithms (REDD1, REDD2, REDD3)

One of the RED's problems is tuning its parameters ($D_{\max}$, qw, $\min threshold$, $\max threshold$). The RED algorithm needs to configure its parameters to specific values in order to guarantee a satisfactory performance. In order to overcome such a drawback, three methods are proposed based on RED [46] and FL [71, 87, 88, 89, 125]. The proposed FL methods do not depend on $\min threshold$ and $\max threshold$ positions as congestion boundaries, rather they utilise a FIP as a congestion detector and controller [71, 78, 93]. In addition, the proposed FL algorithms are named REDD1 [3], REDD2 and REDD3. Section 5.6 experimentally compares the proposed FL methods with RED with respect to different performance measures (mql, T, D, P_L, D_p, P_{Loss}). The results indicate that the proposed methods outperform RED when a high congestion is present, and they lose fewer packets than RED due to overflow of router buffers. The statistical analysis results (σ^2, σ, 95% CI, lower limit for 95% CI, upper limit for 95% CI) for each performance measure in the

proposed FL methods are very small, indicating the results for each performance measure are not widely dispersed from their mean.

5. Discrete-time queue analytical models based on different AQM algorithms

Due to relatively little research using discrete-time queue analytical approach, i.e. [59, 60, 83, 118], in the literature, different discrete-time queue analytical models [1, 4, 6, 7] based on various AQM methods are proposed in this thesis. Additionally, these models are proposed in order to analyse and compare their performance with the traditional AQM methods in detecting congestion in wired networks.

The proposed analytical models identify the congestion at the router buffers incipiently by dropping packets, and reduce packet arrival probability to alleviate the congestion. Moreover, these analytical models treat both single packet arrivals and departures. The proposed analytical models control congestion by either 1) Decreasing the value of packet arrival probability to another constant and lower value. Therefore, the D_p value increases from probability value to another higher and constant one, or 2) Decreasing the value of packet arrival probability linearly. Thus, the D_p value increases linearly. The developed models are called DRED-Alpha, DRED-Linear, BLUE-Alpha, BLUE-Linear, RED-Alpha, RED-Linear, GRED-Alpha, and GRED-Linear.

Every proposed analytical model has been compared with its original AQM algorithm concerning to the above mentioned performance measures to identify the one that derives

better performance, for example RED-Alpha and Linear models are compared with the RED algorithm. Other goals for developing these models are to obtain a satisfactory performance when a high congestion is present, and some of the developed models, i.e. DRED-Alpha, RED models and GRED models maintain their mql and D at values smaller than the original algorithms.

1.7 Outline of the Thesis

This thesis consists of seven chapters, and is organised as follows: The literature review on congestion control methods, i.e. TCP and AQM methods, is introduced in Chapter 2.

Chapter 3 compares five AQM methods using simulation (RED, GRED, ARED, DRED and BLUE) with reference to different performance measures (mql, T, D, P_L, D_P and P_{Loss}).

Chapter 4 proposes a new AQM approach based on RED, called Dynamic RED (REDD) to detect and control the congested router buffers in the early stages of congestion. In addition, a comparison of the proposed REDD and both RED and ARED with reference to the above performance measures is presented in Chapter 4.

Chapter 5 presents the development of new AQM approaches based on FL and the RED algorithm. The FL methods are called REDD1, REDD2 and REDD3. This chapter

also compares the proposed FL algorithms and RED according to the above performance measures.

Some discrete-time queue analytical models based on RED, GRED, DRED and BLUE, are proposed in Chapter 6. These proposed models are named DRED-Alpha, DRED-Linear, BLUE-Alpha, BLUE-Linear, RED-Alpha, RED-Linear, GRED-Alpha and GRED-Linear. Furthermore, in Chapter 6, each analytical model is compared with its original AQM method concerning the above performance measures. For example, the proposed RED-Alpha and Linear models are compared with RED.

Finally, the conclusions of this thesis and future research suggestions are presented in Chapter 7.