A New Algorithm for Controlling a Buffer with Time-Varying Arrival Rate

Rasha Fares, Mike Woodward
Department of Computing, University of Bradford,
United Kingdom

Abstract

Due to the increase in network services including real-time video and audio applications, Quality of Service (QoS) became a crucial issue for today's Internet users. In order to guarantee QoS to diverse Internet services, it is important to employ effective buffer management schemes at Internet routers. Effectively constraining the mean queue length to a specified level is a key QoS requirement. Controlling the size of the buffer is determined by the dynamics of TCP's congestion control. Therefore, controlling a buffer with time-varying arrival rate can be done by controlling the mean delay or the buffer size around a specified level. The aim of this paper is to propose a new adaptive prediction algorithm for Active Queue Management (AQM) that is simple to implement and can maintain the mean queue length around its target. The algorithm has been modified in order to bind the delay at a constant value when the arrival rate varies with time. This paper presents a feedback control strategy that operates on a buffer which incorporates a moveable threshold. An algorithm is developed to control the buffer by dynamically adjusting the threshold, which In turn, controls the effective arrival rate by randomly dropping packets. The feasibility of the model is examined using both theoretical analysis and simulation.

1. Introduction

Internet traffic congestion occurs when the aggregate demand exceeds the capacity of the available resources. Congestion results in packets being dropped or lost from the network during transmission. The main reason for dropping these packets is that there is not enough buffer space to accommodate all the transmitted packets. Using large buffers can absorb more bursty traffic but will increase the end-to-end delay as well, which will decrease the overall network performance. This has made sizing router buffers [1] one of the critical issues in present networks.

The aim of this work is to propose a new adaptive prediction algorithm for Active Queue Management (AQM) that is simple to implement and can maintain the mean queue length (MQL) at a constant value when the arrival rate varies. This is done using a control strategy for controlling the MQL through a buffer with time-varying arrival rate. The algorithm uses a feedback control strategy that depends on the instantaneous queue length to adjust the effective arrival rate at appropriate times by dynamically adjusting the queue threshold. The instantaneous queue length has been used rather than using the average queue length because the possibility of buffer overflow will be reduced if congestion is detected using the instantaneous queue length [2].

Constraining the mean delay to a specified value is a key Quality of Service (QoS) requirement and one of the most important considerations for real-time services. Bounding delay not only applies in a TCP network [3], but also in other kinds of networks such as wireless networks [4]. Therefore, an efficient mechanism to control the delay is vitally important if delay is to be constrained to specified value and jitter is to be minimized.

The remainder of this paper is organized as follows: An overview of the related work is presented in Section 2. The proposed model is described and analyzed in Section 3. Section 4 presents and explains the feedback control strategy. While Section 5 provides the performance metrics used, followed by the simulation results in Section 6. Section 7 gives a performance validation and, finally, conclusions and future work are given in Section 8.

2. Related Work

Due to the heterogeneous nature of real world traffic and the rapid development of the Internet, various buffer management mechanisms have been proposed to control traffic congestion and satisfy specified QoS requirements.

The traditional technique used to control traffic congestion has been Drop Tail [5]. Drop Tail tends to penalize bursty connections by dropping all the incoming packets only when the buffer is full. This will continue until the number of packets in the queue is below the maximum queue length and congestion is eliminated. This method has two main drawbacks: 'Lock–Out' which is a result of global synchronization, and 'Full Queue' which results in high packet delay [6] .

A number of AQM mechanisms have been proposed to overcome the weaknesses of the Drop Tail technique. AQM techniques notify traffic sources of congestion in order to reduce their transmission rates to avoid congestion and to reduce both queueing delay and packet loss. For example, the Early Random Drop (ERD) [7] and Random Early Detection (RED) [8] mechanisms are

variations of the Random Drop mechanism [9] aimed at avoiding congestion by predicting when it will occur rather than reacting to it.

Many significant modifications have been done on RED in order to improve its performance such as Adaptive RED (ARED) [10], BLUE [11], Random Early Marking (REM) [12] and Double Slope RED (DSRED) [13]. Most of these studies mainly focus on when and how to drop the arriving packets and to reduce RED sensitivity to parameter settings but they rely on static thresholds which can be restrictive when they operate with sources with varying arrival rates.

All these buffer management algorithms usually operate in conjunction with TCP. The essence of the TCP congestion control mechanism is that the TCP source adjusts its window size based on implicit feedback about the congestion [14]. In networks with large Round Trip Times (RTTs) the TCP congestion control has slower response to congestion. Thus it is desirable that the source be notified early so as to adjust its rate preemptively in order to avoid congestion.

In addition, AQM strategies such as ARED, fail to maintain the MQL around a target value when the arrival rate greatly varies with time [15, 16]. This particularly applies when the arrival rate exceeds the service rate when the MQL can assume high values.

3. Proposed Model

A basic assumption is made that the arrival rate is varying with time and between changes follows a Poisson arrival process. To model this we use a two state Markov Modulated Poisson Process (MMPP) source to provide a time-varying arrival rate in the system. The abrupt changes in the state of the MMPP could be considered to approximately mimic those characteristics of TCP changing the length of its congestion windows.

MMPP [17, 18] is a doubly stochastic Poisson process where the Poisson arrival rate is defined by the state of a Markov chain. The arrival process has two distinct states, state1 and state2. When the arrival process is in state1, it generates arrivals that follow a Poisson distribution with rate (λ_{11}). Similarly, when the arrival process is in state2, it generates arrivals that follow a Poisson distribution with rate (λ_{22}). The transition rate from state1 to state 2 is (γ_1), and form state2 to state1 is (γ_2).

The MMPP is characterized by the transition rate matrix Q of the modulating Markov chain and the arrival rate matrix Λ as follows:

$$Q = \begin{bmatrix} -\gamma_{11} & \gamma_{12} \\ \gamma_{21} & -\gamma_{22} \end{bmatrix} \quad (1)$$

$$\Lambda = \begin{bmatrix} \lambda_{11} & 0 \\ 0 & \lambda_{22} \end{bmatrix} \quad (2)$$

In the proposed continuous-time queueing system, the time has been divided into slots of equal length. These slots (time windows) are assumed large enough to accommodate a relatively large number of events (arrivals and departures). It is also assumed that the RTT from the arrival process to the queue and back to the arrival process is less than one time window. This assumption has been considered to enable the arrival process to switch states from one time window to the next.

The time window length (*TWL*) is assumed to be much smaller than the mean time in each state. This is to allow the system to assume a steady state between changes in the arrival rate, on average, so for most time windows the arrival process will be in the same state.

4. The Feedback Control Strategy

It is assumed that the buffer has a finite capacity of K packets, including the server, with two thresholds (L_1) and (L_2) as shown in Figure 1. The queueing discipline is First-In First-Out (FIFO). When the number of packets in the buffer is less than the minimum threshold (L_1), there is no dropping and the source operates normally. If the number of packets exceeds the maximum threshold (L_2), then the excess packets will be dropped. However, by assuming that the transit time from the controller back to the source is less than the mean packet inter-arrival time, in the majority of cases it should be possible to signal the source to stop sending packets when a full buffer is detected (L_2 packets). Packet transmission can commence after the next departure (service completion). In this way the majority of packet loss due to buffer overflow might be avoided. If the number of packets in the system falls between the first threshold (L_1) and the second threshold (L_2), then the arriving packets are dropped with dropping probability $pd(t)$.

$$pd(t) = \max_p \frac{(Q(t) - L_1)}{(L_2 - L_1)} \quad (3)$$

$\max_p$ is the maximum dropping probability and $Q(t)$ is the instantaneous queue length at any time t.

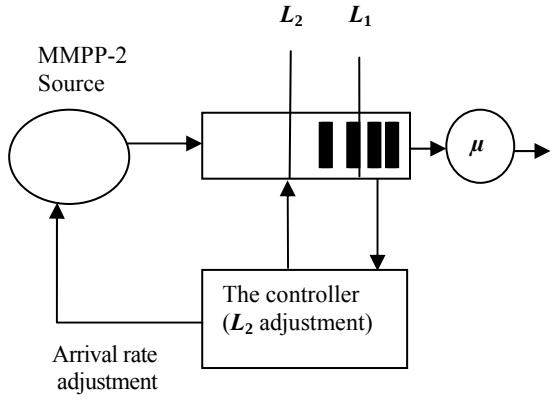

Figure 1. Schematic diagram of the proposed model

The parameters used in the feedback control mechanism are:

MQL: Mean Queue Length

TMQL: Target Mean Queue Length

MMQL: Measured Mean Queue Length

D_T: Target Mean Delay

D_M: Measured Mean Delay

λ_1: Measured mean arrival rate

λ_2: Measured mean arrival rate at L_2

L_1: first threshold (fixed)

L_2: second threshold (moveable over each time window)

The basic idea for the controller is to measure the mean queue length and the mean arrival rate over each time window (*W*). The mean arrival rate is measured by counting the number of arrivals within each window and dividing it over the window length.

These measurements are used to calculate the new position of the threshold L_2 for the next time window (*W*+1) in order to maintain the mean queue length at the required Value. The algorithm has been modified to calculate the new position of L_2 in order to maintain the mean delay at the required target value.

5. Performance Metrics

The queueing model used can be considered as a modification of a MMPP/M/1/K queue. However, setting the time window length less than the mean time the arrival process remains in each state (for example $TWL=0.1/\gamma_1$), then over most time windows the model can be viewed as a modification of the M/M/1/K queue. Due to the use of two thresholds in the model, the queue needs to be considered in two parts in order to calculate the steady state probabilities. The state transition diagram for the proposed system with the two thresholds L_1 and L_2 is shown in Figure 2. The first threshold L_1 is fixed and should be initialized at the beginning of the simulation. The second threshold L_2 can be adjusted to any position in the queue.

The balance equations of the continuous-time finite queue can be obtained through the state transition diagram (c.f. Figure 2). The equilibrium probabilities can be expressed in terms of $P_{(0)}$ as follows:

5.1. Controlling the MQL

From state (0) to state (L_1)

$$\lambda_1 \, p_{(0)} = \mu \, p_{(1)} \tag{4}$$

$$p_{(1)} = \frac{\lambda_1}{\mu} \, p_{(0)}$$

(5) $\lambda_1 \, p_{(1)} = \mu \, p_{(2)}$ (6)

$$p_{(2)} = \frac{\lambda_1}{\mu} \, p_{(1)} = \frac{\lambda_1}{\mu} * \frac{\lambda_1}{\mu} \, p_{(0)} = (\frac{\lambda_1}{\mu})^2 \, p_{(0)} \tag{7}$$

From (5) and (7), the general equation is:

$$p_{(x)} = \rho^x \, p_{(0)} \qquad , x = 0,1,\cdots, L_1 \tag{8}$$

where $\rho = \frac{\lambda_1}{\mu}$.

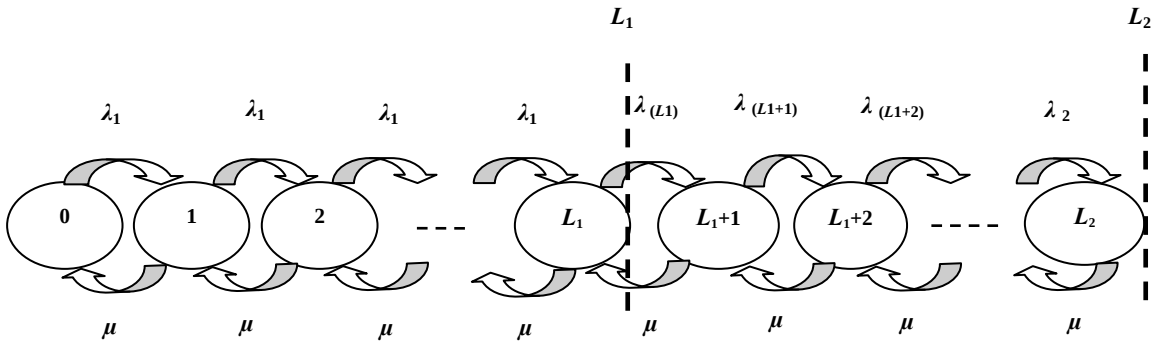

Figure 2. The state transition diagram

From state (L_1+1) to state (L_2)

From (8)

$$p_{(L_1)} = \rho^{L_1}\ p_{(0)} \tag{9}$$

$$\lambda_{(L_1)}\ p_{(L_1)} = \mu\ p_{(L_1+1)}$$

$$(10)\ p_{(L_1+1)} = \frac{\lambda_{(L_1)}}{\mu}\ p_{(L_1)} \tag{11}$$

From (9) and (11)

$$p_{(L_1+1)} = \frac{\lambda_{(L_1)}}{\mu} * \rho^{L_1}\ p_{(0)} \tag{12}$$

$$\lambda_{(L_1+1)}\ p_{(L_1+1)} = \mu\ p_{(L_1+2)} \tag{13}$$

$$p_{(L_1+2)} = \frac{\lambda_{(L_1+1)}}{\mu} p_{(L_1+1)} \tag{14}$$

From (12) and (14)

$$p_{(L_1+2)} = \frac{\lambda_{(L_1+1)}}{\mu} * \frac{\lambda_{(L_1)}}{\mu} * \rho^{L_1}\ p_{(0)} \tag{15}$$

Generalizing this:

$$p_{(L_1+i)} = \prod_{j=0}^{i-1}\left(\frac{\lambda_{L_1+j}}{\mu}\right) * \rho^{L_1}\ p_{(0)}\ , i = 1,2,\cdots, L_2 - L_1 \tag{16}$$

With reference to Figure 3, equation (16) can be written in terms of the slope of the arrival rate characteristics as follows:

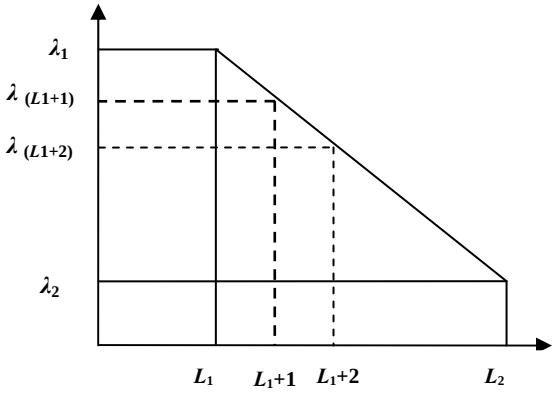

Figure 3. The change in the arrival rate due to packet dropping

$$\text{Slope} = \Delta = \frac{\lambda_1 - \lambda_2}{L_2 - L_1} = \frac{\lambda_1 - \lambda_{(L_1+1)}}{L_1 + 1 - L_1} \tag{17}$$

$$\Delta = \lambda_1 - \lambda_{(L_1+1)} \tag{18}$$

$$\lambda_{(L_1+1)} = \lambda_1 - \Delta \tag{19}$$

$$\Delta = \frac{\lambda_1 - \lambda_{(L_1+2)}}{L_1 + 2 - L_1} \tag{20}$$

$$\Delta = \frac{\lambda_1 - \lambda_{(L_1+2)}}{2} \tag{21}$$

$$\lambda_{(L_1+2)} = \lambda_1 - 2\Delta \tag{22}$$

In general:

$$\lambda_{(L_1+k)} = \lambda_1 - k\Delta \quad , k = 0,1,\ldots, L_2 - L_1 \tag{23}$$

From (16) and (23), the equilibrium probability can be solved in terms of $P_{(0)}$:

$$p_{(L_1+y)} = \prod_{i=1}^{y} \frac{(\lambda_1 - (i-1)\Delta)}{\mu} * \rho^{L_1} \; p_{(0)}$$
$$, \; y = 1,2,\cdots, L_2 - L_1 \tag{24}$$

By using the normalization equation $\sum_{i=0}^{L_2} p_i = 1$,

$P_{(0)}$ can be obtained as follows:

$$1 = p_{(0)}\left[\sum_{x=0}^{L_1} \rho^x + \rho^{L_1} \sum_{y=1}^{L_2-L_1} \prod_{i=1}^{y} \frac{(\lambda_1 - (i-1)\Delta)}{\mu}\right] \tag{25}$$

By applying the summation formula for the geometric series

$$\sum_{x=0}^{L_1} \rho^x = \frac{1-\rho^{L_1+1}}{1-\rho} \tag{26}$$

Substituting (26) in (25)

$$p_{(0)} = \frac{1}{\left[\frac{1-\rho^{L_1+1}}{1-\rho} + \rho^{L_1} \sum_{y=1}^{L_2-L_1} \prod_{i=1}^{y} \frac{(\lambda_1 - (i-1)\Delta)}{\mu}\right]} \tag{27}$$

The special case for $P_{(0)}$ when $\lambda_1=\mu$ is given by:

$$p_{(0)} = \frac{1}{\left[(1+L_1) + \sum_{y=1}^{L_2-L_1} \prod_{i=1}^{y} \frac{(\lambda_1 - (i-1)\Delta)}{\mu}\right]} \tag{28}$$

The *MQL* for this finite buffer is given by:

$$MQL = E[N] = \sum_{x=0}^{L_1} x p_{(x)} + \sum_{n=L_1+1}^{L_2} n p_{(n)} \tag{29}$$

From (8) and (24)

$$MQL = p_{(0)}\left[\sum_{x=0}^{L_1} x\rho^x + \rho^{L_1} \sum_{n=L_1+1}^{L_2} n * \prod_{i=1}^{n-L_1} \frac{(\lambda_1 - (i-1)\Delta)}{\mu}\right] \tag{30}$$

Now L_2 is the only unknown in the following equation:

$$MQL - p_{(0)}\left[\sum_{x=0}^{L_1} x\rho^x + \rho^{L_1} \sum_{n=L_1+1}^{L_2} n * \prod_{i=1}^{n-L_1} \frac{(\lambda_1 - (i-1)\Delta)}{\mu}\right] = 0 \tag{31}$$

Equation (31) forms the core of the feedback control algorithm. The bisection method [19] is then used in order to find the roots of this equation to obtain the threshold position L_2 for the next time window (W+1). Then L_2 should be moved to the new position that keeps the *MQL* around its target.

5.2. Controlling the mean delay

The target delay (D_T) can be obtained by applying Little's law as follows:

$$D_T = \frac{MQL}{throughput} = \frac{MQL}{\mu(1 - p_{(0)})} \tag{32}$$

From (30) and (32)

$$D_T = \frac{p_{(0)}\left[\sum_{x=0}^{L_1} x\rho^x + \rho^{L_1} \sum_{n=L_1+1}^{L_2} n * \prod_{i=1}^{n-L_1} \frac{(\lambda_1 - (i-1)\Delta)}{\mu}\right]}{\mu(1 - p_{(0)})} \tag{33}$$

Now L_2 is the only unknown in the following equation:

$$D_T - \frac{p_{(0)}\left[\sum_{x=0}^{L_1} x\rho^x + \rho^{L_1} \sum_{n=L_1+1}^{L_2} n * \prod_{i=1}^{n-L_1} \frac{(\lambda_1 - (i-1)\Delta)}{\mu}\right]}{\mu(1 - p_{(0)})} = 0 \tag{34}$$

The bisection method is then used in order to find the roots of equation (34) to obtain the new position for L_2 for the next time window (W+1). Then L_2 should be moved to the new position that keeps the mean delay around its target.

6. Simulation Results

A discrete event simulation has been implemented using Java programming to assess the performance of the proposed model. The simulation time is divided into time windows of fixed length.

6.1. MQL results

The bisection method has been used to solve equation (31) at the end of each time window to find the new position for $L2$ that attempts to bound the *MQL* to its target value. The parameters used have been initialized as follows: λ_{11}=10, λ_{22}=6, μ=5, γ_1=0.02, γ_2=0.01, L_1=5, $\max_p$=0.1, w_q=0.002, *TMQL*=10 and K=40. Figure 4 represents the measured *MQL* (*MMQL*) compared with the target *MQ*L (*TMQL*) at *TWL*=10. The *MMQL* calculated as the area under the Q(t) curve between the beginning and the end of the simulation, where Q(t) denotes the number of customers in the queue at time (t) [20].

The *MMQL* achieved is 13.09135 and the error variance between *TMQL* and *MMQL* is 9.973281.

The error variance is quite high and this is a result of using a small *TWL*=10. Figure 5 shows that the error variance can be reduced by increasing the length of the time window, with its minimum value around 60 after which it becomes constant. *TWL* =60 can therefore be used as the optimum length of the time window.

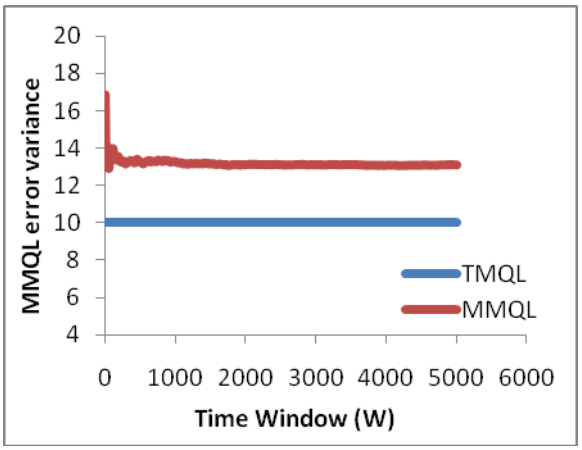

Figure 4. *MMQL* compared with *TMQL* at *TWL*=10

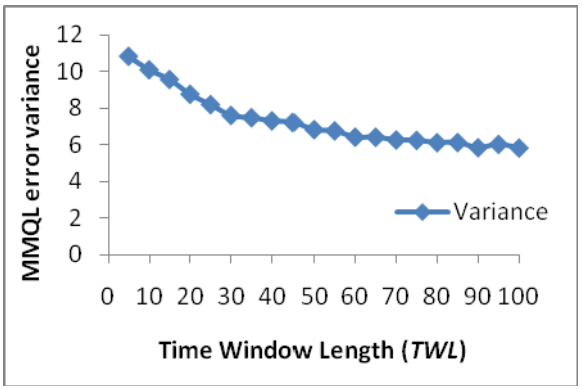

Figure 5. Variance of *MMQL* error vs. Time Window Length (*TWL*)

Another factor that impacts on the measurement of the variance is the amount of error in calculating the *MMQL*. The *MMQL* error in time window W is calculated as $TMQL_W - MMQL_W$. Then $MMQL_{W+1}$ can be expressed as follows:

$$\frac{MMQL_W + MMQL_{W+1}}{2} = TMQL_W \qquad (35)$$

$$MMQL_{W+1} = 2TMQL_W - MMQL_W \qquad (36)$$

$MMQL_{W+1}$ is the predicted value of the measured *MQL* at time window (W+1) based on the current value of the $MMQL_W$ at time window (W). In order to reduce the error in measuring the *MQL*, $MMQL_{W+1}$ is used as the new target for each time window. In this case the results attained are shown in Figure 6.

Compared with the results in Figure 4, The *MMQL* achieved was 11.38484 and the variance between *TMQL* and *MMQL* is 1.918091, which is a much smaller value.

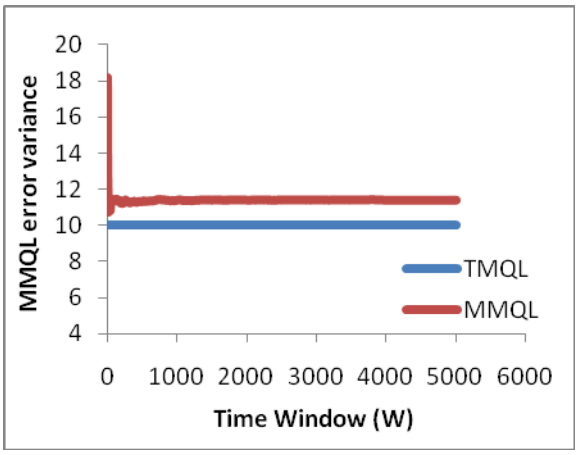

Figure 6. *MMQL* compared with *TMQL* at *TWL*=60

6.2. Mean delay results

Equation (34) has been solved using the bisection method in order to find the roots for L_2 that keeps the target mean delay around its target value. The parameters used have been initialized as follows: λ_{11}=10, λ_{22}=6, μ=5, γ_1=0.02, γ_2=0.01, L_1=5, $\max_p$=0.1, w_q=0.002, D_T=5 and K=40. The two arrival rates have been chosen higher than the service rate to demonstrate the effectiveness of the algorithm in constraining an increasing delay.

Figure 7 represents the measured mean delay (D_M) compared with the target mean delay (D_T). The measured mean delay is calculated as the time spent in the system by all packets divided by the total number of packets served at the end of each time window [20]. By using a *TWL*=10 it is noticeable that the measured mean delay is approaching the target mean delay which is (5). The mean delay achieved is 5.422202 and the variance of delay error is 0.283505. The results of this are shown in Figure 7.

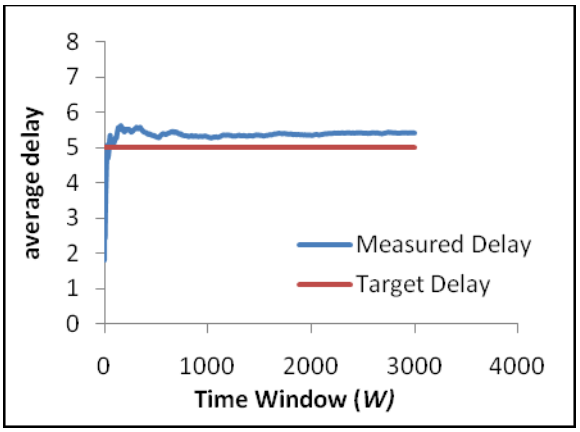

Figure 7. Measured mean delay compared with target mean delay at *TWL*=10

Finding the optimum window length is an important issue and impacts on the accuracy of the measurements. Using a very long window is good in having enough arrivals to accurately measure the mean but these arrivals will most likely be at different arrival rates since the MMPP source might change states within the wide window. Thus changing the arrival rates will not give accurate measurements of the actual mean arrival rate. While using too small window gives better measurements for the actual mean arrival rate as the source probably is less likely to change its state within the window but it is still not very accurate as there will not be many arrivals to measure the mean. This means that the accurate tracking of changes in the arrival rate is another factor that impacts on the performance of the algorithm. The ideal situation is to find the optimum window length that tracks the variations in the arrival rate as closely as possible so that the arrival rate can be measured accurately.

In order to examine the effect of changing the *TWL* on the variance of measurements, different time window lengths have been used. Figure 11 shows the delay error variance calculated at different time window lengths with 95% confidence intervals based on ten trials for each value. The delay error variance represents the error in measuring the delay (D_T - D_M) within each time window and is calculated using Equation (37):

$$\text{Delay error variance} = \frac{1}{N}\sum_{W=1}^{N}(D_T - D_M)^2 \quad (37)$$

N represents the maximum number of time windows used.

From Figure 8 it is noticeable that the delay error variance takes high values if the time window is too short or too long and it reaches its minimum value at *TWL*=7. So *TWL*=7 can be used as the optimum length of the time window. Compared with the results obtained in Figure 7 the mean delay achieved in Figure 9 was 5.147966 and the variance is 0.166576, which is a smaller value. This also shows that the measured delay can be successfully maintained around its target value if an appropriate length is chosen for the time window.

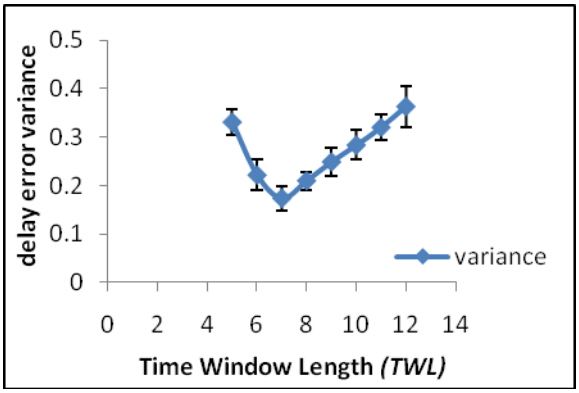

Figure 8. Variance of measured delay error vs. *TWL*

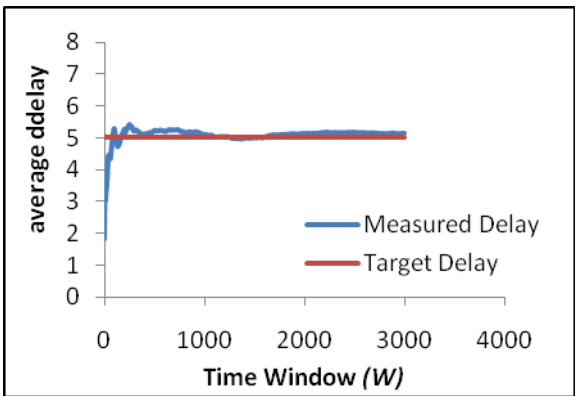

Figure 9. Measured mean delay compared with target mean delay at *TWL*=7

6.3. Using Pareto switchover

Self-similarity and Long Range Dependence (LRD) are the characteristics of some types of modern network traffic. In order to examine the performance of the model in controlling the mean delay under LRD and self-similar conditions, a Pareto switchover has been used in the former MMPP source to replace the exponential one. This means the transition from state1 to state2 and from state2 to state1 will follow the heavy tailed Pareto distribution. Running the model under LRD conditions means that the source might spend a much longer time in one of the states than the other within a limited period. If this is a high

arrival rate state then these situations can lead to congestion.

By setting the arrival rates of the MMPP source to high values λ_{11}=10, λ_{22}=6 and by using a *TWL*=10, the error variance obtained=1.487159 the delay error variance is high because the Pareto switched source stays long times in the high rate states and this causes the delay to be higher than the target value. This is because the target delay value was too low to achieve with the high arrival rate.

By setting one of the arrival rates of the MMPP to a high value λ_{11}=10 and setting the other rate to a low value λ_{22}=4 while the other parameters are kept without change and by using a *TWL*=9 the measured delay is close to the target delay value (5) and the variance=0.249665 as shown in Figure 10. The measured mean delay reached the target mean delay as a result of using a low arrival rate.

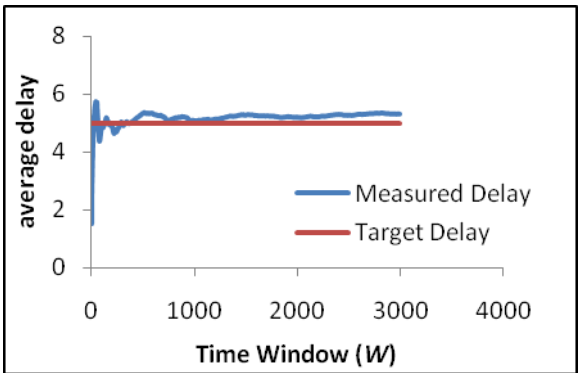

Figure 10. Measured delay compared with target mean delay at *TWL*=9 using LRD source

7. Performance Validation

In order to test and validate the analytical model this is compared with a simulation running in a steady state. In the test model instead of using a dynamic moving threshold, two fixed thresholds have been used to ensure that the model works effectively in a steady state between changes in the arrival rate. For validation purposes the source used has an arrival rate that follows a Poisson distribution since an MMPP source in the dynamic performance evaluation. For all the results obtained the parameters used are: μ=5, P_{max}=0.1, w_q=0.002, queue limit K=40, L_1=5 and L_2=15.

Figure 11 represents the normalized throughput obtained from the analytical model compared with the simulation model and it is clear that they are matching. Figure 12 shows that the mean delay obtained from the simulation is also matches well with the mean delay obtained using the analytical model. Figure 13 also indicates a good match between the dropping probability results obtained from simulation and analytical model.

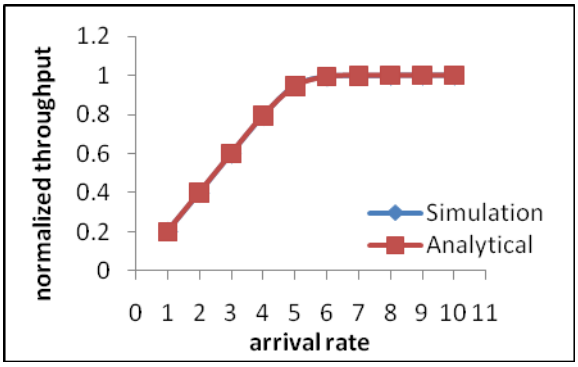

Figure 11. The normalized throughput vs. traffic load

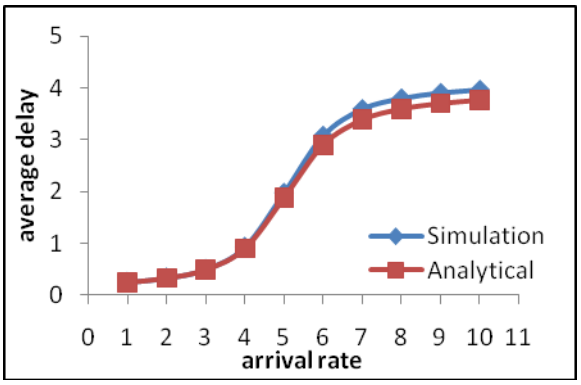

Figure 12.The mean delay vs. traffic load

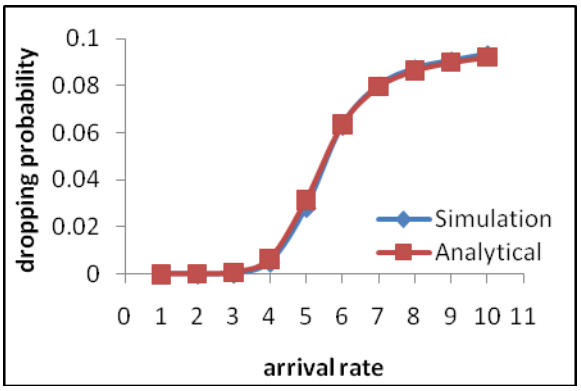

Figure 13.The dropping probability vs. traffic load

8. Conclusions and Future Work

This paper introduces a new algorithm for controlling a buffer with time-varying arrival rate. The proposed approach uses a feedback control strategy to adjust a queue threshold dynamically which, in turn, controls the effective arrival rate by

randomly dropping packets. In practice, if the system is operating under TCP, then these packet drops will cause the source to slow down as TCP responds by reducing its congestion window size. An equation has been developed that relates the threshold position to the target mean queue length over each time window. The model presented can also be used to maintain average delay constraints for delay sensitive applications like real-time services for Internet applications.

The performance of the model depends critically on the length of a time window and also on the ability to accurately detect the changes in the arrival rate. The optimum time window lengths for mean queue length and mean delay are very different, with a much shorter optimum window length required for the mean delay. This is likely due to the mean delay having a much greater sensitivity than mean queue length to changes in the moveable threshold position. This clearly indicates the need for selecting different window sizes depending on the performance metric of interest that is to be controlled. These issues have been investigated by applying the algorithm under different arrival rate conditions and different time window lengths. The algorithm has also been evaluated for a source with Pareto switchover times instead of the exponential ones in order to examine the effects of LRD and self-similarity, since LRD characterizes some types of modern network traffic. The steady state performance of the model has been validated by demonstrating a good match between simulation and analytical models.

The algorithm might be generalized for other arrival processes and for different queuing disciplines. Also, the algorithm might be modified to be used with multi-class traffic, although this will almost certainly require separate buffers for each type of traffic due to the obvious interdependencies between different traffic classes in a shared buffer environment, which would make an equilibrium state extremely difficult to achieve. Also the use of a sliding window instead of a fixed one could be investigated.

9. Acknowledgements

Rasha Fares gratefully acknowledges the provision of studentship from the Egyptian Ministry of Higher Education.

10. References

[1] A. Guido, K. Isaac, and M. Nick, "Sizing router buffers," *SIGCOMM Comput. Commun. Rev.*, vol. 34, pp. 281-292, 2004.

[2] Z. Wei, T. Liansheng, and P. Gang, "Dynamic queue level control of TCP/RED systems in AQM routers," *Comput. Electr. Eng.*, vol. 35, pp. 59-70, 2009.

[3] P. Hsiao, H. T. Kung, and K. Tan, "Active delay control for TCP," presented at Global Telecommunications Conference, 2001. GLOBECOM '01. IEEE, San Antonio, TX, USA, 2001.

[4] H. Liu and M. El Zarki, "Delay and synchronization control middleware to support real-timemultimedia services over wireless PCS networks," *IEEE Journal on Selected Areas in Communications*, vol. 17, pp. 1660-1672, 1999.

[5] C. Brandauer, G. Iannaccone, C. Diot, T. Ziegler, S. Fdida, and M. May, "Comparison of Tail Drop and Active Queue Management Performance for Bulk-Data and Web-Like Internet Traffic," *Proceedings of the Sixth IEEE Symposium on Computers and Communications*, pp. 122 -129, 2001.

[6] B. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, C. Partridge, L. Peterson, K. Ramakrishnan, S. Shenker, J. Wroclawski, and L. Zhang, "Recommendations on Queue Management and Congestion Avoidance in the Internet," RFC 2309 April 1998.

[7] E. Hashem, "Analysis of random drop for gateway congestion control," in *Department of Electrical Engineering and Computer Science*, vol. M.S. thesis: Massachusetts Institute of Technology, 1989.

[8] S. Floyd and V. Jacobson., "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Netw.*, vol. 1, pp. 397-413, 1993.

[9] A. Mankin, "Random drop congestion control," in *Proceedings of the ACM symposium on Communications architectures & protocols*. Philadelphia, Pennsylvania, United States: ACM Press, 1990.

[10] S. Floyd, R. Gummadi, and S. Shenker, "Adaptive RED: An Algorithm for Increasing the Robustness of RED's Active Queue Management," *Technical Report*, 2001.

[11] W.-c. Feng, D. D. Kandlur, D. Saha, and K. G. Shin, "BLUE: A New Class of Active Queue Management Algorithms ", 1999.

[12] S. Athuraliya, V. H. Li, S. H. Low, and Q. Yin., "REM: Active Queue Management," *IEEE Network*, vol. 15, 2001.

[13] B. Zheng and M. Atiquzzaman, "DSRED: an active queue management scheme for next generation networks," in *Proceedings of the 25th Annual IEEE Conference on Local Computer Networks*: IEEE Computer Society, 2000.

[14] M. Welzl, *Network congestion control : managing Internet traffic*: WileyBlackwell 2005.

[15] Y. ZHENG, M. LU, and Z. FENG, "Performance Evaluation of Adaptive AQM Algorithms in a Variable

Bandwidth Network," *IEICE Trans Commun (Inst Electron Inf Commun Eng)*, vol. E86-B, pp. 2060-2067, 2003.

[16] H. Jaesung, J. Changhee, and B. Saewoong, "Active queue management algorithm considering queue and load states," *Comput. Commun.*, vol. 30, pp. 886-892, 2007.

[17] W. Fischer and K. Meier-Hellstern, "The Markov-modulated Poisson process (MMPP) cookbook," *Perform. Eval.*, vol. 18, pp. 149-171, 1993.

[18] R. O. Onvural, *Asynchronous Transfer Mode Networks: Performance Issues*, 2nd ed: Artech House, 1995.

[19] C. F. Gerald and P. O. Wheatley, *Applied Numerical Analysis*, 6th ed: Addison Wesley Publishing Company, 1999.

[20] A. M. Law and W. D. Kelton, *Simulation Modeling and Analysis*, 3rd ed: McGraw-Hill, 2000.

