ACCESS CONTROL FOR MPEG VIDEO APPLICATIONS USING NEURAL NETWORK AND SIMULATED ANNEALING

N. U. AHMED AND HONG YAN

Received 3 March 2004

We present a dynamic model for access control mechanism used in computer communication network applied to MPEG video transmission over Internet. This model is different from those developed in the previous works related to this topic. In our model, token buckets supported by data buffers are used to shape incoming traffic and one multiplexor, serving all the token pools, multiplexes all the conforming traffic. The model is governed by a system of discrete nonlinear difference equations. We use neural network as the feedback controller which receives at its input (measurable) available information and provides at its output the optimal control. The simulated annealing algorithm is used to optimize the system performance by adjusting the weights. For illustration, we present numerical results which show that the system performance of MPEG video server can be improved by using neural network and simulated annealing approach.

1. Introduction

Currently, more and more multimedia applications such as video are transmitted or delivered on the Internet in order to provide quality of service (QoS) guarantees to real-time applications. The Internet Engineering Task Force (IETF) has proposed integrated service (Intserv) and differentiated service (DiffServ). In particular, token bucket (TB) as a traffic controller has been incorporated in the design of the network architecture.

In this paper, we focus our efforts on the study of the TB as a traffic shaper. A TB is associated with a traffic source in order to regulate the rate at which the source may send its traffic into the network. The terminology of TB as a shaper is that tokens are generated into the bucket at a rate determined by the volume of incoming traffic and the state of the network. When the bucket is full of tokens, newly arriving tokens are rejected. If, upon a packet arrival, enough tokens are available, the token content of the bucket is decreased according to the packet size; the packet is then classified as belonging to the conforming traffic and sent into the network. On the other hand, if there are not enough tokens in the bucket, the packet is buffered in the data buffer (DF) and waits for a fresh supply of

Mathematical Problems in Engineering 2004:3 (2004) 291–304
2000 Mathematics Subject Classification: 93A30, 34A36, 93C55
URL: http://dx.doi.org/10.1155/S1024123X04403019

tokens in the bucket. When the buffer is full, the arriving packet can not be accepted by the buffer and so is discarded.

TB mechanism has been employed to characterize the traffic generated by MPEG video in [4, 6, 8]. In [4], Alam et al. study the effects of a traffic shaper at the video source on the transmission characteristics of MPEG video streams while providing delay and bandwidth guarantees. In [6], Lombaedo et al. have applied the TB model to evaluate the probability of marking nonconforming data packets for the transmission of MPEG video on the Internet. The authors of [8] developed an algorithm to evaluate the TB parameters which characterize the video stream.

In this paper, we develop a dynamic model for MPEG video transmission over network using TB access control mechanism. The system includes buffers and a shared multiplexor controlled by a neural network.

Our model differs from most of the previous studies in two major points. First, we focus on the modelling of the TB algorithm as a shaper rather than a policer as in [1, 2, 3]. Second, we propose the use of neural network and simulated annealing (SA) technique to control token generation rate in order to optimize the system performance.

The paper is organized as follows. In Section 2, a dynamic system model is presented. The system model consists of a number of TBs (one for each source) with DFs and one multiplexor shared by all the TBs. System state is defined in detail. In Section 3, we define an appropriate objective function and propose a control strategy by employing the neural network. We use SA algorithm to optimize system performance. In Section 4, we present numerical results which show improved performance gained by use of our proposed control. A brief conclusion is presented in Section 5.

2. Traffic model and system model

In this paper, we present a dynamic model based on the basic philosophy of IP networks. Before describing the system model, we define the following symbols which are used throughout the paper:

(1) for $x, y \in \mathbb{R}$, we define

$$x \wedge y \equiv \operatorname{Min}\{x, y\}, \qquad x \vee y \equiv \operatorname{Max}\{x, y\}; \tag{2.1}$$

(2) for $x, y \in \mathbb{R}^n$, let $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$. We define

$$x \wedge y \equiv (x_1 \wedge y_1, \ldots, x_n \wedge y_n), \qquad x \vee y \equiv (x_1 \vee y_1, \ldots, x_n \vee y_n); \tag{2.2}$$

(3)

$$I(S) = \begin{cases} 1, & \text{if the statement } S \text{ is true,} \\ 0, & \text{otherwise.} \end{cases} \tag{2.3}$$

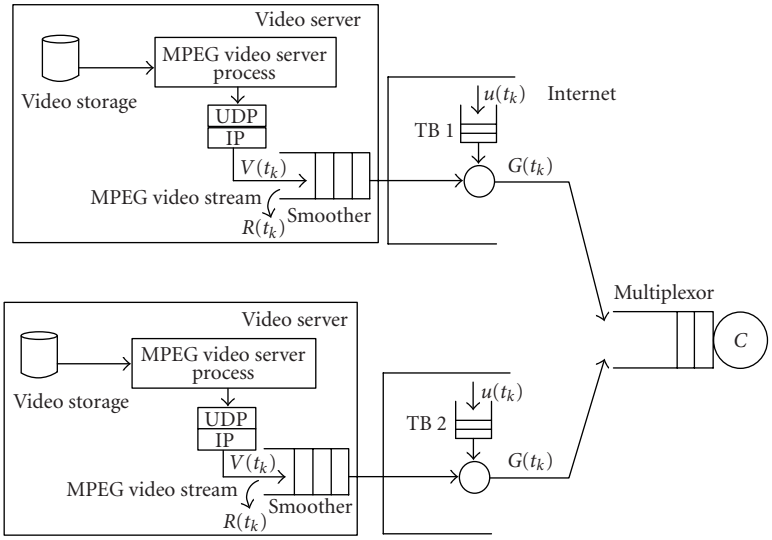


Figure 2.1. MPEG video transmission over Internet.

2.1. Traffic model. The traffic in this paper is MPEG-encoded video streams which is transmitted by a video server to clients over the network. MPEG video sequences are stored in video storage. The server can retrieve data from this storage and transfer it. Video streams are packetized in a UDP/IP protocol suite. Here, we consider each MPEG-encoded frame packetized by the UDP/IP of a size of 576 bytes. In other words, each frame content is divided by 576 into one or more packets of size equal to or less than 576 bytes. Therefore, in our traffic model, a number of IP packets (which may originate from different frames such as (I, P, B)) per frame interval are emitted. Let I_k, $k = 0,1,2,\ldots,K$, be the frame intervals $I_k \equiv [t_k, t_{k+1})$, which are assumed to be of equal length. Let $V(t_k)$ denote the number of IP packets which the frame has during frame interval $I_k \equiv [t_k, t_{k+1})$.

2.2. System model. We consider that the system involves n traffic sources, which are video servers, and that each source is assigned to a TB with a DF that regulates the traffic rate. All the traffic sources share one and the same multiplexor having finite buffer size Q and one outgoing link having capacity C. The system model is shown in Figure 2.1.

Let T denote the physical size (capacity) of TBs and $\hat{T}$ the DF (smoother) size. The source $V(t_k)$ denotes the number of packets entering the DF during the time interval $I_k \equiv [t_k, t_{t+1})$ and $G(t_k)$ denotes the conforming packets, which are allowed to enter into the multiplexor by the TBs, after TB shaping during the same period. Let $u(t_k)$ denote the number of tokens generated during the frame interval I_k. In case of multiple clients and servers, these quantities should be considered as vectors. In this case, the system state is defined by the contents of the TBs, the token buffers, and the multiplexor. We denote this by the $(2n+1)$-dimensional vector

$$(\rho, \hat{\rho}, q). \tag{2.4}$$

Here, for n TBs, the state is described by an n-dimensional vector-valued function $\rho = (\rho_1, \rho_2, \ldots, \rho_n)'$; for n DFs, we have n-dimensional vector $\hat{\rho} = (\hat{\rho}_1, \hat{\rho}_2, \ldots, \hat{\rho}_n)'$. The scalar (one-dimensional vector) q denotes the state of the multiplexor.

First, we determine the temporal evolution of the state (number of tokens in token pool) of the TB as follows:

$$\rho(t_{k+1}) = \rho(t_k) + \{u(t_k) \wedge [T - \rho(t_k)]\} - G(t_k). \tag{2.5}$$

This expression is based on simple material balance. The state of each TB at time t_{k+1} is determined by its previous state, the number of tokens accepted during the frame interval $[t_k, t_{k+1})$, and the number of tokens consumed during the same period $[t_k, t_{k+1})$. The consumption equals the conformed traffic $G(t_k)$.

Similarly, the state of the DFs, denoting the number of packets in each DF, is described by

$$\hat{\rho}(t_{k+1}) = \hat{\rho}(t_k) + \{V(t_k) \wedge [\hat{T} - \hat{\rho}(t_k)]\} - G(t_k). \tag{2.6}$$

Here the number of packets at the current stage is determined by its previous state, the number of packets accepted during the frame interval, and the number of packets conformed by TB.

In these equations, G represents the conforming traffic, that is, the traffic matching with the available tokens in the TBs. This is given by $G(t_k)$ as

$$\begin{aligned} G(t_k) = {} & \{\hat{\rho}(t_k) + \{V(t_k) \wedge [\hat{T} - \hat{\rho}(t_k)]\}\} \\ & \wedge \{\rho(t_k) + \{u(t_k) \wedge [T - \rho(t_k)]\}\}. \end{aligned} \tag{2.7}$$

This expression simply states that the conforming traffic is given by the minimum of the available number of tokens in the token pool and the available number of packets in the DF. In other words, the TB can serve traffic which matches the available tokens. This traffic is called the conforming traffic.

Now we consider the multiplexor. The dynamics of the multiplexor queue is given by the following equation:

$$q(t_{k+1}) = \{[q(t_k) - C\tau] \vee 0\} + \left\{\left[\sum_{i=1}^{n} G_i(t_k)\right] \wedge [Q - [[q(t_k) - C\tau] \vee 0]]\right\}. \tag{2.8}$$

The first term on the right-hand side of expression (2.8) describes the leftover traffic in the queue at time t_{k+1} after the traffic has been injected into the network. The second term represents all the conforming traffic accepted by the multiplexor during the same period of time.

In summary, the complete system model is given by the vector difference equation (2.5), (2.6), and the scalar difference equation (2.8).

Now, we discuss packet losses at the DF and multiplexor. Some packets may be dropped at the DF because of the limitation of the DF size and the token generation rate. Therefore, the losses at the DF at time t_k are given by

$$\mathbb{R}(t_k) = \{V(t_k) - [\hat{T} - \hat{\rho}(t_k)]\} I\{V(t_k) \geq [\hat{T} - \hat{\rho}(t_k)]\}. \tag{2.9}$$

When the available (data) buffer space exceeds the incoming traffic, there will be no packet losses at the DF. Otherwise, in general, some losses would occur. From (2.8), it is clear that if the multiplexor (of size Q) does not have enough space to store all the conformed traffic, the excess is discarded. The traffic losses in the multiplexor at time t_k, denoted by $L(t_k)$, are then given by

$$L(t_k) = \left[\sum_{i=1}^{n} G_i(t_k)\right] - \left\{\left[\sum_{i=1}^{n} G_i(t_k)\right] \wedge [Q - [[q(t_k) - C\tau] \vee 0]]\right\}. \tag{2.10}$$

The term $\{[\sum_{i=1}^{n} G_i(t_k)] \wedge [Q - [[q(t_k) - C\tau] \vee 0]]\}$ represents conforming traffic accepted by the multiplexor. If the available buffer space is large enough to accept all the conforming traffic, no multiplexor losses would occur. Otherwise, some part of the conforming traffic must be dropped.

3. Neural network/simulated annealing algorithm for performance optimization

We have seen above several losses that may occur in the network. In our system, losses may occur at DF and also at the multiplexor. Further, there are delays in DF and the multiplexor due to time spent in the queue before being served. Considering all these factors, we can define the following objective function:

$$J(u) \equiv \sum_{k=0}^{K} \alpha(t_k) L(t_k) + \sum_{i=1}^{n} \sum_{k=0}^{K} \beta_i(t_k) \mathbb{R}_i(t_k) + \sum_{k=0}^{K} \gamma(t_k) \hat{\rho}(t_k) + \sum_{k=0}^{K} \lambda(t_k) q(t_k). \tag{3.1}$$

The first term of the objective functional describes the weighted packet losses at the multiplexor; the second term represents the weighted losses at the DFs; the third term gives the weighted cost associated with service delay at the DF; and the last term is the weighted cost associated with queuing delay at the multiplexor. These last two terms are approximate measures of service delay. Relative weights to various losses are denoted by the parameters $\alpha(t_k)$, $\beta_i(t_k)$, $\gamma(t_k)$, $\lambda(t_k)$, $i = 1, 2, \ldots, n$, which are nonnegative functions of time and can be assigned by network designers to reflect different concerns and scenarios as necessary.

To find the optimal control for the dynamic system (equations (2.5), (2.6), (2.8)) with the objective functional (3.1), we may use dynamic programming (DP) or genetic algorithm (GA) as proposed in a recent paper [1]. This gives rise to the Bellman equation of DP. Note that because of the presence of logical functions in the dynamic equations, the associated Hamilton-Jacobi-Bellman (HJB) equation will have discontinuous coefficients. This is a much more difficult problem to solve (see [1]). Here we propose a sufficiently efficient but simple control strategy by using the neural network and SA algorithm [5, 7]. We use SA to find the optimal weights that minimize the cost function.

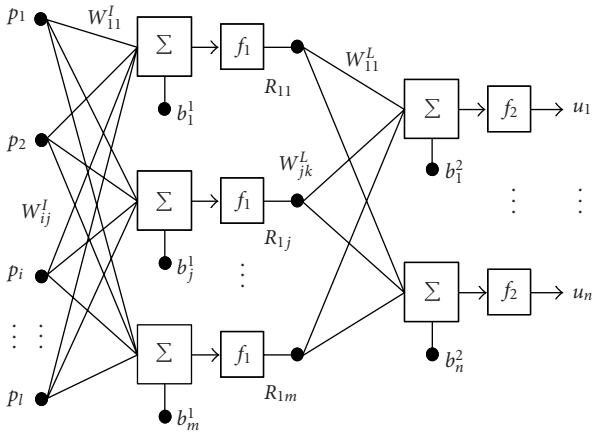


Figure 3.1. 2-layer neural network.

3.1. Neural network as controller. In principle, one can use Bellman's DP equations to find optimal feedback controls. Often it involves difficult nonlinear partial differential equations in multidimensional spaces which have to be solved offline to determine the control law. Clearly, this is computationally intensive. If one is satisfied with suboptimal strategies, this can be avoided by using simple neural network with inputs being the available information and outputs being the controls. Once this is integrated with the system, the only variables that have to be chosen optimally are the weights of the neural network. This can be done relatively easily using SA. In this paper, we have used the following neural network structure with p denoting the input vector and u the output vector representing the control. The parameters shown in Figure 3.1 are the weights which can be adjusted for optimization as discussed later. All these can be compactly described by the following parameterized nonlinear transformation:

$$u \equiv N(W,p), \tag{3.2}$$

where $u_r = N_r(W,p)$, $r = 1,2,\ldots,n$, and the input-state vector $p = (V,\rho,\hat{\rho},q)$. If this control law is inserted in the state equations (2.5), (2.6), (2.8), the corresponding cost functional (see expression (3.1)) may be written as a function of the weight vector W:

$$\begin{aligned} J(W) \equiv & \sum_{k=0}^{K} \alpha(t_k) L(t_k,W) + \sum_{i=1}^{n} \sum_{k=0}^{K} \beta_i(t_k) \mathbb{R}_i(t_k,W) \\ & + \sum_{k=0}^{K} \gamma(t_k) \hat{\rho}(t_k,W) + \sum_{k=0}^{K} \lambda(t_k) q(t_k,W), \end{aligned} \tag{3.3}$$

where now the state obviously depends on the choice of W. Thus the original optimal control problem has been transformed into a parameter optimization problem. More

Table 4.1. MPEG-4 Simpsons statistics specification.

Max frame size	Min frame size	Peak rate μ_p	Mean rate μ
15234 bytes	555 bytes	3.047 Mbps	1.490 Mbps

precisely, the problem is to find a $W^o \in \mathbb{R}^d$, $d = m(\ell+1)+n(m+1)$, that minimizes the cost functional $J(W)$ given by (3.3) subject to the dynamic constraints (2.5), (2.6), (2.8), and (3.2). For this purpose, one can use DP and GA as in [1]. But because of the curse of dimensionality presented by DP, computational time is extremely large. To avoid the computational complexity of DP, we have chosen to accept suboptimal policies which can be determined by use of SA algorithm in a reasonable computing time. In other words, we find the lowest possible cost achievable in a finite number of iterations and declare the corresponding weights as being optimum.

4. Numerical results

For numerical simulation, we consider the simple scenario: one MPEG server emits IP packets with the flow controlled by one TB transmitting conformed outputs into the multiplexor.

4.1. Specification of traffic trace. The traces used in our experiments correspond to the traffic statistics of one MPEG-4 sequence. Here, we choose 12-second MPEG-4 trace of "Simpson" as our traffic (http://trace.eas.asu.edu). Each group of pictures (GoP) contains 12 frames with a frame rate of 25 frames/second. The GoP pattern can be characterized as IBBPBBPBBPBB. I-frame stands for "intraframe" which is the most basic type of frames used in video compression. P-frame, "predictive frame," is more complicated than I-frames. P-frames are built from a previous I- or P-frame. B-frame stands for "bidirectional frame." The basis of B-frames is the logical extension of the idea of P-frames. Whereas P-frames are built from the previous frame, B-frames are built from two frames, typically, one or both are P-frames. B-frames are completely independent of each other. The video traffic statistics which is used in our experiment is given in Table 4.1. Since the maximum frame size is 15234, a video frame is segmented into a number of packets equal to or less than 27. The trace is shown in Figure 4.1.

4.2. Parameters used in simulation experiment. In this paper, we assume UDP/IP protocol suite. Therefore, each frame is packetized by UDP/IP protocol into more IP packets with a dimension of 576 bytes or less. We further assume that one token takes one packet. We consider the tradeoff between two actual losses (packet losses at the DF and the multiplexor) and the cost associated with waiting time in the DF and multiplexor buffer. The weights assigned to these losses are arbitrary and depend on the QoS desired. The service provider and the network management are free to choose these weights according to specified QoS. For our numerical experiments, we choose $\alpha(t_k) = 10$, $\beta_i(t_k) = 5$, $\gamma(t_k) = 0.05$, and $\lambda(t_k) = 0.05$, for all i and all t_k. State initialization is set as $\rho(t_0) = 0$, $\hat{\rho}(t_0) = 0$, and $q(t_0) = 0$. The system configurations are specified as follows:

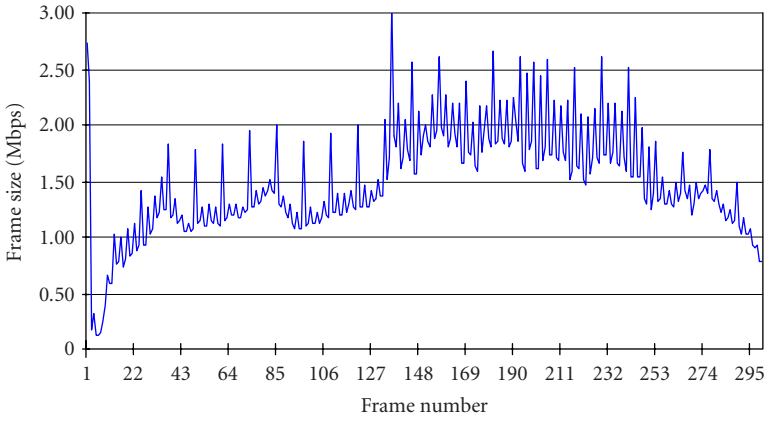


Figure 4.1. 7-second MPEG-4 trace (Simpsons).

(i) $T = 15000$ bytes; the TB has been set to allow maximum frame size to pass through;
(ii) $C = 1.0$ Mbps, 1.49 Mbps, and 2.0 Mbps;
(iii) $\hat{T} = 30000$ bytes; the DF size has been set for twice the maximum frame size;
(iv) $Q = 17000$ bytes; the multiplexor has been set large enough to be able to accommodate the maximum frame;
(v) $\tau = 40$ milliseconds; this represents the frame interval (frames are generated at a rate of 25 frames per second).

For our experiment, we choose the structure of a neural network as follows:

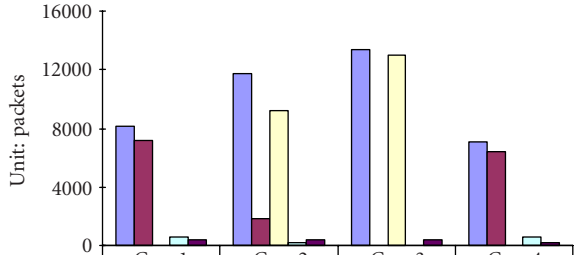
(a) number of inputs: 4;
(b) number of outputs: 1;
(c) number of layers: 2;
(d) number of neurons in layer 1: 3;
(e) number of neurons in layer 2: 1;
(f) transfer function in layer 1: tangent sigmoid;
(g) transfer function in layer 2: positive linear.

As to the parameters for SA, we use geometric decrement method, which sets the next temperature t_n as a positive multiple of the current temperature t_c, that is, $t_n = \phi t_c$, with $0 < \phi < 1$. In this experiment, we take $\phi = 0.6$.

4.3. Simulation results and analysis. Here we present numerical simulation results by studying the following cases.

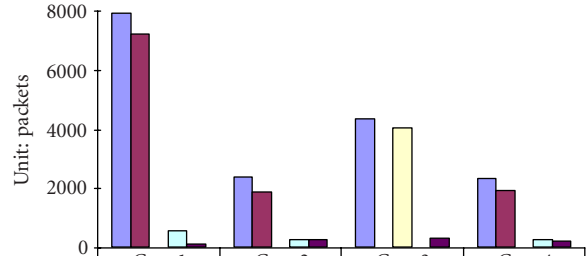
(1) Case 1: open loop $u = 1$ Mbps ($\leq$ mean rate).
(2) Case 2: open loop $u = \mu = 1.49$ Mbps (mean rate).
(3) Case 3: open loop $u = \mu_p = 3.047$ Mbps (peak rate).
(4) Case 4: optimal control based on neural network providing variable bit rate.

Here, μ denotes the mean traffic rate and μ_p the peak rate.



	Case 1	Case 2	Case 3	Case 4
Total cost	8126	11703	13369	7064
Buffer loss	7210	1845	0	6354
Mul loss	0	9200	12953	0
Buffer delay	553	242	0	543
Mul delay	363	416	416	170

Figure 4.2. Dependence of cost on control strategy ($C = 1$ Mbps).



	Case 1	Case 2	Case 3	Case 4
Total cost	7882	2363	4323	2337
Buffer loss	7210	1845	0	1905
Mul loss	0	0	4040	0
Buffer waiting	553	242	0	250
Mul waiting	119	276	283	182

Figure 4.3. Dependence of cost on control strategy ($C = 1.49$ Mbps).

4.3.1. Dependence of cost on control strategies. Figures 4.2 and 4.3 show the system performance corresponding to four different control strategies. The results of Figure 4.3 are based on link capacity $C = 1$ Mbps and those of Figure 4.4 are based on link capacity $C = 1.49$ Mbps. The results show that Case 4, which corresponds to optimal control based on neural network, offers the minimum cost despite the variation of the output link capacity. Cases 1, 2, and 3, which correspond to open-loop control strategies, are inferior to feedback control strategy provided by neural network (Case 4). In the case of $C = 1.0$ Mbps (see Figure 4.2), the output link capacity is equal to or less than the token generating rates (Cases 1, 2, and 3). In this situation, as the token generating rate increases, in spite of reduced TB losses, more and more packets are dropped by the multiplexor thereby increasing the overall cost.

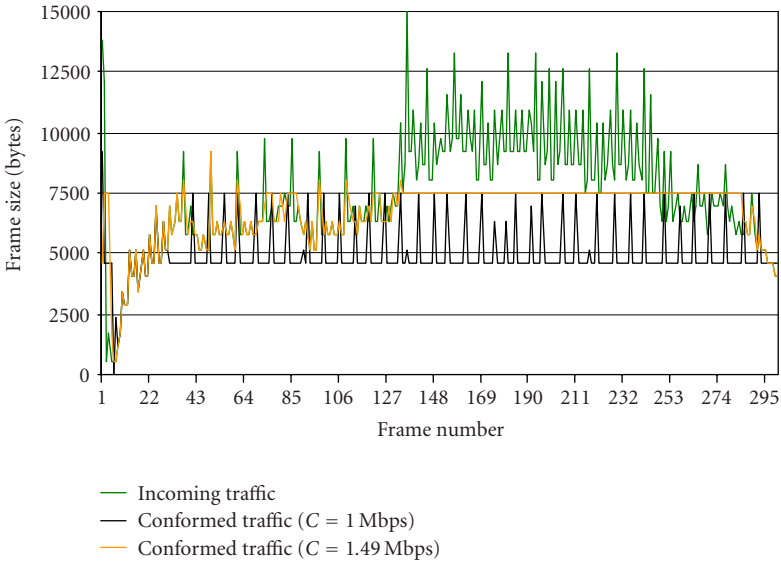


Figure 4.4. Incoming traffic versus conformed traffic ($C = 1$ Mbps, $C = 1.49$ Mbps).

In the case of $C = 1.49$ Mbps (see Figure 4.3), we observe that the (total) cost corresponding to Case 2 is very close to that of Case 4. This is because the token generating rate in Case 2 is the same as the traffic mean rate and the outgoing link capacity. When the traffic mean rate is equal to the token generating rate, the DF losses are small. Since in this case, the input rate of the multiplexor equals the outgoing link capacity, no losses occur in the multiplexor either. Hence the performance of the mean rate control and the optimal control are close.

4.3.2. Performance analysis. In this section, we present the temporal performance of the feedback system (based on optimized neural network) by comparing the data flow in the TB and the DF.

Comparison of incoming traffic with conformed traffic. Figure 4.4 presents, for different link capacities ($C = 1.0,\ 1.49$ Mbps), the traces of the incoming traffic along with the conformed traffic resulting from a control strategy based on optimized neural network. It is natural that the conformed traffic depends on the outgoing link capacity. In the case of $C = 1.0$ Mbps, which is less than the mean incoming traffic rate, the feedback control tries to follow the incoming traffic but, because of shortage of bandwidth, fails to satisfy the demand. In the case of $C = 1.49$ Mbps (equal to the mean traffic rate), when the traffic rate is much higher than the mean rate during some time periods, the network saturates and serves at its maximum capacity. Therefore, during these busy periods, the conformed traffic is maintained at the saturation level. For $C = 2$ Mbps, Figure 4.5 shows the conformed traffic trace. In this case, the capacity $C = 2$ is greater than the mean traffic rate and, as a result, the conformed traffic trace, shown in Figure 4.5, almost matches with

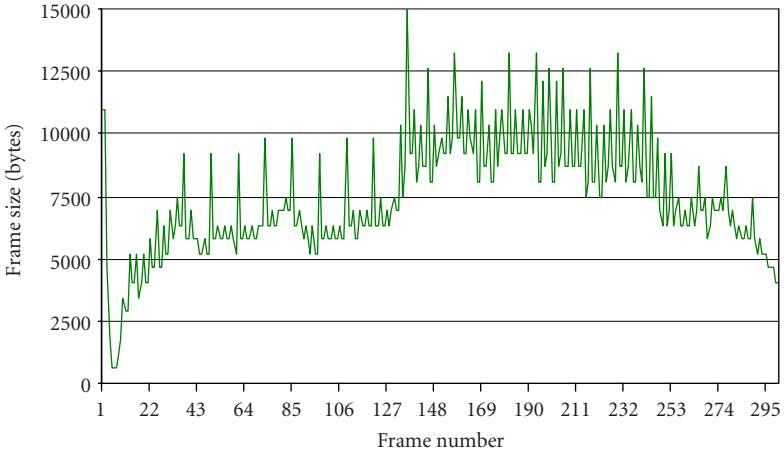


Figure 4.5. Conformed traffic ($C = 2$ Mbps).

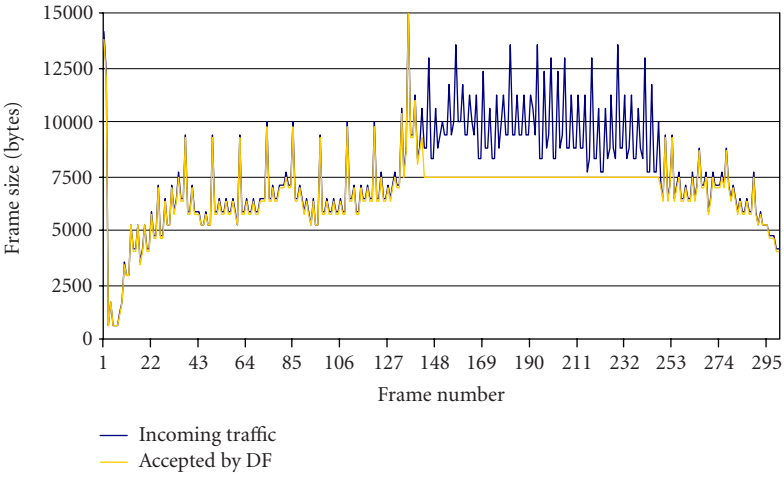


Figure 4.6. Incoming traffic versus traffic accepted by DF ($C = 1.49$ Mbps).

the incoming traffic (Figure 4.4). This is because the service rate is large enough to serve the incoming traffic with less losses.

Comparison of incoming traffic with accepted traffic. Figure 4.6 shows the traces of incoming traffic along with the traffic accepted by the DF which is based on the link capacity (equal to the mean traffic rate). Again we observe that during some time periods (see Figure 4.6, 145–250), when the incoming traffic rate is greater than the link capacity (equivalent to the mean traffic rate), the network experiences congestion and during these periods, the network serves at its maximum capacity (7450 bytes within one frame interval). As a result, the DF drops packets whenever the rate exceeds this level.

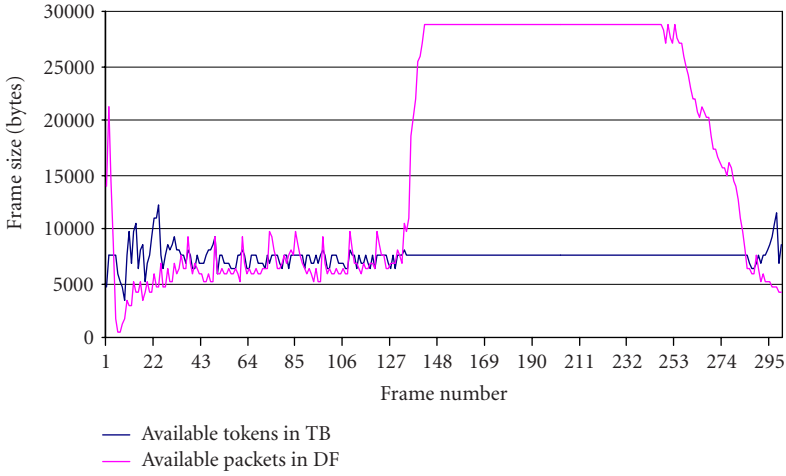


Figure 4.7. Available tokens in TB versus available packets in DF ($C = 1.49$ Mbps).

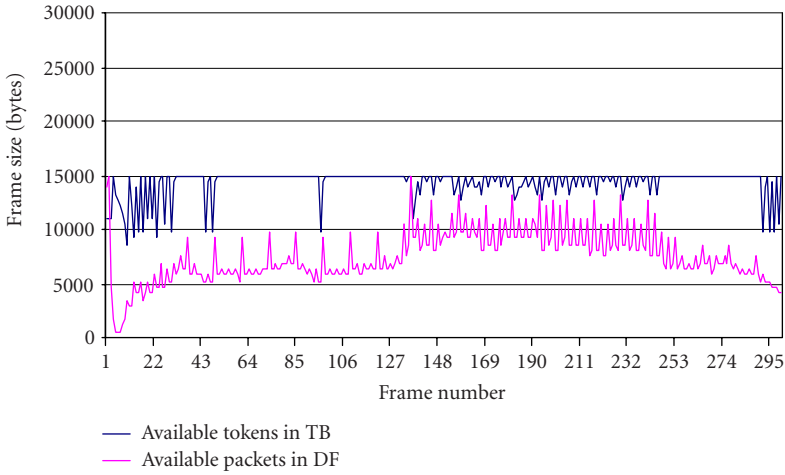


Figure 4.8. Available tokens in TB versus available packets in DF ($C = 2.0$ Mbps).

Comparison of available tokens with packets in DF. Figures 4.7, 4.8, and 4.9 give us the traces of available tokens in the TB and available packets in the DF based on optimal control strategy corresponding to different link capacities. In the case of $C = 1.49$ Mbps (see Figure 4.7), we notice that when the traffic rate is greater than the mean rate, the DF is full ($\hat{T} = 30K$), and during the same period, the available tokens do not exceed the number of packets the network can serve. For $C = 2$ Mbps (Figure 4.8) and $C = 3.047$ Mbps (Figure 4.9), the available tokens in the token pool are larger than the available packets in the DF. In these cases, losses are significantly reduced as expected.

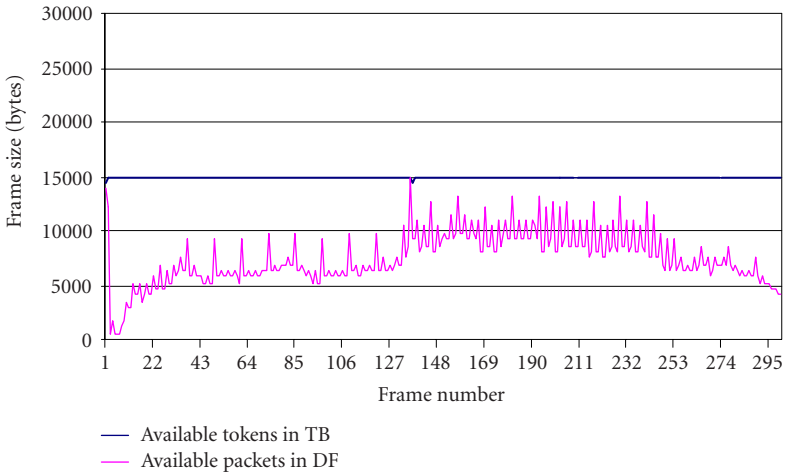


Figure 4.9. Available tokens in TB versus available packets in DF ($C = 3.047$ Mbps).

5. Conclusion

In this paper, we have constructed a dynamic system model based on token bucket algorithm with supporting buffers and multiplexors. We have proposed a feedback control strategy using neural network. We define an objective functional reflecting all the system losses including queuing delays. Using simulated annealing algorithm, we minimize the cost (objective) functional giving the optimal control law (neural network with optimum weights). The numerical results indicate that a neural-network-based control law properly optimized can produce improved system performance.

References

[1] N. U. Ahmed, B. Li, and L. Orozco Barbosa, *Optimization of computer network traffic controllers using a dynamic programming/genetic algorithm approach*, working paper, 2002.

[2] N. U. Ahmed, Q. Wang, and L. Orozco Barbosa, *Systems approach to modeling the token bucket algorithm in computer networks*, Math. Probl. Eng. **8** (2002), no. 3, 265–279.

[3] N. U. Ahmed, H. Yan, and L. Orozco-Barbosa, *Performance analysis of the token bucket control mechanism subject to stochastic traffic*, Dyn. Contin. Discrete Impuls. Syst. Ser. B Appl. Algorithms **11** (2004), no. 3, 363–391.

[4] M. F. Alam, M. Atiquzzaman, and M. A. Karim, *Effects of source traffic shaping on MPEG video transmission over next generation IP networks*, Proc. IEEE International Conference on Computer Communications and Networks (Massachusetts), 1999, pp. 514–519.

[5] N. Jakatdar, X. Niu, and C. J. Spanos, *A neural network approach to rapid thin film characterization*, Flatness, Roughness, and Discrete Defects Characterization for Computer Disks, Wafers, and Flat Panel Displays II, LASE '98 Precision Manufacturing Technologies Photonics West, vol. 3275, SPIE, California, 1998, pp. 163–171.

[6] A. Lombaedo, G. Schembra, and G. Morabito, *Traffic specifications for the transmission of stored MPEG video on the Internet*, IEEE Trans. Multimedia **3** (2001), no. 1, 5–17.

[7] S. Moins, *Implementation of a simulated annealing algorithm for Matlab*, preprint, 2002, http://www.ep.liu.se/exjobb/isy/2002/3339/.

[8] S.-H. Park and S.-J. Ko, *Evaluation of token bucket parameters for VBR MPEG video transmission over the Internet*, IEEE Transactions on Communications **E85-B** (2002), no. 1, 43–51.

N. U. Ahmed: School of Information Technology and Engineering and Department of Mathematics, University of Ottawa, 161 Luis Pasteur, Ottawa, Ontario, Canada K1N 6N5
E-mail address: ahmed@site.uottawa.ca

Hong Yan: 207-304 Sherk Street, Leamington, Ontario, Canada N8H3W
E-mail address: lily_yanhong@yahoo.com