REAL-TIME FEEDBACK CONTROL OF COMPUTER NETWORKS BASED ON PREDICTED STATE ESTIMATION

N. U. AHMED AND HUI SONG

We present a real-time feedback control strategy to optimize the dynamic performance of a computer communication network. In previous studies closely related to this topic, feedback delay, arising from communication delay, was shown to degrade system performance. Considering this negative impact of delay, we propose a new control law which predicts the traffic in advance and exercises control based on the predicted traffic. We demonstrate through simulation experiments that the predictive feedback control law substantially improves the system performance.

1. Introduction

With the explosive growth of the Internet over the past twenty years, excessive network congestion occurs frequently. This kind of congestion results in packet losses, transmission delay, and delay variation. Moreover, the increasing integrated applications (e.g., web surfing, teleconferencing, and e-commerce) are sensitive to delay and packet losses. Therefore, it becomes important to efficiently manage and optimally control the traffic, and to ultimately offer a guarantee of quality of service (QoS) for multimedia applications.

A number of different control mechanisms have been proposed to solve these problems. The token bucket (TB) algorithm is one of the methods widely used in the network access control field (see [14, 16, 17]; see also http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/12cgcr/qos_c/qcpart4/qcpolts.htm, and it can dynamically allocate bandwidth and efficiently minimize packet losses. Previous studies [1, 2] used TB to construct a dynamic system model. Additionally, different control strategies were proposed to manage traffic flow into the backbone network. The results showed that the feedback control laws can improve network performance by improving throughput, reducing packet losses, and relaxing congestion. On the other hand, in [2], it was observed that the system performance was highly degraded in the presence of feedback delay (arising from communications). Due to the time delay, what we capture in real time is the lagged or delayed traffic information. Control based on delayed information leads to excessive degradation of network performance. Thus, in practice, its impact cannot be

ignored and must be taken into consideration and compensated for. However, in [1, 2] no attempt was made to deal with this problem and solve the impact of the communication delay.

Traffic prediction methods have been widely used in network management [4, 6, 7, 8, 9, 12, 13, 15]. By use of prediction techniques, that is, forecasting the future behavior of the traffic, one can effectively prevent traffic jams, traffic congestion, and network crashes. In [4], a method combining the single-step-ahead (SSP) and multi-step-ahead (MSP) predictors was developed to forecast MPEG-4 video traffic trace, which employed a nonlinear estimation tool and offline training. To solve the admission control problem in wireless mobile networks, the authors of [7] predicted cell bandwidth requirements subject to the constant bit-rate traffic. In [15], the evolutionary genetic approach (EGA) was used to predict the short-term bandwidth demand in virtual paths (VPs), which enabled intelligent controllers to improve transmission efficiency. Inspired by these ideas, we have applied prediction techniques to solve the problem encountered in [1, 2].

For this purpose, we propose a real-time feedback control mechanism based on the predicted state and traffic. The traffic and state information are predicted for different values of prediction times based on their past history (the traffic history measured online). An accurate prediction for the future traffic and state (short-term prediction) is able to provide better control compensating for time delay. Thus the impact of time delay can be minimized and the system performance improved.

In this paper, we developed an online predictor based on the principle of the least mean square error (LMSE), which is one of the simplest methods. It was noted in [6] that LMSE can achieve better accuracy compared to those complex long-memory predictors [9, 13] for online measurements. Without the requirement of complex computation, it can be implemented at a high speed. As a result of traffic prediction, the system performance degradation due to delay is reduced by use of proper control actions. According to our results, it is possible to optimize the system performance and minimize the cost function by implementing the new method.

The paper is organized as follows. Section 2 briefly describes TB-based system model in current practice. In Section 3, a cost functional is defined for network performance evaluation. Section 4 presents different control strategies (feedback with time delay and without delay). In Section 5, we propose a real-time feedback control based on the predicted state and traffic. An LMSE predictor is introduced here. Traffic used for simulation is illustrated in Section 6. After choosing different predictor parameters, we seek to optimize the network performance and reduce the system losses. Simulation results will be discussed in Section 7, where we present results showing improvements on system performance using predicted state feedback control. In Section 8, conclusion and suggestions for future work are discussed.

2. System model with control

In order to understand and solve the performance-related problems in computer communication network, it is critical to build a dynamic model of the information flow through the system. Further, the basic statistical properties of measured trace data must be known.
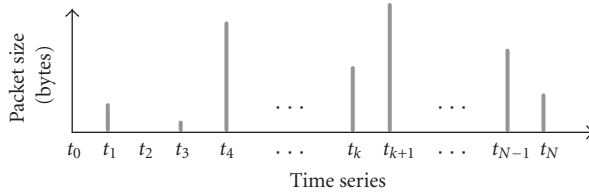
Figure 2.1. A general model.

Over the last two decades, a number of traffic models have been proposed and studied in the traffic management area. Traditional characterization of the Internet traffic is based on the Poisson process (which exhibits short-range dependence), Bernoulli process, or more generally doubly stochastic Poisson process (DSPP).

A recent study [11] shows that network traffic has self-similarity characteristics and long-range dependence. Self-similarity means that a certain property of traffic behavior is preserved over space and/or time scales, and long-range dependence is said to exhibit long-term correlations which decay at rates slower than exponential ones. On the other hand, the correlation functions of traditional traffic models decay exponentially or faster.

In this paper, a general model is constructed to simulate the incoming traffic illustrated in Figure 2.1, which is similar to those in [1, 2]. A discrete time series represents a traffic $\{V(t_k),\ k = 0, 1, 2, \ldots, K\}$ generated by the users' applications, where $V(t_k)$ denotes the packet size (measured in bytes) arriving during the non-overlapping time intervals $[t_k, t_{k+1})$. This can be modeled as a marked point process where the event times are the points and the sizes of the packets are the marks. For simplicity we assume that the time intervals are of equal length.

The following notations are used throughout the paper:

(1) $\{x \wedge y\} \equiv \text{Min}\{x, y\}$, $\{x \vee y\} \equiv \text{Max}\{x, y\}$, for $x, y \in \mathbb{R}$;

(2) $\{x \wedge y\} \equiv \{x_i \wedge y_i,\ i = 1, 2, \ldots, N\}$, $\{x \vee y\} = \{x_i \vee y_i,\ i = 1, 2, \ldots, N\}$, for $x, y \in \mathbb{R}^n$;

(3)

$$I(S) = \begin{cases} 1 & \text{if the statement } S \text{ is true,} \\ 0 & \text{otherwise.} \end{cases} \tag{2.1}$$

2.1. Basic model for a token bucket. A TB is described by a pair of parameters (token generation rate u and bucket capacity T). TBs are usually implemented to shape or police traffic between the host and the network or between the routers (see [14, 16, 17]; see also   http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/12cgcr/qos_c/qcpart4/qcpolts.htm). Here TB is used as a traffic policer. Figure 2.2 illustrates a TB model used in our previous work. The incoming tokens keep accumulating until they reach the capacity of the bucket. The number of tokens offered during the kth time interval is denoted by $u(t_k)$ and the number of tokens (status) in TB is $\rho(t_k)$. If the incoming tokens exceed the capacity, tokens in excess of the capacity will be dropped. Thus the acceptable tokens can be represented by $u(t_k) \wedge [T - \rho(t_k)]$.
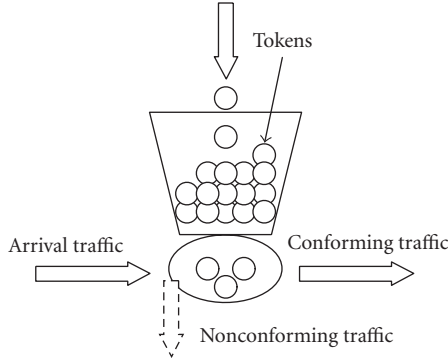
Figure 2.2. A token bucket model.

If the packet size of the arriving traffic $V(t_k)$ is less than the number of tokens available, the traffic is marked as conforming (traffic) and is immediately passed on to the network for queuing up in the multiplexor. At the same time, a number of tokens equal to the size of the packet is taken out of the token pool. Thus the state of TB at any time is determined by the algebraic sum of three terms: tokens left over from the previous time interval, new tokens added, and tokens consumed during the current time interval. As a result, the dynamics governing the status of the TB is given by the following expression:

$$\rho(t_{k+1}) = \rho(t_k) + \{u(t_k) \wedge [T - \rho(t_k)]\} \\ - V(t_k)I\{V(t_k) \leq [\rho(t_k) + u(t_k) \wedge [T - \rho(t_k)]]\}. \tag{2.2}$$

Note that the last term in the above equation gives the conforming traffic denoted by

$$g(t_k) = V(t_k)I\{V(t_k) \leq [\rho(t_k) + u(t_k) \wedge [T - \rho(t_k)]]\}, \tag{2.3}$$

and hence the nonconforming traffic is given by

$$r(t_k) = V(t_k) - g(t_k). \tag{2.4}$$

2.2. Complete system model. To simulate a network, we construct a mathematical model comprised of N individual users (traffic streams), served by N corresponding TBs, all of which are coupled to a multiplexor connected to an outgoing link having (bandwidth) capacity C. This is illustrated in Figure 2.3 [1, 2].

Each TB implements its algorithm to police the arriving packet. The nonconforming traffic streams are dropped while all the conforming traffic are multiplexed and queued up for entering the multiplexor. As a matter of fact, not all conforming traffic from TBs will be accepted because of the size limitation of the multiplexor (buffer size Q) and the link capacity (speed) of the accessing node. If the sum of these traffics exceeds the multiplexor size, some part of the conforming traffic may be dropped. The discarded traffic is defined as the traffic loss at the multiplexor $L(t_k)$.
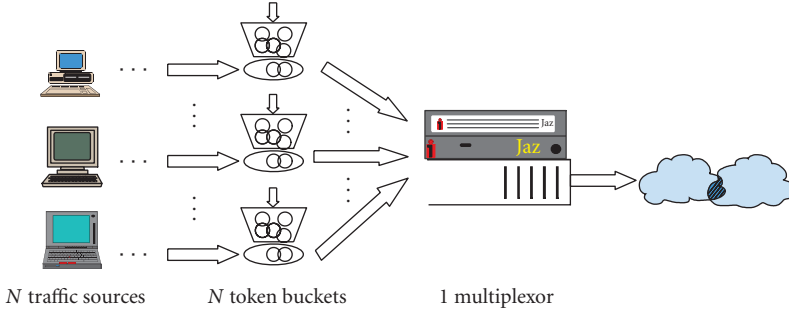
Figure 2.3. Dynamic system model.

The dynamics for a single TB is given by (2.2). For a system consisting of N TBs, served by a single multiplexor, the associated variables are given by multidimensional vectors from $\mathbb{R}^N$ which are listed as follows:

$$
\begin{aligned}
T &\equiv (T_1, T_2, \ldots, T_N)', \\
\rho(t_k) &\equiv (\rho_1(t_k), \rho_2(t_k), \ldots, \rho_N(t_k))', \\
V(t_k) &\equiv (V_1(t_k), V_2(t_k), \ldots, V_N(t_k))', \\
u(t_k) &\equiv (u_1(t_k), u_2(t_k), \ldots, u_N(t_k))', \\
g(t_k) &\equiv (g_1(t_k), g_2(t_k), \ldots, g_N(t_k))', \\
r(t_k) &\equiv (r_1(t_k), r_2(t_k), \ldots, r_N(t_k))'.
\end{aligned}
\tag{2.5}
$$

Since the sources are independent, the dynamics of the system of TBs is now given by a system of identical equations

$$
\begin{aligned}
\rho_i(t_{k+1}) = \rho_i(t_k) &+ \{u_i(t_k) \wedge [T_i - \rho_i(t_k)]\} \\
&- V_i(t_k) I\{V_i(t_k) \le [\rho_i(t_k) + u_i(t_k) \wedge [T_i - \rho_i(t_k)]]\},
\end{aligned}
\tag{2.6}
$$

where $i = 1, 2, \ldots, N$.

The conforming traffic is accumulated in the multiplexor. The state of the multiplexor, denoted by $q(t_k)$, is given by the volume of traffic waiting in the buffer for service during the kth time interval. Based on the sum of all the conforming traffic and the available buffer space, the multiplexor can only accept the traffic given by

$$
\sum_{i=1}^{N} g_i(t_k) \wedge [Q - ([q(t_k) - C * \tau] \vee 0)].
\tag{2.7}
$$

Thus the state dynamics of the multiplexor can be written as follows:

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

The first term on the right-hand side represents packets leftover from the previous $(k - 1)$th time interval, the second term represents packets accepted by the multiplexor during the kth time interval. This is a scalar equation governing the dynamics of the queue in the multiplexor. Thus the dynamics of the whole access control protocol is governed by the system of $(N + 1)$ difference equations (2.6) and (2.8). This leads to the following nonlinear state-space model:

$$X(t_{k+1}) \equiv F(t_k, X(t_k), u(t_k), V(t_k)), \tag{2.9}$$

where $X \equiv (\rho, q)'$ denotes the state vector, u is the control vector, V represents the input traffic vector, and F is the state transition operator determined by the expressions on the right-hand sides of equations (2.6) and (2.8).

3. Objective function and performance measures

3.1. Objective function.   In general, the traffic loss at the TBs during the kth time interval is given by

$$L_T(t_k) \equiv \sum_{i=1}^{N} r_i(t_k) \equiv \sum_{i=1}^{N} [V_i(t_k) - g_i(t_k)], \tag{3.1}$$

while the multiplexor loss during the same time interval is given by

$$L_M(t_k) \equiv \sum_{i=1}^{N} g_i(t_k) - \sum_{i=1}^{N} g_i(t_k) \wedge [Q - ([q(t_k) - C * \tau] \vee 0)]. \tag{3.2}$$

In addition to these losses, it is also important to include a penalty for the waiting time or time spent on the queue before being served. For simplicity we assume that it is linearly proportional to queue length.

Adding all these, we obtain the cost functional. Since the incoming source (or user demand) is a random process, we must compute the average cost as being the expected value of the sum of all the costs described above. This is given by

$$J(u) \equiv E\left\{ \sum_{k=0}^{K} \alpha(t_k) L_M(t_k) + \sum_{k=0}^{K} \beta(t_k) L_T(t_k) + \sum_{k=0}^{K} \gamma(t_k) q(t_k) \right\}, \tag{3.3}$$

where u is the control law which determines the state of the system and hence the individual losses and finally the total cost. The functions α, β, γ represent the relative weights or relative importance given to each of the three distinct losses.

Since the exact stochastic characterization of our traffic is not available or is unknown, the Monte Carlo method is employed to compute the expected values of the performance measures. For applying the Monte Carlo [5, 10] technique, we let N_s denote the number of samples used and let $\Omega \equiv \{w_j, j = 1, 2, 3, \ldots, N_s\}$ denote the elementary events or sample

paths with finite cardinality N_s. The objective functional (3.3) is then given by

$$J(u) \cong \frac{1}{N_s} \sum_{j=1}^{N_s} \left\{ \sum_{k=0}^{K} \alpha(t_k) L_M(t_k, \omega_j) + \sum_{k=0}^{K} \beta(t_k) L_T(t_k, \omega_j) + \sum_{k=0}^{K} \gamma(t_k) q(t_k, \omega_j) \right\}. \quad (3.4)$$

The first term of the expression gives the average weighted loss at the multiplexor, the second gives that for TBs, and the last one is the penalty assigned to the average waiting time in the multiplexor.

3.2. Performance measure. In addition to the objective functional, we also use the network utilization to evaluate the overall system performance. The total amount of data successfully transferred to the network is marked as the actual transferred traffic, while the maximum transfer capacity is given by the link capacity. Utilization is then defined as the ratio of the two, measured in percentage by

$$\eta = \frac{\sum_{k=0}^{K} \sum_{i=1}^{N} V_i(t_k) - \left[\sum_{k=0}^{K} L_M(t_k) + \sum_{k=0}^{K} L_T(t_k) \right]}{C(t_K - t_0)} \times 100. \quad (3.5)$$

4. Feedback controls

4.1. General feedback control law. Unlike open-loop controls which do not consider the status of the network resources shared by competing users, the feedback control mechanism (closed-loop control) exercises controls based on available information on the current traffic and system state [8, 12, 18]. By analyzing the current status of the system, the feedback controller can make proper decisions on resource allocation to achieve superior performance. In general, a feedback control law without communication (plus service) delay is of the form

$$u(t_k) \equiv G(V(t_k), \rho(t_k), q(t_k)), \quad k = 0, 1, 2, \ldots, K - 1, \quad (4.1)$$

where G is a suitable function (to be determined) that maps the available data into a control action. The data may be given by the collection $\{ V \in \mathbb{R}^N, \rho \in \mathbb{R}^N, q \in \mathbb{R}^1 \}$ which denotes the input traffic, state of TBs, and the multiplexor, respectively.

In the presence of communication and service delay, if the same control law G is used, the actual control action would be different. This is given by the following expression, which is nothing but the delayed version u^d of the control u without delay:

$$u^d(t_k) \equiv G(V(t_{k-m_1}), \rho(t_{k-m_2}), q(t_{k-m_3})), \quad k = 0, 1, 2, \ldots, K - 1, \quad (4.2)$$

where m_i, $i = 1, 2, 3$, denotes the number of time slots by which information reaching the controller is delayed. It is clear from this expression that the current control is decided on the basis of past status information and therefore cannot be expected to be as effective
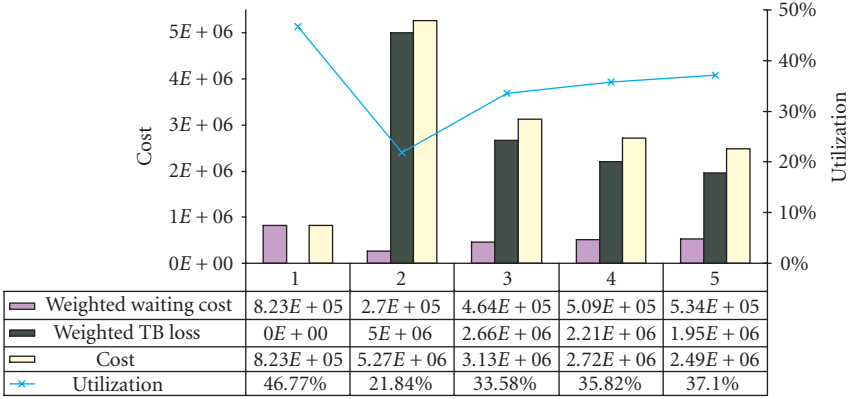
	1	2	3	4	5
Weighted waiting cost	$8.23E + 05$	$2.7E + 05$	$4.64E + 05$	$5.09E + 05$	$5.34E + 05$
Weighted TB loss	$0E + 00$	$5E + 06$	$2.66E + 06$	$2.21E + 06$	$1.95E + 06$
Cost	$8.23E + 05$	$5.27E + 06$	$3.13E + 06$	$2.72E + 06$	$2.49E + 06$
Utilization	46.77%	21.84%	33.58%	35.82%	37.1%

Figure 4.1. System performance (costs and utilization, Cases 1–5)—Bellcore traffic.

	1	2	3	4	5
Weighted waiting cost	$5.14E + 05$	$9.53E + 02$	$4E + 03$	$3.1E + 04$	$3.78E + 04$
Weighted losses at TB	$0E + 00$	$3.44E + 06$	$2.99E + 06$	$1.04E + 06$	$6.36E + 05$
Cost	$5.14E + 05$	$3.44E + 06$	$2.99E + 06$	$1.07E + 06$	$6.74E + 05$
Utilization	30.08%	12.88%	15.14%	24.9%	26.9%
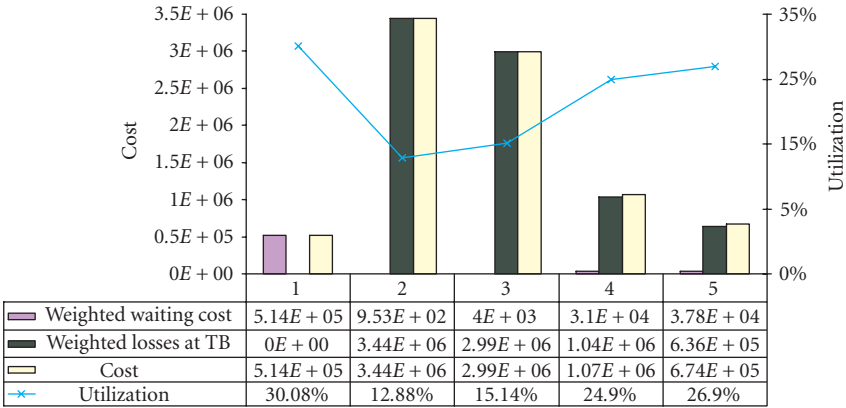
Figure 4.2. System performance (costs and utilization, Cases 1–5)—DSPP1.

as the control without delay. In fact, system performance is significantly degraded if this control is used; see Case 2 in Figures 4.1, 4.2, and 4.3.

4.2. A simple control law in the absence of time delay. To avoid cell losses at the multiplexor and monopoly by any users, we include a permission parameter and compute the maximum permissible allocation as follows. The maximum possible allocation for the ith user is given by

$$\Theta_i(t_k) \equiv \left\{ \frac{V_i(t_k)}{\sum_{i=1}^{N} V_i(t_k)} \wedge \frac{e_i}{N} \right\} [Q - [(q(t_k) - C * \tau) \vee 0]], \qquad (4.3)$$
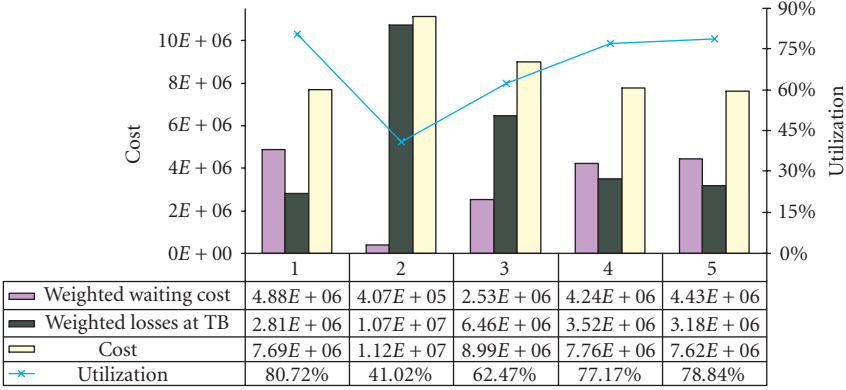
	1	2	3	4	5
Weighted waiting cost	$4.88E + 06$	$4.07E + 05$	$2.53E + 06$	$4.24E + 06$	$4.43E + 06$
Weighted losses at TB	$2.81E + 06$	$1.07E + 07$	$6.46E + 06$	$3.52E + 06$	$3.18E + 06$
Cost	$7.69E + 06$	$1.12E + 07$	$8.99E + 06$	$7.76E + 06$	$7.62E + 06$
Utilization	80.72%	41.02%	62.47%	77.17%	78.84%

Figure 4.3. System performance (costs and utilization, Cases 1–5)—DSPP2.

where each user's share is determined by the smaller of the fraction of his demand and the fraction of the resource allocation permitted times the available multiplexor space. Comparing with the real traffic $V_i(t_k)$, the true allocation for the ith user during the kth time interval is given by

$$A_i(t_k) = \{\Theta_i(t_k) \wedge V_i(t_k)\}. \tag{4.4}$$

If $1 \leq e_i < N$ (for one special i), the network provider withdraws the ith user's full permission and allows other users to utilize the supplementary bandwidth. Thus by an appropriate choice of this parameter, the network provider can control monopoly and even assign priorities.

A simple feedback control law that was suggested in our previous papers [1] was found to be very effective. This is given by the following expression:

$$u_i(t_k) = G_i(V(t_k), \rho(t_k), q(t_k)) \equiv \{A_i(t_k) - \rho_i(t_k)\} I\{A_i(t_k) \geq \rho_i(t_k), A_i(t_k) = V_i(t_k)\}, \tag{4.5}$$

where A_i is given by the expressions in (4.4).

4.3. Feedback control in the presence of time delay. Considering all the delays (labeled as communication delay) in real time, the actual control action is given by

$$u_i^d(t_k) \equiv \begin{cases} G_i(V(t_k), \rho(t_k), q(t_k)) & \text{for } 0 \leq m_j < 1, \ j = 1, 2, 3, \ i = 1, 2, \ldots, N, \\ G_i(V(t_{k-m_1}), \rho(t_{k-m_2}), q(t_{k-m_3})) & \text{for } m_j \geq 0, \ j = 1, 2, 3, \ i = 1, 2, \ldots, N, \end{cases} \tag{4.6}$$

where delay less than one time slot, $0 \leq m < 1$, is simply considered as no delay.

5. Prediction scheme and predictive feedback control

In the presence of feedback delay, it is difficult to make a proper decision to effectively allocate network resources or prevent traffic congestion on the basis of delayed information, which results in degradation of system performance [2]. To reduce the impact of delay, traffic prediction, that is, accurately providing the statistical characteristics of traffic, becomes one of the key issues in network control engineering.

5.1. Prediction scheme. Recently a number of traffic predictors have been proposed [4, 6, 7, 8, 9, 12, 13, 15]. However, those predictors either realize the offline control or focus only on constant bit-rate traffic. Here we develop a real-time predictor which is model-free and independent of the statistical properties of the traffic. The algorithm proposed to improve the system performance is called predictive feedback control (PFC). Monitoring the past history of the system (the traffic and the system state), we are able to make short time prediction of the future traffic and system state. By applying the predicted information to the feedback control law in place of delayed information, excessive packet losses and waste of network resources can be prevented.

Our aim is to find a model-free predictor which does not require excessive computation and can be implemented online. The experiments carried out by Ghaderi, Capka and Boutaba [6, 7] have demonstrated that the LMSE predictor can achieve satisfactory performance. The LMSE predictor does not require much computation and is simple to implement. This adaptive predictor is independent of the traffic model (statistics) and is useful for both the short-range- and long-range-dependent processes. In this study, we apply this technique for traffic prediction.

Let $V(t_k)$, $k = 0, 1, 2, \ldots, K$, denote any history of the traffic process measured in terms of packets. Our objective is to predict the process by some specified units of time ahead, denoted by T_d, based on a segment (window) of the past history of the process denoted by W_s. This is mathematically described by the following expression:

$$\hat{V}(t_k) = \sum_{r=T_d}^{W_s+T_d} \alpha_r V(t_{k-r}), \quad T_d \geq 0, \ W_s \geq 0, \ T_d + W_s < k, \tag{5.1}$$

where T_d denotes the number of time slots (ahead of the current time), and W_s denotes the length of observation, which is also called observation window width. This is the number of past samples used to predict the future traffic. Here the vector α denotes the weight or importance given to past samples observed.

The major objective of traffic prediction is to minimize the mean square difference (error) between the predicted traffic and the actual traffic measured. The choice of the weight vector α determines the level of prediction error. In order to determine the best weight vector, we define the estimation error as follows:

$$J(\alpha) \equiv E\|V(t_k) - \hat{V}(t_k)\|^2. \tag{5.2}$$

Then

$$J(\alpha) = E\left\|V(t_k) - \sum_{r=T_d}^{T_d+W_s} \alpha_r V(t_{k-r})\right\|^2$$

$$= E\|V(t_k)\|^2 - 2\sum_{r=T_d}^{T_d+W_s} \alpha_r E(V(t_{k-r}), V(t_k)) \qquad (5.3)$$

$$+ \sum_{l=T_d}^{T_d+W_s}\sum_{r=T_d}^{T_d+W_s} \alpha_r \alpha_l E(V(t_{k-r}), V(t_{k-l})),$$

where $(x, y) = \sum x_i y_i$ denotes the standard inner product in $\mathbb{R}^N$. If X, Y are two N-dimensional random vectors having finite second moments, then $E(X, Y)$ denotes the expected value of their inner product. Since there are no constraints on $\alpha \in \mathbb{R}^d$, $(d = W_s)$, differentiating J with respect to α and setting it equal to zero, we obtain

$$\sum_{l=T_d}^{T_d+W_s} \alpha_l E(V(t_{k-l}, V(t_{k-r}))) = E(V(t_{k-r}), V(t_k)). \qquad (5.4)$$

Let A and b denote the matrix and vector as defined below:

$$A \equiv \begin{pmatrix} E(V(t_{k-T_d}), \alpha) & \cdots & E(V(t_{k-T_d}), \beta) & \cdots & E(V(t_{k-T_d}), \gamma) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ E(V(t_{k-(T_d+l)}), \alpha) & \cdots & E(V(t_{k-(T_d+l)}), \beta) & \cdots & E(V(t_{k-(T_d+l)}), \gamma) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ E(V(t_{k-(T_d+W_s)}), \alpha) & \cdots & E(V(t_{k-(T_d+d)}), \beta) & \cdots & E(V(t_{k-(T_d+d)}), \gamma) \end{pmatrix},$$

$$b \equiv \begin{pmatrix} E(\alpha, V(t_k)) \\ \vdots \\ E(\beta, V(t_k)) \\ \vdots \\ E(\gamma, V(t_k)) \end{pmatrix},$$

$$(5.5)$$

where

$$\alpha = V(t_{k-T_d}),$$
$$\beta = V(t_{k-(T_d+l)}), \qquad (5.6)$$
$$\gamma = V(t_{k-(T_d+W_s)}).$$

Then (5.4) can be written compactly as $A\alpha = b$, and, if A is nonsingular, the solution is given by $\alpha = A^{-1}b$. Hence the estimated traffic can be computed by the following expression:

$$\widehat{V}(t_k) = \sum_{r=T_d}^{T_d+W_s} (A^{-1}b)_r V(t_{k-r}), \quad T_d \geq 0, \ W_s \geq 0, \ T_d + W_s < k, \tag{5.7}$$

where $(A^{-1}b)_r$ denotes the rth component of the vector $A^{-1}b$.

5.2. Predictive feedback control. After we have the best mean square estimate of the future traffic T_d steps ahead of current time, based on observation of past history of length W_s, the control law can be adapted in advance corresponding to the predicted states of the future traffic. Assuming that the state (ρ, q) is monitored in real time, to minimize the impact of delay we choose the new control law as

$$\widehat{u}(t_k) \equiv G(\rho(t_k), q(t_k), \widehat{V}(t_k)), \quad 0 < T_d < k - 1. \tag{5.8}$$

6. Basic data used for numerical simulations

Implementation of our system approach to computer communication network is described in this section. Fractional Brownian motion (FBM) is realized by use of Matlab. Using the basic idea of the algorithm introduced in [3], we have developed the traffic models in C++ programming language. Finally, the entire system including feedback control is implemented using Matlab.

6.1. System parameters and configurations. In order to compare the results with previous work [2], we use an identical system to the one in [2]. The system consists of three users (sources) regulated by three TBs, which are served by a multiplexor connected to the outgoing link. Three individual traffic traces, each of 4-second duration, are fed into the three TBs directing conformed traffic to the multiplexor. The experiments also follow the previous assumptions of [2] as follows: (a) the three independent traffic traces share the same statistical characteristics, (b) packet size is measured in terms of bytes and one token is consumed for each byte, (c) the traffic traces last 4 seconds.

While calculating the objective function, we have to consider the weights assigned to the costs associated with the losses at TB, the multiplexor, and waiting time. They are chosen as follows: $\alpha(t_k) = 10$, $\beta(t_k) = 5$, and $\gamma(t_k) = 0.3$ for all t_k.

The system parameters are listed in Table 6.1.

6.2. Specification of traffic traces. Two major types of traffic are used in our simulation: (1) Bellcore trace and (2) DSPP generated by FBM as described below.

The traffic traces used here are partitioned into small time slots, which individually contain the information for 4 seconds and are described in bytes. These 4-second data are used as the traffic from the network users.

Table 6.1. System configuration and parameters.

Parameters	Bellcore traffic	DSPP-1/2 Traffic
T_i, $i = 1, 2, 3$	15180 bytes	15180 bytes
C	8 Mbps	8 Mbps
Q	45540 bytes	45540 bytes
τ	0.005 second	0.005 second
K	800	800
M	100	1000
e_i, $i = 1, 2, 3$	3	3
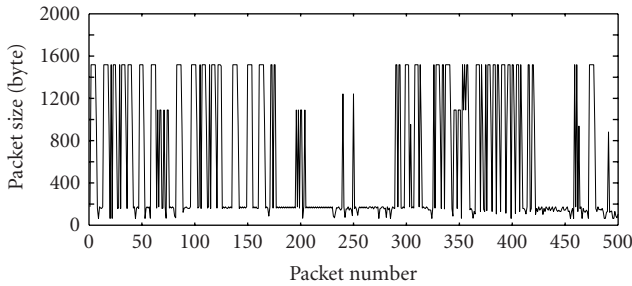
Figure 6.1. Bellcore traffic.

6.2.1. Bellcore traffic trace. Bellcore traffic trace (see http://ita.ee.lbl.gov/html/contrib/ BC.html) was captured on an Ethernet at the Bellcore Morristown Research and Engineering facility, which lasts 3142.82 seconds and contains one million packets. Figure 6.1 shows part of the trace which is divided into 4-second segments.

6.2.2. DSPP traffic traces. We also construct a self-similar traffic model using a nonnegative function of FBM as the input rate of a Poisson process. This yields a DSPP which exhibits self-similarity and long-range dependence properties. FBM denoted by $B_H(t)$ is a self-similar process itself. It depends on the parameter H called the Hurst parameter. In general, its values lie in the interval $(0, 1)$, and it represents a measure of self-similarity in the traffic, and "burstiness." We construct the FBM through the following integral transformation of the standard Brownian motion $\{B(t),\ t \geq 0\}$:

$$B_H(t) = \int_0^t K_H(t - s)dB(s), \quad t \geq 0, \tag{6.1}$$

where $K_H(t)$ is given by

$$K_H(t) = C_H t^{(H-1/2)}, \quad \frac{1}{2} < H < 1, \tag{6.2}$$
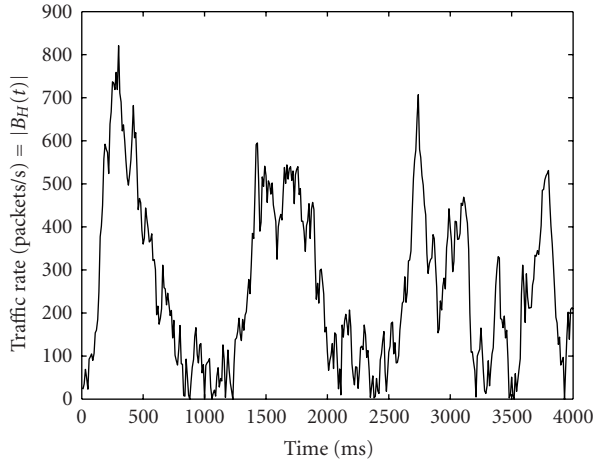
Figure 6.2. DSPP1 intensity ($H = 0.6$).
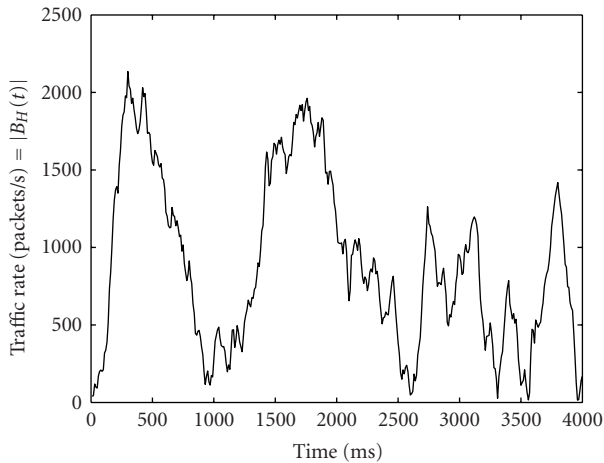
Figure 6.3. DSPP2 intensity ($H = 0.8$).

and C_H is any constant. Then we generate a Poisson process with the intensity function given by the absolute value of the FBM:

$$\lambda(t) \equiv \left| B_H(t) \right|, \quad t \geq 0. \tag{6.3}$$

Figure 6.2 shows $\lambda(t)$ which is generated by the absolute value of the FMB with the Hurst parameter $H = 0.6$ and $C_H = 15$. Another set of $\lambda(t)$ with $H = 0.8$ is plotted in Figure 6.3. They are chosen as the intensity (rate) of a Poisson process. The DSPP traces with the intensities as defined above are generated, and the samples of the traces are plotted in Figures 6.4 and 6.5, respectively.
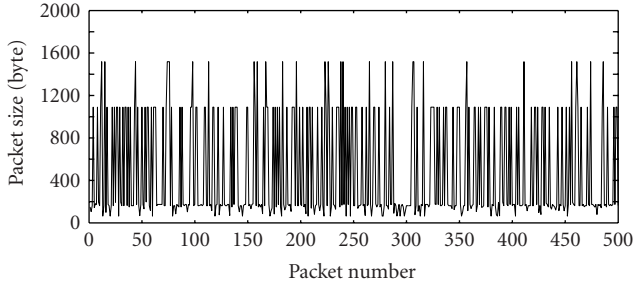
Figure 6.4. DSPP traffic trace ($H = 0.6$).
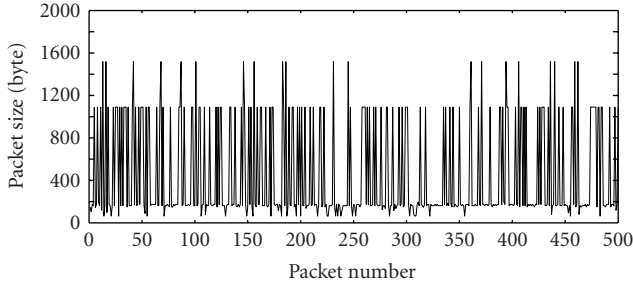
Figure 6.5. DSPP traffic trace ($H = 0.8$).

7. Simulation results and numerical analysis

In this section, we show the prediction performance of the LMSE predictor and the improvement of the overall performance of the system by use of the predictive feedback control.

7.1. Performance of LMSE predictor. To illustrate the dependence of estimation error on the observation window size W_s and the prediction time T_d, we use the Monte Carlo technique to compute the expected value of the (estimation) error given by

$$E(T_d, W_s) = \sqrt{\left(E||\widehat{V}(t_k) - V(t_k)||\right)^2} = \sqrt{\left(\frac{1}{N_s}\sum_{j=1}^{N_s}\left(\widehat{V}(t_k, w_j) - V(t_k, w_j)\right)\right)^2}, \quad (7.1)$$

where w_j denotes the jth sample path and N_s denotes the number of sample paths used.

The inverse of the signal-to-noise ratio (E_{NSR}) is used as another measure to evaluate the quality of prediction results:

$$E_{\text{NSR}} \equiv \left(\text{SNR}\right)^{-1} = \frac{\sum e^2}{\sum \left(V(t_k)\right)^2} = \frac{\left((1/N_s)\sum_{j=1}^{N_s}\left(\widehat{V}(t_k, w_j) - V(t_k, w_j)\right)\right)^2}{\sum \left(V(t_k)\right)^2}. \quad (7.2)$$

Numerical results of prediction based on the LMSE method are presented in the following section for both the Bellcore traffic and DSPP traffic.
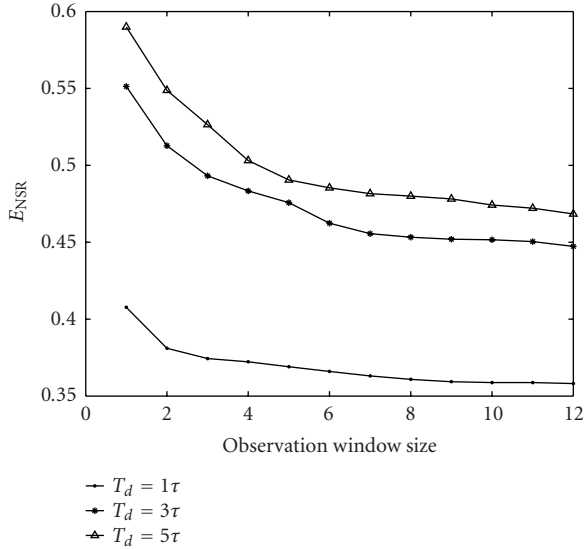
Figure 7.1. Prediction error versus W_s—Bellcore.

7.1.1. Bellcore traffic trace.   Here the performance of the LMSE predictor is evaluated for different observation window sizes and prediction times. Similar studies were reported in [6, 7] for a fixed window size and a fixed prediction time. In contrast, we evaluate the prediction performance for different window sizes and prediction times. Since prediction times required vary with the communication delay causing delayed control actions, our results corresponding to required prediction times can be used to evaluate the performance of the system described by (2.6), (2.8), and (5.8).

Dependence of E_{NSR} on observation window size.   **Figure** 7.1 shows the plots of E_{NSR} as a function of observation window size for fixed prediction times (as parameters). It is clear that for a fixed prediction time, the error decreases with the increase of (observation) window size. This is expected. On the other hand, it is also clear from this figure that, for a fixed window size, prediction error increases with the increase of prediction time. Furthermore, for a fixed prediction time, as the window size increases, the prediction error tends to reach a lower limit possibly greater than zero. This means that by simply increasing the window size we cannot expect to improve the performance beyond a limit.

Dependence of E_{NSR} on prediction time.   In **Figure** 7.2, we plot E_{NSR} as a function of prediction time for fixed values of window size. From the plot, it is clearly seen that for any fixed window size, E_{NSR} increases with the increase of prediction time. This is also expected. Again for a fixed prediction time, as the window size increases, the prediction error decreases.

7.1.2. DSPP ($H = 0.6, 0.8$) traffic traces.   In this subsection, we present the prediction results of the DSPP traffic with the Hurst parameter equal to 0.6 and 0.8, respectively. The result are shown in Figures 7.3, 7.4, 7.5, and 7.6.
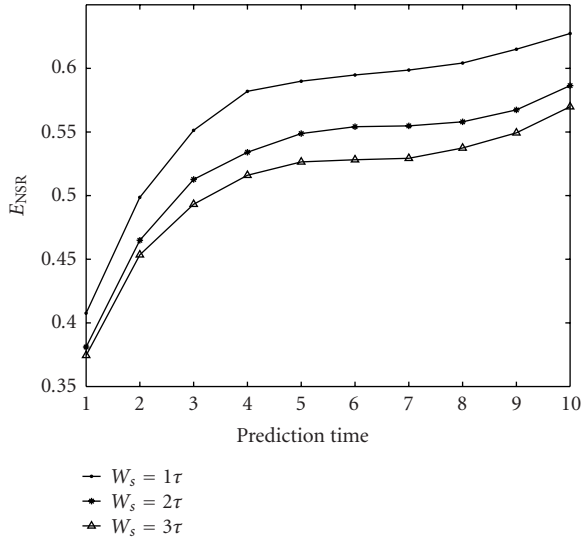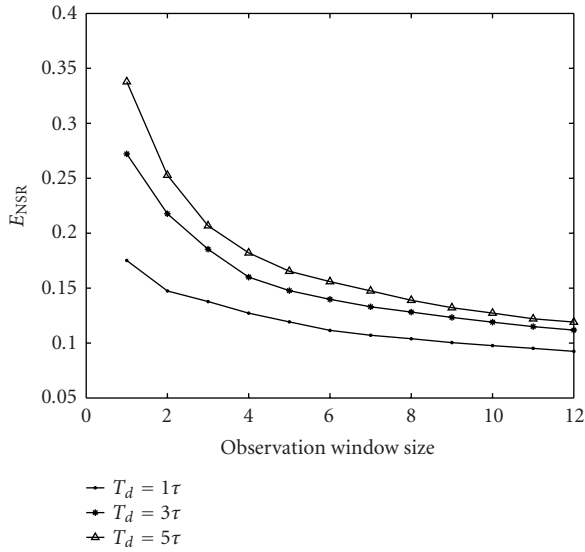
Figure 7.2. Prediction error versus T_d—Bellcore.

Dependence of E_{NSR} on observation window size. Again, it is clear from Figure 7.3, plotted for two different Hurst parameters, that the prediction error decreases when the observation window size increases for a fixed prediction time. This is similar to what was observed in the case of Bellcore traffic. We also find that with the increase of the Hurst parameter, E_{NSR} decreases. Figure 7.4 demonstrates the relationship between E_{NSR} and the Hurst parameter for fixed prediction times and three different window sizes. The result illustrates that the long-range dependence property exists in the traces. The larger the Hurst parameter, the stronger the correlation with the past information resulting in reduced prediction error with increasing observation window size.

Dependence of E_{NSR} on prediction time. Figure 7.5 offers another insight. For any fixed window size, E_{NSR} increases with the increase of prediction time and appears to reach a plateau. As expected, E_{NSR} is smaller for larger Hurst parameters. This is further illustrated in Figure 7.6.

7.2. Simulation results and numerical analysis of system performance. As described in the introduction and Section 4, communication delay in the network, resulting in delay in control actions, adversely affects the performance of the system [2]. By use of predictor controls, we demonstrate improved performance. Here simulation results are presented in Figures 4.1, 4.2, 4.3, 7.7, 7.8, 7.9, and 7.10 to illustrate the improvement of the overall performance of the system.

7.2.1. Dependence of system performance on control policies. In this subsection, we present performance results corresponding to predictive feedback control and compare them with those corresponding to feedback controls with or without delay. The results clearly

(a)
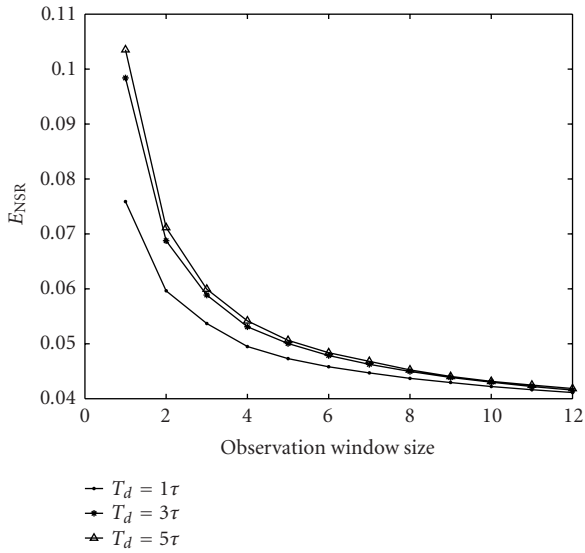
(b)

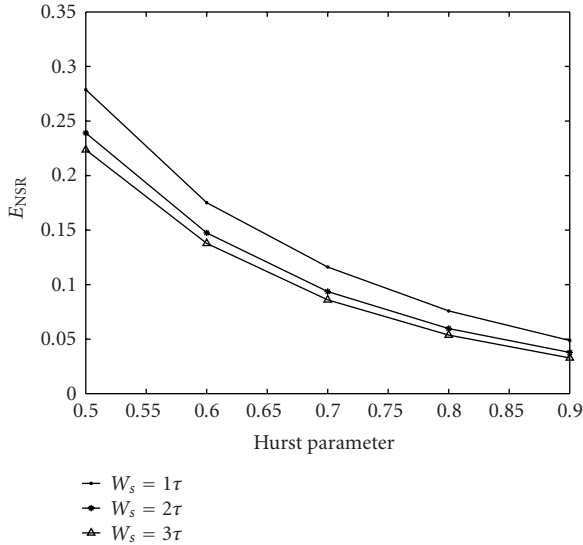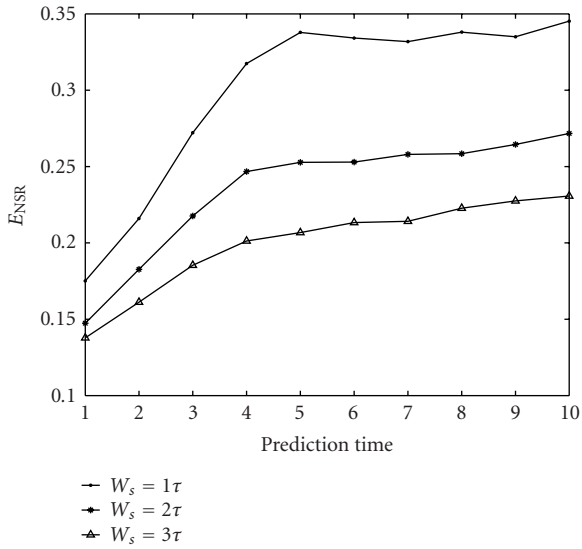Figure 7.3. Prediction error versus W_s—DSPP traffic. (a) $H = 0.6$, (b) $H = 0.8$.

Figure 7.4. Prediction error versus Hurst parameter—DSPP traffic (for $T_d = 1\tau$).

demonstrate that by use of the predictive feedback control law, one can achieve substantial improvement of system performance. We consider the following five cases to compare with the results reported in [2].

(1) Case 1: feedback control (without delay, $T_d = 0$).
(2) Case 2: feedback control (with delay, $T_d = 1$).
(3) Case 3: feedback control (predictive feedback control, $T_d = 1$, $W_s = 1$).
(4) Case 4: feedback control (predictive feedback control, $T_d = 1$, $W_s = 6$).
(5) Case 5: feedback control (predictive feedback control, $T_d = 1$, $W_s = 11$).

Bellcore traffic trace. Numerical results for the system performance are shown in Figure 4.1. Here we have listed the weighted TB losses, waiting-time losses, total cost, and utilization. It is clear from Figure 4.1 that feedback control without (communication) delay (Case 1) achieves the minimum cost and the highest utilization. The worst situation occurs in Case 2 with one unit of communication delay. Provided with the delayed (traffic) information, the controller cannot supply the required number of tokens to match the incoming traffic. This leads to significant packet losses at the TB resulting in degradation of performance and utilization. On the other hand, by use of predictive feedback control law (Cases 3–5), it is possible to reduce the performance degradation significantly. In all these cases we use predictive feedback control law with increasing window size. It is clear (Case 3) that use of this control law substantially improves the performance despite communication delay. This is further improved by use of larger window sizes as seen in Cases 4–5. System utilization given by the thin curve shows that utilization is highest in the absence of communication delay and lowest in its presence (Case 2) and then increases if predictive feedback control is used with increasing window size (Cases 3–5). This is explained further on.

(a)
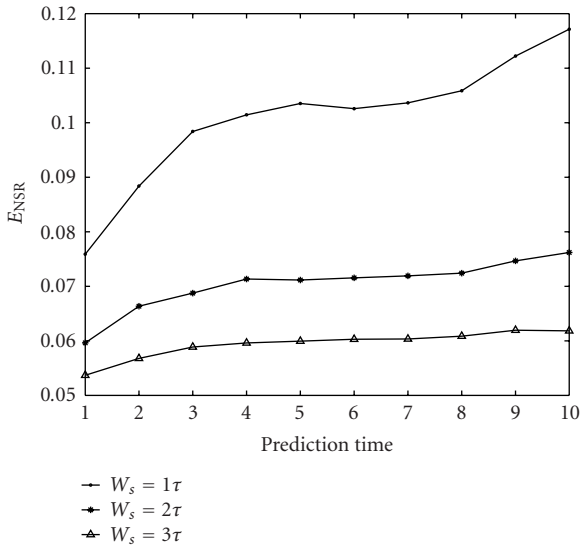
(b)

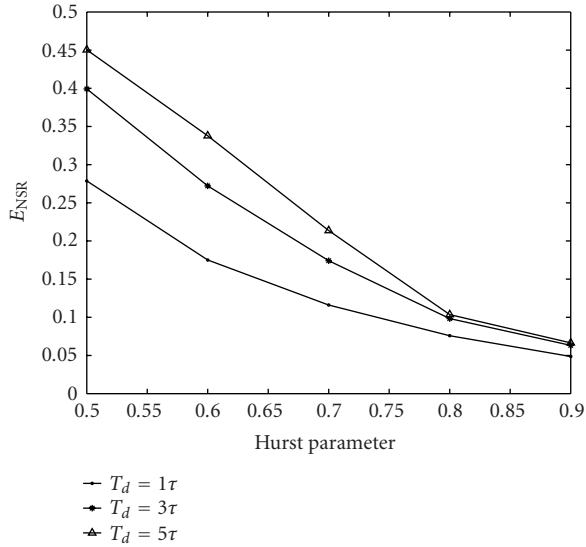Figure 7.5. Prediction error vsersus T_d—DSPP traffic. (a) $H = 0.6$, (b) $H = 0.8$.

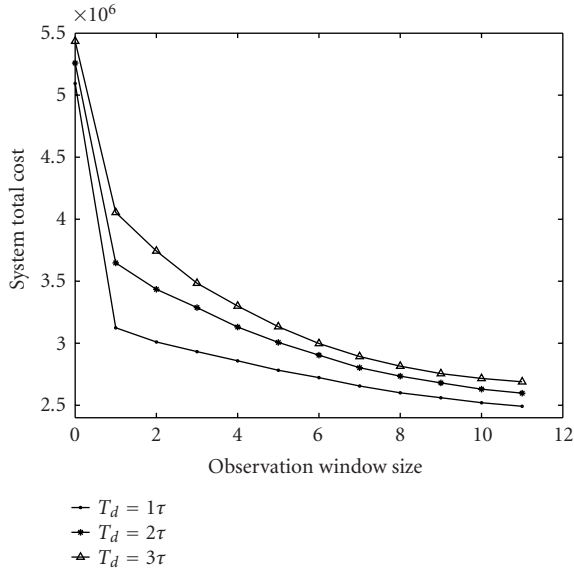Figure 7.6. Prediction error versus Hurst parameter—DSPP traffic (for $W_s = 1\tau$).

Figure 7.7. System cost versus W_s—Bellcore traffic.

DSPP (H = 0.6, 0.8) traffic traces. Here we present performance results corresponding to DSPP traffic using the same five cases as in Bellcore traffic. Performance results corresponding to the traffic with $H = 0.6$ and $H = 0.8$ are shown in Figures 4.2 and 4.3. The results have similar general patterns to those of Bellcore traffic shown in Figure 4.1.
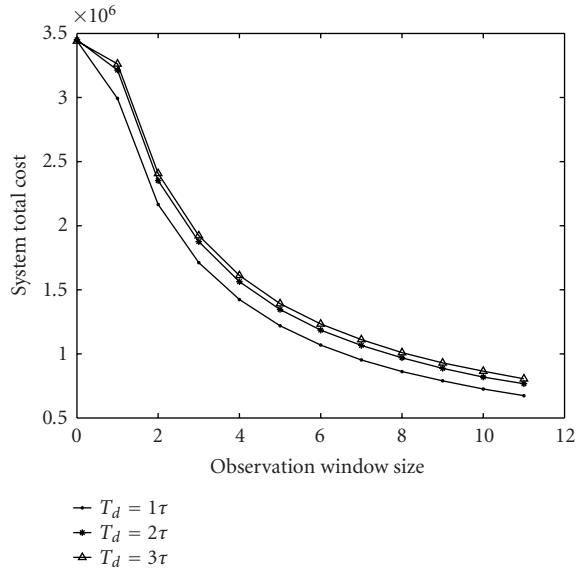
(a)
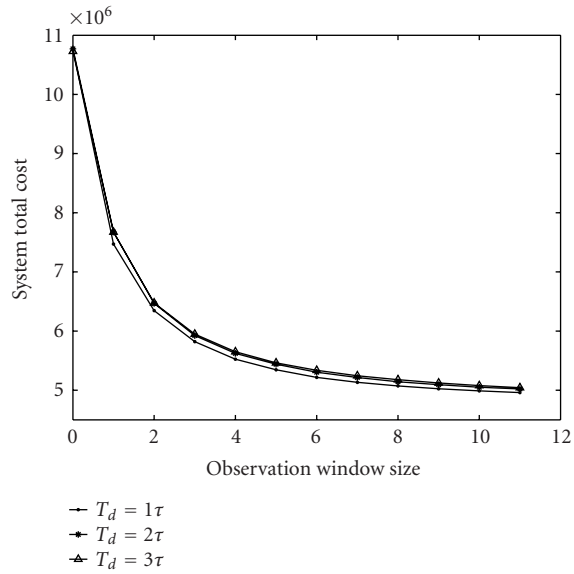
(b)

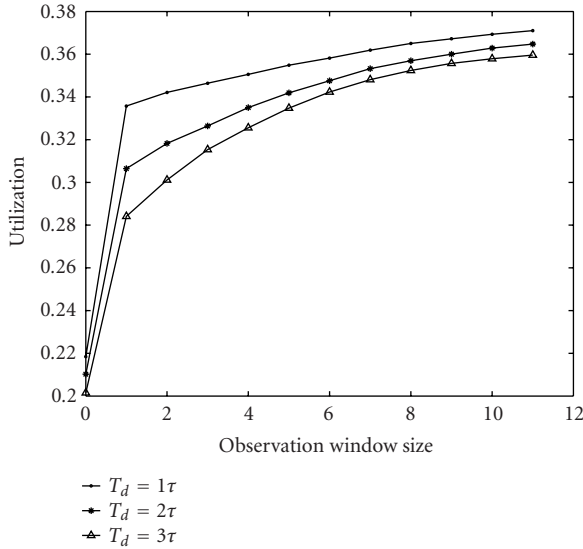Figure 7.8. System cost versus W_s—DSPP traffic. (a) $H = 0.6$, (b) $H = 0.8$.

Figure 7.9. Utilization versus W_s—Bellcore traffic.

Among the five cases, Case 1 maintains the lowest system cost and the highest utilization, and with predictive feedback control law, system performance gets better with increasing window sizes.

Similar results are plotted in Figures 4.2 and 4.3 corresponding to Hurst parameters $H = 0.6$ and $H = 0.8$, respectively. The time average of the expected traffic intensity $\lambda_H(t) \equiv |B_H(t)|$ is given by

$$m_H = \frac{1}{T} \int_0^T E\,|\,B_H(t)\,|\,dt = \frac{C_H}{H(2H+1)}\,T^{2H}. \tag{7.3}$$

For $C_H = 15, T = 4000$ (milliseconds) and $H = 0.6$, $m_H = 2.39 \times 10^5$ and for $H = 0.8$, $m_H = 4.8 \times 10^6$. Note that the waiting cost, losses at TBs, and the total cost shown in the first row of Figure 4.2 are lower than those in Figure 4.3. Also Figure 4.3, corresponding to $H = 0.8$, shows higher utilization. These differences are clearly due to difference in the volume of traffic in the two cases.

7.2.2. Dependence of cost on observation window size. The cost function is a measure of overall system performance. This is plotted as a function of window size for 3 different values of communication delay.

Bellcore traffic trace. Figure 7.7 shows the plots of total cost as a function of the observation window size (W_s) with fixed feedback delay (T_d). It is clear from this figure that system cost decreases with increasing (observation) window size for any given communication delay and it increases with increasing delay for any fixed window size.

DSPP ($H = 0.6, 0.8$) traffic traces. Again in Figures 7.8a and 7.8b, we plot the system cost as a function of the window size for two different values of Hurst parameters. It is clear
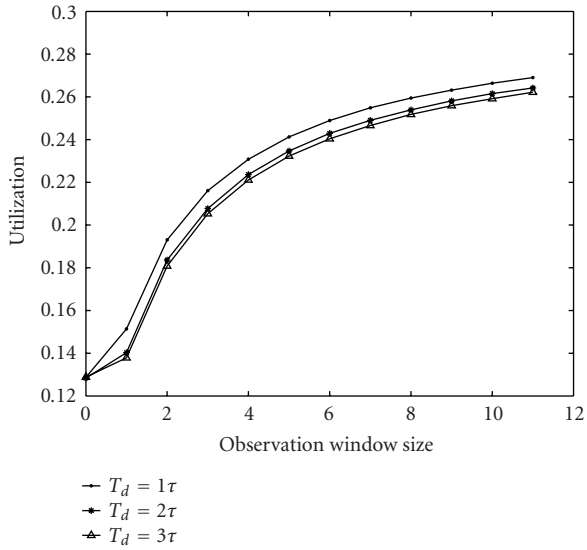
(a)
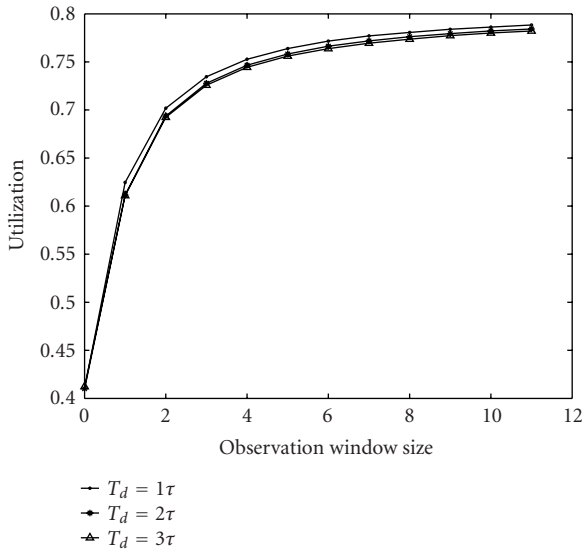
(b)

Figure 7.10. Utilization versus W_s—DSPP traffic. (a) $H = 0.6$, (b) $H = 0.8$.

that these results show similar patterns to those of Bellcore traffic shown in Figure 7.7. Comparing Figures 7.8a and 7.8b, we observe that the cost reduction, with the increase of observation window size, is again more pronounced for larger Hurst parameters. This is due to the fact that the process with larger Hurst parameter has stronger correlation with the past and hence the larger window size contains more useful information for more accurate prediction of the future traffic.

7.2.3. Dependence of utilization on observation window size. The system utilization depends on the volume of traffic successfully transferred to the network. Here we present the utilization as a function of observation window size for three different values of communication delay (Figures 7.9 and 7.10).

Bellcore traffic trace. Figure 7.9 gives another aspect of system performance. It is clear from these curves that utilization increases with increasing window size. Note that as the window size increases to a certain level, the growth of utilization slows down. It illustrates that by simply increasing the window size the performance cannot be improved beyond a limit. Again this is due to the same reason as mentioned in Section 7.2.2.

DSPP ($H = 0.6, 0.8$) traffic traces. Again from Figure 7.10, it is clear that as the window size increases, utilization increases, which has been observed also in Figure 7.9 for Bellcore traffic.

8. Conclusion

In this paper, we have demonstrated that by use of predictive feedback control, it is possible to compensate the impact of communication delay causing performance degradation as reported in [2]. The method presented in this paper improves the overall system performance and prevents network instability. The numerical simulation results presented in Section 7 have shown the effectiveness of the proposed predictive feedback control law. We also explore the relationship between the Hurst parameter of the traffic and prediction performance. It was found that processes with larger Hurst parameter have better prediction performance. In addition, dependence of system cost and utilization on observation window size, corresponding to different values of Hurst parameters, has also been discussed. The results of this paper also lead to a better understanding of the impact of Hurst parameters on network performance. In summary, this work provides a useful tool for design and optimization of future networks using predictive feedback control law thereby avoiding instability.

References

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

[2] N. U. Ahmed, H. Yan, and L. O. 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.

[3] J. Banks, J. S. Carson II, and B. L. Nelson, *Discrete-Event System Simulation*, 2nd ed., Prentice Hall, New Jersey, 1995.

[4] A. Bhattacharya, A. G. Parlos, and A. F. Atiya, *Prediction of MPEG-coded video source traffic using recurrent neural networks*, IEEE Trans. Signal Process. **51** (2003), no. 8, 2177–2190.

[5] G. Cowan, *Statistical Data Analysis*, Oxford University Press, Oxford, 1998.

[6] M. Ghaderi, *On the relevance of self-similarity in network traffic prediction*, Tech. Report CS-2003-28, School of Computer Science, University of Waterloo, Ontario, 2003.

[7] M. Ghaderi, J. Capka, and R. Boutaba, *Prediction-based admission control for DiffServ wireless internet*, IEEE Vehicular Technology Conference (VTC '03), Florida, 2003.

[8] Y. Gong and I. F. Akyildiz, *Dynamic traffic control using feedback and traffic prediction in ATM networks*, Proc. 13th IEEE Networking for Global Communications conference (INFOCOM '94), Toronto, 1994, pp. 91–98.

[9] G. Gripenberg and I. Norros, *On the prediction of fractional Brownian motion*, J. Appl. Probab. **33** (1996), no. 2, 400–410.

[10] J. M. Hammersley and D. C. Handscomb, *Monte Carlo Methods*, Methuen, London, 1965.

[11] W. E. Leland, M. Taqqu, W. Willinger, and D. V. Wilson, *On the self-similar nature of Ethernet traffic*, IEEE/ACM Trans. Networking **2** (1994), no. 1, 1–15.

[12] Y. Lu, T. Abdelzaher, C. Lu, L. Sha, and X. Liu, *Feedback control with queueing-theoretic prediction for relative delay guarantees in web servers*, Proc. IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS '03), Washington, D.C., 2003, pp. 208–217.

[13] Y. Shu, Z. Jin, L. Zhang, L. Wang, and O. W. W. Yang, *Traffic prediction using FARIMA models*, Proc. IEEE International Conference on Communications (ICC '99) Vol. 2, Vancouver, 1999, pp. 891–895.

[14] W. Stallings, *Data and Computer Communications*, 6th ed., Prentice Hall PTR, New Jersey, 2000.

[15] N. Swaminathanaa, J. Srinivasanbb, and S. V. Raghavanaa, *Bandwidth-demand prediction in virtual path in ATM networks using genetic algorithms*, Comput. Commun. **22** (1999), no. 12, 1127–1135.

[16] A. S. Tanenbaum, *Computer Networks*, 3rd ed., Prentice Hall PTR, New Jersey, 1996.

[17] P. Tang and T. Tai, *Network traffic characterization using token bucket model*, Proc. 18th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '99), New York, 1999, pp. 51–62.

[18] B. J. Vickers, M. Lee, and T. Suda, *Feedback control mechanisms for real-time multipoint video services*, IEEE J. Select. Areas Commun. **15** (1997), no. 3, 512–530.

N. U. Ahmed: School of Information Technology and Engineering, University of Ottawa, Ottawa, Ontario, Canada K1N 6N5

E-mail address: ahmed@site.uottawa.ca

Hui Song: School of Information Technology and Engineering, University of Ottawa, Ottawa, Ontario, Canada K1N 6N5

E-mail address: hsong041@uottawa.ca