Congestion Control via Online Sampling

Gang Wu, Edwin K. P. Chong†, and Robert Givan
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN 47907, U.S.A.
{wu15, echong, givan}@ecn.purdue.edu

Abstract— **We consider the congestion-control problem in a communication network with multiple traffic sources, each modeled as a fully-controllable stream of fluid traffic. The controlled traffic shares a common bottleneck node with high-priority cross traffic described by a Markov-modulated fluid (MMF). Each controlled source is assumed to have a unique round-trip delay. We wish to maximize a linear combination of the throughput, delay, traffic loss rate, and a fairness metric at the bottleneck node. We introduce an online sampling-based burst-level congestion-control scheme capable of performing effectively under rapidly-varying cross traffic by making explicit use of the provided MMF model of that variation. The control problem is posed as a finite-horizon Markov decision process and is solved heuristically using a technique called Hindsight Optimization. We provide a detailed derivation of our congestion-control algorithm based on this technique. The distinguishing feature of our scheme relative to conventional congestion-control schemes is that we exploit a stochastic model of the cross traffic. Our empirical study shows that our control scheme significantly outperforms the conventional proportional-derivative (PD) controller, achieving higher utlization, lower delay, and lower loss under reasonable fairness. The performance advantage of our scheme over the PD scheme grows as the rate variance of cross traffic increases, underscoring the effectiveness of our control scheme under variable cross traffic.**

Keywords— **Communication networks, congestion control, traffic models, Markov-modulated fluid, Markov decision processes, online sampling.**

I. Introduction

We study the rate-based congestion control of traffic in a network where a bottleneck node is shared by multiple best-effort traffic sources and other high-priority "cross-traffic" sources. We assume that the best-effort sources can be fully controlled, but that each such source originates at some distance from the bottleneck node, and thus has a control delay. The objective of congestion control is to determine proper and fair transmission rates for the controlled sources to utilize the bandwidth available to the best-effort traffic efficiently at the bottleneck node while at the same time achieving low average queuing delay and a low traffic loss rate under reasonable fairness. We refer to the (varying) bandwidth available for best-effort traffic as the *service rate* of the bottleneck node.

Previous research on best-effort congestion control can be divided into rate-based approaches and credit-based approaches (e.g., [31]). Here we present a rate-based approach, controlling the rates of the best-effort sources rather than allocating credits to those sources. Early rate-based work involves binary feedback [7] and proportional controllers [8] for ATM (Asynchronous Transfer Mode) networks and linear-increase/exponential-decrease controllers for TCP/IP (Transmission Control Protocol/Internet Protocol) networks (see [15] for a recent version). Recent rate-based approaches attempt to achieve better performance by incorporating control-theoretic techniques, including proportional-derivative (PD) controllers [9], [10], [17], [30], and those using optimal control and dynamic game techniques such as linear quadratic (LQ) team, H^∞, and noncooperative game controllers [12], [13], [11].

We motivate our work by noticing that most of the above control schemes are designed for constant or slowly-varying service rates (with the exception of the LQ team and the H^∞ controllers, which do consider short-term variation in the service rate). These controllers aim to balance throughput, delay, and loss by maintaining queue size to a target value. We call these *connection-level* congestion controllers since they assume that the service-rate variation is caused primarily by the joining of new connections and the termination of existing ones—as a result, these controllers are typically evaluated by measuring primarily their response to single isolated step changes in service rate rather than performance over complex rapidly varying traffic. However, burstiness in cross traffic in real networks often occurs at small time scales, i.e., from several milliseconds up to a second [4]. Fast changes in the service rate, coupled with large bandwidth-delay products, often significantly degrade the performance of connection-level controllers. Intuitively, this performance degradation is due in part to the feedback and control delays—the service rate may change even before the adjustment of the traffic transmission rates impacts the bottleneck node, and thus the desired stable queue length may never be obtained. Moreover, all these approaches, assume linearized buffer dynamics at the bottleneck node; i.e., the boundaries of empty and full queues are ignored. This assumption causes stability problems under some bursty service-rate conditions.

We approach the congestion-control problem using an alternative paradigm that alleviates these drawbacks. We assume we are provided with a stochastic model of the cross traffic, and demonstrate a controller that achieves substantial benefits from exploiting this model. We call such controllers *burst-level* congestion controllers. We take a pro-active approach by predicting future service rates using the stochastic model so that our controller can anticipate changes stochastically and act "before" the changes happen.

We model the service rate at the bottleneck node as a Markov-modulated fluid (MMF). MMF models are commonly used to model high-priority QoS-sensitive (quality of service sensitive)

† Correspondence author.

This research is supported by DARPA/ITO under contract F19628-98-C-0051. The equipment for this work was provided in part by a grant from Intel Corporation. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the U. S. Government.

traffic, mostly comprising voice and video [1], [2], [3], [4]. In our setting, the service rate is the total link bandwidth reduced by the portion consumed by the high-priority traffic. Hence, the service rate can also be characterized using essentially the same MMF models. We use a Markovian model for the following reasons. First, Markovian models are very general; in particular, long-range dependent traffic can be approximated arbitrarily well by a Markovian model, perhaps with a large number of states [24]. Second, Markovian models are amenable to powerful decision-theoretic analysis, a strength that we exploit in our approach.

We formulate our congestion-control problem as a discrete-time finite-horizon Markov decision process (MDP) [6]. We formulate a measure of performance over long traces of service-rate variation, balancing throughput, delay, loss, and fairness. At the bottleneck node, we receive a positive reward by forwarding traffic, a negative reward for each time step the traffic spends waiting in buffer for service, a negative reward for traffic lost due to buffer overflow, and a negative reward for differences in traffic arrivals from different controlled sources. The objective is to maximize the average net reward over a finite horizon by choosing proper transmission rates for the controlled sources. We then extend our previously proposed Hindsight Optimization (HO) technique [5] to provide a heuristic solution to the MDP problem. The HO technique has never previously been used to address a problem with an infinite control action space.

The main contribution of this work is to demonstrate that a stochastic model of future service rates can be effectively exploited in congestion control to achieve substantial benefits in throughput, delay, and traffic loss, while maintaining reasonable fairness. A secondary contribution is to provide a specific means to obtain these benefits using a novel congestion-control framework based on online sampling. It remains to be seen whether this specific framework can be realized in practice with current technology, but our work provides both a strong motivation and a useful starting point for seeking a practically-realized congestion control scheme incorporating traffic models.

Our controller achieves significant performance improvements over the PD controller when small time-scale bandwidth variations are present. Although MMF models have been extensively employed in network performance analysis (e.g., [2], [3]), our work is the first to exploit such models for rate-based congestion control. In [12], [13], the authors model cross traffic by an auto-regressive moving-average process corrupted by a sequence of independent and identically distributed random numbers with zero mean and finite variance. Compared with [12], [13], MMF models have more structure, and better performance is therefore expected when such models are available. In [15], the authors incorporate a long-range dependent model into the design of a linear-increase/exponential-decrease controller. Our MMF model yields to a decision-theoretic analysis, as mentioned above, resulting in a controller that is not constrained to be linear-increase/exponential-decrease.

Previous work on congestion control using MDP formulations include [33], [34], [35]. Our work differs from [33], [34], [35] in several ways: 1) Our action space is continuous; 2) our reward structure is more general; 3) we develop an online sampling-based approach to cope with the continuous action and state spaces. Recent work on rollout algorithms (e.g., [36]) provide a means of using simulation to select "good" control actions heuristically, but requires starting with a good heuristic policy.

The remainder of the paper is organized as follows. In Section II, we describe our network model and define the congestion-control problem as an MDP. In Section III, we introduce the HO technique for heuristic MDP control, and present our gradient-based congestion-control algorithm. Section IV presents the simulation results of our controller and the PD controller to enable comparison. Section V concludes the paper.

II. System Model and Problem Description

A. System Model

We consider a network where a single bottleneck node is shared by multiple rate-controlled traffic sources and other high-priority "cross-traffic" sources. The controlled sources transmit at rates specified by a central controller residing at the bottleneck node. Fluid traffic generated by a source has a source-dependent fixed forward delay to reach the bottleneck node. Control signals, periodically generated by the rate controller, travel to each controlled source after a fixed source-dependent backward delay. Thus, associated with each source is a fixed round-trip delay. Without loss of generality, we assume that the round-trip delays are distinct from one another. We notate vectors and their components as follows: for vector $\vec{v}$, we write $v^{(i)}$ for the ith component of $\vec{v}$ when that component is scalar, and $\vec{v}^{\,(i)}$ for that component when it is itself a vector. We also notate the jth component of the ith component of $\vec{v}$ (when $\vec{v}$ is a vector of vectors) as $v^{(i,j)}$. Throughout this paper we use the notation E_X to denote expectation taken with respect to the random variable X. We assume that time is discrete with small time increments δ. We now describe four essential components of our system: the controlled traffic sources, the cross traffic, the bottleneck node, and the congestion controller.

Controlled Sources. We denote the collection of controlled sources by the set $\mathbf{N}$ and let $N = |\mathbf{N}|$ be the cardinality of $\mathbf{N}$, i.e., the number of controlled sources. We assume that in real time the round-trip delay associated with each controlled source is large compared with the time increment δ such that we can express the delay by integral multiples of δ in our discrete-time model with sufficient accuracy. Hence, in discrete time, we denote the round-trip delay of source i as $d^{(i)}$, a positive integer. Without loss of generality we index the sources such that $0 < d^{(1)} < \cdots < d^{(N)}$. We assume that the sources transmit at the controller-specified rates and respond to rate commands instantaneously upon their arrival. This model emulates controlled ABR (available bit rate) traffic in ATM networks and UDP (User Datagram Protocol) traffic in IP networks, which are suitable candidates for rate-based congestion-control schemes.

Cross Traffic. High-priority cross traffic represents, for example, CBR/VBR traffic in ATM networks, or traffic in IP networks receiving high-priority service via the CBQ (class-based queuing) scheme [14]. This cross traffic determines the service rate that controlled traffic experiences at the bottleneck node.

We assume that the cross-traffic process can change at any (discrete) time. For convenience, instead of specifying the cross-

traffic distribution, we specify the "service process," which is the difference between the rate of cross traffic and C, the constant bandwidth of the bottleneck node. We assume that the service process is represented by a Markov chain with state space $\mathbf{S} = \{1, \ldots, m\}$, a transition probability matrix $\mathbf{M} : (\mathbf{S} \times \mathbf{S}) \mapsto [0, 1]$, and a set of distinct rate values $v_1, \ldots, v_m$ (i.e., m is the number of the values that the service rate can take) which are real numbers in the interval $[0, C]$. By "service rate" we mean the amount of fluid traffic that can be served in one time step. When in state s, the service rate is v_s (a constant). Under this assumption, there is a one-to-one correspondence between the states and the service rates. Therefore, measuring service rate suffices to determine the state of the service process.

Bottleneck Node. The bottleneck node has a buffer of finite size. We assume that the best-effort traffic (from the controlled sources) is buffered together, independently of any buffering needed for the QoS-sensitive cross traffic. We denote the size of the buffer by B. As defined above, we denote the bandwidth at the bottleneck node by C.

At each discrete time step, the volume of (fluid) traffic that arrives is the sum of the fluid traffic from all of the controlled sources during the time step—each source contributes fluid arrival equal to the control sent to that source at the time step preceding the current time step by the round-trip delay for that source. We assume that the queue length at the bottleneck node is known at each time step.

Congestion Controller. The controller, residing at the bottleneck node, makes control decisions at each time step. The congestion-control problem is to determine a rate command $u_k^{(i)}$ to relay to source i, $i = 1, \ldots, N$, at time k to achieve some overall performance objective. Our objective is to balance throughput, delay, loss, and fair service to controlled sources, as described formally by the reward function below. When a source receives a command, it transmits at the rate specified by the command until another command is received. The controller can use system observations and a model of the service process to compute rate commands. The rate command for a source at any given epoch impacts the bottleneck node arrivals after a time duration equal to one round-trip delay for that source. Therefore, at each decision-making epoch, the controller needs to compute an appropriate rate command for each source that takes into account the round-trip delays and anticipated service-rate variation. The order of event occurrence at the bottleneck is: decision making, MMF transition, traffic arrival and simultaneous traffic forwarding (according to the new service-rate MMF process state), and then checking for buffer overflow/underflow—we thus assume that the control decision must be taken *before* observing the service rate at the current time step; to model this behavior we note that the MMF state component of the overall system state corresponds to the service rate observed in the previous time step.

B. MDP Problem Formulation

We formulate the congestion-control problem as a Markov decision process (MDP). An MDP consists of an action space, a state space, a state-transition structure, and a reward structure. In the following, we describe each component for our problem.

Action Space. We assume that the transmission rates at the controlled sources are bounded between zero and by a common value $C > 0$. We denote the action space by $\mathbf{U} = [0, C]^N$. At time k the control action is a vector $\vec{u}_k$ of the form $\vec{u}_k = [u_k^{(1)}, \ldots, u_k^{(N)}]$. In ATM networks the lower limit can be set to the minimum cell rate, without significant effect on our techniques.

State Space. The system state has three components. The first is the state of the service process (during service at the previous time step), taking values in $\mathbf{S}$. By the method of discretization described above, the state of the service process at each time step corresponds to the departure rate observed at the previous time step, and the MMF service model will transition before departures occur at the current time step. (This choice models the fact that we cannot know the current cross traffic precisely until after it has occurred and we have had the opportunity to measure it.) The second component is the current queue length l, taking values in $\mathbf{L} = [0, B]$. The third component consists of the control signals that have been issued in the past but whose impact has not yet been felt at the bottleneck node due to the round-trip delays. This control history $\vec{w}$ takes values in $\mathbf{U}^{d^{(N)}}$, where $d^{(N)}$ is the largest among all $d^{(i)}$'s. For example, if the control actions selected over time are $\vec{u}_0, \vec{u}_1, \ldots$ then the control history $\vec{w}_k = (\vec{w}_k^{(1)}, \ldots, \vec{w}_k^{(d^{(N)})})$ at time k is such that $\vec{w}_k^{(i)} = \vec{u}_{k-i}$ and thus $w_k^{(i,j)} = u_{k-i}^{(j)}$. We note that this control history includes unnecessary information in the form of history beyond the round-trip delay for sources closer than delay $d^{(N)}$. This information is included to greatly simplify our notation throughout this paper, but is not truly needed in the intended model or for any of our methods. The complete state space is $\mathbf{X} = \mathbf{S} \times \mathbf{L} \times \mathbf{U}^{d^{(N)}}$.

State Transition. If the state is $\vec{x} = (s, l, \vec{w})$ where $\vec{w} = (\vec{w}^{\,(1)}, \ldots, \vec{w}^{\,(d^{(N)})})$ denotes the control history, and we apply a control $\vec{u}$, the system will make a transition to a new state $\vec{x}^{\,\prime} = (s^\prime, l^\prime, \vec{w}^\prime)$. In the following, we specify how each component of $\vec{x}^{\,\prime}$ depends on $\vec{x}$ and $\vec{u}$.

The service-process state makes a transition from s to $s^\prime$ with probability $P(s, s^\prime)$ given by the $(s, s^\prime)$th entry in the given matrix $\mathbf{M}$—this transition is unaffected by the values of l and $\vec{w}$.

The queue-length component $l^\prime$ depends on $\vec{x}$ as follows. Let $a(\vec{x}) = \sum_{i=1}^{N} w^{(d^{(i)},i)}$ be the aggregate fluid traffic that arrives during the transition from state x to state $x^\prime$ from all controlled sources—this traffic is due to rate commands that were issued to these sources in the past which are now recorded in the state component $\vec{w}$. The queue-length component of the state changes according to the following difference equation, commonly called Lindley's equation:

$$l^\prime = \max\{\min\{l + a(\vec{x}) - v_{s^\prime}, B\}, 0\}.$$

The queue-length component $l^\prime$ does not depend on $\vec{u}$ due to non-zero round-trip delays.

Finally, the control history updates as follows:

$$\vec{w}^{\,\prime(1)} = \vec{u}, \qquad \vec{w}^{\,\prime(i+1)} = \vec{w}^{\,\prime(i)}, \qquad i = 1, \ldots, d^{(N)} - 1.$$

Reward Structure. We define the one-step reward at state $\vec{x}$ by

$$R(\vec{x}) = T(\vec{x}) - \alpha D(\vec{x}) - \beta F(\vec{x}) - \zeta L(\vec{x}), \qquad (1)$$

where $\alpha > \beta$, $0 < \beta < 1/(N-1)$, $\zeta > \beta$, $T(\vec{x})$ is the throughput received at one time step when the system is in state $\vec{x}$, $D(\vec{x})$ is the total queuing delay incurred at that time step, $F(\vec{x})$ is the sum of the absolute pairwise rate differences in arriving traffic from different controlled sources at that time step, and $L(\vec{x})$ is the fluid lost at the current time step due to buffer overflow (after "checking for buffer overflow").

The scaling factors α, β, and ζ reflect our tradeoff preference between throughput, delay, loss, and service fairness. The restrictions on α, β, and ζ above represent a preference-hierarchy among the four terms according to the following order: throughput, delay, loss, and fairness. Because maximizing throughput is typically the first concern in regulating controlled traffic sources, with fairness somewhat subordinate, we are most interested in parameter values satisfying this restriction. Restricting the ranges of α, β, and ζ as shown above allows the analytical selection of a hindsight-optimal control sequence (defined later) more easily. We do not consider the more difficult and less important case of parameter settings that violate this restriction.

The one-step reward $R(\vec{x})$ depends only on the state $\vec{x}$ and not explicitly on the control $\vec{u}$ because any rate command in $\vec{u}$ will not have impact on the bottleneck node until at least $d^{(1)}$ time units later, due to the non-zero round-trip delays $d^{(i)}$. We now provide formal expressions for $T(\vec{x})$, $D(\vec{x})$, $L(\vec{x})$, and $F(\vec{x})$ for completeness as follows:

$$T(\vec{x}) = \min\{l + a(\vec{x}), v_{s'}\} \quad (2)$$

$$D(\vec{x}) = \max\{\min\{l + a(\vec{x}) - v_{s'}, B\}, 0\} \quad (3)$$

$$L(\vec{x}) = \max\{l + a(\vec{x}) - v_{s'} - B, 0\} \quad (4)$$

$$F(\vec{x}) = \sum_{i=1}^{N} \sum_{j=1, j>i}^{N} \left| w^{(d^{(i)},i)} - w^{(d^{(j)},j)} \right|, \quad (5)$$

where s' is the service-rate process state after MMF transition from state s. Note that the throughput, delay, and loss terms of the reward function (and thus the reward itself) are random variables due to their dependence on the random variable s'.

Optimization Goal. Based on the MDP model described above, we can state the congestion-control problem as follows. For a given initial state $\vec{x}_0$, we apply a control $\vec{u}_0$ to the system and receive a reward of $R(\vec{x}_0)$ by serving traffic at the bottleneck node. The system will then make a transition to a new state $\vec{x}_1$, stochastically according to the state-transition structure. We then apply a control $\vec{u}_1$, and so on. After a horizon of H steps, the cumulative reward received (a random variable) is given by

$$W_H(\vec{u}_0, \ldots, \vec{u}_{\tilde{H}-1}) \equiv \sum_{k=0}^{H-1} R(\vec{x}_k),$$

where $\tilde{H} = H - d^{(1)}$, and $\vec{u}_{\tilde{H}-1}$ is the latest control command that can impact the bottleneck node within the horizon H.

Our choice of $\vec{u}_k$ is based on $\vec{x}_k$; that is, we use a "state-feedback" map $\mu_k : \vec{x} \mapsto \vec{u}$ and apply $\vec{u}_k = \mu_k(\vec{x}_k)$. The sequence of maps $\pi = \{\mu_0, \mu_1, \mu_2, \ldots\}$ is called a *policy*. For a given initial state $\vec{x}_0$, the problem is to find a policy that maximizes the objective function

$$V_H^{\pi}(\vec{x}_0) = E\left(W_H(\mu_0(\vec{x}_0), \ldots, \mu_{\tilde{H}-1}(\vec{x}_{\tilde{H}-1}))\right).$$

Given a policy π or a fixed sequence of controls $\vec{u}_0, \vec{u}_1, \ldots$, we denote the (random) state of the system at each time k in $0, 1, \ldots$ by the random variable X_k and the (random) state of the service process at time k by the random variable S_k.

C. Optimal Solution

To describe our approach to the congestion-control problem, we first characterize the optimal congestion-control policy. For a given initial state $\vec{x}$, let

$$V_H^*(\vec{x}) = \max_{\pi} V_H^{\pi}(\vec{x}).$$

Following a standard approach to solving MDPs, we write

$$Q_k(\vec{x}, \vec{u}) = R(\vec{x}) + E(V_{k-1}^*(\vec{x}')), \qquad k = 1, \ldots, H,$$

where the expectation in the right-hand side is with respect to the next state $\vec{x}'$, and $V_{k-1}^*(\vec{x}')$ is the optimal cumulative reward over the $k-1$ time steps starting from the (random) state $\vec{x}'$. A key result in Markov decision theory [6] then states that

$$V_H^*(\vec{x}) = \max_{\vec{u} \in \mathbf{U}} Q_H(\vec{x}, \vec{u}),$$

and a policy $\pi^* = \{\mu_0^*, \mu_1^*, \ldots\}$ is optimal if it satisfies for all k,

$$\mu_k^*(\vec{x}) = \arg\max_{\vec{u} \in \mathbf{U}} Q_{H-k}(\vec{x}, \vec{u}).$$

In particular, for a fixed horizon H, the control $\vec{u}^*$ is an optimal "current" action if it satisfies

$$\vec{u}^* = \mu_0^*(\vec{x}) = \arg\max_{\vec{u} \in \mathbf{U}} Q_H(\vec{x}, \vec{u}). \quad (6)$$

At each control epoch we apply the "current" control action $\vec{u}^*$ in (6). In other words, each control epoch involves optimizing $Q_H(\vec{x}, \vec{u})$ with respect to $\vec{u}$ for a horizon of H into the future. This approach of applying a "moving-horizon" control solution in an online fashion is common in the optimal-control literature, for example in *receding-horizon control* (see, e.g., [19], [20]).

In practice we do not have explicit knowledge of Q_H. Standard techniques can be used to compute Q_H in time polynomial in the size of the state space. However, because we have an implicitly specified state space (specified component by component above), our actual state space is astronomical in size; as a result, these standard techniques cannot be applied in practice. Thus, equation (6) is not directly useful for determining the optimal policy. Our MDP problem does not yield to any other known analytical solution. Instead, we approach the problem by computing an upper-bound estimate of $Q_H(\vec{x}, \vec{u})$. In the next section, we describe a particular approach to solving our optimization problem, based on evaluating candidate actions using such upper-bound estimates of $Q_H(\vec{x}, \vec{u})$.

III. Congestion-control Algorithm Using Hindsight Optimization

A. The Hindsight Optimization Technique

In this subsection, we outline our solution approach, which extends a technique called *hindsight optimization*, first described in [5]. The overall control architecture is illustrated in

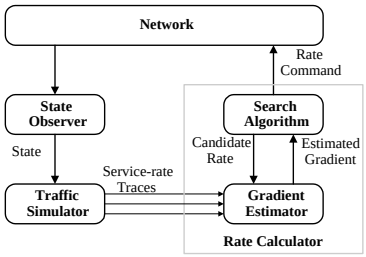

Fig. 1. Congestion-control architecture.

Figure 1. The controller comprises three parts: a state observer, a traffic simulator, and a rate calculator. The state observer is responsible for obtaining the system state $\vec{x}$ by measuring the service rate at each time step (as well as observing the current queue length and storing the recent control history). We assume that the controller has an accurate MMF model of the cross traffic, allowing the state observer to infer the MMF state by measurement (given our assumption that the MMF model state each determines a unique service rate). Hence, the system state is fully observable.

The traffic simulator takes the observed current state $\vec{x}$ and uses it as a starting state to generate a finite number of possible service-rate sequences (traces) using our MMF model. The rate calculator takes these traces and computes a rate command vector $\vec{u}$. The calculation of the rate command vector is based on the following idea. Recall from equation (6) that at any given control epoch and any state $\vec{x}$, the optimal rate command is given by

$$\vec{u}^* = \arg\max_{\vec{u}\in\mathbf{U}} Q(\vec{x},\vec{u}) \tag{7}$$

(we omit the subscript H in $Q_H(\vec{x},\vec{u})$ for brevity). We rely on an estimate $\hat{Q}(\vec{x},\vec{u})$ of the $Q(\vec{x},\vec{u})$ to carry out the above maximization. This estimate is calculated as follows. For each service-rate trace t, we compute the cumulative reward by taking action $\vec{u}$ at state $\vec{x}$ followed by a *trace-optimal* sequence of actions $\vec{u}_1^t, \vec{u}_2^t, \ldots, \vec{u}_{\tilde{H}-1}^t$ for the remaining horizon of $\tilde{H}-1$ time steps. We say that the sequence $\vec{u}_1^t, \vec{u}_2^t, \ldots, \vec{u}_{\tilde{H}-1}^t$ is trace-optimal if this sequence achieves the largest possible cumulative reward under the assumption that the service rate does indeed vary according to the trace under consideration. We call any such trace-optimal sequence a *hindsight-optimal control sequence* and the optimal cumulative reward of any such sequence the *hindsight-optimal value* of the trace—computing such a sequence and its corresponding value is a deterministic optimization problem that is often considerably easier than finding the optimal stochastic control for the online problem. We compute the average of the hindsight-optimal values over the set of n generated traces for the specified initial control $\vec{u}$—this average is our estimate $\hat{Q}_n(\vec{x},\vec{u})$ of $Q(\vec{x},\vec{u})$. In other words, $\hat{Q}(\vec{x},\vec{u})$ and its sampled approximation $\hat{Q}_n(\vec{x},\vec{u})$ are given by

$$\hat{Q}_n(\vec{x},\vec{u}) = \frac{1}{n}\sum_{t=1}^{n} W_t^*(\vec{x},\vec{u}) \tag{8}$$

$$\hat{Q}(\vec{x},\vec{u}) = R(\vec{x}) + E_{S_1,\ldots,S_H} \max_{\vec{u}_1,\ldots,\vec{u}_{\tilde{H}-1}} W_{H-1}(\vec{u}_1,\ldots,\vec{u}_{\tilde{H}-1}). \tag{9}$$

where

$$W_{H-1}(\vec{u}_1,\ldots,\vec{u}_{\tilde{H}-1}) \equiv \sum_{k=1}^{H-1} R(\vec{x}_k),$$

and $W_t^*(\vec{x},\vec{u})$ is the hindsight-optimal value for the trace t as a result of applying control $\vec{u}$ at state $\vec{x}$.

Given the estimate $\hat{Q}(\vec{x},\vec{u})$ of $Q(\vec{x},\vec{u})$ for each action, the hindsight optimization (heuristic) approach is to select and execute the action $\vec{u}$ (a vector) that maximizes this estimate. Previous applications of the hindsight-optimization technique have all involved MDP problems with finite action spaces, unlike our congestion control problem. When the action space is finite, we can simply compute the estimate $\hat{Q}(\vec{x},\vec{u})$ for each action $\vec{u}$, and choose the candidate action associated with the largest such estimate for execution. Here, however, we have an uncountable continuous action space, and cannot compute this estimate for every action. Instead, we extend the hindsight optimization technique by finding a (locally) optimal action using a gradient ascent technique. Because we seek to use gradient ascent to solve this problem, we actually need to analyze traces to find the gradient of $\hat{Q}(\vec{x},\vec{u})$ relative to changes in $\vec{u}$ (rather than to find $\hat{Q}(\vec{x},\vec{u})$ itself). We discuss a method for estimating this gradient from traces below.

To conclude this subsection, we argue that $\hat{Q}(\vec{x},\vec{u})$, if exactly computed (e.g., by infinite sampling), is an upper bound on $Q(\vec{x},\vec{u})$. Let $\vec{x}_1$ denote the next (random) system state after applying control $\vec{u}$ at state $\vec{x}$. Then, $\hat{Q}(\vec{x},\vec{u})$ can be expressed formally as:

$$\begin{aligned}\hat{Q}(\vec{x},\vec{u}) &= R(\vec{x}) + E_{S_1}E_{S_2,\ldots,S_H} \max_{\vec{u}_1,\ldots,\vec{u}_{\tilde{H}-1}} W_1, \quad (10)\\ W_1 &= W_{H-1}(\vec{u}_1,\ldots,\vec{u}_{\tilde{H}-1}).\end{aligned}$$

Note that the sequence $\vec{x}_1,\ldots,\vec{x}_{H-1}$ depends on the controls $\vec{u}_1,\ldots,\vec{u}_{\tilde{H}-1}$ although the notation does not reveal this dependence explicitly. Comparing equation (10) with the following form of $Q(\vec{x},\vec{u})$,

$$\begin{aligned}Q(\vec{x},\vec{u}) &= R(\vec{x}) + E_{S_1} \max_{\mu_1,\ldots,\mu_{\tilde{H}-1}} E_{S_2,\ldots,S_H} W_2, \quad (11)\\ W_2 &= W_{H-1}(\mu_1(\vec{x}_1),\ldots,\mu_{\tilde{H}-1}(\vec{x}_{\tilde{H}-1})),\end{aligned}$$

we can see that $\hat{Q}(\vec{x},\vec{u})$ estimates an upper bound of $Q(\vec{x},\vec{u})$ computed by interchanging expectation and maximization. Note that the "max" in the definition of $\hat{Q}(\vec{x},\vec{u})$ is over sequences of *actions* due to the ability (in $\hat{Q}(\vec{x},\vec{u})$) to apply tailored action sequences to different stochastic futures—in contrast the "max" in $Q(\vec{x},\vec{u})$ occurs outside the expectation, requiring a single *policy* to be selected for all futures. As discussed in [5], the $\hat{Q}(\vec{x},\vec{u})$ upper bound on $Q(\vec{x},\vec{u})$ can be arbitrarily loose. However, these estimates are only used to rank competing candidate actions, and thus it only matters whether or not these estimates preserve the relative values at different states. Our results below give evidence that for congestion control problems the ranking is preserved well enough to make $\hat{Q}(\vec{x},\vec{u})$ useful for selecting an effective control policy.

B. Hindsight-optimal Control Sequences

At each decision-making epoch, we wish to determine a rate vector $\vec{u}^*$ according to (7) using $\hat{Q}_n(\vec{x},\vec{u})$ in place of $Q(\vec{x},\vec{u})$.

For a given $\vec{x}$, we wish to maximize $\hat{Q}_n(\vec{x}, \vec{u})$ with respect to $\vec{u}$. Because the argument $\vec{u}$ is a vector of continuous variables, we can use a search algorithm based on the gradient of $\hat{Q}_n(\vec{x}, \vec{u})$ with respect to $\vec{u}$, which we denote by $\nabla_{\vec{u}} \hat{Q}_n(\vec{x}, \vec{u})$.

Note that from equation (8), we can write $\nabla_{\vec{u}} \hat{Q}_n(\vec{x}, \vec{u})$ as

$$\nabla_{\vec{u}} \hat{Q}_n(\vec{x}, \vec{u}) = \frac{1}{n} \sum_{t=1}^{n} \nabla_{\vec{u}} W_t^*(\vec{x}, \vec{u}),$$

where $\nabla_{\vec{u}} W_t^*(\vec{x}, \vec{u})$ is the gradient of $W_t^*(\vec{x}, \vec{u})$ with respect to $\vec{u}$. Therefore, the calculation of $\nabla_{\vec{u}} \hat{Q}_n(\vec{x}, \vec{u})$ reduces to calculating the gradients on a per-trace basis, i.e., the gradients of the $W_t^*(\vec{x}, \vec{u})$ (hindsight-optimal values). It turns out that these gradients can be computed analytically, as we will show later. Our gradient estimate above is akin to the idea of *infinitesimal perturbation analysis* [27].

Recall that $d^{(1)}$ is the smallest round-trip delay among all $d^{(i)}$'s. We note that if $d^{(1)} \geq H - 1$, then no matter what control sequence we apply, by the end of the horizon H, no action will have impact on the bottleneck node. Therefore, we consider only the nontrivial case where $0 < d^{(1)} < H - 1$.

Suppose that the current time is zero and the current state is $\vec{x}_0 = (s_0, l_0, \vec{w}_0)$. We wish to select as the current control the action $\vec{u}_0^*$ that maximizes $\hat{Q}(\vec{x}_0, \vec{u}_0^*)$. In the following, we first describe how we compute a hindsight-optimal control sequence for a given service-process trace. Based on n such sequences, we then show how to obtain the gradient of $\hat{Q}_n(\vec{x}_0, \vec{u}_0)$, which together with a search algorithm forms our congestion-control algorithm. We now assume that we have generated a specific service-process trace $t = \{v_{s_1}, \ldots, v_{s_H}\}$.

Given the initial state $\vec{x}_0$ and action $\vec{u}_0$, define the trace-relative committed aggregate arrival rate (from all controlled sources due to rate commands in the control history as indicated in the $\vec{w}_0$ component of x_0) at times $k = 0, \ldots, H-1$ for initial control $\vec{u}_0$ by

$$a_k^t(\vec{u}_0) = \sum_{i=1}^{N} I_i(k) w_0^{(d^{(i)}-k,i)}, \text{ where } I_i = 1_{\{0,\ldots,d^{(i)}\}}$$

where we define $\vec{w}_0^{(0)}$ to be $\vec{u}_0$. Define the trace-relative queue-length sequence $\{l_k^t,\ k = 0, \ldots, H\}$ (and the queue length $\{\tilde{l}_{k+1}^t, k = 1, \ldots, H\}$ before "checking for underflow") for initial control $\vec{u}_0$ by

$$\begin{aligned} l_{k+1}^t(\vec{u}_0) &= \max\{\tilde{l}_{k+1}^t(\vec{u}_0), 0\}, \text{ and} \\ \tilde{l}_{k+1}^t(\vec{u}_0) &= \min\{l_k^t(\vec{u}_0) + a_k^t(\vec{u}_0) - v_{s_{k+1}}, B\}, \end{aligned}$$

with $l_0^t(\vec{u}_0) = l_0$ the current queue size.

Notice $u_0^{(i)}$ is the control decision we are going to make at the current time and has not been determined yet; however, in the following proposition, $u_0^{(i)}$ is assumed given because we will assign candidate values to determine the associated $\hat{Q}$ values. For a service-rate trace $t = \{v_{s_1}, \ldots, v_{s_H}\}$, we wish to compute a hindsight-optimal control sequence $\vec{u}_1^t, \ldots, \vec{u}_{\tilde{H}-1}^t$. In choosing such a control sequence, we will need notation for the number of sources that can affect a given time (i.e., sources such that the round-trip delay is less than the specified time)—we write $\tilde{N}_k$ for the number of sources j such that $k \geq d^{(j)}$. The following proposition gives the means to compute the unique hindsight-optimal control sequence analytically.

Proposition 1: Given system state $\vec{x}_0 = (s_0, l_0, \vec{w}_0)$, action $\vec{u}_0$, and a trace of future service rates $v_{s_1}, \ldots, v_{s_H}$, the sequence $\{\vec{u}_k^t,\ k = 1, \ldots, \tilde{H} - 1\}$ with $(u_k^t)^{(i)}$ for source i specified as below is the unique hindsight-optimal control sequence.

$$(u_k^t)^{(i)} = \begin{cases} 0 & \text{when } \tilde{l}_{k+d^{(i)}+1}^t \geq 0, \\ -\tilde{l}_{k+d^{(i)}+1}^t / \tilde{N}_{k+d^{(i)}} & \text{otherwise,} \end{cases}$$

where $\tilde{l}_{k+d^{(i)}+1}^t = \tilde{l}_{k+d^{(i)}+1}^t(\vec{u}_0)$ is a function of $\vec{u}_0$.

A correctness argument for Proposition 1 can be found in the full version of this paper at: dynamo.ecn.purdue.edu/~ngi/).

C. Search Algorithm

At each time step, we wish to determine the control action $\vec{u}_0^*$ that yields the largest estimate $\hat{Q}(\vec{x}_0, \vec{u}_0)$. We use a search algorithm that uses only the gradient of $\hat{Q}$. Let $\nabla_{\vec{u}} \hat{Q}_n(\vec{x}_0, \vec{u}_0)$ represent the gradient of $\hat{Q}_n(\vec{x}_0, \vec{u}_0)$ (with respect to the control action $\vec{u}_0$). The search algorithm is of the form (see, e.g., [16])

$$\vec{u}(k+1) = \vec{u}(k) + \gamma(k) \nabla_{\vec{u}} \hat{Q}_n(\vec{x}_0, \vec{u}(k)), \qquad (12)$$

where $\gamma(k)$ is a positive step size, and the iterate $\vec{u}(k)$ is an estimate of $\vec{u}_0^*$ (more sophisticated algorithms are possible, but appear unnecessary for our purposes). From equation (8), $\nabla_{\vec{u}} \hat{Q}_n(\vec{x}_0, \vec{u}_0)$ is given by

$$\nabla_{\vec{u}} \hat{Q}_n(\vec{x}_0, \vec{u}_0) = \frac{1}{n} \sum_{t=1}^{n} \nabla_{\vec{u}} W_t^*(\vec{x}_0, \vec{u}_0)$$

where $\nabla_{\vec{u}} W_t^*(\vec{x}_0, \vec{u}_0)$ is the gradient of the hindsight-optimal value for the trace t.

There are points where the trace-relative hindsight-optimal value is not differentiable. The use of gradient ascent methods with functions that are not everywhere differentiable has been studied before (e.g., [32]). In practice, we have found that the non-differentiable points in our objective function do not impact the efficacy of the gradient ascent algorithm. In fact, in our empirical study the gradient ascent algorithm never encounters these non-differentiable points. Hence, we do not delve further into this issue.

The result of Proposition 2 below can be used to compute the gradient $\nabla_{\vec{u}} W_t^*(\vec{x}_0, \vec{u}_0)$. Combining this result with the algorithm (12), we have an iterative procedure to compute $\vec{u}_0^*$. In practice, we terminate the algorithm (12) when the gradient $\nabla_{\vec{u}} \hat{Q}_n(\vec{x}_0, \vec{u}(k))$ is sufficiently close to $\vec{0}$. Specifically, we stop when $\|\nabla_{\vec{u}} \hat{Q}_n(\vec{x}_0, \vec{u}(k))\| \leq \varepsilon$, where $\varepsilon > 0$ is a prespecified parameter and we use $\|\vec{v}\|$ to denote the "sup norm" given by $\max_i |v^{(i)}|$. Note that we also need a value $\vec{u}_0(0)$ to initialize the algorithm. For the step size sequence $\{\gamma(k)\}$, a typical and simple choice is to set $\gamma(k)$ to be a small positive constant.

We summarize the search procedure in the following routine. Let $\mathbf{Tr}$ be a given set of future bandwidth traces, $n = |\mathbf{Tr}|$ the cardinality of $\mathbf{Tr}$ as defined before, and $\nabla_{\vec{u}} W_t^*(\vec{x}_0, \vec{u}(k))$ the gradient of the hindsight-optimal value for trace t in $\mathbf{Tr}$, as given by Proposition 2. Let $\vec{d}$ be a vector whose ith entry is $d^{(i)}$.

grad-search$(\mathbf{Tr}, \vec{d})$
1. Initialize $\vec{u}(0)$.
2. For $k = 1, 2, \ldots,$ do
$\nabla_{\vec{u}} \hat{Q}_n(\vec{x}_0, \vec{u}(k)) = (1/n) \sum_{t=1}^{n} \nabla_{\vec{u}} W_t^*(\vec{x}_0, \vec{u}(k))$
$\vec{u}(k+1) = \vec{u}(k) + \gamma(k) \nabla_{\vec{u}} \hat{Q}_n(\vec{x}_0, \vec{u}(k))$
until $|\nabla_{\vec{u}} \hat{Q}_n(\vec{x}_0, \vec{u}(k))| \leq \varepsilon$.
3. Output $\vec{u}(k)$.

The set of traces $\mathbf{Tr}$ and delay parameter vector $\vec{d}$ are listed as arguments of the routine because both are needed in the calculation of $\nabla_{\vec{u}} W_t^*(\vec{x}_0, \vec{u}(k))$. Note that the above algorithm assumes that the state $\vec{x}_0$ is given.

The search algorithm is in fact only a local-search method; the solution obtained depends on the initial condition, and may not be globally optimal. To search for a globally optimal solution, we could employ other familiar search techniques such as simulated annealing, but we have found satisfactory empirical results using only this local search.

D. The Gradient of the Hindsight-optimal Value

This section summarizes our efficient means of computing the gradient $\nabla_{\vec{u}} W^*(\vec{x}_0, \vec{u}_0)$ (a more technical account, including a correctness argument of the gradient, can be found in the full version of this paper at: dynamo.ecn.purdue.edu/~ngi/).

Let the trace-relative projected queue-size trajectories $\{l_k^t, k = 0, \ldots, H\}$ and $\{\tilde{l}_{k+1}^t, k = 1, \ldots, H\}$ be as defined previously. Let $k^{t,i}(\vec{u}_0)$ be the first buffer-underflow or buffer-full time after $d^{(i)} - 1$ when encountering trace t with no additional flow requested after $\vec{u}_0$, given by

$$\begin{aligned} k_{\downarrow}^{t,i}(\vec{u}_0) &= \min\Big\{k : \Big(l_{k+1}^t = 0 \text{ and } d^{(i)} \leq k < H-1\Big) \\ &\qquad \text{or } k = H-1\Big\}, \\ k_{\uparrow}^{t,i}(\vec{u}_0) &= \min\Big\{k : \Big(l_{k+1}^t = B \text{ and } d^{(i)} \leq k < H-1\Big) \\ &\qquad \text{or } k = H-1\Big\}, \text{ and} \\ k^{t,i}(\vec{u}_0) &= \min\Big\{k_{\downarrow}^{t,i}, k_{\uparrow}^{t,i}\Big\}. \end{aligned}$$

Define $\mathbf{N}_i$ to be the set of all sources j in $\mathbf{N}$ with round-trip delays greater than $d^{(i)}$. By our ordering of the sources we have $\mathbf{N}_i = \{j \in \mathbf{N} : j > i\}$. We now introduce notation that divides the sources in $\mathbf{N}_i$ according to whether they have arrival rates at time $d^{(i)}$ that are higher or lower than any particular rate r. For each source $i \in \mathbf{N}$, we partition the set $\mathbf{N}_i$ of "uncontrollable" sources into two subsets, $\mathbf{N}_<^i(r)$ and $\mathbf{N}_>^i(r)$, according to how the arrival rates from those sources at time $d^{(i)}$ compare to the rate r, as follows:

$$\begin{aligned} \mathbf{N}_<^i(r) &= \{j \in \mathbf{N}_i : w_0^{(d^{(j)} - d^{(i)}, j)} < r\}, \\ N_<^i(r) &= |\mathbf{N}_<^i(r)|, \\ \mathbf{N}_>^i(r) &= \{j \in \mathbf{N}_i : w_0^{(d^{(j)} - d^{(i)}, j)} > r\}, \text{ and} \\ N_>^i(r) &= |\mathbf{N}_>^i(r)|. \end{aligned}$$

The gradient $\nabla_{\vec{u}} W^*(\vec{x}_0, \vec{u}_0)$, at points where it exists, is now given as follows.

Proposition 2: Given state $\vec{x}_0$, candidate initial control $\vec{u}_0$, and service-rate trace $t = \{v_{s_1}, \ldots, v_{s_H}\}$, the source i component (for any i) of $\nabla_{\vec{u}} W^*(\vec{x}_0, \vec{u}_0)$ is given by the weighted sum of the following four terms (with the weights 1, $-\alpha$, $-\beta$, and $-\zeta$, respectively), representing the throughput, delay, fairness, and loss components of the change in total reward:

$$\begin{aligned} \nabla_{\vec{u}} T(\vec{x}_0, \vec{u}_0)^{(i)} &= \begin{cases} 1 & \text{if } i = 1 \text{ and } k_{\downarrow}^{t,1}(\vec{u}_0) = d^{(1)} \\ 0 & \text{otherwise} \end{cases} \\ \nabla_{\vec{u}} D(\vec{x}_0, \vec{u}_0)^{(i)} &= k^{t,i} - d^{(i)} \\ \nabla_{\vec{u}} F(\vec{x}_0, \vec{u}_0)^{(i)} &= N_<^i(u_0^{(i)}) - N_>^i(u_0^{(i)}) + \nabla_{\vec{u}} F_1^{(i)} \\ \nabla_{\vec{u}} L(\vec{x}_0, \vec{u}_0)^{(i)} &= \begin{cases} 1 & \text{if } k_{\uparrow}^{t,i} < k_{\downarrow}^{t,i} \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

where

$$\nabla_{\vec{u}} F_1^{(i)} = \begin{cases} 0 & \text{if } i = 1 \\ i-1 & \text{else if } \tilde{l}_{d^{(i)}+1}^t > 0 \\ i + N_>^i(r) - N_<^i(r) & \text{else if } r < u_0^{(i)} \\ -i + N_>^i(r) - N_<^i(r) & \text{else if } r > u_0^{(i)} \end{cases},$$

$$r = -\tilde{l}_{d^{(i)}+1}^t(u_0^{(i)})/(i-1).$$

E. The Congestion-control Algorithm

We conclude this section by summarizing the congestion-control algorithm as follows. At each control epoch, we perform these steps:

1. Observe current system state $\vec{x}_0$;
2. Generate a set $\mathbf{Tr}$ of future service-rate traces;
3. Compute $\vec{u}_0^* =$ **grad-search**$(\mathbf{Tr}, \vec{d})$;
4. Transmit rate command $\vec{u}_0^*$ to sources.

IV. Empirical Results

A. Evaluation Setup

We use the network simulator *ns* version 2 as the basis of our simulation environment. We have modified *ns* to implement our congestion-control algorithm over UDP in *ns* to emulate an ATM network. Accordingly, we set the packet size to be 53 bytes, the size of a standard ATM cell.

Figure 2 illustrates our simulated network configuration. The traffic sent from four sources, indexed by 0 to 3, shares a common bottleneck node, and has a common destination node. All links between the sources and the bottleneck node have bandwidth of 155 Mbps, while the bottleneck link is of only 55 Mbps. The size of the buffer at the bottleneck node is 150 cells. Source 0 represents the source for high-priority cross traffic. Sources 1, 2, and 3 are controlled best-effort traffic sources, which send traffic at the rates determined by the controller residing at the bottleneck node. These three sources are associated with round-trip delays of 20, 30, and 40 ms, respectively, as shown in the figure.

Our empirical study consists of two parts. In the first part, source 0 is composed of ten identical connections, each generating fluid traffic according to a two-state ON-OFF MMF

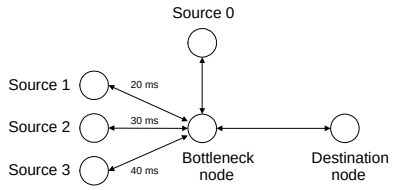

Fig. 2. Network configuration for empirical study.

model. In our experiments, we will be varying the two transmission rates corresponding to the ON and the OFF states, respectively, to let the aggregated cross traffic rate have different variances (ranging from 0 to 72.6 Mbps^2) but the same mean (22 Mbps). This allows us to study the impact of the variance of the cross-traffic rate on system performance. In the two-state MMF model, the expected lengths of the ON and the OFF periods are 400 and 600 ms, respectively; these values were chosen to reflect realistic voice connections, which is the focus of the second part of our empirical study.

In the second part, source 0 consists of one thousand independent and identically distributed voice connections, reflecting a typical scenario arising in networks with mixed voice and data traffic. While most data traffic receives only best-effort service, commercial telecommunication companies have begun to carry voice connections, which require real-time guaranteed service, over packet-switched networks. The dynamics of voice connections are well captured by MMF models [2], [18]. We model a single voice connection by a two-state ON-OFF MMF model, with the expected ON and OFF periods being 400 and 600 ms, respectively. Since a standard voice connection consumes 64 Kbps bandwidth, we set the rate of each voice connection in our simulation to 70.667 Kbps by considering that the actual payload in a 53-byte ATM cell is only 48 bytes.

B. Comparison Metrics

We compare the performance of the controller described in this paper, called the "HO controller" hereafter, with the well-known PD controller (with various values of the target queue size). Our metrics for comparison are utilization, average queuing delay, cell loss rate, and fairness. Utilization is the throughput normalized by the total available service "volume" (the sum of the service rates) over the simulation period at the bottleneck node. The average queuing delay is the total amount of time that all the cells spend waiting in the queue at the bottleneck node divided by the total number of cells forwarded. The cell loss rate is defined as the number of cells (from the controlled sources) lost due to buffer overflow divided by the total number of cells (from the controlled sources) that arrive at the bottleneck node over the simulation period. To define fairness, let T_i denote the total number of cells that arrive from source i. Then, fairness is defined as the sum of the mutual absolute differences between the T_i ($i = 1, 2, 3$) values, normalized by the product of the number of source pairs (3 in our case) and the sum of T_i, $i = 1, 2, 3$. In order words, our fairness metric is the the average pairwise arrival difference per unit arrival.

In most previous papers on rate-based congestion control, e.g., [9], [10], the test metric is the controller's ability to maintain a target queue size. However, by design the HO controller does not aim to maintain a fixed queue size. Thus, we do not evaluate the HO controller's ability to maintain a particular target queue size.

C. Impact of Cross-traffic Variance

In this subsection, we change the variance of the cross traffic rate by varying the transmission rates corresponding to the ON and OFF states but still keep the same mean rate. We investigate how the variance impacts the performance of the HO and the PD controllers.

PD-type congestion controllers have been shown to be generally effective [9], [10], [17], [30]. PD controllers adjust the transmission rate based on the deviation of the queue length from a target value. We tested the PD controller with different target queue sizes. The target queue size reflects the network administrator's tradeoff among utilization, delay, and cell loss rate. A larger queue size indicates the desire for higher utilization at the expense of higher delay and cell loss rate. To maintain fairness, the PD controller issues the same rate command to every controlled source at every decision-making epoch.

We chose the values of the parameters (gains) of the PD controller by first following the design procedure provided in [9] and then by fine tuning manually to obtain the best response possible for a constant cross traffic with rate of 22 Mbps, which is the mean rate of the cross traffic with which we will be carrying out our tests. The queue-length responses achieved by our choice of gains are very similar to those of [10]. Our experience with the PD controller suggests that tuning the gains of a high-order PD controller is nontrivial (the order is determined by the largest round-trip delay; for example, in our experiments the PD controller is of order 43). The stability and quality of the system response depend on many factors, such as the values of gains, the length of the control update interval, and initial conditions.

For the HO controller, we use $n = 200$; i.e., at each decision epoch we generate 200 service-rate traces using the cross-traffic model. Each trace is of length $H = 50$ time intervals. We chose the duration δ of each time interval to be 1 ms. The value of δ was chosen on the one hand to be small enough to capture the fast variation in service rate and on the other hand large enough for affordable computation. In addition, 1 ms is a value of δ small enough to express typical round-trip delays as integral multiples of δ.

With a value of $\delta = 1$ ms, the computational burden still seems nontrivial. Fortunately, many of the computation performed by the HO controller can be carried out in parallel. For example, the calculation of the components in the gradient are independent of each other, and thus these components can be readily computed in parallel. The trace generation can also be carried out in parallel. Bearing this in mind and considering the pace of progress in computation speeds, we believe that real-time implementation of our control algorithm is feasible.

We set $\alpha = 1000$, $\beta = 1/3$, and $\zeta = 1/2$. These values satisfy the restrictions on the values of α, β, and ζ that we have given in defining our objective function, in equation (1).

Figure 3 shows the utilization values achieved by the competing controllers. The PD controller with a target queue size of 50 cells is denoted by PD-50; similarly, PD-10 and PD-1 represent PD controllers with target queue sizes of 10 and 1, re-

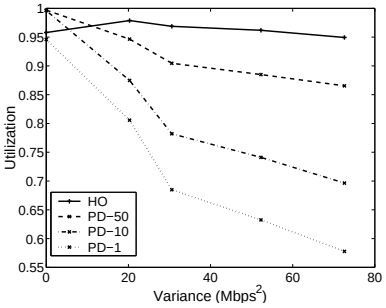

Fig. 3. Plots of utilizations achieved by the HO controller and PD controllers with different target queue sizes versus the variance of cross-traffic rate. The symbol PD-50 stands for the PD controller with a target queue size of 50 cells, etc.

spectively. The horizontal axis is the rate variance of the cross traffic. We see that all controllers achieve high utilization, when constant-rate cross traffic is present (i.e., when rate variance is zero in the figure). As the cross traffic becomes more variable, the utilization values achieved by the PD controllers decrease at rates much faster than that of the HO controller. The reason is that when the cross traffic is highly variable, the PD controllers cannot maintain a stable queue size to ensure satisfactory utilization. In contrast, the HO controller can stochastically "anticipate" changes in service rate and can in turn respond to these changes beforehand. Figure 3 demonstrates the effectiveness of the HO controller in an environment with a highly-variable service rate, a condition which is often found in practical networks. We note that at very low variance the HO controller is outperformed by the most utilization-aggressive (high target queue size) PD controllers, but that this disadvantage disappears quickly with increasing variance. We note that even for very low variance traffic this disadvantage is not without a corresponding benefit—in the next plot we will see that utilization-aggressive PD controller suffer a significant penalty in average delay under these same conditions. The only PD controller that compete with HO in average delay is the PD-1 controller, which shows no utilization advantage over the HO controller, even with low variance in the service rate.

Figure 4 shows plots of the average queuing delays. The HO controller achieves much smaller queuing delays than those of PD-50 and PD-10. PD-1 has queuing delays close to those of HO; however, it does so at the significant cost of much smaller utilizations (see Figure 3). Compared with HO, PD-50 has much larger delay and less utilization; it has more than twice the delay and 8% less utilization in the most variable service rate case, i.e., the right-most point in the figures shown. Figures 4 and 3 suggest that the HO controller can achieve higher utilization with smaller delay compared with the PD controllers with fixed target queue sizes.

Figure 5 shows plots of cell loss rates (CLRs). The CLR for the HO controller is the smallest for all the experiments we carried out due to the fact that the implicit goal of a hindsight-optimal control sequence is to keep a zero queue length, and hence it leaves most of the buffer ready to absorb bursts of incoming traffic. The PD-50 controller, which achieves the closest utilization values to HO among the PD controllers, has a CLR that is at least seven times that of HO in all our experiments.

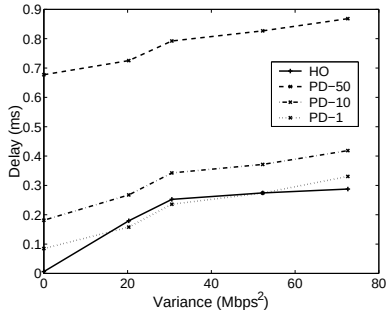

Fig. 4. Plots of average delays.

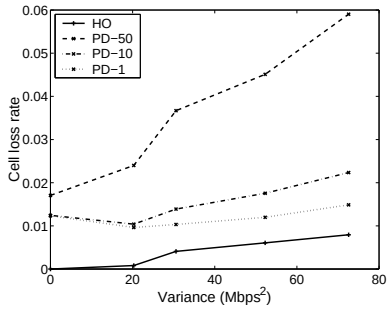

Fig. 5. Plots of cell loss rates.

Figure 6 shows plots of the fairness metric. We see that the HO controller is less fair than the PD controllers. The figure shows that in the most variable service rate case, the average difference between two T_i's is less than 4% of the total arrival, while for the PD's, the unfairness value is negligibly small. The reason that the PD controllers are more fair is that they are "hardwired" to issue to all the controlled sources the same rate command at decision-making epochs, while the HO controller does not have this restriction. Instead, the HO controller makes decisions based on balancing the throughput, delay, loss, and fairness in the reward function (with fairness being lowest in priority), and thus the controller may sacrifice fairness for throughput. For example, in the most variable service rate case, HO gains 8% more throughput than PD-50 with much less delay and cell loss rate.

Moreover, as expected, the HO controller achieves the highest cumulative reward in all the experiments we conducted. However, we do not show any figure here due to space limiataion.

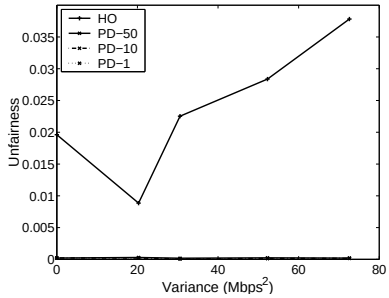

Fig. 6. Plots of fairness metric.

D. Voice Connections As Cross Traffic

In this subsection, cross traffic consists of 1000 identical voice connections. Each voice connection generates fluid traffic according to a two-state MMF model whose parameters are as given in the last subsection. The parameters of the HO and the PD controllers also stay the same. Table 1 summarizes the performance comparison between the HO and the PD controllers. In this experiment, the rate variance of the cross traffic is small (1.2 $Mbps^2$), and therefore the PD controllers achieve good utilizations. However, the HO controller, while maintaining the highest utilization value among the competing controllers, enjoys a much smaller queuing delay and cell loss rate.

Table 1: Performance comparison using 1000 voice connections as cross traffic

Controller	Util	Delay (ms)	CLR	UF
HO	0.988	0.078	0.00	0.0089
PD-50	0.976	0.828	2.39e-3	0.0013
PD-10	0.949	0.238	2.82e-4	0.0013
PD-1	0.896	0.087	6.22e-5	0.0013

Util = Utilization CLR = Cell Loss Rate UF = Unfairness

V. Conclusions

We have introduced an online sampling-based congestion controller to regulate best-effort traffic to achieve high network efficiency. We have demonstrated that exploiting the structure of service-rate models can result in significantly improved network performance.

While the proposed control scheme is promising, two main issues remain to be addressed:

1) Our hindsight-optimization framework is founded on a crisp and powerful decision-theoretic formulation, but little is understood on conditions under which the technique works well.

2) To incorporate a long-range-dependence traffic model into our control scheme is an interesting direction worth pursuing. Such a model can be made Markovian but with a potentially large state space. Managing the size of the state space but still capturing the long-range dependence is important for our approach to apply in this case.

References

[1] A. Anick, D. Mitra, and M. Sondhi, "Stochastic theory of a data handling system with multiple sources," *Bell System Technical J.*, vol. 61, no. 8, pp. 1871–1894, 1982.

[2] N. B. Shroff and M. Schwartz, "Improved loss calculations at an ATM multiplexer," *IEEE/ACM Trans. Networking*, vol. 6, no. 4, pp. 411–421, Aug. 1998.

[3] L. A. Kulkarni and S.-Q. Li, " Performance analysis of a rate-based feedback control scheme," *IEEE/ACM Trans. Networking*, vol. 6, no. 6, pp. 797–810, Dec. 1998.

[4] R. Pazhyannur and R. Agrawal, "Feedback-based flow control of B-ISDN/ATM networks," *IEEE J. Select. Areas Commun.*, vol. 13, no. 7, pp. 1252–1266, Sep. 1995.

[5] E. K. P. Chong, R. L. Givan, and H. S. Chang, "A framework for simulation-based network control via Hindsight Optimization," *39th IEEE Conf. on Decision and Control*, Dec. 2000.

[6] D. P. Bertsekas, *Dynamic Programming and Optimal Control, Volumes 1 and 2*, Athena Scientific, 1995.

[7] K. Ramakrishnan and R. Jain, "A binary feedback scheme for congestion avoidance in computer networks," *ACM Trans. Comput. Syst.*, vol. 8, pp. 158-181, 1990.

[8] L. Roberts, "Enhanced proportional rate control algorithm (PRCA)," in ATM Forum/9494-0735R1, Aug. 1994.

[9] L. Benmohamed and S. M. Meerkov, "Feedback control of congestion in packet switching networks: The case of a single congested node," *IEEE/ACM Trans. Networking*, vol. 1, no. 6, pp. 693–707, Dec. 1993.

[10] A. Kolarov and G. Ramamurthy, "A control-theoretic approach to the design of an explicit rate controller for ABR service," *IEEE/ACM Trans. Networking*, vol. 7, no. 5, pp. 741–753, Oct. 1999.

[11] E. Altman and T. Basar, "Multiuser rate-based flow control," *IEEE Trans. Commun.*, vol. 46, no. 7, Jul. 1998.

[12] E. Altman, T. Basar, and R. Srikant, "Robust rate control for ABR services," in *Proc. IEEE INFOCOM*, Mar. 1998, pp. 166–173.

[13] Z. Pan, E. Altman, and T. Basar, "Robust adaptive flow control in high speed telecommunications networks," in *Proc. 35th IEEE Conf. on Decision and Control*, Dec. 1996, pp. 1341–1346.

[14] S. Floyd and V. Jacobson, "Link-sharing and resource management models for packet networks," *IEEE/ACM Trans. Networking*, vol. 3, no. 4, pp. 365–386, Aug. 1995.

[15] T. Tuan and K. Park, "Multiple time scale congestion control for self-similar network traffic," to appear in *Performance Evaluation*, 1999.

[16] E. K. P. Chong and S. H. Zak, *An Introduction to Optimization*, John Wiley and Sons, New York, 1996.

[17] S. C. Liew and D. C. Tse, "A control-theoretic approach to adapting VBR compressed video for transport over a CBR communication channel," *IEEE/ACM Trans. Networking*, vol. 6, no. 1, pp. 42–55, Feb. 1998.

[18] I. Rubin and K. D. Lin, "A burst-level adaptive input-rate flow control scheme for ATM networks," in *Proc. IEEE INFOCOM*, 1993, vol. 2, pp. 386–394.

[19] D. Q. Mayne and H. Michalska, "Receding horizon control of nonlinear systems," *IEEE Trans. Auto. Contr.*, vol. 35, no. 7, pp. 814–824, Jul. 1990.

[20] J. B. Rawlings and K. R. Muske, "The stability of constrained receding horizon control," *IEEE Trans. Auto. Contr.*, vol. 38, no. 10, pp. 1512–1516, Oct. 1993.

[21] J. M. Jaffe, "Bottleneck flow control," *IEEE Trans. Commun.*, vol. 29, no. 7, pp. 954–962, Jul. 1981.

[22] F. Bonomi and K. Fendick, "The rate-based flow control framework for the Available Bit Rate ATM service," *IEEE Network*, vol. 9, no. 2, pp. 25-39, Mar./Apr. 1995.

[23] K. Bharath-Kumar and J. M. Jaffe, "A new approach to performance-oriented flow," *IEEE Trans. Commun.*, vol. 29, no. 4, pp. 427–435, Apr. 1981.

[24] V. M. Misra and W. B. Gong, "A hierarchical model for teletraffic," in *Proc. 37th IEEE Conf. Decision and Control*, 1998, vol. 2, pp. 1674–1679.

[25] R. Jain, "Congestion control in computer networks: Issues and trends," *IEEE Network*, vol. 4, no. 3, pp. 24–30, May 1990.

[26] R. Pazhyannur and R. Agrawal, "Feedback based flow control in ATM networks with multiple propagation delays," in: *Proc. IEEE INFOCOM*, 1996, vol. 2, pp. 585–593.

[27] Y.-C. Ho and X.-R. Cao, *Perturbation Analysis of Discrete Event Dynamic Systems*, Kluwer Academic Publishers, Boston, 1991.

[28] L. R. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, *Proc. of the IEEE*, vol. 77, no. 2, pp. 257–285, Feb. 1989.

[29] M. L. Littman, *Algorithms for Sequential Decision Making*, Ph.D. Thesis, Department of Computer Science, Brown University, 1996.

[30] I. Lengliz and F. Kamoun "A rate-based flow control method for ABR service in ATM networks," *Computer Networks*, vol. 34, no. 1, pp. 129–138, Jul. 2000.

[31] D. Mitra and J. B. Seery, "Dynamic adaptive windows for high speed data networks with multiple paths and propagation delays," *Computer Networks and ISDN Systems*, vol. 25, no. 6, pp. 663-679, Jan. 1993.

[32] E. K. P. Chong, S. Hui, and S. H. Zak, "An Analysis of a Class of Neural Networks for Solving Linear Programming Problems," *IEEE Trans. Auto. Contr.*, vol. 44, no. 11, pp. 1995–2006, Nov. 1999.

[33] P. Kermani and L. Kleinrock, "Dynamic flow control in store-and-forward computer networks," *IEEE Trans. Commun.*, vol. COM-28, no. 2, pp. 263–271, Feb. 1980.

[34] E. Altman and P. Nain, "Closed-loop control with delayed information," *Performance Evaluation Review*, vol. 20, no. 1, pp. 193–204, Jun. 1992.

[35] J. Kuri and A. Kumar, "Optimal control of arrivals to queues with delayed queue length information," *IEEE Trans. Auto. Contr.* , vol. 40, no. 8, pp. 1444–1450, Aug. 1995.

[36] D. P. Bertsekas and D. A. Castanon, "Rollout algorithms for stochastic scheduling problems," *J. of Heuristics* , vol. 5,pp. 89–108, 1999.