TECHNICAL RESEARCH REPORT

Dynamic Resource Allocation of GPS Queues with Leaky Buckets

by Peerapol Tinnakornsrisuphap, Sarut Vanichpun, Richard J. La

CSHCN TR 2003-24
(ISR TR 2003-48)

The Center for Satellite and Hybrid Communication Networks is a NASA-sponsored Commercial Space Center also supported by the Department of Defense (DOD), industry, the State of Maryland, the University of Maryland and the Institute for Systems Research. This document is a technical report in the CSHCN series originating at the University of Maryland.

Web site http://www.isr.umd.edu/CSHCN/

Dynamic Resource Allocation of GPS Queues with Leaky Buckets

Peerapol Tinnakornsrisuphap, Sarut Vanichpun and Richard J. La
Department of Electrical and Computer Engineering
and Institute for Systems Research
University of Maryland, College Park, MD, 20742
{peerapol,sarut,hyongla}@eng.umd.edu

Abstract

We study the problem of dynamic resource allocation of a GPS server with two traffic classes when the leaky bucket scheme is employed as a traffic policing mechanism. Three popular input traffic models – independent Poisson arrival, autoregressive model, and partially observed traffic (Hidden Markov Model) – are investigated in this paper.

Theoretically, the optimal control can be obtained by a basic dynamic programming algorithm. However, such a solution is computationally prohibitive due to the *curse of dimensionality*. Instead, we propose several heuristic policies with improvements using rollout, parallel rollout, and hindsight optimization techniques under the aforementioned traffic models and show that these techniques can significantly reduce the penalty associated with the delay and dropped packets.

1 Introduction

Communication networks technologies have been evolving rapidly to satisfy the varying demands of new applications. Some emerging applications require that network service providers provide "quality-of-service" (QoS) in the form of delay and/or throughput guarantees. These QoS guarantees can be interpreted as a mutual agreement between the service provider and users.[1] Such an agreement is envisioned to require that a user adhere to an agreed traffic profile monitored through a traffic shaper, while the service provider guarantees certain throughput level, maximum delay, and/or packet loss rate. The problem of designing an efficient scheduling algorithm capable of providing such QoS guarantees can be modeled as a stochastic optimization problem, where the service provider tries to minimize the total price/penalty it needs to pay for packet delays and/or packet losses in violation of the QoS. In this paper we adopt a popular traffic policing mechanism called *leaky bucket* flow control scheme for shaping incoming traffic [4].

[1]Here a user can in fact be a domain that needs to buy a service level agreement (SLA) from another domain to carry its traffic.

In order to minimize the penalty, the network needs to decide the amount of bandwidth assigned to each traffic source. The bandwidth allocation should be dynamically adjusted based on network state and traffic profiles. Ideally, this dynamic bandwidth allocation can be viewed as a dynamic weight allocation in the *generalized processor sharing* (GPS) server with multiclass users, where the input traffic for each class is policed by a leaky bucket flow controller. GPS can be thought of as an idealized version of a commonly-used scheduling algorithm for high-speed switches such as Weighted Fair Queueing [10, 11].

While it is possible to write the state and cost functions, and pose the problem of adjusting the weights of the GPS server with leaky bucket controllers as a classical stochastic dynamic programming (DP) problem, the problem suffers from the *curse of dimensionality* because the size of the state space explodes as the time horizon and the capacity of the server increase. Instead of attempting to compute the optimal policy through DP we study the performance of several heuristic policies with improvements using rollout, parallel rollout, and hindsight optimization under various scenarios.

In this paper, we first describe the model in Section 2, followed by problem formulation in Section 3. A brief description of hindsight optimization, rollout and parallel rollout is given in Section 4 along with that of some heuristic policies to be used as a base policy for rollout. The experimental results are presented in Section 5. Concluding remarks and future work are presented in Section 6.

2 The model

Consider the problem of two traffic classes sharing a GPS server under the assumption that each traffic stream is constrained by a leaky-bucket flow controller before arriving at the GPS server. For each $i = 0, 1$, let $\{x_{i,k}, k = 0, 1, \ldots\}$ denote the integer-valued random traffic process i, i.e., $x_{i,k}$ denotes the number of packets that arrive at the beginning of timeslot k from traffic source i. Assume that the GPS server has a capacity of C packets per timeslot where C is a positive integer. At the beginning of timeslot k the GPS server has to allocate $\phi_{i,k} \in \{0, 1, \ldots, C\}$ to class i queue. Here $\phi_{i,k}$ represent the minimum guaranteed number of packets that can be transmitted from class i queue in timeslot k. We assume $\phi_{0,k} + \phi_{1,k} = C$. Our objective is to find an optimal sequence $\{\phi_{0,k}, k = 0, 1, \ldots\}$ that minimizes a given cost function which will be specified later. We use the following notation throughout the paper. For any $x, a, b \in \mathbb{R}$, we denote $[x]_a^b = \max(\min(x, b), a)$ and $[x]^+ = \max(x, 0)$. A vector $[x_{0,k},\ x_{1,k}]'$ is denoted by x_k.

2.1 Leaky bucket (σ, ρ)

A (σ, ρ) leaky bucket flow controller is a traffic shaper that works as follows: When a packet arrives, the packet is allowed into the network only if a token is available in the leaky bucket. Tokens are generated at a constant rate of ρ tokens/timeslot. The leaky bucket is allowed to store up to σ tokens in a token bucket, and tokens generated when the token bucket has σ tokens are discarded. A packet that finds the token bucket empty must wait till a token becomes available before entering the network. The number of packet arrivals into the network over any period of duration M timeslots is constrained by $\sigma + M \cdot \rho$ under a leaky

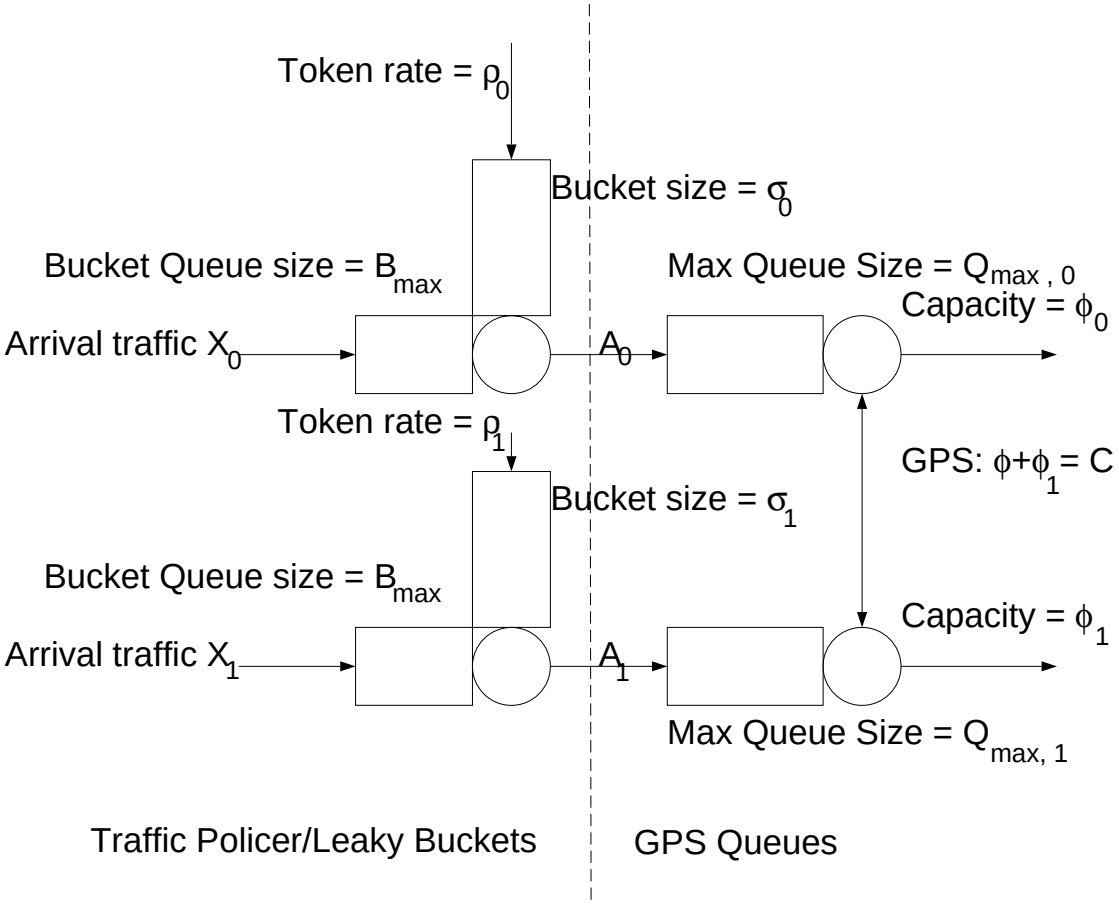

Figure 1: The model

bucket with parameters (σ, ρ).

Let $T_{i,k}, i = 0, 1$, be the number of tokens available at the beginning of timeslot k and $B_{i,k}$ be the number of packets queued in the leaky bucket queue at the beginning of timeslot k. Assume that the leaky bucket queue has a capacity of B_{max} packets. The value of B_{max} is typically small because the traffic arrival to a leaky bucket is supposed to conform to the traffic arrival profile and the network is not penalized for any packets dropped from this queue.

With the leaky bucket (σ_i, ρ_i) scheme, the evolution of $T_{i,k}$ and $B_{i,k}$ is given by

$$\begin{aligned} T_{i,k+1} &= [T_{i,k} + \rho_i - (B_{i,k} + x_{i,k})]_0^{\sigma_i}, \\ B_{i,k+1} &= [B_{i,k} + x_{i,k} - A_{i,k}]_0^{B_{max}} \end{aligned}$$

where $A_{i,k}$ is the output process of the leaky bucket i, i.e., the number of packets leaving the leaky bucket i at the beginning of timeslot k and is given by

$$A_{i,k} = \min(T_{i,k} + \rho_i, B_{i,k} + x_{i,k}) \quad k = 0, 1, \ldots \tag{1}$$

2.2 GPS queue

Let $Q_{i,k},\ i = 0, 1$, denote the queue size of class i traffic at the GPS server at the beginning of timeslot k. Since class i traffic is first passed through the leaky bucket i, the arrival process at the GPS server is given by $\{A_{i,k}, k = 0, 1, \ldots\}$. The server will then transmit as many of the available packets in each class as allowed by $\{\phi_{i,k}, k = 0, 1, \ldots\}$. However, if $Q_{i,k} + A_{i,k} < \phi_{i,k}$, then the other traffic class $1 - i$ can use the remaining capacity, i.e., class $1 - i$ can transmit up to $C - Q_{i,k} - A_{i,k}$ packets in the timeslot. Any unused capacity in the timeslot will be wasted.

Assumption 1. *For any sequence $\{\phi_{0,k}, k = 0, 1, \ldots\}$ such that there exists an integer m and $i \in \{0, 1\}$ that $\phi_{i,m} > Q_{i,m} + A_{i,m}$ and $\phi_{1-i,m} < Q_{1-i,m} + A_{1-i,m}$, there exists a sequence $\{\phi_{i,0}, \phi_{i,1}, \ldots, \phi_{i,m-1}, Q_{i,m} + A_{i,m}, \phi_{i,m+1}, \ldots\}$ for class i traffic which yields the same or lower cost.*

The above assumption is natural in the sense that the server should not assign more bandwidth to one class than needed while the other class has more packets to transmit. This assumption also allows us to capture the evolution of $Q_{i,k}$ by

$$Q_{i,k+1} = [Q_{i,k} + A_{i,k} - \phi_{i,k}]_0^{Q_{max,i}}, \quad k = 0, 1, \ldots, \tag{2}$$

where $Q_{max,i}$ is the maximum queue size of class i traffic. This assumption is true for a large class of cost per stage functions.

2.3 Cost per stage function

The performance of this system can be measured by delay and loss of the packets. The delay cost of class i in timeslot k, is proportional to $Q_{i,k}/\phi_{i,k}$, which is the *expected* number of timeslots needed to empty the current queue of class i traffic, given that the transmission

rate is $\phi_{i,k}$. We also incur a heavy penalty for each packet loss. The number of dropped packets from class i in times slot k equals $[Q_{i,k} + A_{i,k} - \phi_{i,k} - Q_{max,i}]^+$. The cost per stage $h(Q_{0,k}, Q_{1,k}, \phi_{0,k}, \phi_{1,k}, A_{0,k}, A_{1,k})$ is defined to be

$$h(Q_{0,k}, Q_{1,k}, \phi_{0,k}, \phi_{1,k}, A_{0,k}, A_{1,k}) = \sum_{i=0}^{1} (f_i(Q_{i,k}, \phi_{i,k}) + K_i[Q_{i,k} + A_{i,k} - \phi_{i,k} - Q_{max,i}]^+) \quad (3)$$

where $K_i > 0$ is the penalty per each dropped packet from class i and $f : \mathbb{Z}_+^2 \to \mathbb{R}_+$ is the delay penalty function. We now prove that Assumption 1 is true for a certain class of cost functions.

Lemma 2. *Consider a two-class* GPS *server described earlier. Assumption 1 holds for the cost function (3) with the delay penalty function $f_i : \mathbb{Z}_+^2 \to \mathbb{R}_+$, $i = 0, 1$, given by*

$$f_i(Q_i, \phi_i) = \begin{cases} g_i(\max(\frac{Q_i}{\phi_i}, 1)) & \text{if } Q_i > 0 \\ 0 & \text{if } Q_i = 0 \end{cases}, \quad (4)$$

where $g_i : \mathbb{R}_+ \to \mathbb{R}_+$ is an increasing function.

Proof. Assume the condition in Assumption 1, i.e., there exists $i \in \{0, 1\}$ and an integer m such that a sequence $\phi_{\mathbf{i}} = \{\phi_{i,k}, k = 0, 1, \ldots\}$ that $\phi_{i,m} > Q_{i,m} + A_{i,m}$ and $\phi_{1-i,m} < Q_{1-i,m} + A_{1-i,m}$. Consider another sequence $\phi'_{\mathbf{i}} = \{\phi_{i,0}, \phi_{i,1}, \ldots, \phi_{i,m-1}, Q_{i,m} + A_{i,m}, \phi_{i,m+1}, \ldots\}$. We see that up to timeslot $m-1$, both sequences have the same cost and both state spaces are identical up to timeslot m. At timeslot m, we have from the assumption that $\phi_{i,m} > Q_{i,m} + A_{i,m}$, which results in $Q_{i,m}/\phi_{i,m} < 1$ and so the delay cost for the control sequence $\phi_{\mathbf{i}}$ at this stage is $g_i(1) + g_{1-i}(Q_{1-i,m}/\phi_{1-i,m})$ while the cost per stage of $\phi'_{\mathbf{i}}$ is $g_i(1) + g_{1-i}(Q_{1-i,m}/(C - Q_{i,m} - A_{i,m}))$ which is always no greater because $C - Q_{i,m} - A_{i,m} > C - \phi_{i,m} = \phi_{1-i,m}$ and g_i is increasing. Since $\phi_{i,m} > Q_{i,m} + A_{i,m}$, we have $Q_{i,m+1} = 0$ and the remaining capacity $C - Q_{i,m} - A_{i,m}$ will all be utilized by the queue $1-i$. Therefore, $Q_{1-i,m+1} = [Q_{1-i,m} - (C - Q_{i,m} - A_{i,m}) + A_{1-i,m}]_0^{Q_{max}}$ for either the control sequence $\phi_{\mathbf{i}}$ and $\phi'_{\mathbf{i}}$ for class i queue. This indeed concludes the proof and hence the model described by (2) is justified. □

In this paper, for $i = 0, 1$, we consider the delay penalty function, $f_i : \mathbb{Z}_+^2 \to \mathbb{R}_+$, given by

$$f_i(Q_i, \phi_i) = \begin{cases} a_i \frac{Q_i}{\phi_i} & \text{if } \phi_i > 0 \\ M & \text{if } \phi_i = 0 \text{ and } Q_i > 0 \\ 0 & \text{if } \phi_i = 0 \text{ and } Q_i = 0 \end{cases} \quad (5)$$

where $a_i > 0$ and $M > \max_{i=0,1}\{a_i Q_{max,i}\}$ is an upper bound on the cost when $\phi_i = 0$ but $Q_i > 0$. This function approximates the delay penalty function in Lemma 2 and we assume Assumption 1 holds for this function.

3 Problem formulation

We formulate here the finite horizon dynamic programming problem. By letting N be the final timeslot, the goal is to find the optimal policy $\pi = \{\phi_{0,k}, k = 0, \ldots, N-1\}$ where $\phi_{0,k} \in \{0, \ldots, C\}$ such that it minimizes the total cost

$$\mathbf{E}\left[\sum_{k=0}^{N-1} h(Q_{0,k}, Q_{1,k}, \phi_{0,k}, C - \phi_{0,k}, A_{0,k}, A_{1,k})\right]. \tag{6}$$

3.1 Traffic models

3.1.1 Independent traffic model

In this case, for each $i = 0, 1$, the rvs $\{x_{i,k}, k = 0, 1, \ldots\}$ are independent. Thus, the state of the system at each time k has a fixed dimension and is given by $y_k = [Q'_k\ T'_k\ B'_k]'$ and $y_{k+1} = F(y_k, x_k, \phi_k)$ where F is an appropriate map.

In a more realistic traffic model, for each $i = 0, 1$, the rvs $\{x_{i,k}, k = 0, 1, \ldots\}$ are correlated. It is possible that either the number of states is fixed for each timestep k or is increasing as k grows large. As a consequence, two problems arise. First, *the curse of dimensionality* leads to the state space that grows larger as the correlation horizon of the input traffic process increases. Second, since the input process may not be known exactly, we may have imperfect state information. The correlated traffic models with perfect and imperfect state information considered in this paper are Autoregressive (AR) models and Hidden Markov models (HMM), respectively.

3.1.2 Autoregressive model

For each $i = 0, 1$, let $\{x_{i,k}, k = 0, 1, \ldots\}$ be the m-step autoregressive process, i.e.,

$$x_{i,k+1} = \sum_{l=0}^{m-1} \alpha_{i,l} x_{i,k-l} + w_{i,k}, \tag{7}$$

where the rvs $\{w_{i,k}, k = 0, 1, \ldots\}$ are independent. For this system, we assume the perfect state information where, at time k, the past values of $x_{i,k-1}, \ldots, x_{i,0}$ are known for each $i = 0, 1$. If we define the state variable for this system to be $y_k = [Q'_k\ T'_k\ B'_k\ x_{k-1}\ \ldots x_{k-m}]'$, this problem reduces to the basic problem with independent noise $\{w_{i,k}, k = 0, 1, \ldots\}$.

3.1.3 Partially observed traffic model

It is well-known that HMM can capture a variety of interesting input traffic processes, and is widely used in network traffic modeling [1, 9]. Under the HMM traffic, the problem is now one of imperfect state information.

For each source $i = 0, 1$, the HMM traffic has a finite set of states Δ_i, where each state s in Δ_i is associated with a packet arrival distribution G_i^s over $\mathbb{Z}_+$ and a next state transition probability F_i^s over Δ_i, i.e., a state s in Δ_i generates n packets with probability $G_i^s(n)$ and then moves to state s' with probability $F_i^s(s')$.

At each timeslot k, the system can estimate the probability distribution of the belief state as follows: For all $s \in \Delta_i$, let $\Pi_{i,k}(s)$ be a probability estimate that the actual state is s at timeslot k. Given n_i packet arrivals from source i in timeslot k, we update the distribution $\Pi_{i,k+1}$ by applying Bayes' rule, i.e., $\Pi_{i,k+1}(s) = \alpha \sum_{s' \in \Delta_i} G_i^{s'}(n_i) F_i^{s'}(s) \Pi_{i,k}(s')$, where α is a normalizing factor so that $\{\Pi_{i,k+1}(s)\}_{s \in \Delta_i}$ is a probability distribution. Now $\Pi_{i,k} = \{\Pi_{i,k}(s)\}_{s \in \Delta_i}$ can be used to augment the state variable to be $y_k = [Q_k' \; T_k' \; B_k' \; \Pi_k']'$, and the problem reverts back to the basic dynamic programming formulation.

4 Policy selection

A naive approach to our problem is to directly apply the DP algorithm. However, as mentioned earlier, a straightforward DP approach is computationally prohibitive. Instead, we propose some heuristic policies for this problem and improve them using rollout and parallel rollout. These heuristic policies should be optimal in some regions of the state space in order for the rollout or parallel rollout policies to perform well. Also, we consider the hindsight optimization technique for this problem as well.

The basic problem in dynamic programming can be outline as follows [3]: Consider the dynamic system $y_{k+1} = f(y_k, u_k, w_k)$ for $k = 0, \ldots, N-1$ where at each time k, $y_k \in \mathcal{S}$, u_k is the control to be chosen from the nonempty subset $U(y_k)$ of a control space $\mathcal{C}$, and w_k is a random disturbance. The rvs $\{w_k, k = 0, 1, \ldots\}$ are independent. If $g(y_k, u_k, w_k)$ represents the cost at timestep k, we define the total cost under policy $\pi = \{\mu_0, \ldots, \mu_{N-1}\}$, where $\mu_k(y_k) \in U(y_k)$, as

$$J_N^\pi(y_0) = \mathbf{E}\left[\sum_{k=0}^{N-1} g(y_k, \mu_k(y_k), w_k)\right] \tag{8}$$

for each initial state y_0. The objective is to find the optimal policy π^* that minimize the cost (8).

Moreover, we define for each $i = 0, \ldots, N-1$,

$$J_{N-i}^\pi(y_i) = \mathbf{E}\left[\sum_{k=i}^{N-1} g(y_k, \mu_k(y_k), w_k)\right] \quad \text{and}$$

$$J_{N-i}^*(y_i) = \min_\pi (J_{N-i}^\pi(y_i)).$$

The DP algorithm is then given by

$$\begin{aligned} J_0^*(y_N) &= 0 \\ J_{N-i}^*(y_i) &= \min_{u \in U(y_i)} Q_{N-i}(y_i, u), \quad i = 0, \ldots, N-1. \end{aligned}$$

where we have set

$$Q_{N-i}(y_i, u) = \mathbf{E}\left[g(y_i, u, w_i) + J_{N-i-1}^*(f(y_i, u, w_i))\right] \tag{9}$$

to be the *Q-value of control u at y_i for horizon $N-i$*. To allow various approximation in suboptimal controls, we also define

$$Q^J_{N-i}(y_i, u) = \mathbf{E}\left[g(y_i, u, w_i) + J(f(y_i, u, w_i))\right] \tag{10}$$

for any function $J : \mathcal{S} \to \mathbb{R}$. We are now ready to introduce suboptimal controls used in this paper.

4.1 Hindsight optimization

The hindsight optimization approach has been proposed by Chang et. al. [5] (see also [8]). The idea is to find a lower bound on the true Q-value of each control by interchanging the order of expectations and minimizations in (9) for every timestep, i.e.,

$$\hat{Q}_{H_s}(y_0, u_0) = \mathbf{E}\left[g(y_0, u_0, w_0) + \min_{u_1,\ldots,u_{H_s-1}} \sum_{k=1}^{H_s-1} g(y_k, u_k, w_k)\right]$$

where $H_s << N$ is the sampling horizon. By Jensen's inequality, it is easy to see that $\hat{Q}_{H_s}(y_0, u_0) \leq Q_{H_s}(y_0, u_0)$.

If the disturbance sequence $\{w_0, \ldots, w_{H_s-1}\}$ is known, then the minimization in the above equation is simply the deterministic optimization problem and the best sequence of controls $\{u_1, \ldots, u_{H_s-1}\}$ can be chosen after taking action u_0 ("in hindsight"). In some problems, the deterministic hindsight optimization can be solve analytically and can be used to find the value of $\hat{Q}_{H_s}$.

However, in a more complicated problem or in an online fashion, the $\hat{Q}_{H_s}$-value can be estimated using the Monte Carlo simulation. Formally, at timestep k, the hindsight optimization policy selects the control $u_k = \arg\min_{u \in U(y)} \hat{Q}_{H_s}(y, u)$. For a given current state y and every $u \in U(y)$, the $\hat{Q}_{H_s}$-value is computed as follows: First, we generate a number of sample paths $W = \{w_0, \ldots, w_{H_s-1}\}$ and find the $\hat{Q}_{H_s,W}$-value from the deterministic optimization problem. The estimate $\hat{Q}_{H_s}$-value is obtained by averaging over all sample paths.

4.2 Rollout

This is a heuristic method of policy improvement using sampling introduced by Bertsekas [2, 3]. Rollout using sampling is designed to improve a heuristic policy in an online fashion. At each timestep $0 \leq k \leq N-1$, given that the current state is y, the rollout policy with a base policy π selects the control $u_k = \arg\max_{u \in U(y)} Q^{J^\pi}_{H_s}(y, u)$ where

$$Q^{J^\pi}_{H_s}(y, u) = \mathbf{E}\left[g(y, u, w) + J^\pi_{H_s-1}(f(y, u, w))\right].$$

The function $J^\pi_{H_s-1}(f(y, u, w))$ is estimated by using sampling where $H_s << N$ is the sampling horizon and $Q^{J^\pi}_{H_s}(y, u)$ can be computed by averaging over all sample paths.

It has been shown that the cost of using rollout policy with a base policy π is lower than the cost of using the base policy π [3]. However, there are many candidates for the base policy and the choice of the base policy can greatly affect the performance.

4.3 Parallel rollout

Chang, Givan and Chong [6] propose a new technique which generalizes the rollout policy by considering a set of base policies instead of one base policy.

Given a set of base policies Π, at timestep k, the parallel rollout approach selects the control $u_k = \arg\max_{u \in U(y)} Q^V_{H_s}(y,u)$ where

$$Q^V(y,u) = \mathbf{E}\left[g(y,u,w) + V_{H_s-1}(f(y,u,w))\right]$$

and

$$V_{H_s-1}(f(y,u,w)) = \min_{\pi \in \Pi} J^\pi_{H_s-1}(f(y,u,w)).$$

For a given current state y and each control $u \in U(y)$, the Q^V-value can be estimated as follows: First, we generates a number of sample paths of $W = \{w_0, \ldots, w_{H_s-1}\}$. For each sample path, we compute Q^V_W-value by applying the control u at the first stage, then using the simulation to find $J^\pi_{H_s-1,W}(f(y,u,w_0))$ for every policy $\pi \in \Pi$ and taking the minimum of $J^\pi_{H_s-1,W}(f(y,u,w_0))$ over all policies in Π to find $Q^V_W(y,u)$. By averaging over all sample paths, we obtain a good estimate for Q^V-value.

The new policy obtaining from parallel rollout on Π denoted by π_{pr} has been shown [6] to have a better performance than all the base policies $\pi \in \Pi$, i.e., $J^{\pi_{pr}}_N(y) \leq \min_{\pi \in \Pi} J^\pi_N(y)$ for all $y \in \mathcal{S}$.

4.4 Heuristic algorithms

This section presents simple heuristic policies which will be considered as base policies for improvements using rollout and parallel rollout techniques.

4.4.1 Fixed Rate, Constant Rate and Bang-bang policies

The fixed rate a policy is simply a capacity sharing scheme similar to dedicated capacity where the rates allocated to both traffic streams are fixed throughout the horizon, i.e., $\phi_{0,k} = a$ and $\phi_{1,k} = C - a, k = 0, 1, \ldots$ where $a \in \{0, \ldots, C\}$.

The constant rate policy is a special case of the fixed rate policy where the rate assigned to each traffic class corresponds to its relative fluid transmission rate, i.e., the ratio between its average traffic arrival rate and total traffic arrival rate of both classes. With traffic policing mechanism such as leaky-buckets, the long-term average rate is bounded from above by ρ.

The bang-bang policy uses the queue size of the traffic stream to make a decision. If $Q_{0,k} \geq Q_{1,k}$, then $\phi_{0,k} = C - 1$ and $\phi_{1,k} = 1$. Otherwise, $\phi_{0,k} = 1$ and $\phi_{1,k} = C - 1$. This policy tries to maximally reduce the larger queue size.

The bang-bang, constant rate, and the fixed rate a policy for $a = 0, \ldots, C$ form a set of base policies to use with parallel rollout approach.

4.4.2 Square-root of queue size (SqrtQ) policy

Given the delay cost function $f_i(Q_i, \phi_i), i = 0, 1$ in (5), one can find $\phi^\star = (\phi^\star_0, \phi^\star_1) \in \mathbb{R}^2$ that minimizes $f_0(Q_0, \phi_0) + f_1(Q_1, C - \phi_0)$. We summarize this finding in the following claim.

Claim 3. *For the delay cost f_i, $i = 0, 1$ given in (5), the service rate for traffic $i = 0, 1$ which minimizes the total delay cost $\sum_{i=0,1} f_i(Q_i, \phi_i)$ is given by*

$$\phi^\star = \left(\frac{\sqrt{a_0 Q_0}}{\sqrt{a_0 Q_0} + \sqrt{a_1 Q_1}} C, \frac{\sqrt{a_1 Q_1} C}{\sqrt{a_0 Q_0} + \sqrt{a_1 Q_1}} C \right) . \tag{11}$$

with the resulting optimal delay cost

$$\sum_{i=0,1} f_i(Q_i, \phi_i) = \frac{2(\sqrt{a_0 Q_0} + \sqrt{a_1 Q_1})}{C}. \tag{12}$$

Proof. By assuming that $\phi_0, \phi_1 > 0$ and noting that $\phi_1 = C - \phi_0$, the total delay cost can be rewritten as

$$g(\phi_0) = \sum_{i=0,1} f_i(Q_i, \phi_i) = a_0 \frac{Q_0}{\phi_0} + a_1 \frac{Q_1}{C - \phi_0}.$$

By taking the derivative of $g(\phi_0)$ with respect to ϕ_0 and set to 0, we find

$$-a_0 \frac{Q_0}{\phi_0^2} + a_1 \frac{Q_1}{(C - \phi_0)^2} = 0.$$

From the assumption that $0 < \phi_0 < C$, the optimal $\phi_0^\star$ that minimizes the total delay cost is the positive solution of the quadratic equation

$$(a_1 Q_1 - a_0 Q_0)(\phi_0^\star)^2 + 2 a_0 Q_0 C \phi_0^\star - a_0 Q_0 C^2 = 0.$$

With a simple calculation, we obtain

$$\phi_0^\star = \frac{\sqrt{a_0 Q_0}}{\sqrt{a_0 Q_0} + \sqrt{a_1 Q_1}} C$$

and

$$\phi_1^\star = C - \phi_0^\star = \frac{\sqrt{a_1 Q_1}}{\sqrt{a_0 Q_0} + \sqrt{a_1 Q_1}} C.$$

Note that the rates $\phi_0^\star$ and $\phi_1^\star$ are positive if both $Q_0 > 0$ and $Q_1 > 0$.

If for some $i = 0, 1$, $Q_i = 0$, then $f_i(Q_i, \phi_i) = 0$ for any ϕ_i. Hence, we need only to minimize $f_{1-i}(Q_{1-i}, \phi_{1-i})$. In this case, the optimal rates that minimize the total delay cost are simply $\phi_i^\star = 0$ and $\phi_{1-i}^\star = C$ which also identify with (11). Lastly, when $Q_0 = Q_1 = 0$, the total delay cost $\sum_{i=0,1} f_i(0, \phi_i) = 0$ for any ϕ_0. By substitute $\phi_0^\star$ and $\phi_1^\star$ in (11) back into the total delay cost function, we obtain (12). □

Lastly, we note that if $K_1 = K_2$, then the SqrtQ policy minimizes the cost per stage h in (3) with the delay cost f (5) in the situation when both classes experience packet losses or neither class experiences a packet loss. In the implementation of this policy, $\phi_0^\star$ is rounded to the nearest integer. Also, if $\max(Q_0, Q_1) = 0$, then we set ϕ_0, ϕ_1 according to the constant rate policy.

4.4.3 Equal weighted delay policy

This policy attempts to equalize the delay of each queue weighted by its cost, i.e.,

$$a_0 \frac{Q_0}{\phi_0} = a_1 \frac{Q_1}{\phi_1} \Leftrightarrow \frac{\phi_0}{\phi_1} = \frac{a_0 Q_0}{a_1 Q_1} \tag{13}$$

In other words, each queue will contribute the same amount to the delay penalty. From the condition $\phi_0 + \phi_1 = C$, if $\max(Q_0, Q_1) > 0$, then we have

$$\phi_0 = \left(\frac{a_0 Q_0}{a_0 Q_0 + a_1 Q_1} \right) C \quad \text{and} \quad \phi_1 = \left(\frac{a_1 Q_1}{a_0 Q_0 + a_1 Q_1} \right) C. \tag{14}$$

Again, we round ϕ_0 to the nearest integer and set ϕ_0 and ϕ_1 according to the constant rate policy if $Q_0 = Q_1 = 0$.

4.4.4 Look ahead policy

For traffic model such as AR, we can also utilize the additional knowledge of the state information in the policy. For instance, this additional information can be used in conjunction with the ϕ_0 in this timeslot to predict the queue level in the next timeslot and to minimize the cost. More specifically, the best estimate of the noise in timeslot k (ignoring non-linearity) is

$$\hat{x}_{i,k+1} = \max \left(\text{round} \left(\sum_{l=0}^{m-1} \alpha_{i,l} x_{i,k-l} + \mathbf{E}[w_{i,k}] \right), 0 \right).$$

Then, an estimate of the queue size $\hat{Q}_{k+1}$ can be computed for any ϕ_0, and the policy that minimizes the total cost can be computed.

4.4.5 Equal weighted packets policy

This policy is designed to use the knowledge of state information in the case of HMM traffic model. Instead of looking at the delay as in Section 4.4.3, we consider the estimated number of packets from each class in the system. For each $i = 0, 1$, the estimated number of packets from traffic i in timeslot $k+1$ is simply the current queue size plus the rate of the current belief state $s_{i,k}$, where the belief state $s_{i,k}$ is computed from

$$s_{i,k} = \arg \max_{s \in \Delta_i} \Pi_{i,k}(s).$$

Assume that the average rate of each state in Δ_i is given. By letting λ_i be the average rate of the belief state $s_{i,k}$, i.e., $\lambda_i = \sum_{n \in \mathbb{Z}_+} n \cdot G_i^{s_{i,k}}(n)$, the equal weighted packets policy is defined as

$$\phi_i = \text{round} \left(\frac{(Q_i + \lambda_i)}{(Q_0 + \lambda_0) + (Q_1 + \lambda_1)} C \right), \quad i = 0, 1. \tag{15}$$

5 Empirical results

5.1 Experimental set-up

We set up a Monte-Carlo simulation of the model described in Section 2 and the cost per stage in (3) with the delay cost in (5). For each simulation run, we use heuristic policies described in Section 4 to adjust the weight assigned to each GPS traffic class. The simulation parameters are fixed throughout all experiments in this section as follows: $\sigma_0 = \sigma_1 = 5$, $\rho_0 = \rho_1 = 5$, $B_{\max} = 5$, $Q_{\max} = 20$, $C = 10$. The delay cost parameters are $a_0 = 2$ and $a_1 = 1$, while the cost for each dropped packet is set to $K_0 = K_1 = 10$ for both classes. The duration of horizon is $N = 1000$. We start the simulation with the initial conditions $Q_{i,0} = T_{i,0} = B_{i,0} = 0, i = 0, 1$.

Three types of arrival traffic processes are considered here, namely, Poisson, autoregressive, and HMM arrival processes as described in Section 3.1. We consider each of these traffic models in turn.

5.2 Poisson arrival process

We first test the arrival traffic according to Poisson processes. Poisson traffic arises naturally in several situations. For example, it is well known that the multiplexed packet traffic generated by a large number of bursty data sources is well described by a Poisson process.

Poisson traffic can be modeled in discrete-time as follows: in each timeslot, the number of packet arrivals in the timeslot is a Poisson random variable (rv). In our case, we consider the case where the number of packet arrivals to leaky bucket $i, i = 0, 1$, in each timeslot are i.i.d. rvs with a constant rate λ_i. We consider several heuristic policies in the simulations:

(a) Constant rate : Since the traffic arrival for both classes are identically distributed, we choose a constant rate $C/2$ for each class.

(b) Bang-bang policy

(c) Equal Weighted Delay

(d) SqrtQ

(e) Rollout-SqrtQ : In this policy, we use SqrtQ as the base policy for rollout.

(f) ParallelRollout-FixedRate : For this policy, we use the fixed rate policy with $a = 0, \ldots, C$ as base policies for parallel rollout.

(g) ParallelRollout-Heuristic : For this policy, we use 'Constant rate', 'Bang-bang', 'Equal Weighted Delay', and 'SqrtQ' as base policies for parallel rollout.

(h) Hindsight Optimization : We use a standard deterministic dynamic programming algorithm to solve the deterministic optimization problem required for hindsight optimization.

Policy	$\lambda = 1$	$\lambda = 2$	$\lambda = 3$	$\lambda = 4$	$\lambda = 5$
Constant	0.5	13.4	121.4	584.3	1998.2
Bang-bang	74.4	211.5	452.4	1319.0	3985.0
Equal Weighted Delay	0.7	68.1	371.5	874.8	1915.5
SqrtQ	0.6	59.5	357.6	835.0	1868.4
Rollout-SqrtQ	17.0	34.6	113.0	443.1	1341.4
ParallelRollout-FixedRate	16.3	29.6	100.0	397.5	1431.7
ParallelRollout-Heuristic	15.8	31.1	95.9	395.8	1419.8
Hindsight	16.7	31.7	101.0	401.7	1745.2

Table 1: Comparison between the average cost of each policy when the traffic arrival in each timeslot is i.i.d. Poisson with mean λ.

Throughout the simulations in Section 5, for all of the rollout and parallel rollout policies, the sampling horizon for the "noise" process is 5 timeslots and the sampling width is 10 sample paths. For the hindsight optimization, we reduce the sampling horizon to 3 timeslots due to the explosion of the state space which causes even the deterministic optimization to be computationally expensive.

The system is simulated for a total of 10 traffic sample paths, and we compute the average cost for each value of $\lambda = 1, \ldots, 5$. The simulation results are shown in Table 1 and Figure 2.

5.2.1 Discussion

From the simulation results, it is clear that when the arrival rate is small, simple heuristic policies such as constant rate perform well due to the fact that the burstiness of the arrival traffic is almost unaffected by the leaky-bucket flow controller and the capacity of the server is relatively high. As the arrival rate increases, simple heuristic policies become less effective. All three rollout policies incur a similar cost and are significantly better than the base policies. Therefore, rollout substantial reduces the cost with a marginal increase in the computational complexity. While hindsight optimization marginally reduces the cost, it incurs exponential increment in computational complexity. Until a more computationally scalable algorithm for solving the deterministic problem can be found similar to [6], hindsight optimization may not be practical.

5.3 Autoregressive arrival process

In the previous section, we consider the case where traffic arrival in each timeslot is Poisson i.i.d. rv which is a very simplistic traffic model. In order to incorporate a more sophisticate traffic arrival into the problem, we introduce AR traffic as described in Section 3.1.2 into the system. We assume that the exact coefficients of the AR process are known and the number of packet arrivals in previous timeslots (at least equal to the number of taps of the AR filter) is also augmented into the system state. We assume the noise $w_{i,k}$ are i.i.d. Gaussian rvs with a mean of 3 packets and standard deviation of 3 packets. Since the number of packet

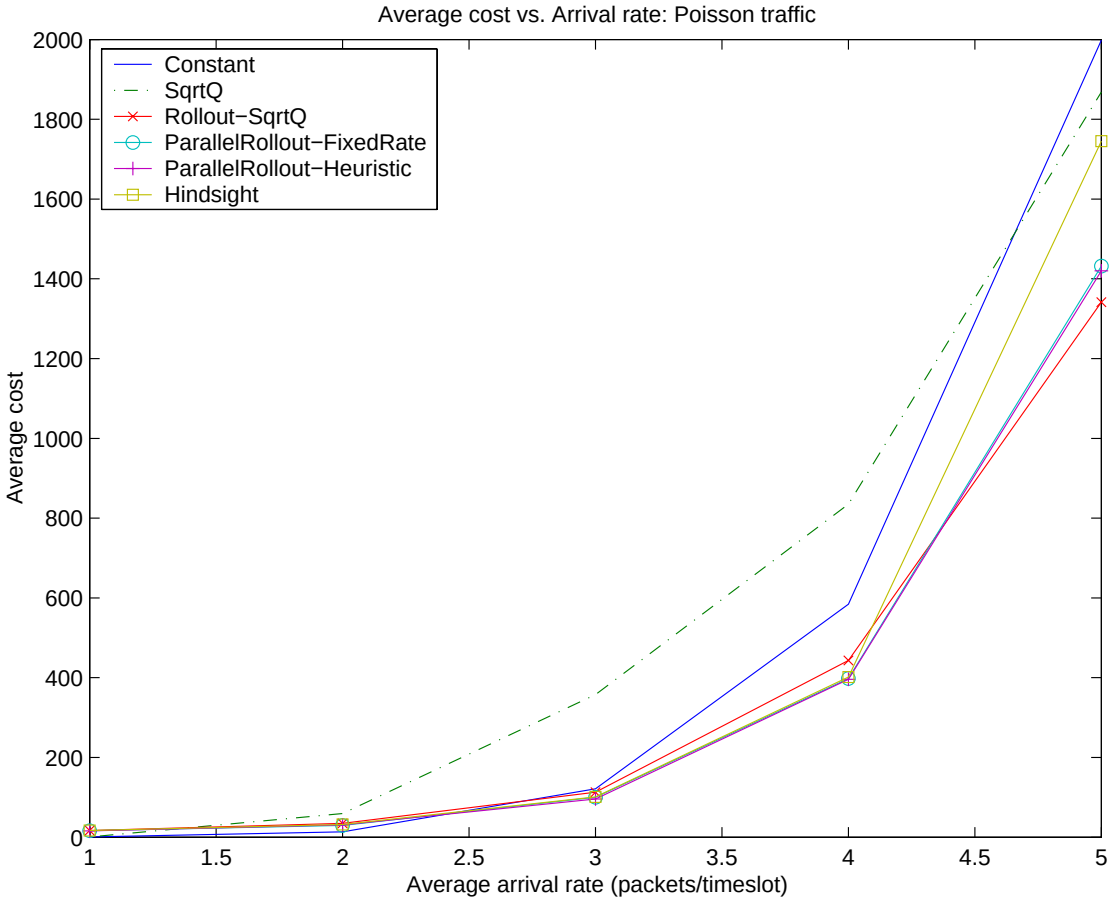

Figure 2: Comparison between different policies for Poisson traffic.

Policy	$(\alpha_0, \alpha_1) = (0.3, 0)$	$(0.3, 0.2)$	$(-0.3, 0)$
Constant	1432.0	2376.3	251.2
Bang-bang	2431.1	4366.2	533.7
Equal Weighted Delay	3137.4	5981.7	368.8
SqrtQ	1116.4	1922.8	336.0
Lookahead	1300.9	2097.1	531.5
Rollout-SqrtQ	912.3	1276.8	374.5
ParallelRollout-FixedRate	1077.3	1676.4	337.6
ParallelRollout-Heuristic	872.0	1341.6	363.3
Hindsight	1040.1	1524.3	395.9

Table 2: Comparison between the average cost (over 5 sample paths) of each policy when the traffic arrival in each timeslot is AR.

arrivals needs to be a non-negative integer, we round $x_{i,k+1}$ in (7) to the nearest integer value. Then, we take the maximum of the resulting value with zero as the number of packet arrivals in timeslot $k+1$.

In addition to all the policies used in Section 5.2, we also study the followings policies for AR models:

(i) Lookahead : This policy exploits the knowledge of state information as discussed in Section 4.4.4

(g') ParallelRollout-Heuristic : In addition to the policies considered before, i.e., 'Constant rate', 'Bang-Bang', 'Equal Weighted Delay', and 'SqrtQ', we also consider 'Lookahead' as a base policy for parallel rollout.

Note that in all of the rollout, parallel rollout, and hindsight optimization policies, the augmented states (i.e., the past packet arrivals and the AR filter coefficients) are also needed to generate the simulated noise sequences. The resulting traffic for three different AR filters are considered in Table 2.

Again, we found a similar conclusion to Section 5.2. We also tested the effect of positive correlations to the cost of the system. The AR filter with $(0.3, 0.2)$ coefficients is more positively correlated than $(0.3, 0)$ and it also incurs much higher cost to the system. However, we note that policies that exploit the information about the AR filter and the past packet arrival patterns experience significantly less increase in the cost. On the other hand, negatively correlated traffic such as $(-0.3, 0)$ typically does not cause the queue to build up. As a consequence, all of the policies perform roughly the same.

5.4 Hidden Markov Model arrival process

In this section, we adopt the HMM arrival processes with the assumption that both traffic classes have the same set of states Δ and same transition probabilities over Δ. To obtain a model that captures the multiple timescales of network traffic behavior, we select a 20-state HMM where each state is associated with a different packet arrival rate. The state space Δ is

Policy	$(l_1, l_2, l_3, l_4) = (1, 2, 4, 5)$	(2,3,5,6)
Constant	684.3	1,027.2
Bang-bang	865.0	1,434.8
Equal Weighted Delay	564.5	798.7
SqrtQ	528.3	808.8
Equal Weighted Packets	329.4	655.3
Rollout-SqrtQ	271.2	465.8
ParallelRollout-FixedRate	280.7	486.4
ParallelRollout-Heuristic	265.8	412.7
Hindsight	270.1	295.4

Table 3: Comparison between the average cost (over 5 sample paths) of each policy when the traffic arrival in each timeslot is HMM.

divided into four different traffic load regions, i.e., $\Delta = \cup_{r=1}^{4} R_r$ and $R_r = \{s_{r,j},\ j = 1, \ldots, 5\}$. The packet arrival distribution for $s_{r,j}$ is a Poisson distribution with parameter $\lambda_{r,j}$, where $\lambda_{r,j} = l_r + U_{r,j}$. Here l_r is a fixed constant for R_r, and $\{U_{r,j}, r = 1, \cdots, 4, j = 1, \cdots, 5\}$ is a collection of i.i.d. [-0.5, 0.5]-uniform rvs.

The transition probabilities $F^s(s')$ of the Markov state are selected at the beginning of simulation and satisfy the following. Suppose that $s = s_{r,j}$. The probability $\sum_{s' \in \Delta \setminus R_r} F^s(s') = 0.1$. Also, $F^s(s) \in [0,\ 0.9]$ and $F^s(s_{r,(j \bmod 5)+1}) = 0.9 - F^s(s)$. Note that the transition within a region forms a circle, which introduces a periodicity in the autocovariance function of the traffic as found in typical video traffic [7].

The policies we consider here are the same as those used in Section 5.2 with some additional policies:

(g") ParallelRollout-Heuristic : In addition to the policies considered in Section 5.2, we also investigate 'Equal Weighted Packets' as a base policy for parallel rollout.

(j) Equal Weighted Packets : By using the state information of HMM models, we can apply this policy with the system as discussed in Section 4.4.5.

In order to provide multiple timescale bursty behavior, we select two HMM models with load level $(l_1, l_2, l_3, l_4) = (1,2,4,5)$ and (2,3,5,6). From the simulation results, we again reach the conclusion that rollout and parallel rollout can significantly reduce the cost compared to the heuristic base policies with a reasonable increase in computational complexity. Moreover, the equal weighted packets policy that takes advantage of the state information has the lowest cost when compared to those of the other heuristic policies. Lastly, we note that in this scenario hindsight optimization performs as well as the rollout and parallel rollout policies eventhough it has a smaller sampling horizon.

6 Conclusions

The research on "quality of service" (QoS) in the networking community has long been considered challenging problems. In this paper, we consider the problem of optimal resource

allocations of the GPS server to two traffic classes when the leaky buckets scheme is employed as a traffic policing at the entrance of the GPS queue. Three traffic models, namely, the independent Poisson arrival, the AR model and the partially observed traffic (HMM) model, are used as the input traffic to the system. From these models, the basic dynamic programming problems are formulated. However, using DP algorithm to compute the optimal policy can be computationally prohibitive. As a consequence, we propose several heuristic policies and investigate three suboptimal controls, namely, rollout, parallel rollout, and hindsight optimization.

For all three traffic models, the performance of simple heuristic policies is inconsistent and sensitive to the system parameters. However, the heuristic policies that exploit the knowledge of the state information perform better than the other simple policies. The rollout, parallel rollout, and hindsight optimization significantly improve the performance of the system. All rollout and parallel rollout policies yield almost similar costs and perform better than the base policies.

The suboptimal controls using rollout, parallel rollout, and hindsight optimization used here significantly improve the performance of the system. All rollout and parallel rollout policies yield almost similar costs and perform better than any of the base policies. However, the ParallelRollout-FixedRate which has the simplest base policies perform poorer than the others. This, indeed, demonstrates the effect of selecting the base policy for rollout.

We conclude that rollout greatly reduces the cost with marginal increases in the computational complexity and it can be implemented online. On the other hand, hindsight optimization gives the same performance as those rollout policies but it incurs exponentially increase in the computational complexity.

The model we considered here contains only two traffic classes with the assumption that both are identically distributed. It is interesting to see how these policies work if both traffic classes are different and their system parameters are different. Furthermore, the effect of system parameters, e.g., Q_{max}, B_{max}, on the performance of the policies should be investigated. In a realistic situation, there are more than 2 traffic classes arriving at the router. Therefore, one basic extension of our problem is to consider the system with N traffic classes. Some of our heuristic policies can be directly applied to this case.

Lastly, this problem can be formulated as the infinite horizon discounted problem where the cost is given by

$$\lim_{N\to\infty} \mathbf{E}\left[\sum_{k=0}^{N-1} \alpha^k h(Q_{0,k}, Q_{1,k}, \phi_{0,k}, C-\phi_{0,k}, A_{0,k}, A_{1,k})\right], \quad 0<\alpha<1.$$

The objective is to find the optimal stationary policy π that minimizes the discounted cost. Our conjecture is that under a simple traffic models, a specific form of the value function can be derived. Moreover, we can study the performance of our policies in the discounted problem.

References

[1] A.T. Anderson, A. Jensen, and B.F. Nielson, "Modelling and performance study of packet-traffic with self-similar characteristics over several time-scales with Markovian

arrival processes (MAP)," in *Proceedings of 12th Mordic Teletraffic Seminar*, pp. 269–283, 1995.

[2] D.P. Bertsekas, "Differential training of rollout policies," in *Proceedings of 35th Allerton Conference on Communication, Control, and Computing*, Allerton Park (IL), 1997.

[3] D.P. Bertsekas, *Dynamic Programming and Optimal Control: 2nd Edition*, Vol. 1, Athena Scientific, Belmont (MA), 2000.

[4] D. Bertsekas, and R. G. Gallagher, *Data Networks, 2nd edition*, Prentice-Hall, 1995.

[5] H.S. Chang, R. Givan, and E.K.P. Chong, "On-line scheduling via sampling," in *Proceedings of the Fifth International Conference on Artificial Intelligence Planning and Scheduling (AIPS2000)*, Brekenridge (CO), April 2000, pp. 62–71.

[6] H.S. Chang, R. Givan, and E.K.P. Chong, "Parallel rollout for online solution of partially observable Markov decision processes," To appear in *Discrete Event Dynamic Systems*. Available at http://www.ece.purdue.edu/~givan/.

[7] D.T. Chen, and M Rieders, "Cyclic Markov modulated Poisson processes in traffic characterization," *Stochastic Models*, vol. 12, no.4, pp. 585–610, 1996.

[8] E.K.P. Chong, R.L. Givan, and H.S. Chang, "A framework for simulation-based network control via hindsight optimization," in *Proceedings of the 39th IEEE Conference on Decision and Control*, Sydney, Australia, December 2000, pp. 1433–1438.

[9] V.M. Misra, and W.B. Gong, "A hierarchical model for teletraffic," in *Proceedings of the 37th IEEE Conference on Decision and Control*, Tampa, Florida, December 1998, pp. 1674–1679.

[10] A.K. Parekh, and R. G. Gallagher, "A generalized processor sharing approach to flow control in integrated services networks: the single-node case," *IEEE/ACM Transaction on Networking*, vol. 1, pp. 344-357, 1993.

[11] A.K. Parekh, and R. G. Gallagher, "A generalized processor sharing approach to flow control in integrated services networks: the multiple node case," *IEEE/ACM Transaction on Networking*, vol. 2, pp. 137-150, 1994.