# Dynamic Congestion-Based Pricing of Bandwidth and Buffer

Nan Jin, *Student Member, IEEE*, Gayathri Venkitachalam, and Scott Jordan, *Member, IEEE*

***Abstract*—We consider pricing of network resources in a reservation-based quality-of-service architecture. The pricing policy implements a distributed resource allocation to provide guaranteed bounds on packet loss and end-to-end delay for real-time applications. Distributed pricing roles are assigned to each user, each network node, and an arbitrager in between the user and the network. When delay constraints are not binding, we investigate two dynamic pricing algorithms using gradient projection and Newton's method to update prices, and prove their convergence. We analyze the performance of the dynamic pricing policies and show that the gradient algorithm using Newton's method converges more quickly and displays only a few small fluctuations. When delay constraints are binding, we investigate subgradient methods which can provide convergence to some range of the optimal allocation.**

***Index Terms*—Congestion-based pricing, dynamic pricing, QoS, resource allocation, utility.**

## I. Introduction

WE ARE concerned in this paper with ensuring quality of service (QoS) to real-time applications in a network with a reservation-based QoS architecture. By *reservation-based QoS architecture*, we include any network architecture that can reserve bandwidth and buffer, whether for a single flow or an aggregate of flows, for the purpose of QoS. Such architectures could potentially include RSVP, MPLS, and ATM. By *real-time applications*, we include any applications whose QoS depends on the amount of reserved bandwidth and buffer. In this paper, we consider packet loss probability and maximum end-to-end delay as the QoS measures.

Common network mechanisms to ensure QoS for real-time applications in a reservation-based QoS architecture include flow control, connection admission control (CAC), and dimensioning. Flow control is usually applied on a time-scale less than or equal to a round trip time, CAC on a time scale greater than a round trip time, and dimensioning on a relatively long time scale. Any such QoS architecture would also include many other elements, potentially including traffic characterization, traffic measurement, application-level QoS characterization, scheduling policies, dropping policies, and traffic smoothing. We focus here on the CAC time-scale, and jointly consider resource reservation and admission control.

Manuscript received December 9, 2003; revised July 4, 2004, and December 2, 2004; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor A. Orda. This work was supported by the National Science Foundation and by the Defense Advanced Research Projects Agency.

The authors are with the Department of Electrical Engineering and Computer Science, University of California, Irvine, CA 92697-2625 USA (e-mail: njin@uci.edu; sjordan@uci.edu).

Digital Object Identifier 10.1109/TNET.2005.861252

Our particular concern is the distributed control of QoS. In a reservation-based QoS architecture, some mechanism must decide what network resources to reserve for each flow (or aggregate of flows). Our premise is that this decision should be based both on how the application values QoS and on congestion in the network. We are looking for a mechanism in which a minimal amount of information is exchanged between the application and the network, and considering how pricing could be used to accomplish this in a distributed fashion.

Pricing has been suggested as a method to signal congestion and distribute the allocation of bandwidth to individual flows or groups of flows [1]–[12]. In this paper, we pose a generic pricing method to control bandwidth and buffer based on differences between demand and supply of each, i.e., what is commonly called congestion-based pricing. We include three aspects that are usually overlooked. First, we consider both the mapping from network resources to QoS and the mapping from QoS to an application's utility, rather than modeling utility directly as a function of network resources. Second, we consider allocation of *both* bandwidth and buffer, based on the realization by many previous researchers that bandwidth and buffer often act as substitutes, in the sense that they can be traded off to obtain the same loss [13], [14]. Third, we allow for the inclusion of multiplexing gains, so that this significant source of network efficiency can be reflected in the prices and therefore encouraged.

The method is generic in that it does not rely upon any specific network architecture, signalling protocol, or traffic characterization, and in that it allows for a broad class of dynamic congestion-based pricing algorithms. It is therefore meant to include a large set of potential congestion-based pricing methods. We do not recommend any particular such method; this would necessarily include consideration of the cost and complexity of the resulting architecture and protocols.

In this paper, we investigate the ability of such congestion-based pricing schemes to allocate bandwidth and buffer and achieve desired levels of loss and delay, with the goal of maximizing the total utility of all users in the network. The paper is organized as follows. In Section II, we review the related literature. We discuss the cases in which delay constraints are not binding and delay constraints are binding separately in Sections III–V and Section VI, respectively. In Section III, each user is modeled as an aggregate of flows with similar traffic characterizations and similar utility functions. Utility is assumed to be a function of loss probability, which in turn depends on the reserved bandwidth and buffer at each node. Our goal is maximize total utility of all users under capacity and delay constraints. We show that when delay constraints are not binding, this optimization problem has a unique solution (under concavity conditions), and that the corresponding shadow costs associated with each resource can be related to user's marginal utilities. In Section IV, we propose an implementation distributed between users and network routers, and prove that an allocation is optimal if and only if all users and routers are in equilibrium. For the network algorithm, two different methods—gradient projection and Newton's method—are discussed, and the convergence of the distributed solution is proven for both cases. In Section V, we propose a second implementation in which we introduce an *arbitrager* layer in between the user and the network. The arbitrager sells QoS (loss and delay) to the users, and purchases from the network the least cost bundle of resources (bandwidth and buffer) that achieves the desired QoS. We prove that an allocation is optimal if and only if all users, arbitragers, and routers are in equilibrium. Finally, considering all three entities involved, we investigate two dynamic pricing policies, in which gradient projection or Newton's method is used to update prices, and present simulation results illustrating the resulting network dynamics. Finally, in Section VI, we investigate the case in which delay constrains are binding. We show that the previously proposed two layer and three layer implementations are still applicable, and prove that if an allocation is optimal, then it must be an equilibrium of the network and users in a two layer implementation, or an equilibrium of the network, users and arbitragers in a three-layer implementation. A subgradient method for the network algorithm is discussed and numerical results are presented.

## II. Related Work

Congestion-based pricing policies typically decide what network resources to reserve for each flow (or aggregate of flows) on the basis of how the application values the network resources (commonly called utility) and on congestion in the network. Congestion, in this context, is typically defined based on the difference between demand for a network resource and its supply. Such pricing approaches therefore typically rely on information from both the application and the network. In our model, we abstract what we believe are the key relationships between these entities. On the application side, we presume the existence of two functions, one which describes each flow (or aggregate of flows), and one which describes how the flow measures satisfaction with its QoS. For the flow characterization, we presume the existence of a function that maps the number of sources in an aggregate and the amount of bandwidth and buffer reserved at each hop into the loss probability that these flows experience. Such a function might be derived using the literature on traffic characterization. Some researchers suggest using multi-parameter models to characterize flows, e.g., Markov-modulated Poisson processes or Markov-modulated fluid flows [15], [16]. Many researchers have suggested using effective bandwidth characterizations [17]–[19]. However, we do not necessarily need to know such a function *a priori*, if we can instead measure the loss experienced, such as often assumed in measurement-based admission control policies [20], [21]. For the QoS satisfaction, we presume the existence of a function that maps the number of sources in an aggregate and the loss probability into a *utility*. This utility can be either interpreted directly as the amount that these sources would be willing to pay for this level of QoS, or indirectly interpreted as a numerical measure of satisfaction. Unfortunately, there is less work in the literature on deriving such functions, as the function would likely be application-specific [22], [23].

On the network side, we presume the existence of a reservation-based QoS architecture. In our approach, we assume the network uses some type of route pinning (i.e., virtual circuit) for real-time applications, e.g., using MPLS or ATM. We also assume the existence of scheduling policies that are capable of assigning bandwidth and buffer to aggregates of flows. Such an architecture would likely specify which flows share resources, and thus determine the multiplexing gains in the network. Research on this topic includes effective bandwidth results that describe multiplexing gains by sharing of bandwidth and buffer [24], [25]. Here we merely presume that such multiplexing gains are incorporated into the flow characterization, whether known or measured.

The mechanism that decides what network resources to reserve for each aggregate should take into account this information from the applications and the network. Congestion-based pricing is often used to signal such information in a distributed fashion with a minimal exchange. There is a significant body of literature on the use of pricing in network operation, in order to accomplish a wide variety of goals. Here, we briefly review some of the pricing literature with goals similar to ours. Early versions of congestion-dependent pricing considered charges per packet. Mackie-Mason [26] introduced a smart market pricing scheme in which each user submits a bid for each packet to transmit. The network transmits all packets whose bid exceeds a cut-off price, which is set to the equilibrium price where demand meets capacity, or to the marginal cost of transmitting one more packet, whichever is applicable. Other proposed congestion-dependent per packet pricing approaches often use an auction to determine the optimal price per packet, resulting in prices that vary with demand. However, per-packet pricing can not easily address any flow-based QoS metrics.

The most common version of congestion-based pricing is to set the price per unit bandwidth according to the marginal cost. Murphy [1] suggested a distributed pricing policy to allocate bandwidth to achieve maximal network efficiency. By incorporating congestion into the cost function, they showed that setting the price equal to the marginal cost results in simultaneous maximization of a network measure and of user surplus. Jiang [2] suggested a distributed pricing policy based on pricing of effective bandwidth, with the goal to maximize total user utility. It was shown that the optimal allocation corresponds to the equilibrium in a distributed process in which users maximize surplus and the network sets prices so that demand equals supply.

Utility and pricing have also been used to express the objective of congestion control schemes. Kelly [8], [9] proposed an optimization framework in which the objective is to maximize the aggregate source utility over their transmission rates. The centralized problem is decomposed into a separate problem for each user, which indicates the willingness-to-pay, and the network, which allocates bandwidths given each user's willingness-to-pay. Many researchers followed Kelly's idea of maximizing aggregate utility and proposed their own variations. Low [10] decomposed the optimization problem such that users choose transmission rates given prices and the network determines the price given the differential between total transmission rate and capacity. In both approaches, the utility function for each user is thought of as determined by the flow control algorithm. La *et al.* [11] investigated a window-based congestion control mechanism where the network adjusts its prices and the users adjust their window sizes such that at an equilibrium the system optimum is achieved. Kar *et al.* [12] proposed algorithms where the network tells the user the number of congested links, while the user adjusts its rate based on its utility function and the network congestion feedback. All of this research concentrates on rate control of elastic traffic in a best-effort network, where utility is expressed as a function of a constant rate, and where QoS of different user classes are not directly considered.

Such approaches can capture QoS metrics such as loss, but do not easily capture delay since buffer is not explicitly modeled. A few congestion-based pricing schemes consider both bandwidth and buffer as resources. In [3], Low proposes bandwidth and buffer as resources to be priced and presents an allocation scheme using burstiness curves which capture the tradeoff between bandwidth and buffer. In [4], Low considers an allocation scheme that provides each user with a fixed minimum and a random extra amount of bandwidth and buffer. Allocations and prices are adjusted to adapt to resource availability and user demands. Keon [5] considers constraints on loss, maximum delay, and blocking, and uses auctions to solve the resulting nonlinear integer programming problem. Each of these approaches rely upon specific traffic and user models, and none includes multiplexing gains.

Finally, some pricing algorithms view the price as the result of a game between users or between the users and the network [6], [7].

These papers point us in the right direction, and we use several of these concepts in our distributed QoS control method. However, the approach presented in this paper includes three aspects that we do not believe have been adequately addressed by this literature. First, we consider both the mapping from network resources to QoS and the mapping from QoS to an application's utility, rather than modeling utility directly as a function of network resources. Second, we consider allocation of *both* bandwidth and buffer. Third, we allow for the inclusion of multiplexing gains.

## III. The Pricing Framework

### A. Network and User Models

The complete model consists of a network model, which describes what type of service the network offers, and a user model, which describes how the user behaves. We presume the existence of a reservation-based QoS architecture using virtual circuits for real-time applications and using scheduling policies that are capable of assigning bandwidth and buffer to aggregates of flows.

Consider a network in which $m$ classes of virtual-circuit real-time traffic reserve network resources on $n$ links. Assume each class can reserve bandwidth on each link it transverses, and buffer at each router it passes through. Specifically, we assume class $j$ can reserve bandwidth $BW_{jl}$ on link $l$, and buffer $BF_{jl}$ at the router just before link $l$. If link $l$ has a total (unidirectional) bandwidth $BW_l$ available to real-time traffic, then the bandwidth reservation allocations must obey $\sum_{j\in\hat{r}_l} BW_{jl} \leq BW_l$, where $\hat{r}_l$ denotes the set of user classes that utilize link $l$. Assume routers are output-buffered and that the total buffer available to real-time traffic with output link $l$ is $BF_l$. The corresponding buffer constraint is therefore $\sum_{j\in\hat{r}_l} BF_{jl} \leq BF_l$. The maximum delay for class $j$ traffic at link $l$ is therefore $D_{jl} = BF_{jl}/BW_{jl}$. We assume that each class can place an upper bound on the total end-to-end delay experienced by its traffic $\sum_{l\in r_j} D_{jl} \leq D_j$ where $r_j$ denotes the set of links that user class $j$ traverses. We define $n_j$ as the number of links on the virtual path for class $j$ traffic. Obviously the total number of links $n \geq \max\{n_1, n_2, \ldots, n_m\}$.

We use the term *user*, or class, to refer to an aggregate of flows with similar traffic characterizations and similar utility functions. The QoS of real-time applications is often considered to be defined by loss, delay, and delay jitter. Here, we assume that these applications require a tight bound on end-to-end delay, but are somewhat elastic with respect to loss. (We do not place a separate bound on delay jitter, beyond that placed on delay.) Utility is thus assumed to be a function of loss probability, which in turn depends on the reserved bandwidth and buffer at each node. The user model consists of two functions: a traffic model, which describes the statistics of each real-time flow aggregate, and a QoS model, which describes how each user measures satisfaction with its QoS.

The traffic model is a function that maps the number of sources in an aggregate and the amount of bandwidth and buffer reserved at each hop into the loss probability that these flows experience. Such a function might be derived using the literature on traffic characterization or measured. Specifically, assume there are $N_j$ independent and identically distributed flows within class $j$. We denote the probability of loss of the multiplexed class on link $l$, $L_{jl}$, as a function of the bandwidth allocated to the class, $BW_{jl}$, the buffer allocated to the class, $BF_{jl}$, and the number of sources $N_j$, as $L_{jl} = g_j(BW_{jl}, BF_{jl}, N_j)$. The loss function for class $j$ traffic, $g_j$, might be given for instance by effective bandwidth results. We presume that this loss function is independent of the link number, which is reasonable if the allocated bandwidth and buffer are sufficient to decouple the effective bandwidths [27]. We also presume that each source within a class experiences the same loss probability. Finally, we presume that the loss function is decreasing, differentiable, and jointly strictly convex in $\{BW_{jl}, BF_{jl}\}$. While some effective bandwidth functions in the literature are not differentiable everywhere convexity has been uniformly empirically found to hold in the literature, including the numerical cases investigated in this paper. In addition, a similar result has been proven for overflow probabilities given by large deviations results in the many source regime [28]. We assume that the loss on each link is small and independent of other losses within the path, given bandwidth and buffer allocations. Therefore, the total end-to-end loss probability for class $j$ is $L_j = \sum_{l\in r_j} L_{jl}$.

The QoS model is a function that maps the number of sources in an aggregate and the loss probability into a *utility*. This utility can be either interpreted directly as the amount that these sources would be willing to pay for this level of QoS, or indirectly interpreted simply as a numerical measure of satisfaction. Specifically, we assume class $j$ derives a satisfaction

$U_j(L_j, N_j)$ from supporting $N_j$ sources with a probability of loss of $L_j$. In this paper, we will assume that $U_j(L_j, N_j)$ is decreasing, twice continuously differentiable, and strictly concave in $L_j$. The utility can also be thought of directly as a function of reserved bandwidth and buffer.

*Theorem 1:* $U_j(L_j, N_j)$ is increasing, twice continuously differentiable, and strictly concave in $\{BW_{jl}, BF_{jl}\}$.

*Proof:* By assumption, the loss on each link, $L_{jl}$, is twice continuously differentiable and jointly convex in $\{BW_{jl}, BF_{jl}\}$. It follows that the total loss, $L_j$, is also twice continuously differentiable and jointly convex in $\{BW_{jl}, BF_{jl}\}$, since positive linear combination of convex functions are convex [29, Prop. 2.1.1]. The utility as a function of bandwidth and buffer can then be viewed as the composition of $U_j(L_j, N_j)$ with $L_j$. Therefore, $U_j(L_j, N_j)$ is twice continuously differentiable in $\{BW_{jl}, BF_{jl}\}$ following the chain rule of the second derivatives. And $U_j(L_j, N_j)$ is also strictly concave in $\{BW_{jl}, BF_{jl}\}$, since the composition of the strictly convex function (defined on a convex set), with a decreasing concave function produces a strictly concave function [29, Prop. 2.1.8]. ■

Throughout this paper, we further assume that marginal utilities with respect to bandwidth and buffer are each infinite when these resources are zero (so that infinite resource prices imply zero demand) and that the Hessian of utility with respect to resources is negative definite.

## B. Optimal Resource Allocation

We now pose our resource allocation problem. We assume that the network attempts to maximize total utility of all active users, by choosing bandwidth and buffer allocations on each link and at each router. The corresponding problem is

$$\max_{BW_{jl}, BF_{jl}} \sum_{j=1}^{m} U_j(L_j, N_j)$$

$$\text{s.t.} \quad \sum_{j \in \hat{r}_l} BW_{jl} \le BW_l, \sum_{j \in \hat{r}_l} BF_{jl} \le BF_l \quad \forall l \tag{1}$$

$$BW_{jl} \ge 0, BF_{jl} \ge 0 \quad \forall j, l \tag{2}$$

$$\sum_{l \in r_j} D_{jl} \le D_j \quad \forall j \tag{3}$$

When none of the delay constraints (3) are binding, we denote this as Problem U; this will be shown to be a concave program in Theorem 2. When at least one of the delay constraints (3) is binding, we denote this as Problem UD; the presence of at least one binding constraint in (3) can render the constraint set (1)–(3) nonconvex. In the remainder of this section and in Sections IV and V, we consider Problem U. In Section VI, we consider Problem UD.

To simplify the notation, let

$$x_j \equiv [BW_{ji_1}, BF_{ji_1}, \ldots, BW_{ji_{n_j}}, BF_{ji_{n_j}}]^t \in \mathbb{R}^{2n_j}$$

denote the set of resource allocations for class $j$, and

$$x \equiv [x_1; x_2; \ldots; x_m] \in \mathbb{R}^{2\tilde{n}}$$

denote resource allocations for all active user classes in the network, where $\tilde{n} \equiv \sum_{j=1}^{m} n_j$ is the total number of resource pairs.

Define $f(x) : \mathbb{R}^{2\tilde{n}} \to \mathbb{R}^{2n}$ as the function that maps a resource allocation into its corresponding constraint vector

$$[BW_1 - \sum_{j \in \hat{r}_1} BW_{j1}, BF_1 - \sum_{j \in \hat{r}_1} BF_{j1}, \ldots, \\ BW_n - \sum_{j \in \hat{r}_n} BW_{jn}, BF_n - \sum_{j \in \hat{r}_n} BF_{jn}]^t.$$

Finally, let $X$ represent the constraint set, namely $X \equiv \{x | f(x) \ge 0, x \ge 0\}$.

*Theorem 2:* Problem U is a concave program, and $\arg\max_x \sum_{j=1}^{m} U_j(L_j, N_j)$ contains a single point.

*Proof:* Problem U is a concave program, by definition, if the feasible region is a convex set and the optimization metric is a concave function. By Theorem 1, $U_j(L_j, N_j)$ is strictly concave in the resource allocation $x$.

When delay constraints are not binding, the feasible set is the intersection of constraint sets in (1) and (2), which are convex since these constraints are linear. Thus, the feasible set is also convex. It follows that Problem U is a concave program.

Furthermore, since the objective function is strictly concave, the optimal solution to the problem is either empty or contains a single point [30, Th. 7.14]. Finally, since the constraint set is a closed set, there must be at least one optimal allocation [30, Th. 3.1]. It follows that the optimal allocation is unique. ■

Establishing that Problem U is a concave program tells us that there is only one local maximum of the total utility over the resource allocation space. This property suggests that the optimal allocation can be found using some type of gradient algorithm. This can be made explicit by characterizing the optimal resource allocation in terms of the shadow costs corresponding to each of the constraints.

*Theorem 3:* The resource allocation $x$ solves Problem U if and only if there exists a set of nonnegative shadow costs $\mu \equiv [\beta_1, \gamma_1, \ldots, \beta_n, \gamma_n] \in \mathbb{R}_+^{2n}$ such that

$$\frac{\partial U_j}{\partial BW_{jl}} = \beta_l, \frac{\partial U_j}{\partial BF_{jl}} = \gamma_l$$

$$\beta_l \left( BW_l - \sum_{j \in \hat{r}_l} BW_{jl} \right) = 0$$

$$\gamma_l \left( BF_l - \sum_{j \in \hat{r}_l} BF_{jl} \right) = 0. \tag{4}$$

The theorem is a direct application of the Kuhn–Tucker theorem under convexity [30, Th. 7.16] and the proof is omitted. Solving such a maximization problem for a network of moderate size can be computationally intensive, and thus it is desirable to distribute the computation to the users and various network levels. The Lagrangian dual problem is often used to serve such a purpose.

## C. Dual Problem

Given Problem U as the primal problem, the dual problem is defined as follows:

$$\min_{\mu} q(\mu) \quad \text{s.t. } \mu \ge 0$$

where the dual function is defined as

$$\begin{aligned} q(\mu) &\equiv \sup_x L(x,\mu) = \sup_x \{U(x) + \mu' f(x)\} \\ &= \sum_{j=1}^{m} \max_{x_j} \left[ U_j(x_j) - \sum_{l\in r_j} \beta_l BW_{jl} - \sum_{l \in r_j} \gamma_l BF_{jl} \right] \\ &\quad + \sum_{l=1}^{n} (\beta_l BW_l + \gamma_l BF_l). \end{aligned} \tag{5}$$

The Lagrangian dual problem is a convex optimization problem, since the objective to be minimized is convex and the constraint set is convex. With Slater's condition being satisfied, i.e., there exists a strictly feasible point $x_0 \in X$ where all the constraints are not binding, we know that strong duality holds. In other words, the best bound that can be obtained from the Lagrange dual function is tight for the primal problem.

This suggests a mechanism to distribute the resource optimization between the user classes and the network in which the network solves the dual problem while the user classes determine the minimal cost resource allocation given a set of prices.

## IV. Distributed Resource Allocation Using Pricing

As we mentioned before, the Lagrangian dual problem uses shadow costs to distribute the resource optimization between the users and the network. An interpretation of the (4) is that each of the shadow costs in $\mu$ is zero if the corresponding constraint is not binding, and positive if the constraint is binding. (We are thus considering that congestion occurs only when demand exceeds supply. An alternative approach would be to define congestion as a function of load.)

We note a few properties of the dual function (5). First, it is everywhere continuously differentiable [31, Prop. 8.1.1], and its gradient with respect to the price vector can be represented as

$$\nabla_\mu q(\mu) = f(x_\mu) \tag{6}$$

where $x_\mu = \arg\max L(x,\mu)$ is the minimum cost allocation. This implies that a gradient feedback algorithm can be used to adjust prices based on the differences between demand and supply.

Second, the first term in the dual function is simply the summation of $m$ terms, each of which involves a maximization only over one class's resources. This implies that the optimal global resource allocation can be achieved by the combination of optimal allocations of each class.

Suppose the network and users follow the following strategies at update number $k+1$:

**Network Algorithm N1**: Update the prices as follows:

$$\mu^{k+1} = [\mu^k + s^k t^k]^+ \tag{7}$$

where $\mu^{k+1}$ is the set of prices to use at update number $k+1$, $s^k$ is a positive scalar step size, $[\cdot]^+$ is the projection on $\mathbb{R}_+^{2n}$, and $t^k$ is any feasible direction that satisfies $sgn(\beta_l^{k+1} - \beta_l^k) = sgn(\sum_{j\in\hat{r}_l} BW_{jl} - BW_l)$ and $sgn(\gamma_l^{k+1} - \gamma_l^k) = sgn(\sum_{j\in\hat{r}_l} BF_{jl} - BF_l)$. The feasible direction therefore allows for any price update rule that results in an increase in a price if the corresponding demand exceeds supply.

**User Algorithm U1**: In each class $j$, choose bandwidth and buffer allocations that maximize surplus, where surplus is defined as utility minus cost:

$$x_j^{k+1} = \arg\max_{x_j} \left[ U_j(x_j) - \sum_{l\in r_j} \beta_l^k BW_{jl} - \sum_{l\in r_j} \gamma_l^k BF_{jl} \right]. \tag{8}$$

We prove that the equilibrium point of this distributed resource allocation process is optimal.

*Theorem 4:* The resource allocation $x$ solves Problem U if and only if it is an equilibrium for Network Algorithm N1 and the User Algorithms U1 for each class.

*Proof:* At equilibrium, the Network Algorithm N1 results in either a binding constraint or an associated shadow cost of zero, namely $\beta_l(BW_l - \sum_{j\in\hat{r}_l} BW_{jl}) = 0$ and $\gamma_l(BF_l - \sum_{j\in\hat{r}_l} BF_{jl}) = 0$.

The utility functions $U_j(x_j)$ are jointly strictly concave in $x_j$, so the surplus is jointly strictly concave in $x_j$. It follows that the equilibrium solution to the User Algorithm U1 for class j results in $\partial U_j/\partial BW_{jl} = \beta_l$, $\partial U_j/\partial BF_{jl} = \gamma_l$.

Therefore, the equations (4) are satisfied, and the equilibrium solution is optimal, if and only if it is an equilibrium for Network Algorithm N1 and the User Algorithms U1 for each class. ■

A few caveats are in order here. We are not suggesting that these algorithms should be implemented in the Internet as written, but only mean to demonstrate what a pricing approach might attempt to accomplish. Any user response would undoubtedly have to be done via a user agent to automate the response, the time scales for each iteration would have to be chosen and the feedback algorithms would have to be designed to guarantee convergence. Also, much additional work would have to be done to make any such approach implementable, including development of signalling protocols, reservation protocols, measurement algorithms, and scheduling policies.

The User Algorithm U1 is an unconstrained multi-dimensional optimization problem, and can be solved using any appropriate ascent direction method. In the Network Algorithm N1, common methods for determining the feasible direction include gradient projection and Newton's method. In the next two subsections, we investigate these two methods.

### *A. Gradient Projection Method*

In this subsection, we investigate the use of a gradient projection algorithm to determine the feasible direction in the price update, namely $t^k = -\nabla_\mu q(\mu^k)$ in (7).

Using (6), this results in

$$\mu^{k+1} = \left[\mu^k - s^k f\left(x_\mu^k\right)\right]^+. \tag{9}$$

We can utilize several different types of step size rules, such as constant step size, diminishing step size, and dynamically chosen step size based on the exact optimal dual value or a suitable estimate. To simplify the discussion, we will use a constant step size in this discussion.

Suppose in each iteration, the network updates prices using (9) and the users allocate resources using (8). In the remainder of this section, we will prove the convergence of this algorithm.

First, a lemma is introduced.

Denote $\mathbb{S}$ as the set of positive shadow costs.

*Lemma 1:* The following two statements are true:

1) For every bounded set $A \subset \mathbb{S}$, there exists some constant $C$ such that $\|\nabla q(\mu_1) - \nabla q(\mu_2)\| \leq C\|\mu_1 - \mu_2\|, \forall \mu_1, \mu_2 \in A$.
2) The set $\{\mu | \mu \in \mathbb{S}, q(\mu) \leq c\}$ is either bounded or empty, $\forall c \in \mathbb{R}$.

*Proof:* We first consider the Hessian of $q(\mu)$. From the dual function (5), we have

$$\nabla_x L(x,\mu) = \nabla_x U(x_\mu) + (\nabla_x f(x_\mu))\,\mu = 0.$$

By differentiating the above equation with respect to $\mu$

$$\nabla_\mu x_\mu \nabla^2_{xx} L(x_\mu,\mu) + (\nabla_x f(x_\mu))^t = 0$$

where $\nabla^2_{xx} L(x_\mu,\mu) = \nabla^2_{xx} U(x_\mu)$, which is symmetric and negative definite by assumption. And

$$\begin{aligned}\nabla^2_{\mu\mu} q(\mu) &= \nabla_\mu x_\mu \nabla_x f(x_\mu) \\ &= -(\nabla_x f(x_\mu))^t \left[\nabla^2_{xx} L(x_\mu,\mu)\right]^{-1} \\ &\quad \times \nabla_x f(x_\mu) \end{aligned} \tag{10}$$

which is a real, symmetric, positive definite matrix of dimension $(2n) \times (2n)$. Since $U(x)$ are twice continuously differentiable and $f(x)$ are continuously differentiable, it is obvious that $q(\mu)$ is also twice continuously differentiable over $\mathbb{S}$. Thus, the gradient of $q(\mu)$ satisfies Lipschitz continuity over any bounded set $A \subset \mathbb{S}$, i.e., the first statement is true.

We also notice that when $\mu \to 0$, $x \to \infty$ and $q(\mu) \to \infty$, and that when $\mu \to \infty$, $x \to 0$ and $q(\mu) \to \infty$. Since $q(\mu)$ is convex and there exists some $\mu \in \mathbb{S}$ such that $q(\mu)$ is minimized, then the set $\{\mu | \mu \in \mathbb{S}, q(\mu) \leq c\}, \forall c \in \mathbb{R}$, is either bounded or empty, i.e., the second statement is true. ■

We can now prove the following theorem.

*Theorem 5:* Consider the algorithms specified in algorithms N1 and U1. Let $\{\mu^k\}$ be a sequence generated by (9), and $\{x^k_j\}$ a sequence generated by (8). Then there exists a positive step size $s^k = s > 0$ such that the limit point of $\{\mu^k\}$ is stationary and solves Problem U.

*Proof:* Let $\mu^0$ denotes the initial vector. We need only consider $\mu \in \mathbb{S}$ rather than $\mu \in \mathbb{R}^{2n}_+$ since by theorem 1, utility is increasing and strictly concave in resources, which implies that at the optima all of the shadow costs are positive.

The level set $A = \{\mu | \mu \in \mathbb{S}, q(\mu) < q(\mu^0)\}$ is bounded following the second statement of Lemma 1. Let $R = \max\{\|\mu\| \; | \mu \in A\}$, $G = \max\{\|\nabla q(\mu)\| \; | \mu \in A\}$, and $B = \{\mu | \; \|\mu\| \leq R + 2G\}$. Using the first statement of Lemma 1, we know that there exists a constant $C_B$ such that $\|\nabla q(\mu_1) - \nabla q(\mu_2)\| \leq C_B\|\mu_1 - \mu_2\|, \forall \mu_1, \mu_2 \in B$. Suppose the step size $s$ satisfies $0 < s < \min\{2, 2/C_B\}$. If $\mu^k \in A$

$$\|\mu^{k+1}\| = \left\|\mu^k - s\nabla q(\mu^k)\right\| \leq R + 2G$$

then $\mu^{k+1} \in B$. By using the descent lemma [32, Prop. A.24], we have

$$\begin{aligned} q(\mu^{k+1}) - q(\mu^k) &\leq \nabla q(\mu^k)'(\mu^{k+1} - \mu^k) + \frac{C_B}{2}\|\mu^{k+1} - \mu^k\|^2 \\ &\leq \left(\frac{C_B}{2} - \frac{1}{s}\right)\|\mu^{k+1} - \mu^k\|^2. \end{aligned}$$

When $s < 2/C_B$, the right-hand side of the above relation is nonpositive. So $q(\mu^{k+1}) < q(\mu^k)$ and $\mu^{k+1}$ stays within the level set $A$. If $\{\mu^k\}$ has a limit point $\mu^*$, then

$$\begin{aligned}\mu^* &= \lim_{k\to\infty} \mu^{k+1} = \lim_{k\to\infty}\left[\mu^k - s\nabla_\mu q(\mu^k)\right]^+ \\ &= \left[\mu^* - s\nabla_\mu q(\mu^*)\right]^+ \end{aligned}$$

which implies $\mu^* \cdot f(x_{\mu^*}) = 0$. And since $x^*_j$ is given by (8), we have $\nabla_{x^*_j} U_j = \mu^*$. By Theorem 4, $\mu^*$ is stationary and solves Problem U. ■

## B. Newton's Method

In this subsection, we investigate the use of Newton's method in place of the gradient projection method to determine the feasible direction and step size in the price update, namely $t^k = -(\nabla^2_{\mu\mu} q(\mu^k))^{-1} \nabla_\mu q(\mu^k)$ and $s^k = 1$ in (7), which results in

$$\mu^{k+1} = \left[\mu^k - \left(\nabla^2_{\mu\mu} q(\mu^k)\right)^{-1} f\left(x^k_\mu\right)\right]^+ \tag{11}$$

where $(\nabla^2_{\mu\mu} q(\mu^k))$ is given by (10).

To simplify the computation of (10), we note that $\nabla^2_{xx} L(x_\mu,\mu)$ is a block diagonal matrix and can be denoted as $\nabla^2_{xx} L(x_\mu,\mu) = Diag(\nabla^2_{x_j x_j} U_j), j = 1,\ldots,m$.

The inverse of a block diagonal matrix equals the matrix of the inverse blocks. Furthermore, we have previously proven that $\left[\nabla^2_{xx} U_j\right]^{-1} = \nabla_{\mu_j} x_j$, the sensitivities of class $j$ resources to bandwidth and buffer prices on route $j$ ([33], Lemma 2). Therefore, $\left[\nabla^2_{xx} L(x_\mu,\mu)\right]^{-1} = Diag(\nabla_{\mu_j} x_j), j = 1,\ldots,m$.

$\nabla_x f(x_\mu)$ is a $2\tilde{n} \times 2n$ routing matrix consisting of $2 \times 2$ identity matrices in positions corresponding to links in each route, and $2 \times 2$ zero matrices in all other positions. It can therefore be shown that (10) reduces to

$$\nabla^2_{\mu\mu} q(\mu) = -\sum_{j=1}^{m} M_j \tag{12}$$

where $M_j$ is the $2n \times 2n$ sensitivity matrix for class $j$, which consists of $2 \times 2$ sub-matrices of the sensitivities of resources on link $l'$ with respect to resource prices of link $l$:

$$(M_j)_{l,l'} = \begin{pmatrix} \frac{\partial BW_{jl'}}{\partial \beta_l} & \frac{\partial BF_{jl'}}{\partial \beta_l} \\ \frac{\partial BW_{jl'}}{\partial \gamma_l} & \frac{\partial BF_{jl'}}{\partial \gamma_l} \end{pmatrix}.$$

*Theorem 6:* Consider the algorithms specified in algorithms N1 and U1. Let $\{\mu^k\}$ be a sequence generated by Newton's method (11) and $\mu^*$ be the local minimum of $q(\mu)$ over the constraint set $\{\mu | \mu \geq 0\}$. Then there exists a $\delta > 0$ such that if $\|\mu^0 - \mu^*\| < \delta$, then $\|\mu^k - \mu^*\|$ converges to zero superlinearly.

The proof is straightforward and is omitted here (see [32, Prop. 2.3.5]). Note that this theorem only guarantees local convergence, as is typical with Newton's method. However, well-known variations can be used to guarantee global convergence since the dual problem is a convex optimization problem.

In comparison to the price update based on gradient projection, we find (as is typical) that Newton's method converges more quickly but requires more information. Specifically, the feasible direction based on gradient projection only requires knowledge of the demands and supplies, while the feasible direction based on Newton's method also requires the sensitivity matrices for each class.

## V. Dynamic Pricing

In the previous section, we proposed an implementation distributed between users and network routers, and proved that an allocation is optimal if and only if all users and routers are in equilibrium. For the network algorithm, two different methods—gradient projection and Newton's method—were discussed. This approach, however, requires that users choose the optimal combination of network resources (bandwidth and buffer) at each link and router based on prices for these resources. This level of detail may seem inappropriate for users (or even application agents) since they are likely to care only about the QoS that results from these resource allocations, not the resource allocations themselves.

In this section, we propose a new distributed implementation in which we introduce an *arbitrager* layer in between the user and the network. The arbitrager sells QoS (loss and delay) to the users, and purchases from the network the least cost bundle of resources (bandwidth and buffer) that achieves the desired QoS. This allows the users (or their agents) to focus on QoS rather than the network resources themselves.

With the new three layer model, we prove that an allocation is optimal if and only if all users, arbitragers, and routers are in equilibrium. Through a set of examples, we demonstrate the result of using algorithms for the arbitrager and the network based on gradient projection and on Newton's method, and present simulation results illustrating the network dynamics.

### *A. Network, User, and Arbitrager*

We now propose a new distributed implementation in which we introduce an arbitrager layer in between the user and the network. The arbitrager sells QoS (loss) to the users, and purchases from the network the least cost bundle of resources (bandwidth and buffer) that achieves the desired QoS. This approach simplifies the task for the users by moving consideration of bandwidth and buffer allocation to the arbitrager.

To simplify the notation, we denote

$$\begin{aligned}
U_g^{(j)} &\equiv \partial U_j / \partial L_j \\
U_{gg}^{(j)} &\equiv \partial U_g^{(j)} / \partial L_j \\
g_w^{(jl)} &\equiv \partial L_j / \partial BW_{jl} \\
g_f^{(jl)} &\equiv \partial L_j / \partial BF_{jl} \\
g_{ww}^{(jl)} &\equiv \partial g_w^{(jl)} / \partial BW_{jl} \\
g_{wf}^{(jl)} &\equiv \partial g_w^{(jl)} / \partial BF_{jl} \\
g_{ff}^{(jl)} &\equiv \partial g_f^{(jl)} / \partial BF_{jl}.
\end{aligned}$$

The optimal shadow costs in (4) can be used to establish that, at equilibrium,

$$\frac{\gamma_l}{\beta_l} = \frac{g_f^{(jl)}}{g_w^{(jl)}} \tag{13}$$

$$-U_g^{(j)} = -\frac{\beta_l}{g_w^{(jl)}} = -\frac{\gamma_l}{g_f^{(jl)}} \quad \forall l. \tag{14}$$

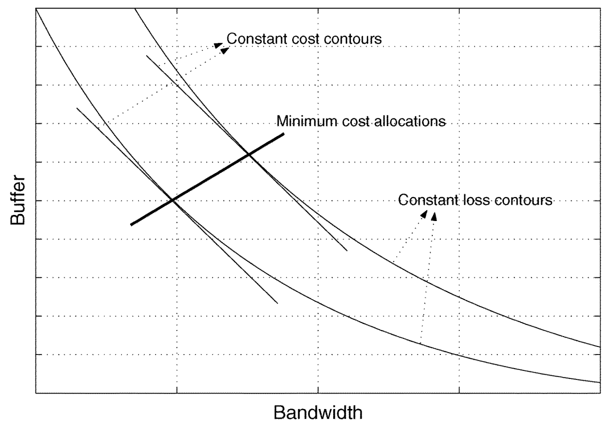

Fig. 1. Minimum cost allocation of bandwidth and buffer when the delay constraint is not binding.

Equation (13) can be interpreted as a user minimization of cost to achieve a desired loss probability on each link. If the prices and losses on each link are known, then (13) indicates that the optimal allocation of bandwidth and buffer on each link are given by finding the tangent point between the constant loss contour and a constant cost line, as illustrated in Fig. 1.

Equation (14) expresses how the user chooses the desired amount of loss, and how to allocate it among the links along the route. At the optimal allocation, the marginal benefit of decreasing loss is equal to the marginal cost. The marginal cost, in turn, is given by the cost of purchasing additional bandwidth and/or buffer, which is guaranteed to be equal. Since the left hand side is independent of $l$, the optimal allocation of loss equalizes the marginal costs among all links. In Fig. 1, this can be viewed as choosing the optimal point along the curve of minimum cost allocations.

Consideration of loss as intermediate variables suggests a new distributed resource allocation method. We now introduce an intermediate layer, called an arbitrager, in between the user and the network. Each network link periodically updates the prices for bandwidth and buffer on that link, based on the differences between the total demands and supplies for bandwidth and buffer. The arbitrager for class $j$ periodically receives a request for a specified loss level from the class $j$ users and finds the minimum cost allocation of bandwidth and buffer on each link on route $j$. The arbitrager also calculates a cost per unit loss, $\eta_j$, and advertises this cost to the class $j$ users. The class $j$ users periodically calculate the desired loss $L_j$ based on the cost $\eta_j$ and the utility function for that class. This process is illustrated in Fig. 2.

The network algorithm (N1) remains the same as above. The arbitrager and user algorithms are formalized as follows.

**Arbitrager Algorithm A2**: In each class $j$, choose the minimum cost bandwidth and buffer allocations that achieve a desired loss $L_j$:

$$\begin{aligned}
x_j^{k+1} = \arg\min_{x_j} \sum_{l \in r_j} &\left[\beta_l^k BW_{jl} + \gamma_l^k BF_{jl}\right] \\
\text{s.t.} \quad \sum_{l \in r_j} &g_j(BW_{jl}, BF_{jl}) \le L_j^k.
\end{aligned} \tag{15}$$

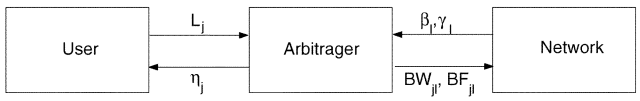

Fig. 2. Communication between user, arbitrager, and network.

Set a price *per unit loss* for class $j$ as the minimum marginal cost:

$$\eta_j^{k+1} \equiv \min_{\forall l} \left( \frac{\beta_l^k}{g_w^{(jl)}}, \frac{\gamma_l^k}{g_f^{(jl)}} \right). \tag{16}$$

**User Algorithm U2**: In each class $j$, choose loss that maximizes consumer surplus, where the surplus is defined as utility minus cost:

$$L_j^{k+1} = \arg\max_{L_j} \left\{ U_j(L_j, N_j) - \eta_j^k L_j \right\}. \tag{17}$$

This distributed process simplifies the task for the users by removing direct consideration of network resources and instead focusing on the QoS parameters of concern. In addition, the arbitrager could be used to guarantee a fixed price for each user connection by acting as an insurance agent who charges a small premium in order to smooth out fluctuations in the resource prices, although we do not consider this here.

*Theorem 7:* The resource allocation $x$ solves Problem U if and only if it is an equilibrium for Network Algorithm N1, the Arbitrager Algorithm A2 for each class, and the User Algorithms U2 for each class.

*Proof:* At equilibrium, the Network Algorithm N2 results in either a binding constraint or an associated shadow cost of zero, namely $\beta_l(BW_l - \sum_{j\in\hat{r}_l} BW_{jl}) = 0$ and $\gamma_l(BF_l - \sum_{j\in\hat{r}_l} BF_{jl}) = 0$.

With fixed prices and surcharges, the Arbitrager A2 is minimizing an increasing linear function on a convex set. Using the Kuhn–Tucker theorem under convexity it follows that a feasible resource allocation is minimal cost if and only if there exists an $\eta_j^* \in R$ such that the Kuhn–Tucker first-order conditions hold: $\eta_j^* = \beta_l/g_w^{(jl)} = \gamma_l/g_f^{(jl)}$ and $\sum_{l\in r_j} g_j(x_j) = L_j$. It follows that, at equilibrium, $\eta_j = \eta_j^*$.

The utility functions $U_j(L_j, N_j)$ are strictly concave and decreasing in $L_j$, so the equilibrium solution to the User Algorithm U1 for each class $j$ results in $U_g^{(j)} = \eta_j$.

Therefore, the equations (4) are satisfied, and the equilibrium solution is optimal, if and only if it is an equilibrium for Network Algorithm N2, the Arbitrager Algorithm A2 for each class, and the User Algorithms U2 for each class. ■

Theorem 7 establishes that the approach using separate user, arbitrager, and network algorithms achieves the maximal utility in a distributed fashion. As with Theorem 4, it is a result on equilibrium, but not on dynamics. Freedom is left to design feedback algorithms that achieve convergence on a time-scale of interest. We have not proven the convergence of this new three layer implementation. However, we believe that given appropriate step sizes, the network algorithm using either gradient projection method or Newton's method to update prices, along with the user and arbitrager algorithms finding optimal solutions, will converge to the equilibrium.

In numerical examples in this section, we consider small networks consisting of classes of on/off flows. Class $j$ consists of $N_j$ i.i.d. on/off fluid flows with Exponentially distributed on and off times. We set the mean on time to 340 ms, the mean off time to 780 ms, and the peak rate to 8 kb/s. We measure bandwidth in multiples of 8 kb/s (the peak rate) and buffer in multiples of 340 B (the mean number of bytes per cycle). We calculate loss using an effective bandwidth function derived by Morrison [19]. The utility function for class $j$ is $U_j(L_j, N_j) = N_j[1 - (L_j/a_j)^2]$, so that utility is proportional to the number of sources, and is a concave decreasing function of loss probability. The parameter $a_j$ determines the application's sensitivity to loss.

### *B. Dynamic Pricing Using Gradient Projection Method*

In this subsection, we consider use of gradient projection to dynamically update resource prices. To simplify the discussion, consider a single link (i.e., $n_j = 1, \forall j$) in the case in which the delay constraints for each class are loose. Then the Network Algorithm using gradient projection for resource prices as presented in (9) reduces to

$$\beta^{k+1} = \max\left\{ \beta^k - K_\beta \left( BW - \sum_j BW_j^k \right), 0 \right\}$$
$$\gamma^{k+1} = \max\left\{ \gamma^k - K_\gamma \left( BF - \sum_j BF_j^k \right), 0 \right\}$$

where $K_\beta$ and $K_\gamma$ are positive step sizes which determine the speed of the convergence. We allow for the step sizes to be different, since the ratio of the prices may be far from 1.

The User Algorithm U2 is a one-dimensional maximization. We assume that each user solves this problem each iteration.

The Arbitrager Algorithm A2, however, is a multi-dimensional constrained optimization. Rather than finding the optimal resource allocation each user and network iteration, we propose using a gradient algorithm to make small adjustments to these each iteration. Thus, in each iteration of the user and network algorithms, the arbitrager will receive updated prices from each network link on its route and an updated desired loss from the user class. In reaction, the arbitrager will make small changes to the resources it buys from the network along the route, and corresponding small changes to the price per unit loss it charges to the user. To specify these changes, it is helpful to separately consider the effects of resource price fluctuations and of changes in desired loss.

Increased loss from the user class should result in a lower allocation of both bandwidth and buffer. In Fig. 3, the minimum cost combination of bandwidth and buffer at a fixed loss probability is shown as point **A**. Let $r \equiv \beta/\gamma$ denote the ratio of the price of bandwidth to the price of buffer. Then point **A** lies on the desired loss contour, and is tangent to a line with slope equal to $-r$. An increase in the desired loss would move the minimum cost combination of bandwidth and buffer to a point **B**, on a lower contour and still tangent to a line with slope equal to $-r$.

Updated prices from the network should result in a new choice of buffer versus bandwidth if $r$ changes. Because the two resources can be traded-off to achieve the same loss, a decrease in $r$ would move the minimum cost combination of

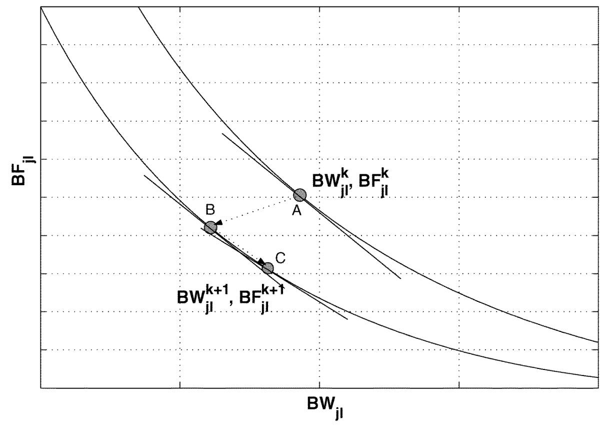

Fig. 3. Income and substitution effects.

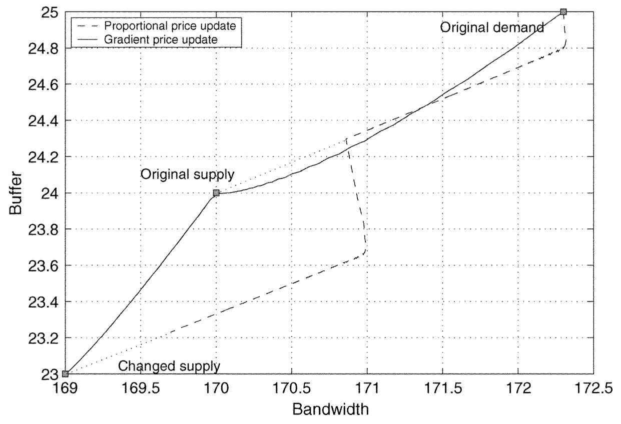

Fig. 4. Dynamic allocation for a single link single class network under a change in supply.

bandwidth and buffer from point **B** to point **C**, on the same loss contour, but with more bandwidth and less buffer.

Let $\Delta L_j^k$ represent the change in desired loss of user class $j$ in iteration $k$, and $\Delta r_j^k$ represent the change in price ratio of class $j$ in iteration $k$. A simple gradient algorithm would react to both changes in prices and changes in loss:

$$\begin{aligned} BW_j^{k+1} &= BW_j^k + \frac{dBW_j}{dL_j}\Delta L_j^k + \frac{dBW_j}{dr_j}\Delta r_j^k \\ BF_j^{k+1} &= BF_j^k + \frac{dBF_j}{dL_j}\Delta L_j^k + \frac{dBF_j}{dr_j}\Delta r_j^k \end{aligned} \tag{18}$$

where $dBW_j/dL_j$ and $dBF_j/dL_j$ are taken along a path with constant prices and where $dBW_j/dr_j$ and $dBF_j/dr_j$ are taken along loss contours. The arbitrager can set the new price *per unit loss* for class $j$ using (16).

In [34], we derive the required sensitivities at constant prices:

$$\frac{dBW_j}{dL_j} = \frac{g_w^{(j)} g_{ff}^{(j)} - g_f^{(j)} g_{wf}^{(j)}}{K^{(j)}}$$
$$\frac{dBF_j}{dL_j} = \frac{g_f^{(j)} g_{ww}^{(j)} - g_w^{(j)} g_{wf}^{(j)}}{K^{(j)}}$$

where $K^{(j)} = \left(g_f^{(j)}\right)^2 g_{ww}^{(j)} - 2g_w^{(j)} g_f^{(j)} g_{wf}^{(j)} + \left(g_w^{(j)}\right)^2 g_{ff}^{(j)}$.

A similar derivation (omitted here) gives the required sensitivities at constant losses: $dBW_j/dr_j = \left(g_f^{(j)}\right)^3/K^{(j)}$, $dBF_j/dr_j = -g_w^{(j)}(g_f^{(j)})^2/K^{(j)}$.

We now investigate the performance of this gradient projection algorithm through a numerical example. We consider a single link single class network with 500 sources that have a sensitivity to loss given by $a_1 = 0.01$. To drive the dynamics, we set the initial demand for bandwidth and buffer, respectively, at (172.3, 25), and the initial supply at (170, 24). The supply is held fixed for the first portion of the experiment. In Fig. 4, we show the resulting bandwidth and buffer allocations as the dashed curve.

At first, both bandwidth and buffer prices rise quickly, since demand exceeds supply for both. However, the price per unit buffer rises more quickly than the price per unit bandwidth, and therefore the price ratio $r$ falls. The effect of decreases in $r$, the substitution effect, initially dominates the income effect. As a result, initially the allocated buffer falls while the allocated bandwidth increases, approximately following a loss contour. After a relatively small number of iterations, the substitution and income effects become of the same order. As a result, both bandwidth and buffer allocations now decrease toward the supply.

When the demand is approximately (170.8, 24.3), we change the supply to (169, 23). (The change in supply may represent additional allocation to other classes not modeled here.) If the supply had not been modified, then the resource allocation would follow the dotted curve, converging to the initial supply of (170, 24). The modified supply causes an abrupt change in the resource allocation. The price ratio $r$ quickly decreases, resulting in movement along the current loss contour toward higher bandwidth and lower buffer. Soon, the substitution and income effects are of the same order, and both bandwidth and buffer allocations decrease toward the new supply.

The step sizes $K_\beta$ and $K_\gamma$ determine both the speed of convergence and the slope of the resource allocation path when both the substitution and income effects are significant. A clever choice of these constants can equalize the time scales on which the two terms in (18) work. This would result in a much more direct path from demand to supply. However, such clever choices of $K_\beta$ and $K_\gamma$ are dependent upon system parameters, and we believe that it may be difficult to find a method that sets them appropriately under a wide variety of loads.

## C. Dynamic Pricing Using Newton's Method

In this subsection, we consider use of Newton's method to dynamically update resource prices. We will consider a network with two links and two classes in the case in which the delay constraints for each class are loose. The Network Algorithm using Newton's method for resource price adjustments was given by (11). The required gradients in the sensitivity matrix $M_j$ are provided in [33, Th. 7]. These sensitivities would have to be calculated by the arbitrager and passed to the network (in addition to the bandwidth and buffer demands).

In the gradient projection method, the resource allocation path usually does not converge directly due to the differences in the time scales on which the substitution and income effects applied. To rectify this issue, we propose that the Arbitrager Algorithm also use Newton's method to determine adjustments to the resource allocation and price per unit loss.

The arbitrager wishes to find the minimum cost resource allocations that achieve a desired loss $L_j$, denoted $C_j(L_j)$. This can be represented as a constrained minimization problem:

$$C_j(L_j) = \min_{\{L_{jl}\}} \sum_{l=1}^{n_j} C_{jl}(L_{jl}), \quad \text{s.t.} \ \sum_{l=1}^{n_j} L_{jl} = L_j$$

where $C_{jl}(L_{jl})$ represents the minimum cost required to achieve a loss $L_{jl}$ on route $j$ and link $l$

$$C_{jl}(L_{jl}) \equiv \min_{BW_{jl}, BF_{jl}} (\beta_l BW_{jl} + \gamma_l BF_{jl}), \\ \text{s.t.} \ \ g_j(BW_{jl}, BF_{jl}) = L_{jl} \qquad (19)$$

Equivalently, we can represent this as an unconstrained minimization problem:

$$C_j(L_j) = \min_{L_{j1},\ldots,L_{jn_j-1}} \sum_{l=1}^{n_j} C_{jl}(L_{jl})$$

where $L_{jn_j} = L_j - \sum_{l=1}^{n_j-1} L_{jl}$.

From our previous work [33, Th. 1] we know that the minimum cost, $C_j(L_j)$, is a decreasing and convex function of loss probability, $L_j$. The minimization problem is therefore a convex problem. We propose to use several iterations of Newton's method to determine the resource allocation and associated loss probabilities in between each single iteration of the user and network algorithms. Denote the set of achieved loss probabilities on route $j$ at arbitrager subiteration $i$ as $\overrightarrow{L_j^i} = (L_{j1}, \ldots, L_{jn_j-1})^t$, and the corresponding total cost of obtaining these loss probabilities as $C_j(\overrightarrow{L_j^i})$. Newton's method can then be applied as follows:

$$\overrightarrow{L_j^{i+1}} = \overrightarrow{L_j^i} - \left[\nabla^2 C_j \left(\overrightarrow{L_j^i}\right)\right]^{-1} \nabla C_j \left(\overrightarrow{L_j^i}\right). \qquad (20)$$

Each time the arbitrager receives a desired loss from the user and a set of prices from the network, it executes several iterations of (20) to find the loss distribution that minimizes the total cost. From this loss distribution, the arbitrager determines the resource allocation by solving the 2-dimensional minimization in (19). The arbitrager also computes a cost per unit loss $\eta_j$ and a sensitivity matrix $M_j$, and sends this information to the user and the network, respectively.

It only remains to determine the gradient and Hessian required in (20). In our previous work [33], we determined the required sensitivities:

$$\frac{\partial C_j}{\partial L_{jl}} = \frac{\beta_l}{g_w^{(jl)}} - \frac{\beta_{n_j}}{g_w^{(jn_j)}}$$

$$\frac{\partial^2 C_j}{\partial L_{jl}^2} = \frac{\partial^2 C_{jl}}{\partial L_{jl}^2} + \frac{\partial^2 C_{jn_j}}{\partial L_{jn_j}^2}, \qquad \frac{\partial^2 C_j}{\partial L_{jl} \partial L_{jl'}} = \frac{\partial^2 C_{jn_j}}{\partial L_{jn_j}^2}$$

where $l, l' = 1, 2, \ldots, n_j - 1$ and $l \neq l'$, and

$$\frac{\partial^2 C_{jl}}{\partial L_{jl}^2} = -\frac{\beta_l}{g_w^{(jl)}} \frac{g_{ww}^{(jl)} g_{ff}^{(jl)} - \left(g_{wf}^{(jl)}\right)^2}{K^{(jl)}}.$$

The resulting bandwidth and buffer allocations using this dynamic pricing algorithm, for the single link single class experiment considered in the previous subsection, is shown in Fig. 4 as the solid curves. As expected, use of Newton's method results in both a more direct convergence path and quicker convergence.

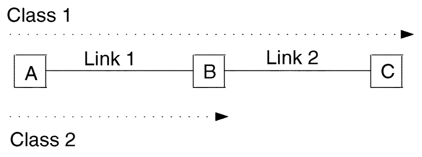

Fig. 5. Network topology with two links and two classes of sources.

The improvement in dynamics, however, comes at the cost of additional complexity. When the Network Algorithm is based on gradient projection, it only needs the resource demands from the arbitrager. However, when the Network Algorithm is based on Newton's method, it also needs from the arbitrager a gradient matrix containing sensitivity information. These sensitivities may in turn require the arbitrager to obtain information from the user, or alternatively use an estimation procedure such as perturbation analysis.

In order to further investigate the performance of this approach, we directly consider the effect of a change in the number of users upon the resource allocation for each class. For this experiment, we consider a network with two links and two classes, shown in Fig. 5. The bandwidth supply on the links are set to 370 and 170, respectively, and the output buffer available at routers A and B are set to 80 each. Class 1 sources have a sensitivity to loss defined by $a_1 = 0.01$, and class 2 sources have a sensitivity to loss defined by $a_2 = 0.1$. A single class 2 source therefore has the same traffic characterization as a class 1 source, but is less sensitive to loss.

We start by examining convergence of the resource allocation for each class on link 1. The number of class 1 sources is held fixed at $N_1 = 500$, while the number of class 2 sources starts at $N_2 = 630$. The initial desired losses of each class are set so that the initial demands for bandwidth and buffer are (165.00, 41.98) for class 1 on link 1 and (203.90, 37.92) for class 2.

In Figs. 6 and 7, we show the resulting bandwidth and buffer allocations on link 1, for class 1 and class 2, respectively. The resource allocations for both class 1 and class 2 converge directly to the optimal allocations for each class, denoted *first optimal allocation* in the figures.

After these resource allocations converge to values near the optima, we increase the number of class 2 sources from 630 to 631. The immediate effect of the new user entering the network is first felt at the class 2 arbitrager, who must buy additional bandwidth and buffer to achieve the same loss for the higher number of sources under the same resource prices. As a result, we see a jump in class 2's resource allocation from (204.48, 38.21) to (204.80, 38.24), shown as a dashed line in Fig. 7. We note that resource allocations per source now are less than resource allocations per source in the previous iteration, because class 2 has a higher multiplexing gain with one more user coming in.

Following this immediate effect, the network algorithm increases both resource prices on link 1 since the demands are now higher than the supplies. The users in each class react by accepting higher losses, and the arbitragers for each class corre-

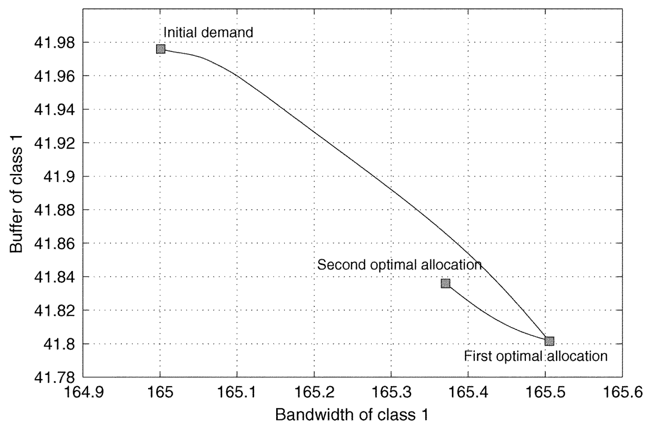

Fig. 6. Dynamic allocation of class 1 on link 1 under a change in the number of class 2 users.

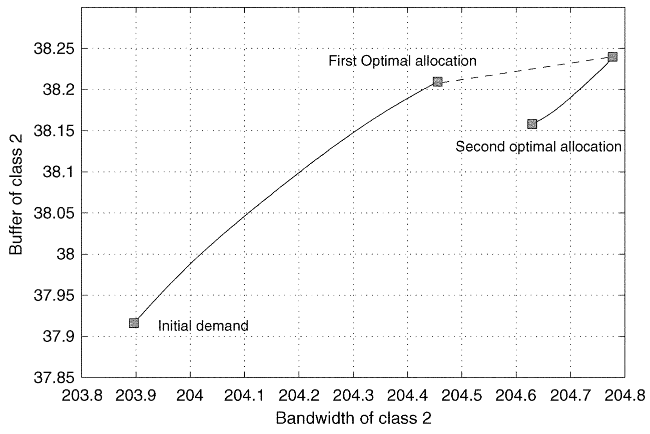

Fig. 7. Dynamic allocation of class 2 on link 1 under a change in the number of class 2 users.

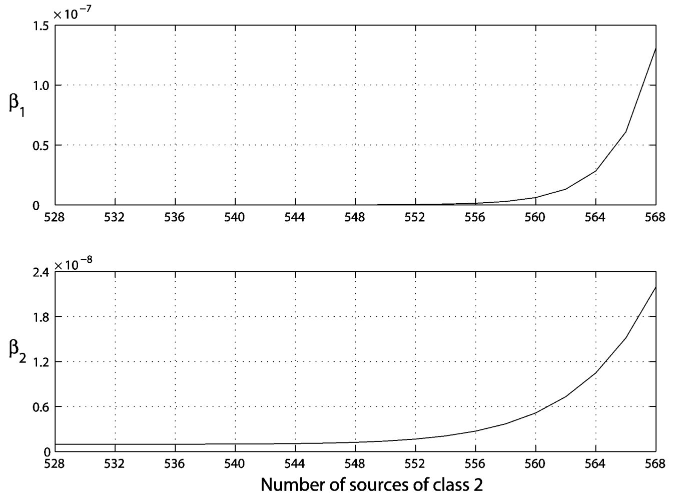

Fig. 8. Prices of bandwidth on link 1 and link 2.

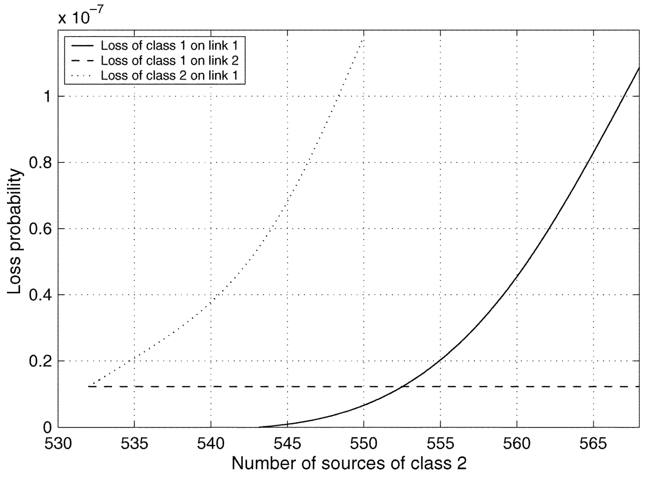

Fig. 9. Loss probabilities of class 1 and 2 on each link.

spondingly modify the resource allocations. This process continues until the allocations converge to the *second optimal allocations* as shown in the figures. The end result of an additional class 2 user is that both class 1 and class 2 users are allocated a combination of bandwidth and buffer per user which results in a higher loss. In this particular case, the final optimal allocation presents a decreasing bandwidth but increasing buffer per class 1 source, and a decreasing bandwidth and buffer per class 2 source.

We now investigate the variation in prices, loss probabilities and utility over a wide range of $N_2$, with $N_1$ held constant at 500. When the number of class 2 sources is small $(N_2 < 530)$, both classes can obtain very low loss probabilities on link 1, and correspondingly the price for bandwidth on link 1, $\beta_1$, is very low. On link 2, where the bandwidth supply is lower, the price for bandwidth, $\beta_2$, is higher than on link 1, and correspondingly class 1's loss is also higher. The utility per source is near its maximum value, and thus the total sensitivity increases nearly linearly with $N_2$ in this range.

As the number of class 2 sources exceeds 530, the competition for resources on link 1 becomes significant. Prices for link 1 resources slowly start to increase. The effect is felt first by class 2 sources, which are less sensitive to loss and therefore more willing to scale back demand. We illustrate the bandwidth prices on each link in Fig. 8; the buffer prices, which are not shown, follow similar trends. As $N_2$ exceeds 560, the prices for link 1 resource escalate more quickly to counteract the increased demand. Both classes are now accepting higher loss on link 1. In addition, class 1 sources try to obtain lower loss on link 2 to compensate, driving up prices for resources on link 2.

The effect of these price increases on loss is evident in Fig. 9. When the number of class 2 sources is small, the loss of class 1 is dominated by loss on link 2. As $N_2$ increases, loss on link 1 increases, eventually becoming the dominant component of class 1's total loss. On link 1, class 1 and 2 face the same resource prices. When $N_2 > N_1$, class 2 obtains a higher multiplexing gain than class 1, and hence the price per unit loss for class 2 is cheaper, i.e., $|\eta_2| < |\eta_1|$. However, class 2 is less sensitive to loss, and this dominates the effect of the multiplexing gain, so the loss of class 2 on link 1 is always larger than the loss of class 1.

## VI. Binding Delay Constraints

In this section, we investigate the case in which at least one delay constraint is binding, namely Problem UD. Whereas Problem U is a concave program by Theorem 2, Problem UD [which is equivalent to Problem U plus the delay constraints

(3)] is *not* a concave program. (Whether the set of constraints remains a convex set depends on the set of routes.)

In this section, we revisit Theorems 2–7 under Problem UD. Theorem 2 guaranteed a unique optimal resource allocation under Problem U. With delay constraints binding, there remains at least one optimal resource allocation, but it is no longer guaranteed to be unique.

Without binding delay constraints, Theorem 3 states that the optimal allocation occurs if and only if the optimal price set is used. Furthermore, the optimal prices on a particular link are equal to the marginal utilities with respect to bandwidth and buffer for each class passing through that link.

When delay constraints are binding, at the optimal allocation there still exists a corresponding optimal price set. However, there are two significant differences. First, the prices charged to each class on a particular link must be differentiated on the basis of the effect the corresponding allocation has upon the delay constraints. Second, satisfaction of these Kuhn–Tucker conditions no longer guarantees optimality. We express these relationships in Theorem 8.

*Theorem 8:* If resource allocation $x^*$ solves Problem UD, then there exists a set of nonnegative shadow costs $\mu^* \equiv [\beta_1^*, \gamma_1^*, \ldots, \beta_n^*, \gamma_n^*, \alpha_1^*, \ldots, \alpha_m^*] \in \mathbb{R}_+^{2n+m}$ such that

$$\frac{\partial U_j}{\partial BW_{jl}} = \beta_l^* + \hat{\beta}_{jl}^*, \; \frac{\partial U_j}{\partial BF_{jl}} = \gamma_l^* + \hat{\gamma}_{jl}^*,$$

$$\beta_l^* \left( BW_l - \sum_{j \in \hat{r}_l} BW_{jl} \right) = 0, \; \gamma_l^* \left( BF_l - \sum_{j \in \hat{r}_l} BF_{jl} \right) = 0$$

$$\alpha_j^* \left( D_j - \sum_{l \in r_j} D_{jl} \right) = 0$$

where $\hat{\beta}_{jl} \equiv \alpha_j(\partial D_{jl}/\partial BW_{jl}) = -\alpha_j(BF_{jl}/BW_{jl}^2)$ and $\hat{\gamma}_{jl} \equiv \alpha_j(\partial D_{jl}/\partial BF_{jl}) = \alpha_j/BW_{jl}$.

*Proof:* We know that a global maximum $x^*$ exists to Problem UD by Weierstrass Theorem (c.f. [30]). The theorem follows by the Kuhn–Tucker theorem if constraint qualification holds everywhere on the feasible set [30, p. 152]. Define $f(x) : \mathbb{R}^{2\tilde{n}} \rightarrow \mathbb{R}^{2n+m}$ as the function that maps a resource allocation into its corresponding constraint vector

$$\Big[BW_1 - \sum_{j \in \hat{r}_1} BW_{j1}, BF_1 - \sum_{j \in \hat{r}_1} BF_{j1}, \ldots, BW_n - \sum_{j \in \hat{r}_n} BW_{jn}, BF_n - \sum_{j \in \hat{r}_n} BF_{jn}, D_1 - \sum_{l=1}^{n_1} D_{1l}, \ldots, D_m - \sum_{l=1}^{n_m} D_{ml}\Big]^t.$$

The gradient of the constraint function, $\nabla_x f(x_\mu)$, is a $2\tilde{n} \times (2n+m)$ matrix which consists of two parts. The left part is the $2\tilde{n} \times 2n$ routing matrix (discussed previously), and the right part, with a dimension of $2\tilde{n} \times m$, corresponds to the gradients of the delay constraints. By analyzing $\nabla_x f(x_\mu)$, it is easy to show that $\rho(\nabla_x f_E(x_\mu)) = |E|, \forall x \in X$, where $|E|$ is the cardinality of the set of effective constraints and $f_E$ is a vector of effective constraints.

Finally, since the above two conditions hold, there exists a $\mu^*$ such that $(x^*, \mu^*)$ is a critical point of the Lagrange function of Problem UD; in other words, there exists a $\mu^*$ such that the Kuhn–Tucker first and second conditions are met [30, Th. 6.1]. ■

We interpret $\hat{\beta}_{jl}$ and $\hat{\gamma}_{jl}$ as surcharges for bandwidth and buffer, respectively, to class $j$ traffic on link $l$ if the delay constraint for that class is binding.

For Problem U, Theorem 4 provides a manner to distribute the tasks between each user and each network router. For Problem UD, the network algorithm (denoted $\mathrm{N1}'$) is the same as N1, except that the price vector now includes shadow costs for the each delay constraint. (Updates of these associated prices per unit delay must be provided by an agent that knows about the delay for that class, which will be the arbitrager for that class in the three-level model.) The user algorithm (denoted $\mathrm{U1}'$) is similar to U1, but now includes a charge for delay, namely

$$x_j^{k+1} = \arg\max_{x_j} \left[ U_j(x_j) - \sum_{l \in r_j} \beta_l^k BW_{jl} - \sum_{l \in r_j} \gamma_l^k BF_{jl} - \sum_{l \in r_j} \alpha_j^k \frac{BF_{jl}}{BW_{jl}} \right]$$

is used in place of (8). The new version of Theorem 4 becomes:

*Theorem 9:* If the resource allocation $x^*$ solves Problem UD, then it must be an equilibrium for Network Algorithm $\mathrm{N1}'$ and User Algorithms $\mathrm{U1}'$ for each class.

The proof is straightforward and is omitted. The theorem tells us that equilibrium is a necessary condition for the solution of Problem UD.

Dynamics for Problem U were described by Theorems 5 and 6 for the gradient projection and Newton's methods, respectively. When delay constraints are binding, the dual function may not be everywhere continuously differentiable. The descent direction, therefore, is not uniquely defined at some points. The solution to this is typically to use subgradient methods. At points at which the gradient of $q(\mu)$ is not defined, the feasible direction $-t^k$ is given by any of the subgradients of $q(\mu)$ at $\mu_k$, i.e.,

$$\mu^{k+1} = \left[\mu^k - s^k f\left(x_\mu^k\right)\right]^+$$

where $f(x_\mu^k)$ is one of the subgradients of $q(\mu)$ at $\mu_k$. Convergence results similar to Theorems 5 and 6 can be proven using well-known properties of subgradient methods. For instance, the gradient projection algorithm used earlier can be replaced by a subgradient method with a constant step size, which is now guaranteed to converge to within some range of the optimal value [31]. Due to space limitations, we omit the details.

We now consider the design of an arbitrager. As we did in the case in which delay constraints are not binding, we can consider loss as intermediate variables and introduce an arbitrager layer in between the user and the network. When a delay constraint is binding, i.e., $\alpha_j > 0$, surcharges are added, with $\hat{\beta}_{jl} < 0$ and $\hat{\gamma}_{jl} > 0$. Correspondingly, the constant cost contours become steeper, as demonstrated in Fig. 10. The result is that the arbitrager will choose higher bandwidth and lower buffer to achieve similar loss. In addition, the marginal costs on delay will allow the arbitrager to allocate the total delay $D_j$ among the links on route $j$.

For Problem U, Theorem 7 provides a manner to distribute the tasks between users, arbitragers, and the network. For Problem UD, the network algorithm remains $\mathrm{N1}'$ (except that logically

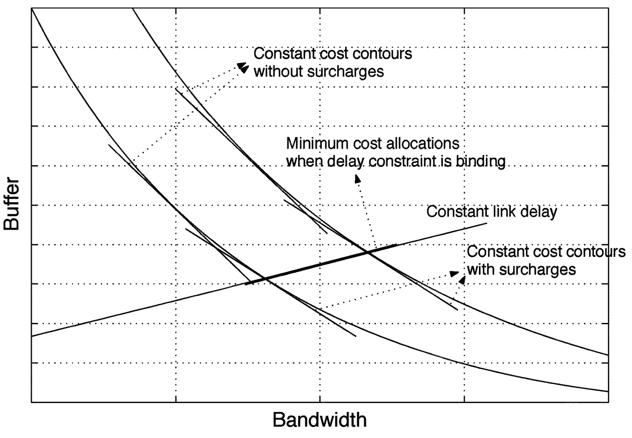

Fig. 10. Minimum cost allocation of bandwidth and buffer when the delay constraint is binding.

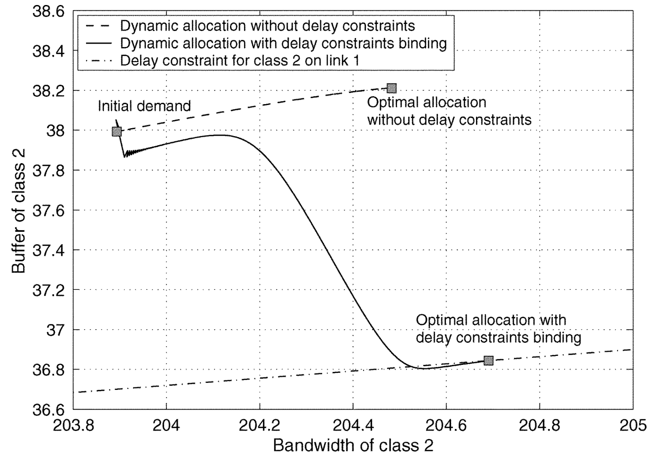

Fig. 11. Dynamic allocation for class 2 on link 1.

the update of the associated prices per unit delay is done by the arbitrager for that class). The arbitrager algorithm (denoted A2′) becomes

$$x_j^{k+1} = \arg\min_{x_j} \sum_{l \in r_j} \left[ \beta_l^k BW_{jl} + \gamma_l^k BF_{jl} + \alpha_j^k \frac{BF_{jl}}{BW_{jl}} \right].$$

The user algorithm remains the same as U2. The resulting combination gives us:

*Theorem 10:* If the resource allocation $x^*$ solves Problem UD, then it must be an equilibrium for Network Algorithm N1′, the Arbitrager Algorithm A2′ for each class, and the User Algorithms U2 for each class.

The proof is straightforward and is omitted.

In the remainder of this section, we return to our two class, two link numerical example presented in Section V-B. The parameters remain the same. Without a delay constraint, the optimal bandwidth and buffer allocations result in total delays of $D_1^* = 0.723$ for class 1 $D_2^* = 0.186$ for class 2. We now we add delay constraints of $D_1 \leq 0.65$ and $D_2 \leq 0.18$, both of which are tighter than corresponding optimal delays under Problem U. We use the user, arbitrager, and network algorithms presented in this section, with a constant step size subgradient method.

In Fig. 11, we show the resulting bandwidth and buffer allocation for class 2 on link 1. Without a delay constraint, the optimal

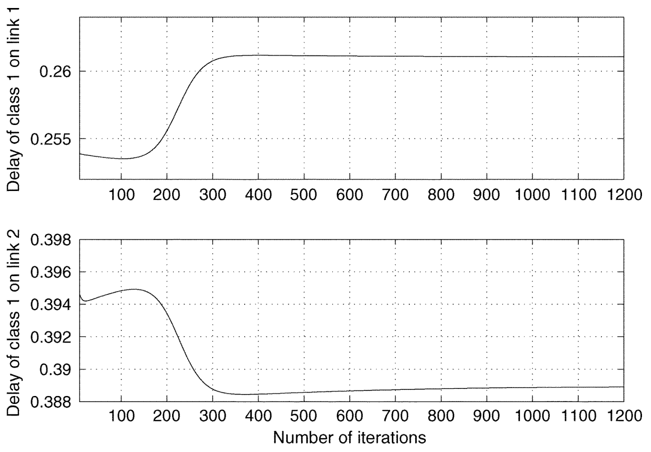

Fig. 12. Dynamic delay of class 1 along link 1 and link 2.

bandwidth and buffer allocations were (204.48,38.21). This allocation violates the delay constraint for class 2, and therefore the algorithms increase the price per unit buffer and decrease the price per unit bandwidth charged to class 2 on link 1, until the allocation satisfies the delay constraint. At optimality, the allocation for class 2 becomes (204.70, 36.84). Consequently, class 1 uses less bandwidth and more buffer on link 1. This raises the delay for class 1 on link 1. To compensate, class 1 buys less buffer on link 2 in order to satisfy its delay constraint. These dynamics are illustrated in Fig. 12.

We note that in this experiment there is an unique optimum allocation, and the users, arbitragers, and network all converge to this point. Although the presence of the delay constraint makes Problem UD not a concave program, we have not found a numerical example in which there exist multiple optimum allocations or in which the gradient projection algorithm does not globally converge.

## VII. Conclusion

We have assumed the existence of a reservation-based QoS architecture that uses shadow-cost pricing. In particular, we considered a pricing policy which implements a distributed resource allocation to provide guaranteed bounds on packet loss and end-to-end delay for real-time applications. Distributed pricing roles consist of three entities: user, network, and an arbitrager layer in between the user and the network.

When delay constraints are not binding, we have constructed two simple dynamic pricing policies using gradient projection method and Newton's method for network price adjustments. We have proven the convergence of these two methods when pricing based resource allocation is distributed between user and network. When delay constraints are binding, we have investigated subgradient methods which can provide convergence to some range of the optimal allocation. Numerical results provide some insight into how such algorithms respond when the number of users or resource supplies change. In practice, the rate of convergence of these algorithms must be compared to the rate of change of other network dynamics.

In the approach presented in this paper, we considered congestion-based pricing in a reservation-based QoS architecture e.g., IntServ. The method is generic in that it does not rely upon any specific network architecture, signalling protocol, or traffic characterization, and in that it allows for a broad class of dynamic congestion-based pricing algorithms. Many challenges remain before any such congestion-based pricing could be implemented in real networks including design of automated user agents, delineation of time scales, design of feedback algorithms, and development of measurement algorithms, and consideration of the cost and complexity of the resulting architecture and protocols.

A related research question is how to accomplish congestion-based pricing in a priority-based QoS architecture such as diffServ. Such an architecture often groups traffic into different priority classes, and performs scheduling policies, dropping policies, and traffic smoothing on each priority class. We believe pricing can be used to implement congestion control and resource allocation in these frameworks, and are considering this in current research.

## References

[1] J. Murphy, L. Murphy, and E. C. Posner, "Distributed pricing for embedded ATM networks," in *Proc. Int. Teletraffic Congr.*, 1994, pp. 1053–1062.

[2] H. Jiang and S. Jordan, "The role of price in the connection establishment process," *Eur. Trans. Telecommun.*, vol. 6, no. 4, pp. 421–429, Jul.–Aug. 1995.

[3] S. Low and P. Varaiya, "A new approach to service provisioning in ATM networks," *IEEE/ACM Trans. Netw.*, vol. 1, no. 5, pp. 547–553, Oct. 1993.

[4] S. Low, "Equilibrium bandwidth and buffer allocations for elastic traffics," *IEEE/ACM Trans. Netw.*, vol. 8, no. 3, pp. 373–383, Jun. 2000.

[5] N. J. Keon and G. Anandalingam, "Optimal pricing for multiple services in telecommunications networks offering quality-of-service guarantees," *IEEE/ACM Trans. Netw.*, vol. 11, no. 1, pp. 66–80, Feb. 2003.

[6] H. Yaiche, R. R. Mazumdar, and C. Rosenberg, "A game theoretic framework for bandwidth allocation and pricing in broadband networks," *IEEE/ACM Trans. Netw.*, vol. 8, no. 5, pp. 667–678, Oct. 2000.

[7] X. Cao, H. Shen, R. Milito, and P. Wirth, "Internet pricing with a game theoretical approach: concepts and examples," *IEEE/ACM Trans. Netw.*, vol. 10, no. 2, pp. 208–216, Apr. 2002.

[8] F. Kelly, "Charging and accounting for bursty connections," in *Internet Economics*, L. W. McKnight and J. P. Bailey, Eds. Cambridge, MA: MIT Press, 1997, pp. 253–278.

[9] A. M. F. P. Kelly and D. Tan, "Rate control for communication networks: shadow prices, proportional fairness and stability," *J. Oper. Res. Soc.*, vol. 49, pp. 237–252, 1998.

[10] S. Low and D. Lapsley, "Optimization flow control, i: basic algorithm and convergence," *IEEE/ACM Trans. Netw.*, vol. 7, no. 6, pp. 861–875, Dec. 1999.

[11] R. J. La and V. Anantharam, "Utility-based rate control in the internet for elastic traffic," *IEEE/ACM Trans. Netw.*, vol. 10, no. 2, pp. 272–286, Apr. 2002.

[12] K. Kar, S. Sarkar, and L. Tassiulas, "A simple rate control algorithm for maximizing total user utility," in *Proc. IEEE INFOCOM*, 2001, pp. 133–141.

[13] S. Low and P. Varaiya, "Burstiness bounds for some burst reducing servers," in *Proc. IEEE INFOCOM*, 1993, pp. 2–9.

[14] R. Cruz, "A calculus for network delay, part 1: networks elements in isolation," *IEEE Trans. Inf. Theory*, vol. 37, no. 1, pp. 114–131, Jan. 1991.

[15] P. Skelly, M. Schwartz, and S. Dixit, "A histogram-based model for video traffic behavior in an ATM multiplexer," *IEEE/ACM Trans. Netw.*, vol. 1, no. 4, pp. 446–459, Oct. 1993.

[16] W. Leland, M. Taqqu, W. Willinger, and D. Wilson, "On the self-similar nature of Ethernet traffic (extended version)," *IEEE/ACM Trans. Netw.*, vol. 2, no. 1, pp. 1–15, Feb. 1994.

[17] M. Montgomery and G. DeVeciana, "On the relevance of time scales in performance oriented traffic characterizations," in *Proc. IEEE INFOCOM*, 1996, pp. 513–520.

[18] A. W. Berger and W. Whitt, "Effective bandwidths with priorities," *IEEE/ACM Trans. Netw.*, vol. 6, no. 4, pp. 447–460, Aug. 1998.

[19] J. Morrison, "Asymptotic analysis of a data-handling system with many sources," *SIAM J. Appl. Math.*, vol. 49, no. 2, pp. 617–637, Apr. 1989.

[20] J. Qui and E. W. Knightly, "Measurement-based admission control with aggregate traffic envelopes," *IEEE/ACM Trans. Netw.*, vol. 9, no. 2, pp. 199–210, Apr. 2001.

[21] C. A. Courcoubetis, A. Dimakis, and G. D. Stamoulis, "Traffic equivalence and substitution in a multiplexer with applications to dynamic available capacity estimation," *IEEE/ACM Trans. Netw.*, vol. 10, no. 2, pp. 217–231, Apr. 2002.

[22] L. Breslau and S. Shenker, "Best-effort versus Reservations: A simple comparative analysis," in *ACM SIGCOMM*, 1998, pp. 3–16.

[23] J. L.-T. Park, J.-W. Baek, and W.-K. Hong, "Management of service level agreements for multimedia Internet service using a utility model," *IEEE Commun. Mag.*, vol. 39, no. 5, pp. 100–106, May 2001.

[24] A. Elwalid and D. Mitra, "Effective bandwidth of general Markovian traffic sources and admission control of high-speed networks," *IEEE/ACM Trans. Netw.*, vol. 1, no. 3, pp. 329–344, Jun. 1993.

[25] A. Elwalid, D. Mitra, and R. H. Wentworth, "A new approach for allocating buffers and bandwidth to heterogeneous, regulated traffic in an atm node," *IEEE J. Sel. Areas Commun.*, vol. 13, no. 6, pp. 1115–1127, Aug. 1995.

[26] J. K. MacKie-Mason and H. Varian, "Pricing congestible network resources," *IEEE J. Sel. Areas Commun.*, vol. 13, no. 7, pp. 1141–1149, Sep. 1995.

[27] G. D. Veciana, C. Courcoubetis, and J. Walrand, "Decoupling bandwidths for networks: a decomposition approach to resource management," in *Proc. IEEE INFOCOM*, vol. 2, 1994, pp. 466–473.

[28] K. Kumaran, M. Mandjes, and A. Stolyar, "Convexity properties of loss and overflow functions," *Oper. Res. Lett.*, vol. 31, no. 2, pp. 95–100, 2003.

[29] J.-B. Hiriart-Urruty and C. Lemaréchal, *Convex Analysis and Minimization Algorithms*. New York: Springer-Verlag, 1993.

[30] R. K. Sundaram, *A First Course in Optimization Theory*. Cambridge, MA: Cambridge Univ. Press, 1996.

[31] D. P. Bertsekas, A. Nedic, and A. E. Ozdaglar, *Convex Analysis and Optimization*. Belmont, MA: Athena Scientific, 2003.

[32] D. P. Bertsekas, *Nonlinear Programming*, 2nd ed. Belmont, MA: Athena Scientific, 1999.

[33] N. Jin and S. Jordan, "The effect of bandwidth and buffer pricing on resource allocation and QoS," *Internet Economics, Special Issue Computer Networks*, vol. 46, no. 1, pp. 53–71, 2004.

[34] N. Jin and S. Jordan, "Sensitivity of optimal quality of service to bandwidth and buffer prices," in *Proc. IEEE Conf. Decision and Control*, Dec. 2003, pp. 1580–1585.

**Nan Jin** (S'04) received the B.S degree from Nanjing University, Nanjing, China, in 1997, and the M.S. degree from the Acoustics Institute, Chinese Academy of Sciences, Beijing, China, in 2000, both in electrical engineering. She is currently working toward the Ph.D. degree in the Department of Electrical Engineering and Computer Science, University of California, Irvine.

Her research interests include Internet pricing, system modeling of communication networks, network performance evaluation, protocols, and architecture design.

**Gayathri Venkitachalam**, photograph and biography not available at the time of publication.

**Scott Jordan** (S'86–M'90) received the B.S./A.B., M.S., and Ph.D. degrees from the University of California, Berkeley, in 1985, 1987, and 1990, respectively.

From 1990 until 1999, he served as an faculty member at Northwestern University, Evanston, IL. Since 1999, he has served as an faculty member at the University of California, Irvine. His research interests currently include pricing and differentiated services in the Internet, and resource allocation in wireless multimedia networks.