Optimization Flow Control, I: Basic Algorithm and Convergence

Steven H. Low, *Senior Member, IEEE* and David E. Lapsley

Abstract— **We propose an optimization approach to flow control where the objective is to maximize the aggregate source utility over their transmission rates. We view network links and sources as processors of a distributed computation system to solve the dual problem using gradient projection algorithm. In this system sources select transmission rates that maximize their own benefits, utility minus bandwidth cost, and network links adjust bandwidth prices to coordinate the sources' decisions. We allow feedback delays to be different, substantial and time–varying, and links and sources to update at different times and with different frequencies. We provide asynchronous distributed algorithms and prove their convergence in a static environment. We present measurements obtained from a preliminary prototype to illustrate the convergence of the algorithm in a slowly time–varying environment.**

Keywords— **Optimization flow control, congestion pricing, gradient projection, asynchronous algorithm, Convergence**

I. Introduction

It seems better to serve elastic traffics [31] with variable bandwidth using, in the context of ATM for instance, Available Bit Rate (ABR) rather than Constant Bit Rate (CBR) service. Indeed this folklore can be formally proved in the following abstract model: suppose a network offers fixed and variable bandwidth to a set of elastic sources and price them according to excess demand, and the sources freely purchase them to maximize their own benefits. The interpretation is that a source that desires only fixed bandwidth in the model would subscribe to CBR in practice and a source that desires both fixed and variable bandwidth would subscribe to ABR with a minimum cell rate guarantee. We show in [23], [24] that at equilibrium, where all sources are at their optimality and demand equals supply, *every* source desires a *strictly* positive amount of variable bandwidth. This observation provides perhaps another motivation for end–to–end flow control, for reactive flow control, where sources adjust their transmission rates in response to changes in network conditions, is a practical way to provision variable bandwidth.

The purpose of this paper is to propose an optimization approach to flow control, where the control mechanism is derived as a means to optimize a global measure of network performance. We will present synchronous and asynchronous algorithms, and prove their convergence in a static network. We will then describe a prototype and present experimental measurements to illustrate the algorithm's convergence in a slowly time–varying environment.

A. Summary

Consider a network that consists of a set L of unidirectional links of capacities c_l, $l \in L$. The network is shared by a set S of sources, where source s is characterized by a utility function $U_s(x_s)$ that is concave increasing in its transmission rate x_s. The goal is to calculate source rates that maximize the sum of the utilities $\sum_{s\in S} U_s(x_s)$ over x_s subject to capacity constraints. Solving this problem centrally would require not only the knowledge of all utility functions, but worse still, complex coordination among potentially all sources due to coupling of sources through shared links. Instead we propose a decentralized scheme that eliminates this requirement and adapts naturally to changing network conditions. The key is to consider the dual problem whose structure suggests treating the network links and the sources as processors of a distributed computation system to solve the dual problem using gradient projection method. Each processor executes a local algorithm, communicates its computation result to others, and the cycle repeats.

The algorithm takes the familiar form of reactive flow control. Based on the local *aggregate* source rate each link $l \in L$ calculates a 'price' p_l for a unit of bandwidth at link l. A source s is fed back the scalar price $p^s = \sum p_l$, where the sum is taken over all links that s uses, and it chooses a transmission rate x_s that maximizes its own benefit $U_s(x_s) - p^s x_s$, utility minus the bandwidth cost. These individually optimal rates $(x_s(p^s), s \in S)$ may not be socially optimal for a general price vector $(p_l, l \in L)$, i.e., they may not maximize the aggregate utility. The algorithm it-

Appeared in IEEE/ACM Transactions on Networking, 7(6), pp. 861–874, Dec. 1999. This is a living document, with corrections and acknowledgments, that is available at: http://netlab.caltech.edu/pub.

S. H. Low is with the Department of Electrical & Electronic Engineering, University of Melbourne, Australia, Email: slow@ee.mu.oz.au. Work done when D. E. Lapsley was with the Department of Electrical & Electronic Engineering, University of Melbourne, Australia. He is now with Ericsson, Australia, Email: lapsley@ee.mu.oz.au. The first author acknowledges the support of the Australian Research Council under grants S499705 and A49930405, the second author acknowledges the Australian Commonwealth Government and ATERB for their scholarships, and both acknowledge the financial support of Melbourne IT, Australia.

eratively approaches a price vector $(p_l^*, l \in L)$ that aligns individual and social optimality such that $(x_s(p^{*s}), s \in S)$ indeed maximizes the aggregate utility.

The algorithm is *partially asynchronous* [5, Chapter 6] in which the sources and links may compute based on outdated information, they may communicate at different times and with different frequencies, and the communication delays may be substantial, different and time–varying. We prove that as long as the intervals between updates are bounded the algorithm converges to yield the optimal rate.

In equilibrium sources that share the same links do not necessarily equally share the available capacity. Rather their shares reflect how they value the resources as expressed by their utility functions and how their usage implies a cost on others. This could be a basis to provide differentiated services in terms of different rate allocations.

The basic algorithm is derived and its convergence proved in a static environment where link capacities and the set of active sources remain unchanged. The algorithm generalizes directly to the case of time-varying environment. We present measurements from our prototype that illustrate the convergence of the algorithm when network condition changes.

The paper is structured as follows. In Section II we present the optimization problem and its dual that motivate our approach. In Section III we derive a synchronous algorithm and describe its convergence. This algorithm and its convergence proof are extended to an asynchronous setting in Section IV. In Section V we remark on fairness and pricing. In Section VI we present experimental results on convergence obtained from our prototype. Proofs of convergence are in the two Appendices.

B. Extensions

We now comment on past works and extensions. The basic algorithm has been presented in [20] and a preliminary prototype is briefly discussed in [19]. In this paper we analyze its convergence and fairness properties through analysis and implementation. The basic algorithm requires communication of link prices to sources and source rates to links, and hence cannot be implemented on the Internet. This communication requirement is greatly simplified in [25], [21], as follows. In [25] we describe a way for links to estimate source rates using local information and prove that optimality is still maintained. This eliminates the need for explicit communication from sources to links. In the reversed direction we proposed a method in [21] that accomplishes the communication from links to sources using only binary feedback. This can be implemented using the proposed ECN (Explicit Congestion Notification) bit in the IP header [9], [27]. These two simplifications are combined into a flow control scheme we call REM (Random Early Marking), a variant of RED [10], that not only stabilizes network queues but also tracks a global optimum. REM is made more robust in the face of large feedback delays by having links take weighted averages of past prices [1]. REM and its enhancements are explained in Part II of this paper.

The value of the optimization model presented in this paper is twofold. First, though it may not be possible, nor critical, that optimality is exactly attained in a real network, the optimization framework offers a means to explicitly steer the *entire* network towards a desirable operating point. We will see below that flow control can be regarded as a distributed computation over the network, and hence the behavior of the network *as a whole* becomes easily understandable. Second it is useful to treat practical flow control schemes simply as implementations of a certain optimization algorithm. The optimization model then makes possible a systematic method to design and refine these schemes, where modifications to a flow control mechanism are guided by modifications to the optimization algorithm. For instance, it is well known that Newton algorithm has much faster convergence than gradient projection algorithm. By replacing the gradient projection algorithm presented in this paper by the Newton algorithm we derive in [2] a practical Newton–like flow control scheme that can be proved to maintain optimality, has the same communication requirement as the basic scheme here but enjoys a much better convergence property. We have also applied pole placement technique in linear control to the model here to stabilize its transient in the face of large feedback delays. This has led to a more robust REM presented in [1].

C. Related works

An extensive literature exists on flow control, including the original TCP flow control [15] and recent enhancement in [10], the binary feedback schemes of, e.g., [28], [6], two–bit feedback scheme of [22], the control theoretic approach of, e.g., [3], [29], [7], etc. Also see a recent review in [14].

A key premise of optimization based flow control [11], [13], [8], [16], [17], [12], [20], [19], [25], [21] is that sources with different valuation of bandwidth should react differently to network congestion. All these works motivate flow control by an optimization problem and derive their con-

trol mechanisms as solutions to the optimization problem. They differ in their choice of objective functions or their solution approaches, and result in rather different flow control mechanisms to be implemented at the sources and the network links. Our model is closest to that in [16], [17]. Indeed both their work and ours have the same objective of maximizing aggregate source utility. In [16], [17] this objective is decomposed into optimization subproblems for the network and the sources, and they propose a different mechanism for its solution where each source chooses a willingness to pay and the network allocates rates to these sources in a way that is proportionally fair. An interesting feature of their approach is that it allows the users to decide their payments and receive what the network allocates, whereas in our approach, the users decide their rates and pay what the network charges. See a more detailed comparison in Remark 3 after Algorithm A1 in Section III.

II. The optimization problem

In this section we state the optimization problem that leads to our congestion control framework, and suggest a solution approach. Algorithms to solve the problem will be given in the following sections.

A. Primal problem

Consider a network that consists of a set $L = \{1, \ldots, L\}$ of unidirectional links of capacity c_l, $l \in L$. The network is shared by a set $S = \{1, \ldots, S\}$ of sources. Source s is characterized by four parameters $(L(s), U_s, m_s, M_s)$. The path $L(s) \subseteq L$ is a set of links that source s uses, $U_s : \Re_+ \to \Re$ is a utility function, $m_s \geq 0$ and $M_s < \infty$ are the minimum and maximum transmission rates, respectively, required by source s. Source s attains a utility $U_s(x_s)$ when it transmits at rate x_s that satisfies $m_s \leq x_s \leq M_s$. We assume U_s is increasing and strictly concave in its argument. Let $I_s = [m_s, M_s]$ denote the range in which source rate x_s must lie and $I = (I_s, s \in S)$ be the vector. For each link l let $S(l) = \{s \in S \mid l \in L(s)\}$ be the set of sources that use link l. Note that $l \in L(s)$ if and only if $s \in S(l)$.

Our objective is to choose source rates $x = (x_s,\ s \in S)$ so as to:

$$\textbf{P:} \quad \max_{x_s \in I_s} \quad \sum_s U_s(x_s) \tag{1}$$

$$\text{subject to} \quad \sum_{s \in S(l)} x_s \ \leq \ c_l, \quad l = 1, \ldots, L. \tag{2}$$

The constraint (2) says that the aggregate source rate at any link l does not exceed the capacity. A unique maximizer, called the primal optimal solution, exists since the objective function is strictly concave, and hence continuous, and the feasible solution set is compact.

Though the objective function is separable in x_s, the source rates x_s are coupled by the constraint (2). Solving the primal problem (1–2) directly requires coordination among possibly all sources and is impractical in real networks. The key to a distributed and decentralized solution is to look at its dual.

B. Dual problem

Define the Lagrangian

$$\begin{aligned} L(x,p) &= \sum_s U_s(x_s) - \sum_l p_l \Big(\sum_{s \in S(l)} x_s - c_l \Big) \\ &= \sum_s \Big(U_s(x_s) - x_s \sum_{l \in L(s)} p_l \Big) + \sum_l p_l c_l. \end{aligned}$$

Notice that the first term are separable in x_s, and hence $\max_{x_s} \sum_s (U_s(x_s) - x_s \sum_{l \in L(s)} p_l) = \sum_s \max_{x_s} (U_s(x_s) - x_s \sum_{l \in L(s)} p_l)$. The objective function of the dual problem is thus (e.g., [5, Section 3.4.2], [26])

$$D(p) \ = \ \max_{x_s \in I_s} L(x,p) \ = \ \sum_s B_s(p^s) + \sum_l p_l c_l$$

where

$$B_s(p^s) = \max_{x_s \in I_s} \ U_s(x_s) - x_s p^s \tag{3}$$

$$p^s = \sum_{l \in L(s)} p_l \tag{4}$$

and the dual problem is:

$$\textbf{D:} \quad \min_{p \geq 0} \quad D(p). \tag{5}$$

The first term of the dual objective function $D(p)$ is decomposed into S separable subproblems (3–4). If we interpret p_l as the price per unit bandwidth at link l then p^s is the total price per unit bandwidth for all links in the path of s. Hence $x_s p^s$ represents the bandwidth cost to source s when it transmits at rate x_s, and $B_s(p^s)$ represents the maximum benefit s can achieve at the given price p^s. We shall see below that this scalar p^s summarizes all the congestion information source s needs to know. A source s can be induced to solve maximization (3) by bandwidth charging. For each p, a unique maximizer, denoted by $x_s(p)$, exists since U_s is strictly concave.

Since U_s are concave and the constraints (2) are linear there is no duality gap and dual optimal prices, which are Lagrange multipliers, exist [4, Propositions 5.2.1 and 5.1.4]. In general $(x_s(p), s \in S)$ may not be primal optimal, but if $p^* \geq 0$ is dual optimal then $(x_s(p^*), s \in S)$ is indeed primal

optimal provided also that it is primal feasible and complementary slackness is satisfied [4, Proposition 5.1.5]. Hence we will focus on solving the dual problem (5). Once we have obtained p^* the primal optimal source rates $x^* = x(p^*)$ can be computed by individual sources s by solving (3), a simple maximization; see (6) below. The important point to note is that, given p^*, individual sources s can solve (3) *separately without the need to coordinate with other sources.* In a sense p^* serves as a coordination signal that aligns individual optimality of (3) with social optimality of (1).

C. Notations and assumptions

Unless otherwise specified, z usually denotes a vector whose ith component is some z_i defined before z is introduced. For a vector or matrix z, z^T denotes its transpose. For a set A, $|A|$ denotes its cardinality. For a vector z, $||z||_2$ denotes the Euclidean norm, $||z||_1 = \sum_i |z_i|$, $||z||_\infty = \max_i |z_i|$, and $||z||$ without subscript denotes any norm. For a matrix z, $||z||$ denotes the corresponding induced norm.

We will sometimes represent the information $L(s)$ and $S(l)$ in terms of a routing matrix R whose (l,s)th entry is $R_{ls} = 1$ if $l \in L(s)$ (or $s \in S(l)$), and 0 otherwise.

For each source s, $p^s = \sum_{l \in L(s)} p_l$, the sth component of $p^T R$, is the (path) bandwidth price that s faces. For each link l, $x^l = \sum_{s \in S(l)} x_s$, the lth component of Rx, is the aggregate source rate at link l.

Let $x_s(p)$ be the unique maximizer in (3). We will abuse notation and use $x_s(\cdot)$ both as a function of scalar price $p \in \Re_+$ and of vector price $p \in \Re_+^{|L|}$. When p is a scalar, by the Kuhn–Tucker theorem, $x_s(p)$ is given by

$$x_s(p) \quad = \quad [U_s^{'-1}(p)]_{m_s}^{M_s} \tag{6}$$

where $[z]_a^b = \min\{\max\{z, a\}, b\}$. Here $U_s^{'-1}$ is the inverse of U_s', which exists over the range $[U_s'(M_s), U_s'(m_s)]$ since U_s' is continuous and U_s *strictly* concave (condition C1 below). Indeed $x_s(p)$ is the demand function in microeconomics. It is illustrated in Figure 1. When p is a vector, $x_s(p) = x_s(p^s) = x_s(\sum_{l \in L(s)} p_l)$. The meaning should be clear from the context. Let $x(p) = (x_s(p), s \in S)$.

Assumptions on the utility functions are:

C1: On the interval $I_s = [m_s, M_s]$, the utility functions U_s are increasing, strictly concave, and twice continuously differentiable. For feasibility, assume $\sum_{s \in S(l)} m_s \le c_l$ for all l.

C2: The curvatures of U_s are bounded away from zero on I_s: $-U_s''(x_s) \ge 1/\overline{\alpha}_s > 0$ for all $x_s \in I_s$.

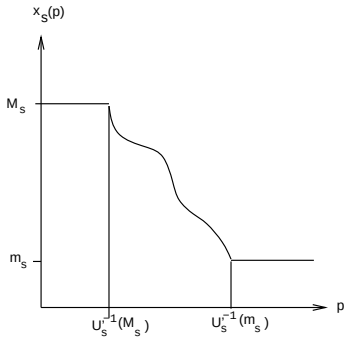


Fig. 1. Source rate $x_s(p)$ as a function of (scalar) price p.

III. Synchronous Distributed Algorithm

In this section we present the basic synchronous algorithm and prove its convergence under conditions C1 and C2. This algorithm and its convergence proof form the basis of the asynchronous algorithm and the proof of its convergence, to be described in the next section.

We will solve the dual problem using gradient projection method (e.g., [26], [5]) where link prices are adjusted in opposite direction to the gradient $\nabla D(p)$:

$$p_l(t+1) \quad = \quad [p_l(t) - \gamma \frac{\partial D}{\partial p_l}(p(t))]^+ \tag{7}$$

Here $\gamma > 0$ is a stepsize, and $[z]^+ = \max\{z, 0\}$. Recall that $x_s(p)$ denotes the unique maximizer in (3). Then

$$D(p) \quad = \quad \sum_s (U_s(x_s(p)) - x_s(p)p^s) + \sum_l p_l c_l.$$

Since U_s are *strictly* concave, $D(p)$ is continuously differentiable ([5, pp.669]) with derivatives given by

$$\frac{\partial D}{\partial p_l}(p) \quad = \quad c_l - x^l(p) \tag{8}$$

where $x^l(p) := \sum_{s \in S(l)} x_s(p)$ is the aggregate source rate at link l. Substituting (8) into (7) we obtain the following price adjustment rule for link $l \in L$:

$$p_l(t+1) \quad = \quad [p_l(t) + \gamma(x^l(p(t)) - c_l)]^+ \tag{9}$$

This indeed is consistent with the law of supply and demand: if the demand $x^l(p(t)) = \sum_{s \in S(l)} x_s(p(t))$ for bandwidth at link l exceeds the supply c_l, raise price $p_l(t)$; otherwise reduce price $p_l(t)$. As with (3) the decentralized nature of (9) is striking: though the dual problem is not separable in p, given aggregate source rate $x^l(p(t))$ that goes through link l, the adjustment algorithm (9) is completely distributed and can be implemented by individual links using only local information.

This suggests *treating the network links l and the sources s as processors in a distributed computation system* to solve the dual problem (5). In each iteration, sources s individually solve (3) and communicate their results $x_s(p)$ to links $l \in L(s)$ on its path. Links l then update their prices p_l according to (9) and communicate the new prices to sources s, and the cycle repeats. We summarize.

Algorithm A1: Synchronous Gradient Projection
Link l's algorithm:
At times $t = 1, 2, \ldots$, link l:
1. Receives rates $x_s(t)$ from all sources $s \in S(l)$ that go through link l.
2. Computes a new price

$$p_l(t+1) \quad = \quad [p_l(t) + \gamma(x^l(t) - c_l)]^+$$

where $x^l(t) = \sum_{s \in S(l)} x_s(t)$.
3. Communicates new price $p_l(t+1)$ to all sources $s \in S(l)$ that use link l.
Source s's algorithm:
At times $t = 1, 2, \ldots$, source s:
1. Receives from the network the sum $p^s(t) = \sum_{l \in L(s)} p_l(t)$ of link prices in its path.
2. Chooses a new transmission rate $x_s(t+1)$ for the next period: [1]

$$x_s(t+1) \quad = \quad x_s(p^s(t))$$

where $x_s(\cdot)$ is given by (6).
3. Communicates new rate $x_s(t+1)$ to links $l \in L(s)$ in its path.

Remarks:
1. As noted in Section I-B a link l requires the aggregate source rates $x^l(t)$ and a source s the path price $p^s(t)$ for their updates. This communication can be greatly simplified, leading to the REM algorithm discussed in [21], [1].
2. Newton's method, where the direction of price adjustment is the negative gradient scaled by the inverse of the Hessian $\nabla^2 D$, typically has a much faster convergence than the gradient projection algorithm. However, this, according to Lemma 3 below, requires that a link know the (second derivative of the) utility functions of nonlocal sources, and hence is not practical. In [2] we describe and prove the optimality of a practical Newton–like algorithm that enjoys a better performance.

[1] Here, we abuse notation and use $x_s(\cdot)$ both as a function of time, to denote source rate at time t under Algorithm A1, and as a function of price given by (6). The meaning should be clear from the context.

3. Our work is closely connected to Kelly's as described in [16], [17], [12]. Both solve the same optimization problem (1–2), but they differ in the solution approach which leads to different flow control algorithms, which in turn lead to different marking implementation of the algorithms.
The approach taken in [16], [17] decomposes problem (1–2) into a user subproblem and a network subproblem. The user subproblem is to choose a willingness–to–pay w_s given the path price p^s in order to maximize its benefit, and the network subproblem is to choose source rates $(x_s, s \in S)$ given users' willingness–to–pay vector $(w_s, s \in S)$ in order to maximize $\sum_s w_s \log x_s$. It is shown in [16] that there exist path prices $(p^s, s \in S)$, source rates $x = (x_s, s \in S)$ and willingness–to–pay $(w_s, s \in S)$ with $w_s = p^s x_s$ such that w_s solves user s's subproblem, and x solves the network subproblem and the system (primal) problem (1–2). Our approach is simply to solve the dual of problem (1–2) using gradient projection algorithm.
Major effort in [17] is to solve the network subproblem, or equivalently, the dual of the network subproblem (not to be confused with our dual problem **D** in (5))[2]. To this end they propose the following primal algorithm

$$\frac{d}{dt}x_s(t) \quad = \quad \gamma\left(w_s - x_s(t)\sum_{l \in L(s)} p_l(t)\right) \qquad (10)$$

$$p_l(t) \quad = \quad f_l\left(\sum_{s \in S(l)} x_s(t)\right) \qquad (11)$$

to solve a relaxation of the network subproblem, and the following dual algorithm to solve a relaxation of its dual (given w_s)

$$\frac{d}{dt}p_l(t) \quad = \quad \gamma\left(\sum_{s \in S(l)} x_s(t) - q_l(p_l(t))\right) \qquad (12)$$

$$x_s(t) \quad = \quad \frac{w_s}{\sum_{l \in L(s)} p_l(t)} \qquad (13)$$

The rate adjustment (10) has the attractive feature of multiplicative decrease and additive increase common in several popular flow control schemes. Either algorithm (10–11) or (12–13) can be used to compute the equilibrium source rates.
Our gradient projection algorithm is closer to Kelly's dual algorithm (12–13). Indeed in the special case where $U_s(x_s) = w_s \log x_s$ our algorithm A1 reduces to (12–13), provided we take $q_l(p_l(t)) = c_l$ in (12) though this choice of $q_l(\cdot)$ does not satisfy certain conditions required for the stability proof in [17].

[2] Of course if $U_s(x_s) = w_s \log x_s$, then our primal problem (1–2) and its dual (5) are equivalent to their network subproblem and its dual.

In [17] the nonnegativity constraint on the source rates and link prices is relaxed in (10–13). This allows a simple and elegant stability proof via a Lyapunov argument. In our case, the projection to the positive quadrant complicates the stability analysis considerably (see the Appendices). In a sense the dual objective function $D(p)$ can be thought of as a Lyapunov function for the discrete time system (9) *provided the stepsize γ is sufficiently small.*

In [12] a marking scheme is proposed to implement the primal algorithm (10–11), where the marks convey to a source the charge $x_s(t)p^s(t)$ and the source adjusts its rate to equalize the charge with its willingness–to–pay w_s. In Part II of this paper we describe a marking scheme that implements our algorithm where the marks allow a source to estimate its path price $p^s(t)$ that is needed in its rate adjustment. In view of the remark above this can also be regarded as a marking implementation of Kelly's dual algorithm (12–13) for a specific utility function.

Our first main result states that Algorithm A1 generates a sequence that approaches the optimal rate allocation, provided conditions C1–C2 are satisfied. These conditions imply that ∇D is Lipschitz which guarantees the convergence of gradient projection algorithms. Define $\overline{L} := \max_{s\in S} |L(s)|$, $\overline{S} := \max_{l\in L} |S(l)|$, and $\overline{\alpha} := \max\{\overline{\alpha}_s,\ s \in S\}$. In words $\overline{L}$ is the length of a longest path used by the sources, $\overline{S}$ is the number of sources sharing a most congested link, and $\overline{\alpha}$ is the upper bound on all $-U_s''(x_s)$; see Section II-C.

Theorem 1: Suppose assumptions C1–C2 hold and the stepsize satisfies $0 < \gamma < 2/\overline{\alpha}\overline{L}\overline{S}$. Then starting from any initial rates $m \le x(0) \le M$ and prices $p(0) \ge 0$, every accumulation point (x^*, p^*) of the sequence $(x(t), p(t))$ generated by Algorithm A1 is primal–dual optimal.

Proof. See Appendix I. ∎

Though there is a unique maximizer x^* to the primal problem there may be multiple dual optimal prices because at optimality only the *sum* of link prices is constrained, $U_s'(x_s^*) = \sum_{l\in L(s)} p_l^*$. Theorem 1 does not guarantee convergence to a unique pair (x^*, p^*), though any convergent subsequence yields the optimal rate allocation x^*.

We now comment on the convergence rate when the dual optimal price p^* is unique. Then letting $\tilde{p}(t) = p(t) - p^*$ be the deviation from the unique limit point, it can be shown that the price process $p(t)$ linearized around p^* satisfies

$$\tilde{p}(t+1) \quad = \quad (I - \gamma RB(p^*)R^T)\tilde{p}(t)$$

where $B(p) = Diag\ (\beta_s(p), s \in S)$ is an $S \times S$ diagonal matrix with diagonal elements $\beta_s(p)$ defined by (24) in Appendix I. Hence the rate of convergence near the equilibrium is determined by the spectral radius of the (positive definite for small γ) matrix $I - \gamma RB(p^*)R^T$.

IV. Asynchronous distributed algorithm

The synchronous model of the last section assumes that updates at the sources and the links are synchronized to occur at times $t = 1, 2, \ldots$. In this section we will extend the model to an asynchronous setting which better resembles the reality of large networks. In such networks sources may be located at different distances from the network links. Network state (prices in our case) may be probed by different sources at different rates, e.g., the Resource Management (RM) cells in an ATM networks are sent at different rates by different sources. Feedbacks may reach different sources after different, and variable, delays. These complications make our distributed computation system consisting of links and sources asynchronous. In such a system some processors may compute faster and execute more iterations than others, some processors may communicate more frequently than others, and the communication delays may be substantial and time–varying.

We now present the asynchronous version of Algorithm A1 and prove its convergence. Our asynchronous model and the convergence proof follow the approach of [32] and belong to the class of partially asynchronous algorithms discussed in [5, Chapter 7]. See comments after Theorem 2 below.

Let $T_l^1 \subseteq \{1, 2, \ldots, \}$ be a set of times at which link l adjusts its price based on its current knowledge of source rates. At times $t \notin T_l^1$, link prices are unchanged, i.e., $p_l(t+1) = p_l(t), t \notin T_l^1$. Similarly let $T_s^2 \subseteq \{1, 2, \ldots, \}$ be a set of times at which source s updates its rate. At times $t \notin T_s^2$, $x_s(t+1) = x_s(t)$.

At times $t \in T_l^1$ link l computes an estimate $\lambda_l(t)$ of the gradient and updates its price according to

$$p_l(t+1) \quad = \quad [p_l(t) - \gamma\lambda_l(t)]^+ \tag{14}$$

The estimate $\lambda_l(t)$ is computed using aggregate past source rates at link l (cf. (8)):

$$\lambda_l(t) \quad = \quad c_l - \hat{x}^l(t) \tag{15}$$

$$\hat{x}^l(t) \quad = \quad \sum_{s\in S(l)} \hat{x}_{ls}(t) \tag{16}$$

$$\hat{x}_{ls}(t) \quad = \quad \sum_{t'=t-t_0}^{t} a_{ls}(t', t)\ x_s(t'), \qquad s \in S(l) \tag{17}$$

with

$$\sum_{t'=t-t_0}^{t} a_{ls}(t',t) \quad = \quad 1, \qquad \forall t, \forall l, s \text{ with } s \in S(l). \quad (18)$$

In (15–16), $\hat{x}^l(t) = \sum_{s \in S(l)} \hat{x}_{ls}(t)$ is the aggregate estimated source rates. The estimate $\hat{x}_{ls}(t)$ of individual source rate is the weighted average of its past rates (see (17–18)). It depends on (l, s, t) and can be different for different link–source pairs (l, s) and at different times t. It includes the possibility of information arriving at link l out of order. This model is very general and allows in particular the following two popular types of policies:

- **Latest data only:** only the last received rate $x_s(\tau)$, for some (possibly *unknown*) $\tau \in \{t - t_0, ..., t\}$, is used to estimate $\hat{x}_{ls}(t)$, i.e., $a_{ls}(t', t) = 1$ if $t' = \tau$ and 0 otherwise.
- **Latest average:** only the average over the latest k received rates is used in estimate $\hat{x}_{ls}(t)$, i.e., $a_{ls}(t', t) > 0$ for $t' = \tau - k + 1, \ldots, \tau$ and 0 otherwise, for some (possibly *unknown*) $\tau \in \{t - t_0, ..., t\}$.

The interpretation in both cases is that rates $x_s(t')$ for $t' > \tau$ have not been received at link l by time t, and rates $x_s(t')$ for $t' < \tau$ or for $t' \leq \tau - k$ have been discarded.

At times $t \in T_s^2$ source s computes an estimate $\hat{p}^s(t)$ of path price and updates its rate according to

$$x_s(t) \quad = \quad x_s(\hat{p}^s(t)) \quad (19)$$

where $x_s(\cdot)$ is given by (6), and

$$\hat{p}^s(t) \quad = \quad \sum_{l \in L(s)} \hat{p}_{ls}(t) \quad (20)$$

$$\hat{p}_{ls}(t) \quad = \quad \sum_{t'=t-t_0}^{t} b_{ls}(t',t)\, p_l(t'), \qquad l \in L(s) \quad (21)$$

with

$$\sum_{t'=t-t_0}^{t} b_{ls}(t',t) \quad = \quad 1, \qquad \forall t, \forall l, s \text{ with } l \in L(s). \quad (22)$$

In (19–20) the source computation is the same as in the synchronous case except that it is based on its current estimate $\hat{p}^s(t)$ of path prices. As in the link algorithm the estimated link price $\hat{p}_{ls}(t)$ is obtained by 'averaging' over the past available prices (see (21–22)), and can depend on (l, s, t). Again the 'averaging' model is very general and include the policy of using only the last received price or the average over the last k prices; see above.

Note that (17) and (21) above tacitly assume that the one-way delay between any (l, s) pair is no more than t_0.

We now present the asynchronous algorithm A2. A2 is similar to A1 except that communications are not coordinated and computations are carried out using possibly outdated information.

Algorithm A2: Asynchronous Gradient Projection
Link l's algorithm:
1. From time to time link l receives source rates from sources that go through link l. Link l replaces the oldest rates in its local memory with the newly received rates.
2. At each update time $t \in T_l^1$, link l computes an estimate $\lambda_l(t)$ of $\partial D / \partial p_l(p(t))$ (see (15–18) above) and adjusts its price according to

$$p_l(t+1) \quad = \quad [p_l(t) - \gamma \lambda_l(t)]^+$$

At times $t \notin T_l^1$, $p_l(t+1) = p_l(t)$.
3. From time to time link l communicates the current price to sources that go through link l.
Source s's algorithm:
1. From time to time source s receives bandwidth prices fed back from links in its path. Source s replaces the oldest prices in its local memory with the newly received ones.
2. At each update time $t \in T_s^2$ source s chooses a new rate based on its current estimate $\hat{p}^s(t)$ of path price (see (20–22) above):

$$x_s(t+1) \quad = \quad x_s(\hat{p}^s(t))$$

It then transmits at this rate until the next update, i.e., $x_s(t+1) = x_s(t)$ for $t \notin T_s$.
3. From time to time source s communicates the current source rate to links in its path.

This concludes our description of Algorithm A2. We now turn to its convergence.

Let $\bar{x}_s(t)$ be the *ideal* rate if source s knows the exact price $p^s(t)$ at time t instead of its estimate $\hat{p}^s(t)$:

$$\bar{x}_s(t) \quad = \quad x_s(p^s(t)) \quad (23)$$

where $x_s(\cdot)$ is given by (6) and $p^s(t)$ evolves according to Algorithm A2. Our second main result states that the difference between the various estimates and their true values converges to zero and that Algorithm A2 yields the optimal rate allocation, provided the following additional assumption is satisfied:

C3: For all links l and sources s, the time between consecutive updates (i.e., the difference between consecutive elements of T_l^1 or T_s^2) is bounded.

Theorem 2: Suppose assumptions C1–C3 hold. Provided that the stepsize γ is sufficiently small, then starting from any initial rates $m \leq x(0) \leq M$ and prices $p(0) \geq 0$, every accumulation point (x^*, p^*) of the sequence $(x(t), p(t))$ generated by Algorithm A2 is primal–dual optimal. Moreover, for all sources s, the error in price estimation $||\hat{p}^s(t) - p^s(t)||$ and rate calculation $||\hat{x}_s(t) - \bar{x}_s(t)||$ converges to zero, and the error $||\lambda(t) - \nabla D(p(t))||$ in gradient estimation by the links converges to zero.

Proof. See Appendix II. ■

As in [32], the key to the proof is to show that the price adjustment (14) remains in the decent direction and hence the value of the dual objective function is decreased in each iteration. The proof in our case is somewhat more complicated because, since our minimization is a dual problem, the gradient estimate $\lambda(t)$ depends on previous prices $p(\tau), \tau \leq t$, in a more complex way through (15–22). Moreover a critical assumption there (equation (3.11) in [32] which is needed to derive their equations (A.6) and (A.9)), that is natural in the routing context there, has no equivalent in our context, and hence other properties of our algorithm need to be exploited in order to prove that descent direction is maintained (see Lemma 4(c–e) and Lemma 5 in the Appendix II).

V. Fairness, quasi–stationarity, and pricing

In this section we comment on some fairness and implementation issues.

A. Fairness

A proportionally fair rate vector is defined in [16] as a feasible rate vector $(\hat{x}_s, s \in S)$ such that for any other feasible vector $(x_s, s \in S)$, the aggregate of proportional changes is nonpositive:

$$\sum_{s \in S} \frac{x_s - \hat{x}_s}{\hat{x}_s} \quad \leq \quad 0$$

The primal optimal solution $(x^*_s, s \in S)$ is proportionally fair when all user utilities are logarithmic, $U_s(x_s) = \log x_s$, $s \in S$. As shown in [17], this follows from the optimality condition: for all feasible x,

$$\sum_{s \in S} \frac{\partial U_s}{\partial x_s}(x^*_s)(x_s - x^*_s) \quad = \quad \sum_{s \in S} \frac{x_s - x^*_s}{x^*_s} \quad \leq \quad 0$$

If user utilities are all equal but not necessarily logarithmic, then the following properties on homogeneous sources follow from (6).

Theorem 3: Suppose condition C1 holds, and, for all $s \in S$, $U_s(x_s) = U(x_s)$, $m_s = m$ and $M_s = M$. Let $(x^*_s, s \in S)$ be the primal optimal rate vector.

(a) If sources s_1 and s_2 share the same path, $L(s_1) = L(s_2)$, then $x^*_{s_1} = x^*_{s_2}$.

(b) If the path of s_1 is a subset of s_2, $L(s_1) \subseteq L(s_2)$, then $x^*_{s_1} \geq x^*_{s_2}$.

(c) More generally, suppose p is a dual optimal price vector. If $p^{s_1} \leq p^{s_2}$ then $x^*_{s_1} \geq x^*_{s_2}$, and equality holds if and only if $p^{s_1} = p^{s_2}$.

We now comment on these properties. Theorems 2 and 3(a) imply that sources that visit the same set of links but have different propagation delays will have equal equilibrium rate. If s_1 and s_2 share the same path but one has a higher marginal utility, say, $U'_{s_1}(x) \geq U'_{s_2}(x)$ for all x, then $x^*_1 \geq x^*_2$. Hence the choice of utility function implements priority among connections with the same path.

Theorem 3(b) implies that our scheme discriminates against long connections. We emphasize, however, that by 'long' we mean connections that go through more links, not necessarily those merely having higher propagation delays in accessing the network. This is natural from the perspective of maximizing the aggregate utility: since all utility functions are identical, the longer a connection the more resources it consumes for each unit of increase in aggregate utility, and hence short connections should be favored. If this is undesirable, it can be remedied by weighting the utility functions. Indeed almost any desirable rate vector can be attained in equilibrium by appropriate choice of utility functions; see Theorem 4 below.

Theorem 3(c) justifies treating the bandwidth price $p^s = \sum_{l \in L(s)} p_l$ as a measure of congestion that s faces: the higher the congestion the lower the rate.

A rate vector x^* is called *feasible* if it satisfies the capacity constraint (2). It is called *attainable* if there exist utility functions U_s that satisfy condition C1 for which the unique primal optimal rate vector is x^*. A link l is called *saturated with respect to* x^* if $\sum_{s \in S(l)} x^*_s = c_l$. Assume:

C4: every link l has a single–link connection, i.e., for each l, there exists a source $s(l)$ with $L(s(l)) = \{l\}$.

We can restrict utility functions to be of the form $a_s \log x_s$ or $-\frac{1}{2}(M_s - x_s)^2$ and choose the parameters a_s or M_s appropriately to achieve almost any desirable allocation in a static network.

Theorem 4: Suppose C1 and C4 hold, and suppose utility functions are $U_s(x_s) = a_s \log x_s$, $x_s > 0$, for all s (or $U_s(x_s) = -\frac{1}{2}(M_s - x_s)^2$, $0 \leq x_s \leq M_s$, for all s). Then a feasible rate vector x^* is attainable provided all links are

saturated and, for all s,

$$a_s = \sum_{l \in L(s)} \frac{x_s^*}{x_{s(l)}^*} a_{s(l)}$$

$$(\text{ or } \quad M_s - x_s = \sum_{l \in L(s)} M_{s(l)} - x_{s(l)} \ )$$

Proof. A feasible rate vector x^* is primal optimal if and only if the Kuhn–Tucker condition

$$U_s'(x_s^*) = \sum_{l \in L(s)} p_l^*$$

and the complementary slackness are satisfied. Since all links are saturated by assumption the complementary slackness holds. By C4, we can express $p_l^* = U_{s(l)}'(x_{s(l)}^*)$. Substituting this into the Kuhn–Tucker condition, using the appropriate utility functions, yields the theorem. ■

Theorem 4 implies in particular that a maxmin fair or proportionally fair rate can be attained by appropriate choice of utility functions.

B. Time-varying Environment

Algorithms A1 and A2 are derived, and their convergence proved, assuming that the link capacities c_l, the set S of sources and their utility functions U_s are unchanged. However the algorithms extend directly to the case when these quantities are time-varying. They have the important virtue of not requiring to be restarted when network condition changes.

Each source that comes on board executes the same source algorithm (A1 or A2) with the time invariant utility function $U_s(\cdot)$ replaced by the current utility $U_s(\cdot; t)$ at time t. Each link executes the same link algorithm, except that in computing $\nabla D(p(t))$ at time t in Step 2, the *current* link capacity $c_l(t)$ is used in place of the constant capacity c_l and the set $S(l; t)$ of currently active sources through link l is used in place of the constant set $S(l)$.

If the change in link capacities and sources is slow relative to the convergence of the algorithm, the algorithm tracks the moving optimal rates. This is illustrated by the experimental measurements presented in Section VI below.

C. Pricing and traffic control

Though network feedbacks are discussed in terms of bandwidth 'prices' they may or may not be part of the charge a user pays. If they do then bandwidth charging provides an incentive for the sources to choose socially (primal) optimal rates. In addition to encouraging efficient sharing of resources, pricing for network services also serves other functions. If congestion pricing interferes too much with these functions, then the 'prices' discussed in this paper, should be regarded simply as a control signal to guide sources' decisions.

VI. Experimental results

In this section we briefly summarize a user–space implementation of the basic algorithm and present experimental measurements that illustrate its convergence in a slowly time-varying environment. Detail description of the prototype can be found in [19].

A. Overview of implementation

Our experimental network consists of 2 IBM compatible PCs (Pentium 233 Mhz) running the FreeBSD 2.2.5 operating system. Each PC was equipped with 64 Mbytes of RAM and 100 Mbps PCI ethernet cards. The PCs were connected via ethernet. Implementing the protocol involved writing two applications: *ofc* client application, and *ofcd* routing demon. We refer to our algorithm as OFC.

Two instances of the *ofc* client application are required for each connection: a source instance operating in ACTIVE mode and a destination instance operating in PASSIVE mode. Whenever the OFC transport protocol is used the *ofcd* demon must be run on all computers that have OFC clients (sources or destinations) and on intermediate computers. The *ofc* client processes communicate with each other via the *ofcd* routing demons. All OFC clients transmit their packets to the routing demon on their host, which then either forwards the packet to another machine, or delivers it to a client process on the host. The OFC demons are also responsible for calculating the price on their outgoing links, and placing this price in special control packets as they pass through.

Each computer has a standard internet protocol stack consisting of TCP/UDP running on IP which sits above the network device drivers. The OFC protocol runs on top of the UDP layer, with OFC packets transported across the network on UDP connections. OFC packets are 500 bytes long and consist of a 10 byte header, 1 byte end of packet marker, and a 489 byte data payload. The header contains, among other things, fields that indicate payload type, bandwidth price, and source rate.

An in-kernel implementation of the protocol would have a better performance, but this would require recompiling the kernel of every machine on which we want to implement OFC. A user–space implementation is much more portable: we only need to recompile the application software and execute it on the target machine. We opted for portability over performance. We have tried a number of different ar-

chitectures and designs, and found that the design with the best performance was a single context, monolithic implementation (see [18]) where all of the packet processing was performed within a single thread. See [19] for more details.

B. Convergence

We now present two sets of experimental results and compare them with theoretical prediction. As expected, the bottlenecks of our testbed, which are links in our theoretical model, are not the transmission medium (ethernet) but the host processing. This set of bottlenecks can be represented by the *logical* network in Figure 2.

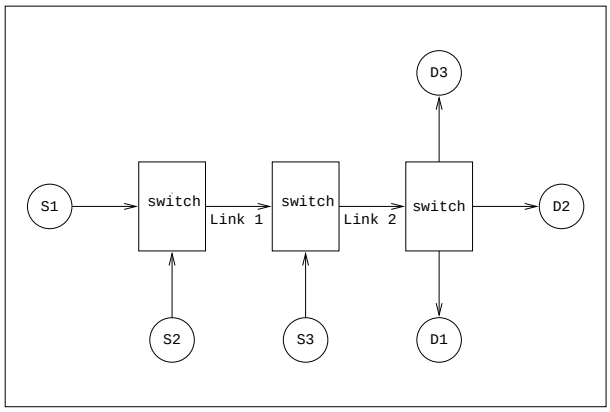


Fig. 2. Logical topology. Source Si transmits to destinations Di, $i = 1, 2, 3$.

Experiment 1: homogeneous sources. Each source transmitted data for a total of 120 s with their starting times staggered by intervals of 40 s: source 1 started transmitting at time 0, source 2 at time 40 s, and source 3 at time 80 s. The utility functions of the sources were set to $a_s \log(1 + x_s)$, with a_s equal to 1×10^4 for all sources $s = 1, 2, 3$. The stepsize γ used by the router to adjust its link prices was set to 1.5×10^{-2}. Client applications as well as routers dumped receive/transmit statistics to file every 500 ms. The routers also calculated new prices every 500 ms. The target bandwidth was set at 200 packets per 500 ms measuring interval (1.6 Mbps).

Figure 3(a) shows the destination receive rates for each source. The sum of the traces is constant at about 200 packets per measuring interval which was the target value set at the routers. The destination receive rates varied in accordance with the changes in link prices in Figure 3(b). From 80 s to 120 s when all sources were active each destination was receiving data at the same rate, and that the longer connections S1-D1 and S2-D2 were not discriminated against. This was because link 1 was not saturated and hence had zero price.

Also shown in both graphs is the steady state rate and price calculated by solving the primal and dual problems in Section II. Note that in Figure 3(b) the measured prices are link prices and the theoretical price is the path price which should equal the sum of the link prices. We see that the prototype behaved as expected and that, provided network conditions vary slowly, our algorithm tracks the optimum.

Experiment 2: heterogeneous sources. The setup in this experiment is the same as in Experiment 1, except that the utility function of source 3 has $a_3 = 2 \times 10^4$, double that of sources 1 and 2.

Figures 4(a) and 4(b) show respectively the destination receive rates and the link prices. As in Experiment 1, the source rates adjusted dynamically as new sources started or stopped transmitting. Again, note the close fit between the theoretical and the measured traces. Due to its higher marginal utility, source 2 gained twice as much bandwidth as each of sources 1 and 3, and caused the price on link 2 to be pushed higher than in Experiment 1. It suggests that our algorithm can support differentiated service in terms of different shares of resource allocation.

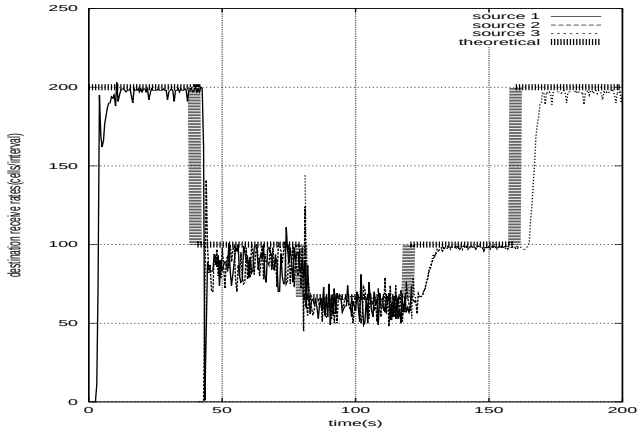
VII. Conclusion

We have described an optimization approach to reactive flow control, and derived a simple asynchronous distributed algorithm. We allow the sources and network links to communicate and update their controls asynchronously at different times, with different frequencies, and after substantial and random delays. The algorithm is provably convergent to the global optimal when network conditions are static and seems to track the optimum when network conditions vary slowly. The scheme has desirable fairness properties and is extensible to a multicasting environment.

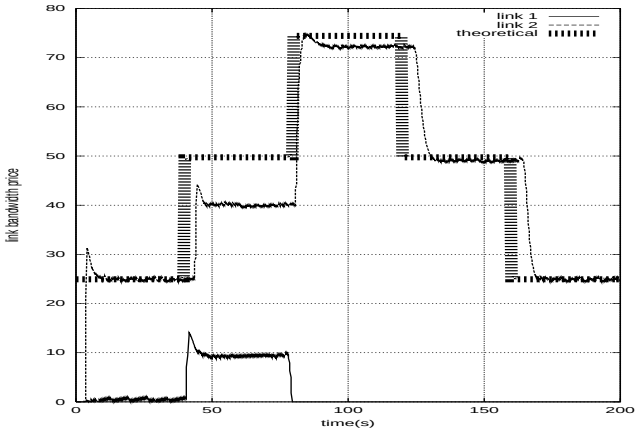
The algorithm presented in this paper requires communication between sources and links. As noted in Section I a practical implementation using only binary feedback from links to sources is the REM scheme described in [21] and Part II of this paper. The abstract model here serves as a convenient framework to systematically refine REM, as illustrated in [1], [2].

Acknowledgments: We are grateful to F. Kelly, D. Mitra, J. Tsitsiklis, and A. Weiss for very helpful discussions. The first author would also like to thank B. Doshi and Y. T. Wang of Bell Laboratories, Lucent Technologies, for their hospitality during a visit in 1997 where part of this work was done.

We thank the following people for pointing out some

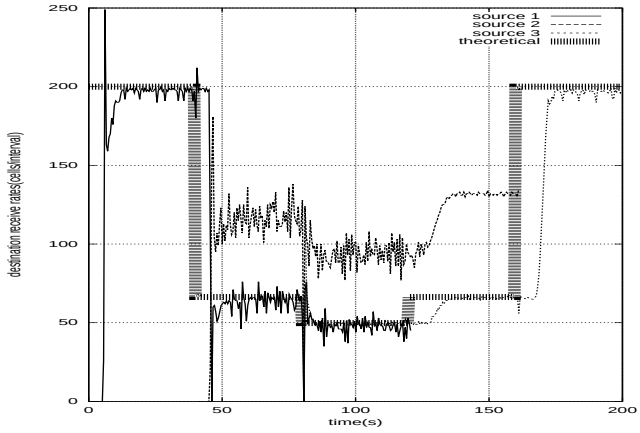


(a) Destination Receive Rates

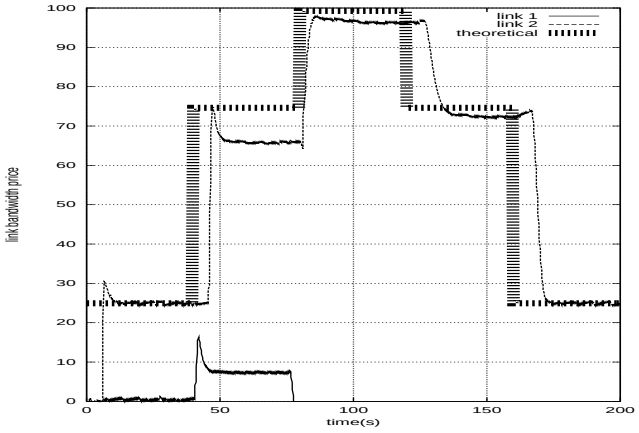


(b) Link Bandwidth Prices

Fig. 3. Experiment 1 - homogeneous sources. Heavy lines are theoretical rates and prices and light lines are measured ones. The sum of user rates is approximately 200 cells per 500 ms measurement interval. In (a) the theoretical rate is the optimal rate x_s for each source (all have identical utility). In (b) the theoretical price is the path price of the longest connection that was on and should roughly equal the sum of the measured link prices.



(a) Destination Receive Rates



(b) Link Bandwidth Prices

Fig. 4. Experiment 2 - heterogeneous sources. Heavy lines are theoretical rates and prices and light lines are measured ones. In (a) the theoretical rate is for source 1 from time = 0 to 120, and for source 2 for $t = 120$ to 160 and source 3 for $t \geq 160$. In (b) the theoretical price is the path price of the longest connection that was on and should roughly equal the sum of the measured link prices.

errors after publication: Wing Chow of Melbourne University, Edward Fan of UCLA, and Andrzej Karbowski of NASK, Poland.

Appendix I: Proof of Theorem 1

We will often use vector notation when it is more convenient. We start with the basic properties of the dual objective function that follow directly from C1.

Lemma 1: Under assumption C1, the dual objective function $D(p)$ is convex, lower bounded, and continuously differentiable.

For any price vector p in $\Re_+^L$ define $\beta_s(p)$ by

$$\beta_s(p) \quad = \quad \begin{cases} \frac{1}{-U_s''(x_s(p))} & \text{if } U_s'(M_s) \leq p^s \leq U_s'(m_s) \\ 0 & \text{otherwise} \end{cases} \tag{24}$$

where p^s is defined in (4) and $x(p)$ is the unique maximizer of (3). Let $B(p) = Diag\ (\beta_s(p), s \in S)$ be the $S \times S$ diagonal matrix with diagonal elements $\beta_s(p)$. Note that from assumption C2 in Section II-C for all $p \geq 0$

$$0 \ \leq \ \beta_s(p) \quad \leq \quad \overline{\alpha}_s \ < \ \infty \tag{25}$$

Recall the routing matrix R defined in Section II-C.

Lemma 2: Under condition C1, the Hessian of D is given by $\nabla^2 D(p) = RB(p)R^T$, where it exists.

Proof. Let $\frac{\partial x}{\partial p}(p)$ denote the $S \times L$ Jacobian matrix whose (s,l) element is $\frac{\partial x_s}{\partial p_l}(p)$. When it exists,

$$\frac{\partial x_s}{\partial p_l}(p) \quad = \quad \begin{cases} \frac{R_{ls}}{U_s''(x_s(p))} & \text{if } U_s'(M_s) \le p^s \le U_s'(m_s) \\ 0 & \text{otherwise} \end{cases}$$

Using (24) we have

$$\left[\frac{\partial x}{\partial p}(p)\right] \quad = \quad -B(p)R^T \tag{26}$$

Now from (8) we have $\nabla D(p) = c - Rx(p)$ and hence

$$\nabla^2 D(p) \quad = \quad -R\left[\frac{\partial x}{\partial p}(p)\right]$$

which together with (26) yields the result. ■

Recall $\overline{L}$, $\overline{S}$, and $\overline{\alpha}$ defined in Section III before Theorem 1.

Lemma 3: Under conditions C1–C2, ∇D is Lipschitz with

$$||\nabla D(q) - \nabla D(p)||_2 \quad \le \quad \overline{\alpha}\overline{LS}\ ||q-p||_2$$

for all $p, q \ge 0$.

Proof. Using Lemma 2, we will show that $||\nabla^2 D(p)||_2 = ||RB(w)R^T||_2 \le \overline{\alpha}\overline{LS}$. [3] The lemma then follows from [30, Theorem 9.19].[4]

Now (see e.g. [5, pp. 635])

$$||RB(w)R^T||_2^2 \le ||RB(w)R^T||_\infty \cdot ||RB(w)R^T||_1$$

i.e., $||RB(w)R^T||_2$ is upper bounded by the product of the maximum row sum and the maximum column sum of the $L \times L$ matrix $RB(w)R^T$. Since $RB(w)R^T$ is symmetric, $||RB(w)R^T||_1 = ||RB(w)R^T||_\infty$, and hence

$$\begin{aligned} ||RB(w)R^T||_2 \quad &\le \quad ||RB(w)R^T||_\infty \\ &= \quad \max_l \sum_{l'} [RB(w)R^T]_{ll'} \\ &= \quad \max_l \sum_{l'} \sum_s \beta_s(w) R_{ls} R_{l's} \\ &= \quad \max_l \sum_s \beta_s(w) R_{ls} |L(s)| \end{aligned}$$

where $|L(s)|$ is the number of links in the path of source s. By definition $|L(s)| \le \overline{L}$, $\beta_s(w) \le \overline{\alpha}$, and hence

$$||RB(w)R^T||_2 \quad \le \quad \overline{\alpha}\overline{L} \max_l\ |S(l)| \ \le \ \overline{\alpha}\overline{LS}$$

as desired. ■

These lemmas establish our first main result.

Proof of Theorem 1. The dual objective function D is lower bounded and ∇D is Lipschitz from Lemmas 1 and 3. Then any accumulation point p^* of the sequence $\{p(t)\}$ generated by the gradient projection algorithm for the dual problem is dual optimal; see [5, pp.214].

Let $\{p(t), t \in T\}$ be a subsequence converging to p^*. At least one exists since it is easy to show that the level set $\{p \ge 0 \mid D(p) \le D(p(0))\}$ of D is compact and that the sequence $\{D(p(t))\}$ is decreasing in t and hence in the level set, provided $0 < \gamma < 2/\overline{\alpha}\overline{LS}$ (Lemma 3 and [5, pp.214]). To show that the subsequence $\{x(t) = x(p(t)), t \in T\}$ converges to the primal optimal source rate $x^* = x(p^*)$, note that $U_s'(x_s)$ is defined on a compact set $[m_s, M_s]$. Moreover it is continuous and one-to-one (because of the *strict* concavity of U_s) and hence its inverse is continuous on $[U_s(M_s), U_s(m_s)]$ [30, Theorem 4.17]. From (6), $x(p)$ is continuous. Therefore $\lim_{t\to\infty, t\in T} x(t) = x(p^*)$. ■

Appendix II: Proof of Theorem 2

Define $\pi(t)$ as $\pi(t) = p(t+1) - p(t)$. Let $\overline{p}(t) = (p_l(t'), l \in L, t' = t - t_0, \ldots, t)$ be the vector of prices at times $t - t_0, \ldots, t$. For any vector $z \in \Re_+^{L(t_0+1)}$, let z_{lt} denote its (l,t)th component. Given any $\overline{p} \in \Re_+^{L(t_0+1)}$, define $u_s(\cdot;\overline{p}) : \Re_+^{L(t_0+1)} \to \Re_+$ by

$$u_s(\epsilon;\overline{p}) \quad = \quad U_s'^{-1}(\epsilon^T \overline{p}) \ = \ U_s'^{-1}(\sum_{(l,t)} \epsilon_{lt}\overline{p}_{lt}) \tag{27}$$

We assume that conditions C1–C3 hold.

We start with a collection of useful facts. Recall the bound α_s on $U_s''(x_s)$ defined in assumption C2 of Section II-C, and the gradient estimate $\lambda_l(t)$ defined in (15–18).

Lemma 4: (a) For all t, $\lambda^T(t)\pi(t) \le -\frac{1}{\gamma}||\pi(t)||_2^2$.
(b) There exists a constant $A_1 > 0$ such that, for all $p \ge 0$ and all q, we have $q^T \nabla^2 D(p) q \le 2A_1 ||q||_2^2$.
(c) For any $\overline{p} \in \Re_+^{L(t_0+1)}$, $0 \le \partial u_s / \partial \epsilon_{lt}(\epsilon;\overline{p}) \le \bar{\alpha}_s \overline{p}_{lt}$, where it exists.
(d) For all t,
$|U_s'^{-1}(\hat{p}^s(t)) - U_s'^{-1}(p^s(t))| \le \bar{\alpha}_s \sum_{t'=t-t_0}^{t-1} \sum_l |\pi_l(t')| R_{ls}$.
(e) For all t,
$|U_s'^{-1}(p^s(t)) - U_s'^{-1}(p^s(\tau))| \le \bar{\alpha}_s \sum_{t'=\tau}^{t-1} \sum_l |\pi_l(t')| R_{ls}$.

Proof.
(a) For $t \in T_l^1$, applying the projection theorem ([4, Proposition 2.1.3]) to the scalar $p_l(t+1) = [p_l(t) - \gamma\lambda_l(t)]^+$ we have

$$(p_l(t) - \gamma\lambda_l(t) - p_l(t+1))(p_l(t) - p_l(t+1)) \quad \le \quad 0$$

[3] Where $\nabla^2 D(w)$ may not exist, at points where $w^s = U_s'(m_s)$ or $w^s = U_s'(M_s)$ for some s, derivatives should be replaced by convex subgradients in the proof. Then Lemma 3 and Theorem 1 hold. For simplicity we will ignore these issues in this paper.

[4] This is pointed out by Edward Fan of UCLA in March 2002 and corrects an error in a previous version.

and hence $\lambda_l(t)\pi_l(t) \le \frac{1}{\gamma}\pi_l^2(t)$. This inequality holds trivially for $t \notin T_l^1$, and hence for all l

$$\lambda_l(t)\pi_l(t) \le -\frac{1}{\gamma}\pi_l^2(t), \qquad \text{for all } t$$

Summing over l yields the desired result.

(b) By Lemma 2, $\nabla^2 D(p)$ is symmetric and positive semidefinite, and hence [4, Appendix A] $q^T\nabla^2 D(p)q \le \rho(\nabla^2 D(p))||q||_2^2$ where $\rho(\nabla^2 D(p))$ is the largest eigenvalue of the matrix $\nabla^2 D(p)$. We claim that $\rho(\nabla^2 D(p))$ is bounded for all p, because from Lemma 2,

$$\begin{aligned}\rho(\nabla^2 D(p)) &\le \text{trace}(RB(p)R^T)\\ &= \sum_s \beta_s(p)|L(s)| \le \bar{\alpha}\bar{L}\bar{S}\end{aligned}$$

Here the first inequality follows from the fact that the trace of a matrix is the sum of all its eigenvalues and that eigenvalues of a positive semidefinite matrix are all nonnegative. The second inequality follows from (25) and the definition of $\bar{\alpha}$, $\bar{L}$ and $\bar{S}$.

(c) The claim follows from chain rule and (25).

(d) Now $U_s'^{-1}(\hat{p}^s(t)) = u_s(\epsilon(t);\overline{p}(t))$ and $U_s'^{-1}(p^s(t)) = u_s(1(t);\overline{p}(t))$, where u_s is defined in (27), $1(t) \in \Re_+^{L(t_0+1)}$ is defined by

$$1_{lt'}(t) = \begin{cases} 1 & \text{if } l \in L(s), t' = t \\ 0 & \text{otherwise}\end{cases}$$

and $\epsilon(t) \in \Re_+^{L(t_0+1)}$ is defined by

$$\epsilon_{lt'}(t) = \begin{cases} b_{ls}(t',t) & \text{if } l \in L(s) \\ 0 & \text{otherwise}\end{cases}$$

Hence by the mean value theorem and applying part (c) of the lemma, we have, for some $\tilde{\epsilon}$,

$$\begin{aligned}&|U_s'^{-1}(\hat{p}^s(t)) - U_s'^{-1}(p^s(t))|\\ =\ &|\sum_{(l,t')} \frac{\partial u_s}{\partial \epsilon_{lt'}}(\tilde{\epsilon};\overline{p}(t))(1_{lt'}(t) - \epsilon_{lt'}(t))|\\ \le\ &\bar{\alpha}_s|\sum_{(l,t')} \overline{p}_{lt'}(t)(1_{lt'}(t) - \epsilon_{lt'}(t))|\\ \le\ &\bar{\alpha}_s \sum_{l\in L(s)} |p_l(t) - \sum_{t'=t-t_0}^{t} b_{ls}(t',t)p_l(t')|\\ \le\ &\bar{\alpha}_s \sum_{l\in L(s)} \max_{t-t_0\le t'\le t} |p_l(t) - p_l(t')|\\ \le\ &\bar{\alpha}_s \sum_{l\in L(s)} \max_{t-t_0\le t'\le t} \sum_{\tau=t'}^{t-1} |\pi_l(\tau)|\\ \le\ &\bar{\alpha}_s \sum_{t'=t-t_0}^{t-1} \sum_{l} |\pi_l(t')|R_{ls}\end{aligned}$$

(e) We have by the mean value theorem and (25)

$$\begin{aligned}&|U_s'^{-1}(p^s(t)) - U_s'^{-1}(p^s(\tau))|\\ \le\ &\bar{\alpha}_s \sum_{l\in L(s)} |p_l(t) - p_l(\tau)|\\ \le\ &\bar{\alpha}_s \sum_{l\in L(s)} \sum_{t'=\tau}^{t-1} |\pi_l(t')|\end{aligned}$$

■

The next lemma bounds the error in gradient estimation in terms of the successive price change $\pi(t) = p(t+1) - p(t)$.

Lemma 5: There exists a constant $A_2 > 0$ such that

$$||\nabla D(p(t)) - \lambda(t)|| \le A_2 \sum_{t'=t-2t_0}^{t-1} ||\pi(t')|| \qquad (28)$$

Proof. From (8), (15–17) and (23) we have

$$\begin{aligned}&[\nabla D(p(t)) - \lambda(t)]_l\\ =\ &\hat{x}^l(t) - \bar{x}^l(t)\\ =\ &\sum_{s\in S(l)} (\sum_{t'=t-t_0}^{t} a_{ls}(t',t)x_s(t') - \bar{x}_s(t))\end{aligned}$$

Hence for some constant $A_2' > 0$ we have

$$\begin{aligned}&||\nabla D(p(t)) - \lambda(t)||\\ \le\ &A_2' \max_l \sum_{s\in S(l)} |\sum_{t'=t-t_0}^{t} a_{ls}(t',t)x_s(t') - \bar{x}_s(t)|\\ \le\ &A_2' \max_l \sum_{s\in S(l)} \max_{t-t_0\le t'\le t} |x_s(t') - \bar{x}_s(t)|\\ \le\ &A_2' \max_l \sum_{s\in S(l)} \max_{t-t_0\le t'\le t} |U_s'^{-1}(\hat{p}^s(t')) - U_s'^{-1}(p^s(t))|\end{aligned}$$

where the last inequality follows from (19) and (23) and from the fact that projection is nonexpansive [4, Proposition 2.1.3]. Applying Lemma 4 (d) and (e), we have

$$\begin{aligned}&||\nabla D(p(t)) - \lambda(t)||\\ \le\ &A_2' \max_l \sum_{s\in S(l)} \max_{t-t_0\le t'\le t} |U_s'^{-1}(p^s(t)) - U_s'^{-1}(p^s(t'))|\\ &\quad + |U_s'^{-1}(p^s(t')) - U_s'^{-1}(\hat{p}^s(t'))|\\ \le\ &A_2' \max_l \sum_{s\in S(l)} \max_{t-t_0\le t'\le t} \bar{\alpha}_s \left\{\sum_{\tau=t'}^{t-1}\sum_{l'} |\pi_{l'}(\tau)|R_{l's}\right.\\ &\quad \left. + \sum_{\tau=t'-t_0}^{t'-1}\sum_{l'} |\pi_{l'}(\tau)|R_{l's}\right\}\\ =\ &A_2' \max_l \sum_{s\in S(l)} \bar{\alpha}_s \max_{t-t_0\le t'\le t} \sum_{\tau=t'-t_0}^{t-1}\sum_{l'} |\pi_{l'}(\tau)|R_{l's}\\ \le\ &A_2' \max_l \sum_{s\in S(l)} \bar{\alpha}_s \sum_{\tau=t-2t_0}^{t-1} ||\pi(\tau)||_1\end{aligned}$$

$$\leq \quad A_2'\bar{\alpha}\bar{S}\sum_{\tau=t-2t_0}^{t-1}||\pi(\tau)||_1$$

where the third inequality follows from $\sum_{l'}|\pi_{l'}(\tau)|R_{l's} \leq ||\pi(\tau)||_1$. The proof is complete since norms in finite dimensional vector space are all equivalent. ■

The next lemma shows that $\pi(t)$ converges to zero.

Lemma 6: Provided γ is sufficiently small we have for all t, $D(p(t+1)) \leq D(p(0)) - (\frac{1}{\gamma} - A_1 - (2t_0+1)A_2)\sum_{\tau=0}^{t}||\pi(\tau)||^2$, where A_1 and A_2 are the constants in Lemmas 4 and 5 respectively. Hence $||\pi(t)|| \to 0$ as $t \to \infty$.

Proof. Applying Lemma 4 (a) and (b) to the second order Taylor expansion of $D(p(t+1))$, we have for some $q(t) \geq 0$

$$\begin{aligned}
& D(p(t+1)) \\
= \quad & D(p(t)) + (\nabla D(p(t)))^T\pi(t) + \frac{1}{2}\pi^T(t)\nabla^2 D(q(t))\pi(t) \\
\leq \quad & D(p(t)) + (\nabla D(p(t)) - \lambda)^T\pi(t) \\
& \quad + \lambda^T(t)\pi(t) + A_1||\pi(t)||^2 \\
\leq \quad & D(p(t)) + ||\nabla D(p(t)) - \lambda|| \cdot ||\pi(t)|| \\
& \quad - (\frac{1}{\gamma} - A_1)||\pi(t)||^2
\end{aligned}$$

Applying Lemma 5 we have

$$\begin{aligned}
D(p(t+1)) \quad \leq \quad & D(p(t)) - (\frac{1}{\gamma} - A_1)||\pi(t)||^2 \\
& + A_2\sum_{t'=t-2t_0}^{t-1}||\pi(t')|| \cdot ||\pi(t)|| \\
\leq \quad & D(p(t)) - (\frac{1}{\gamma} - A_1)||\pi(t)||^2 \\
& + A_2\sum_{t'=t-2t_0}^{t}||\pi(t')||^2 \qquad (29)
\end{aligned}$$

where the last inequality holds because the convex function $\sum_i y_i^2 + z^2 - \sum_i y_i z$ attains a unique minimum over $\{(y_i, z)|y_i \geq 0, z \geq 0\}$ at the origin.[5] Summing (29) over all t we have

$$\begin{aligned}
D(p(t+1)) \quad \leq \quad & D(p(0)) - (\frac{1}{\gamma} - A_1)\sum_{\tau=0}^{t}||\pi(\tau)||^2 \\
& + A_2\sum_{\tau=0}^{t}\sum_{t'=\tau-2t_0}^{\tau}||\pi(t')||^2 \\
\leq \quad & D(p(0)) - \left(\frac{1}{\gamma} - A_1 - (2t_0+1)A_2\right) \\
& \sum_{\tau=0}^{t}||\pi(\tau)||^2 \qquad (30)
\end{aligned}$$

as desired.

Since the above inequality holds for all $\gamma > 0$, we can choose γ sufficiently small such that

$$\frac{1}{\gamma} - A_1 - (2t_0+1)A_2 > 0$$

Then, since $D(p(t))$ is lower bounded (Lemma 1), we must have, letting $t \to \infty$, $\sum_{t=0}^{\infty}||\pi(t)||^2 < \infty$, and hence

$$||\pi(t)|| \quad \longrightarrow \quad 0 \qquad \text{as } t \to \infty. \qquad (31)$$

■

These lemmas establish Theorem 2.

Proof of Theorem 2. We first prove that the various errors due to asynchronism all converge to zero. For all sources s we have from (20–21)

$$\begin{aligned}
|\hat{p}^s(t) - p^s(t)| \quad \leq \quad & \sum_{l\in L(s)}\max_{t-t_0\leq t'\leq t}|p_l(t') - p_l(t)| \\
\leq \quad & \sum_{t'=t-t_0}^{t}\sum_{l\in L(s)}|\pi_l(t')| \\
\leq \quad & \sum_{t'=t-t_0}^{t}||\pi(t')||_1
\end{aligned}$$

$$\frac{A_2}{2}\sum_{t'=t-2t_0}^{t-1}||\pi(t')||^2$$

Proceeding as in the original proof, we have, on summing over all t,

$$\begin{aligned}
D(p(t+1)) \quad \leq \quad & D(p(0)) - \left(\frac{1}{\gamma} - A_1 - t_0A_2\right)\sum_{\tau=0}^{t}||\pi(\tau)||^2 \\
& + \frac{A_2}{2}\sum_{\tau=1}^{t}\sum_{t'=\tau-2t_0}^{\tau-1}||\pi(t')||^2 \\
\leq \quad & D(p(0)) - \left(\frac{1}{\gamma} - A_1 - 2t_0A_2\right)\sum_{\tau=0}^{t}||\pi(\tau)||^2
\end{aligned}$$

which should replace (30) in the original proof. The rest of the proof goes through with an obvious change to the bound on γ.

[5] This statement "$\sum_i y_i^2 + z^2 - \sum_i y_i z$ attains a unique minimum over $\{(y_i, z)|y_i \geq 0, z \geq 0\}$ at the origin" is incorrect. The error and its correction is communicated to us by Andrzej Karbowski in his paper *Correction to Low and Lapsley's article "Optimization Flow Control, I: Basic Algorithm and Convergence"* available at `http://www.ia.pw.edu.pl/~karbowsk/pub/papers`. The correction does not affect the statement of Theorem 2 but allows the bound on the stepsize γ to be slightly larger than that derived in the original proof.

The correction uses the fact that

$$\sum_{i=1}^{n}y_i z \quad \leq \quad \frac{1}{2}nz^2 + \frac{1}{2}\sum_{i=1}^{n}y_i^2$$

The last inequality in (29) should then be replaced by

$$D(p(t+1)) \quad \leq \quad D(p(t)) - \left(\frac{1}{\gamma} - A_1 - t_0A_2\right)||\pi(t)||^2 \ +$$

which by (31) converges to zero as $t \to \infty$. From (19), (23) and (6), $x_s(t)$ and $\bar{x}_s(t)$ are projections of $U_s'^{-1}$ onto $[m_s, M_s]$. Since projection is nonexpansive [4, Proposition 2.1.3], we have

$$\begin{aligned} |x_s(t) - \bar{x}_s(t)| &\leq |U_s'^{-1}(\hat{p}^s(t)) - U_s'^{-1}(p^s(t))| \\ &\leq \bar{\alpha}_s \sum_{t'=t-t_0}^{t-1} ||\pi(t')||_1 \end{aligned}$$

where the last inequality follows from Lemma 4 (d). Hence by (31), $|x_s(t) - \bar{x}_s(t)| \to 0$ for all s. The error $||\lambda(t) - \nabla D(p(t))||$ in gradient estimation converges to zero by Lemma 5 and (31).

We now show that every accumulation point of the sequence $\{p(t)\}$ generated by Algorithm A2 minimizes the dual problem. Let p^* be an accumulation point of $\{p(t)\}$. At least one exists since the level set $\{p \geq 0 \mid D(p) \leq D(p(0))\}$ of D is compact and that the sequence $\{D(p(t))\}$ is in the level set provided γ is sufficiently small (see (30)). Moreover, since the interval between consecutive updates is bounded (condition C3), p^* is also an accumulation point of $\{p(t), t \in \cap_l T_l^1\}$. Let $\{t_k\} \subseteq \cap_l T_l^1$ be a sequence such that $\{p(t_k)\}$ converges to p^*. Since ∇D is continuous (Lemma 1) and $||\lambda(t) - \nabla D(p(t))|| \to 0$ (Lemma 5), we have

$$\lim_k \lambda(t_k) = \lim_k \nabla D(p(t_k)) = \nabla D(p^*).$$

Hence

$$\begin{aligned} [p^* - \gamma \nabla D(p^*)]^+ - p^* &= \lim_k [p(t_k) - \gamma \lambda(t_k)]^+ - p(t_k) \\ &= \lim_k \pi(t_k) = 0 \end{aligned}$$

where the last equality follows from (31). Then the projection theorem [4, Proposition 2.1.3]) implies

$$\gamma [\nabla D(p^*)]^T (p - p^*) \geq 0, \qquad \text{for all } p \geq 0$$

which, due to the concavity of D, implies that p^* minimizes D over $p \geq 0$.

By duality $x^* = x(p^*)$ is the unique primal optimal rate. We now show that it is a limit point of $\{x(t)\}$ generated by Algorithm A2. Consider the subsequence $\{x(t_k)\}$. Since it is in the compact set $\Pi_s[m_s, M_s]$, there exists a sequence $\{t_n\} \subseteq \{t_k\} \cap \cap_s T_s^2$ such that $x(t_n)$ converges. Since $||x(t) - \bar{x}(t)|| \to 0$, we have

$$\lim_n x(t_n) = \lim_n \bar{x}(t_n) = \lim_n x(p(t_n)) = x(p^*)$$

where the second equality follows from (23). This completes the proof. ∎

References

[1] Sanjeewa Athuraliya, David Lapsley, and Steven Low. An Enhanced Random Early Marking Algorithm for Internet Flow Control. In *Proceedings of IEEE Infocom*, March 2000.

[2] Sanjeewa Athuraliya and Steven Low. Optimization flow control with Newton–like algorithm. In *Proceedings of IEEE Globecom'99*, December 1999.

[3] L. Benmohamed and S. M. Meerkov. Feedback control of congestion in store-and-forward networks: the case of a single congested node. *IEEE/ACM Transactions on Networking*, 1(6):693–707, December 1993.

[4] D. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1995.

[5] Dimitri P. Bertsekas and John N. Tsitsiklis. *Parallel and distributed computation*. Prentice-Hall, 1989.

[6] Flavio Bonomi, Debasis Mitra, and Judith B. Seery. Adaptive algorithms for feedback-based flow control in high-speed wide-area ATM networks. *IEEE Journal on Selected Areas in Communications*, 13(7):1267–1283, September 1995.

[7] S. Chong, R. Nagarajan, and Y.-T. Wang. Designing stable ABR flow control with rate feedback and open loop control: first order control case. *Performance Evaluation*, 34(4):189–206, December 1998.

[8] Costas Courcoubetis, Vasilios A. Siris, and George D. Stamoulis. Integration of pricing and flow control for ABR services in ATM networks. *Proceedings of Globecom'96*, November 1996.

[9] S. Floyd. TCP and Explicit Congestion Notification. *ACM Computer Communication Review*, 24(5), October 1994.

[10] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Trans. on Networking*, 1(4):397–413, August 1993.

[11] R. G. Gallager and S. J. Golestani. Flow control and routing algorithms for data networks. In *Proceedings of the 5th International Conf. Comp. Comm.*, pages 779–784, 1980.

[12] R. J. Gibbens and F. P. Kelly. Resource pricing and the evolution of congestion control. *Automatica*, 35, 1999.

[13] Jamal Golestani and Supratik Bhattacharyya. End-to-end congestion control for the Internet: A global optimization framework. In *Proceedings of International Conf. on Network Protocols (ICNP)*, October 1998.

[14] E. J. Hernandez-Valencia, L. Benmohamed, R. Nagarajan, and S. Chong. Rate control algorithms for the ATM ABR service. *European Transactions on Telecommunications*, 8:7–20, 1997.

[15] V. Jacobson. Congestion avoidance and control. *Proceedings of SIGCOMM'88, ACM*, August 1988. An updated version is available via ftp://ftp.ee.lbl.gov/papers/congavoid.ps.Z.

[16] F. P. Kelly. Charging and rate control for elastic traffic. *European Transactions on Telecommunications*, 8:33–37, 1997. http://www.statslab.cam.ac.uk/~frank/elastic.html.

[17] Frank P. Kelly, Aman Maulloo, and David Tan. Rate control for communication networks: Shadow prices, proportional fairness and stability. *Journal of Operations Research Society*, 49(3):237–252, March 1998.

[18] Srinivasan Keshav. *An Engineering Approach to Computer Networking*. Addison-Wesley, 1997.

[19] David E. Lapsley and Steven H. Low. An IP Implementation of Optimization Flow Control. In *Proceedings of the Globecom'98*, November 1998.

[20] David E. Lapsley and Steven H. Low. An optimization approach to ABR control. In *Proceedings of the ICC*, June 1998.

[21] David E. Lapsley and Steven H. Low. Random Early Marking for Internet Congestion Control. In *Proceedings of IEEE Globecom'99*, December 1999.

[22] David E. Lapsley and Michael Rumsewicz. Improved buffer efficiency via the No Increase flag in EFCI flow control. In *Proceedings of the IEEE ATM '96 Workshop*, August 1996.

[23] Steven H. Low. Equilibrium allocation and pricing of variable resources among user-suppliers. *Performance Evaluation*, 34(4), December 1998.

[24] Steven H. Low. Equilibrium allocation of variable resources for elastic traffics. In *Proceedings of INFOCOM'98*, San Francisco, CA, USA, March 1998.

[25] Steven H. Low. Optimization flow control with on-line measurement. In *Proceedings of the ITC*, volume 16, June 1999.

[26] David G. Luenberger. *Linear and Nonlinear Programming, 2nd Ed.* Addison-Wesley Publishing Company, 1984.

[27] K. K. Ramakrishnan and S. Floyd. A Proposal to add Explicit

Congestion Notification (ECN) to IP. Internet draft draft-kksjf-ecn-01.txt, July 1998.

[28] K. K. Ramakrishnan and R. Jain. A binary feedback scheme for congestion avoidance in computer networks with a connectionless network layer. *Proceedings of SIGCOMM'88, ACM*, August 1988.

[29] G. Ramamurthy and A. Kolarov. Application of control theory for the design of closed loop rate control for ABR service. In *Proceedings of ITC 15*, pages 751–760, 1997.

[30] W. Rudin. *Principles of Mathematical Analysis*. McGraw-Hill Inc., third edition, 1976.

[31] Scott Shenker. Fundamental design issues for the future internet. *IEEE Journal on Selected Areas in Communications*, 13(7):1176–1188, 1995.

[32] John N. Tsitsiklis and Dimitri P. Bertsekas. Distributed asynchronous optimal routing in data networks. *IEEE Transactions on Automatic Control*, 31(4):325–332, April 1986.

Steven. H. Low (SM 99) received his B.S. degree from Cornell University and PhD from the University of California – Berkeley in 1992, both in electrical engineering. He has been a consultant to NEC in the USA in 1991 and was with AT&T Bell Laboratories, Murray Hill, from 1992 to 1996. He joined the University of Melbourne, Australia, in 1996 as a Senior Lecturer. He has held visiting academic positions in the US and Hong Kong, and has consulted with companies in Australia and the US. He was a co-recipient of the IEEE William R. Bennett Prize Paper Award in 1997 and the 1996 R&D 100 Award. He is on the editorial board of *IEEE/ACM Transactions on Networking*. He has been a guest editor of the *IEEE Journal on Selected Area in Communications*, on the program committee of several conferences. His research interests are in the control and optimization of communications networks and protocols, and network security and privacy. His email is: slow@ee.mu.oz.au.

David E. Lapsley obtained the B. Sc. degree in Computer Science and the B. E.(hons) degree in Electrical and Computer Systems Engineering from Monash University, Melbourne, Australia. He is completing his Ph. D. at the University of Melbourne, Australia. He is now with Ericsson's Advanced Services Application Centre as an IT Specialist. His research interests include congestion control in packet switched networks and active networking. His email is: lapsley@ee.mu.oz.au.