Window-Based Congestion Control with Heterogeneous Users

Richard J. La and Venkat Anantharam
Department of Electrical Engineering and Computer Sciences
University of California at Berkeley
{hyongla, ananth}@eecs.berkeley.edu

Abstract—
We investigate the fundamental problem of achieving the system optimal rates, which maximize the total user utility, in a distributed network environment using only the information available at the end hosts. This is done by decomposing the overall system problem into subproblems for the network and for the individual users and introducing an incentive-compatible pricing scheme. The users are to solve the problem of maximizing individual net utility, which is their utility less the amount they pay. This is done using a window based algorithm. We provide an algorithm for the network to adjust its prices and the users to adjust their window sizes such that at an equilibrium the system optimum is achieved. It is notable that our algorithm does not require any explicit feedback from the network and can be deployed over the Internet with modifications only at the end hosts. Our scheme is incentive compatible in that there is no benefit to the users to lie about their utilities.

I. Introduction

With continuing growth of the Internet both in size and in the number of users one of the challenging problems network designers face is how to provide a fair and efficient allocation of the available bandwidth. In the current Internet most connections use variants of the Transmission Control Protocol (TCP), which is a window-based congestion control mechanism. However, as is widely recognized, TCP does not generally lead to a fair or efficient allocation of bandwidth among the connections [3], [19], [5]. Another problem with the current situation is that users cannot dynamically indicate their preferences even if they are willing to pay higher prices for the service they desire at some time. This argues for some form of dynamic pricing in the network. Another argument for pricing is that the Internet is increasingly being used in a potentially noncooperative way, and so might need pricing mechanisms to ensure that users do not misbehave. There has already been a good deal of work addressing these issues of providing quality of service in accordance with users' willingness to pay and ensuring system friendly behaviour on the part of the users. Notable seminal papers in this area include schemes based on time and volume measurements [11], and those based on per-packet charges [22]. There is also interesting empirical research showing that flat rate charging leads to inefficiency, whereby a large number of low-usage users end up subsidizing a small number of high-usage users [4]. This study can be viewed as an argument for the desirability of usage-based pricing. A good discussion on the market structure that would result from usage-based pricing is given in [17].

The work in this paper is based on a mathematical formulation of the system problem as a problem of maximizing the aggregate utility of the users. It is motivated by work of Kelly, who made the fundamental observation that this problem can be decomposed into separate subproblems for the network and for the individual users [9], if one adopts a pricing mechanism. In the papers that build on this work, two classes of rate allocation mechanisms are proposed to achieve the system optimum [10], [7]. One should also note that Low *et al.* have approached the same problem, using the gradient based algorithm [15], [1], and have proposed a form of marking mechanism, called Random Early Marking (REM), as a way of arriving at the system wide optimal solution.

Another basic source for the work in this paper is the work Mo and Walrand [20], who investigated the problem of designing a fair window-based end-to-end rate allocation mechanism in a network. One of the things they show is that there is a well defined mapping between window sizes of the users and their equilibrium rates. Also, working with a notion of fairness called (p, α)-proportional fairness, which generalizes the well known notions of max-min fairness and proportional fairness, they design a family of window based algorithms for the individual users which converge to window sizes resulting in rates that achieve (p, α)-proportional fairness.

Our goal is to find window based algorithms that will solve the system problem of maximizing the aggregate utility of the users. This is as in our earlier work [13] where we had the users adopt Mo and Walrand's $(p, 1)$-proportionally fair algorithm while updating their target queue sizes based on their utilities. In the earlier work, we showed that such an algorithm does achieve the system optimum rates, since it has a unique equilibrium, which is the unique fixed point of the mapping from the rates corresponding to target queue sizes p to the equilibrium rates of $(p, 1)$-proportionally fair algorithm given p. In the earlier work we established the convergence result only in the case of a network having a single bottleneck. In this work we present an algorithm that is shown to converge to the system optimal rates in any general network. This algorithm can be thought of a natural extension of $(p, 1)$-proportionally fair algorithm. Moreover, combined with a pricing scheme we propose it is shown that our scheme is incentive compatible in that users do not have an incentive to lie about their utilities.

The rest of the paper is organized as follows. In section II we motivate our problem and describe the model. Section III describes Mo and Walrand's work. We present our pricing scheme and algorithm in section IV, which is followed by simulation results in section V. We discuss how the backlog in the network

can be controlled using our pricing scheme in section VI. We finish by drawing some conclusions in section VII.

II. Motivation, Background & Model

In this section we first motivate the problem we consider by surveying some of the currently most popular congestion control mechanism used in the Internet and discussing some of their problems. We then give a precise mathematical formulation of the model in the context of which we will carry out the analysis of our problem, and in the context of which we will present our congestion control proposal.

A. Motivation

The Internet was designed to be decentralized and to rely on disciplined behavior from the users. Since the users have to share the available bandwidth in the network, and given the infeasiblity of a a centralized authority mandating the rate allocation, it is necessary to have rate allocation algorithms implemented at the end hosts. This is a classical problem and there are a large number of such algorithms, of which some recent ones of relevance to this work are are [10], [20], [15], [8]. In general all such proposals be roughly categorized into two classes: rate-based algorithms and window-based algorithms. A rate-based algorithm directly controls the transmission rate of the connection, based on either feedback from the network [10] or on measurements taken at the end host. A window-based algorithm adjusts the congestion window size, which is the maximum number of outstanding packets that the user has not yet received acknowledgments for, in order to in effect control the transmission rate and also control the backlog inside the network associated to the user (connection). Since window based algorithms have a better handle on the backlog in the network that rate based algorithms, we view them as desirable (this may be controversial, but is the thesis of this paper).

The most widely deployed flow/congestion control mechanism in the current Internet is the Transmission Control Protocol (TCP), which is a window-based congestion control mechanism. There are, of course, several variants of TCP. It is widely recognized that TCP does not necessarily lead to a fair or efficient rate allocation of the available bandwidth. TCP[1] as currently implemented suffers from a high packet loss rate and a delay bias [14], [19], [2]. The high packet loss rate is a consequence of periodic oscillation of the window sizes and aggressive slow start, while the delay bias is the result of a discrepancy in window update rates among different connections. In order to address these issues Brakmo *et al.* [2] have introduced another version of TCP, named TCP Vegas, with a fundamentally different bandwidth estimation scheme. They have demonstrated that TCP Vegas results in a lower packet loss rate, which in turn leads to higher efficiency, and have suggested that TCP Vegas does not suffer from a delay bias, thus leading to a more fair rate allocation[2]. This was later proved to be the case by Mo *et al.* [19] in the case that there is only a single bottleneck in the network. Also, Low *et al.* have shown that TCP Vegas leads to an allocation that is weighted proportionally fair, given that there is sufficient buffer space in the network [16].

Mo and Walrand [20] in another paper, have proposed and studied another fair window-based end-to-end congestion control mechanism, which is similar to TCP Vegas but has a more sophisticated window updating rule. They have shown that proportional fairness[3] can be achieved by their $(p, 1)$-proportionally fair algorithm and that max-min fairness can be achieved as a limit of (p, α)-proportionally fair algorithms[4] as α goes to ∞ [20].

Clearly fairness is a desirable property for a rate allocation mechanism. There is, however, another aspect of a rate allocation problem that is of increasing importance as the Internet becomes increasingly widely used and thus has a diverse population of users with different valuations for the service they get. We suggest that a good rate allocation mechanism should not only be fair, but should also allocate the available bandwidth in such a way that the overall utility of the users is maximized. Our goal in this paper is to address how this might be achieved using window based rate allocation algorithms. In this paper we describe a pricing mechanism that achieves these goals without requiring knowledge of the users' utility functions and without requiring any explicit feedback from the network.

It is well known that many desirable rate allocations, such as a proportionally fair rate allocation, are a solution to a form of optimization problem [1], [20], [12]. Hence, an algorithm like ours may be used to achieve any such rate allocation in the network.

B. Model and Background

We consider a network with a set $\mathcal{J}$ of resources or links and a set $\mathcal{I}$ of users. Let C_j denote the finite capacity of link $j \in \mathcal{J}$. Each user has a fixed route $\mathcal{J}_i$, which is a non-empty subset of $\mathcal{J}$.[5] We define a zero-one matrix A, where $A_{i,j} = 1$ if link j is in user i's route $\mathcal{J}_i$ and $A_{i,j} = 0$ otherwise. When the throughput of user i is x_i, user i receives utility $U_i(x_i)$. We assume that the utility $U_i(x_i)$ is an increasing, strictly concave and continuously differentiable function of x_i over the range $x_i \geq 0$. Furthermore, we assume that the utilities are additive so that the aggregate utility of rate allocation $x = (x_i, i \in \mathcal{I})$[6] is $\sum_{i \in \mathcal{I}} U_i(x_i)$. Let $U = (U_i(\cdot), i \in \mathcal{I})$ and $C = (C_j, j \in \mathcal{J})$. In this paper we study the feasibility of achieving the maximum total utility of the users in a distributed environment, using only the information available at the end hosts. Under our model this problem can be formulated as follows [9].

SYSTEM(U,A,C):

$$\begin{aligned} maximize \quad & \sum_{i \in \mathcal{I}} U_i(x_i) \\ subject\ to \quad & A^T x \leq C \\ over \quad & x \geq 0 \end{aligned} \tag{1}$$

[1] The most popular versions of TCP in the Internet are TCP Tahoe and Reno.

[2] Fairness refers to max-min fairness that is discussed in section II-B.

[3] Proportional fairness is introduced by Kelly in [9], and is discussed in the next subsection.

[4] (p, α)-proportional fairness is defined in section III.

[5] We assume that there is a routing mechanism and do not discuss the issue of routing in this paper.

[6] Throughout the paper we assume that all vectors are column vectors.

The first constraint in the problem says that the total rate through a resource cannot be larger than the capacity of the resource. Given that the system knows the utility functions U of the users, this optimization problem may be mathematically tractable. However, in practice not only is the system not likely to know U, but also it is impractical for a centralized system to compute and allocate the users' rates due to the large scale of the network. Hence, Kelly in [9] has proposed to consider the following two simpler problems.

Suppose that given the price per unit flow λ_i, user i selects an amount to pay per unit time, p_i, and receives a flow $x_i = \frac{p_i}{\lambda_i}$.[7] Then, the user's optimization problem becomes the following [9].

$USER_i(U_i; \lambda_i)$:

$$\begin{array}{ll} maximize & U_i(\frac{p_i}{\lambda_i}) - p_i \\ over & p_i \geq 0 \end{array} \quad (2)$$

The network, on the other hand, given $p = (p_i, i \in \mathcal{I})$, attempts to maximize the sum of weighted log functions $\sum_{i\in\mathcal{I}} p_i \log(x_i)$. Then the network's optimization problem can be written as follows [9].

NETWORK(A,C;p) :

$$\begin{array}{ll} maximize & \sum_{i\in\mathcal{I}} p_i \log(x_i) \\ subject\ to & A^T x \leq C \\ over & x \geq 0 \end{array} \quad (3)$$

Note that the network does not require the true utility functions $(U_i(\cdot), i \in \mathcal{I})$, and pretends that user i's utility function is $p_i \cdot \log(x_i)$ to carry out the computation. It is shown in [9] that one can always find vectors $\lambda^* = (\lambda_i^*, i \in \mathcal{I}), p^* = (p_i^*, i \in \mathcal{I})$, and $x^* = (x_i^*, i \in \mathcal{I})$ such that p_i^* solves $USER_i(U_i; \lambda_i^*)$ for all $i \in \mathcal{I}$, x^* solves $NETWORK(A, C; p^*)$, and $p_i^* = x_i^* \cdot \lambda_i^*$ for all $i \in \mathcal{I}$. Further, the rate allocation x^* is also the unique solution to $SYSTEM(U, A, C)$. This suggests that the problem of solving $SYSTEM(U, A, C)$ can be achieved by an algorithm that solves $NETWORK(A, C; p(t))$ for a given $p(t)$ at time t on a smaller time scale, and drives $p(t)$ to p^* on a larger time scale.

Another important aspect of a rate allocation mechanism is fairness. There are many different definitions for fairness, the most commonly accepted one being max-min fairness. A rate allocation is max-min fair if a user's rate cannot be increased without decreasing the rate of another user who is already receiving a smaller rate. In this sense, max-min fairness gives an absolute priority to the users with smaller rates. However, often in order to achieve a max-min fair allocation the optimality of the network needs to be sacrificed as discussed in [18]. In order to handle this tradeoff between fairness and optimality Kelly [9] has proposed another definition of fairness. A vector of rates $\breve{x} = (\breve{x}_i, i \in \mathcal{I})$ is said to be *weighted proportionally fair*[8] with weight vector p if $\breve{x}$ is feasible, i.e., $\breve{x} \geq 0$ and $A^T \breve{x} \leq C$, and for any other feasible vector x, the following holds:

$$\sum_{i\in\mathcal{I}} p_i \frac{x_i - \breve{x}_i}{\breve{x}_i} \leq 0$$

Note that a rate allocation x solves $NETWORK(A, C; p)$ if and only if it is weighted proportionally fair with weight vector p.

III. Mo and Walrand's Work

A TCP connection adjusts its rate by updating its window size, based on the estimated congestion state of the network. There are several different versions of TCP, which can be categorized into two classes, based on their bandwidth estimation schemes. TCP Tahoe and Reno use packet losses as an indication of congestion in the network, while TCP Vegas [2] and the algorithm proposed by Mo and Walrand [20] use the estimated queue size to adjust the congestion window size[9].

Over the years researchers have observed that the most widely used versions of TCP, which are TCP Tahoe and Reno, exhibit several undesirable characteristics such as a delay bias and a high packet loss rate [5], [6], [19]. In order to deal with these issues Mo and Walrand [20] have investigated the existence of a fair window-based end-to-end congestion control algorithm that updates the congestion window size more intelligently.

We first present a fluid model of the network that describes the relationship between the window sizes, rates, and queue sizes. Throughout the paper we assume that the switches exercise the first-in-first-out (FIFO) service discipline. This model can be represented by the following equations:

$$A^T x - C \leq 0 \quad (4)$$
$$Q(A^T x - C) = 0 \quad (5)$$
$$X(AC'q + \bar{d}) = w \quad (6)$$
$$x \geq 0, q \geq 0, \quad (7)$$

where

$$C = (C_1, ..., C_J)^T, q = (q_1, ..., q_J)^T,$$
$$\bar{d} = (\bar{d}_1, ..., \bar{d}_I)^T, w = (w_1, ..., w_I)^T,$$
$$X = diag(x), C' = diag(C_1^{-1}, ..., C_J^{-1}), Q = diag(q),$$

w_i is the congestion window size of user i, $\bar{d}_i$ is the round trip propagation delay of route $\mathcal{J}_i$ not including the queueing delay, and q_j denotes the backlog at link j buffer. For simplicity of analysis we assume that the buffer sizes are infinite. The first condition in (4) represents the capacity constraint. The second constraint says that there is backlogged fluid at a resource only if the total rate through it equals the capacity. The third constraint follows from that the window size of connection i should equal

[7] This is equivalent to selecting its rate x_i and agreeing to pay $p_i = x_i \cdot \lambda_i$.

[8] This is expressed by the phrase *rates per unit charge are proportionally fair* in [9]

[9] We call these loss-based and queue-based TCP, respectively.

the sum of the amount of fluid in transit and the backlogged fluid in the buffers, i.e.,

$$w_i = x_i \cdot \bar{d}_i + q^i,$$

where q^i denotes connection i's total backlog in the buffers.

Let $W = [0, w_{1,max}] \times \cdots \times [0, w_{I,max}]$, where $w_{i,max}$ is connection i's maximum congestion window size announced by the receiver and is assumed to be sufficiently large, and $\mathcal{X} = [0, \min_{j \in \mathcal{J}_1} C_j] \times \cdots \times [0, \min_{j \in \mathcal{J}_I} C_j]$. It has been shown in [20], [18] that the rate vector x is a well defined function of the window sizes w, and we can define a function that maps a window size vector $w \in W$ to a rate vector $x \in \mathcal{X}$.[10]

Under the $(p,1)$-proportionally fair algorithm by Mo and Walrand [20] each connection has a target queue size $p_i > 0$ and attempts to keep a total of p_i packets at the switch buffers. Let $d_i(t), w_i(t)$, and $x_i(t)$[11] denote the round trip delay with queueing delay[12], the congestion window size, and the rate of connection i at time t, respectively. Suppose that each connection i has a fixed target queue size p_i. Connection i updates its window size $w_i(t)$ according to the following differential equation

$$\frac{d}{dt} w_i(t) = -\kappa \frac{\bar{d}_i}{d_i(t)} \frac{s_i(t)}{w_i(t)},$$

where κ is some positive constant, and $s_i(t) = w_i(t) - x_i(t) \cdot \bar{d}_i - p_i$. Under this algorithm the window sizes converge to a unique point w^* such that for all $i \in \mathcal{I}$

$$w_i^* - x_i(w^*) \cdot \bar{d}_i = p_i,$$

where $x_i(w^*)$ is connection i's throughput when the window sizes are w^*. They show that at the unique stable point w^* of the algorithm the resulting allocation $x(w^*)$ is weighted proportionally fair.

Suppose that users update their window sizes according to the following system of differential equations

$$\frac{d}{dt} w_i(t) = -\kappa x_i(t) s_i(t) u_i(t),$$

where

$$s_i(t) = w_i(t) - x_i(t) \cdot \bar{d}_i - \frac{p_i}{(x_i(t)+1)^{\alpha-1}}$$

and

$$u_i(t) = \bar{d}_i - (\alpha - 1) \frac{p_i}{(x_i(t)+1)^{\alpha}}.$$

This algorithm is called a (p, α)-proportionally fair algorithm[13]. They prove that the above algorithm converges to a unique stable point of the system for all fixed α and the max-min fair allocation is achieved as a limit as $\alpha \to \infty$.

[10] In the rest of the paper despite the abuse of notation we denote the corresponding rates given the window sizes w by $x(w)$.

[11] We use $x_i(t)$ to denote $x_i(w(t))$ when there is no confusion.

[12] User i's queueing delay at time t is defined to be $\sum_{j \in \mathcal{J}_i} \frac{q_j(t)}{C_j}$, where $q_j(t)$ is the backlog at resource j at time t. Hence, $d_i(t) = \frac{w_i(t)}{x_i(t)} = \bar{d}_i + \sum_{j \in \mathcal{J}_i} \frac{q_j(t)}{C_j}$.

[13] A vector of rates $\hat{x} = (\hat{x}_i, i \in \mathcal{I})$ is (p, α)-proportionally fair if it is feasible and for any other feasible vector x, we have

$$\sum_{i \in \mathcal{I}} p_i \frac{x_i - \hat{x}_i}{\hat{x}_i^{\alpha}} \leq 0.$$

IV. Pricing Scheme & Algorithm

The algorithms suggested in [10], [7], [15] are based on the assumption that the network can provide the necessary feedback to the users, and users adjust their *rates* based on the feedback information. However, in the current Internet, many, if not most, connections use TCP, which is a window-based congestion control mechanism, to control their rates. There are also several arguments for preferring a window-based congestion control algorithm over a rate-based algorithm. The main argument has to do with robustness to estimation errors. Suppose that users use a rate-based congestion control algorithm. If they have incorrect estimates of the available bandwidth and release packets into the network at a rate that is much higher than they should, the network could temporarily experience an extremely high packet loss rate due to buffer overflows and may take a long time to recover from it. A window-based congestion control algorithm not only controls the transmission rate, but also limits the maximum number of outstanding packets according to the congestion window size. This helps alleviate the long term effect of the inaccurate estimation of the available bandwidth by the users. This is an important advantage of a window-based algorithm. In an open system such as the Internet, it is important to control the number of packets the connections can keep within the network for stability and to ensure that no users can arbitrarily penalize other users by increasing their rates during congestion periods due to incorrect estimates.

In this section we adopt the same fluid model used by Mo and Walrand and propose a pricing scheme and a window-based congestion control mechanism that can be used to solve the system problem. We prove the convergence of the algorithm to the system optimal rates.

A. Pricing Scheme

Section III tells us that, given $(p_i, i \in \mathcal{I})$, the users can reach a solution to *NETWORK(A,C;p)* using a window-based congestion control mechanism, namely the $(p,1)$-proportionally fair algorithm of Mo and Walrand. The challenge now is to design a mechanism that drives the users to the right p^* where the resulting rate allocation solves $SYSTEM(U, A, C)$. In this subsection we describe a simple pricing mechanism that can achieve this with the algorithm given in the following subsection. We show that the price a user pays can be directly computed by the user without any feedback from the network, by using the same mechanism already built into TCP.

When the total rate through a link is strictly smaller than its capacity there is no congestion, as each user receives its desired rate without contention from other users. However, when the total rate reaches the link capacity, any increase in a congestion window size by a user i in an attempt to increase its own rate results in increased backlog at the resource. This leads to

higher queueing delay at the resource. If the users are delay sensitive this increase in queueing delay represents an increase in the implicit[14] cost the users pay through larger delay. From the perspective of the network, this increase in queueing delay may be interpreted as an increase in undesirable delay cost for the system. In other words, the queueing delay caused by a user can be interpreted as the delay cost it imposes on other users [17].

Suppose that the network attempts to recover the increase in system cost due to queueing delay through a pricing mechanism as follows. Let $q_j, j \in \mathcal{J}$, denote the backlog at resource j. When the total rate through the resource is strictly smaller than its capacity, we assume that there is no backlog, from (5). When resource $j \in \mathcal{J}$ is congested, i.e., the total rate through it equals its capacity, the resource charges a price, where the price per unit flow g_j is the queueing delay at the resource, i.e., $g_j = \frac{q_j}{C_j}$. Hence, the total price per unit flow for a user with route $\mathcal{J}_i \subset \mathcal{J}$ is $\sum_{j\in\mathcal{J}_i} g_j$, and the total price per unit time user i pays is $x_i \cdot \sum_{j\in\mathcal{J}_i} g_j$. Then, the net benefit or the objective function of the user is given by

$$U_i(x_i) - x_i \cdot \sum_{j\in\mathcal{J}_i} g_j = U_i(x_i) - x_i \cdot \sum_{j\in\mathcal{J}_i} \frac{q_j}{C_j} \tag{8}$$

We have, however, claimed that no information is explicitly fed back to the end hosts from the network. Hence, the switches are not allowed to send any information regarding the current price per unit flow back to the end users.

In order to maximize the objective function in (8) given the *current prices per unit flow*, a user only needs to know the total price per unit flow of its route, but not the price at each resource. We now show that users using TCP can compute their prices per unit flow without any help from the network. Suppose that each connection knows the propagation delay of its route. In practice this is done by keeping track of the minimum round trip time of the packets [2], [19]. Suppose that the target queue sizes of the users are given by $p = (p_i, i \in \mathcal{I})$, and the users' window sizes converge to the stable point of TCP, where the resulting rates are weighted proportionally fair with the weight vector p. Then, the price per unit time user i pays, h_i, can be computed as follows:

$$h_i = x_i \cdot \sum_{j\in\mathcal{J}_i} \frac{q_j}{C_j} = \sum_{j\in\mathcal{J}_i} q_j \cdot \frac{x_i}{C_j} = \sum_{j\in\mathcal{J}_i} q_j^i = p_i, \tag{9}$$

where q_j^i is connection i's queue size at resource j. Here we have implicitly assumed that the queue size of each connection at a congested resource is proportional to its rate, which is a consequence of the assumption that the switches exercise FIFO service discipline. Equation (9) tells us that the users can infer the current price from the queueing delay and use their target queue sizes to solve the user problem with our pricing scheme.[15] This eliminates the need for any feedback from the network. One thing to note is that the rate of a user depends not only on its own target queue size, but also on those of other users. Hence, the prices per unit flow at the resources depend on the congestion level in the network or the demand of the users.

One important aspect of a pricing scheme is fairness. The price a connection pays for using resource $j \in \mathcal{J}$ should be proportional to its rate. In other words, the price per unit flow for each connection at a resource, should be the same. This is obviously the case with the above pricing scheme. Moreover, it is easy to see that when a new user comes into the network, the price the new user pays is exactly the increase in the total system cost, i.e., the increase in the total queueing delay experienced by the packets. This can be seen from that the price user i pays per unit time equals its target queue size p_i and the total system cost per unit time is given by

$$\sum_{j\in\mathcal{J}} C_j \cdot g_j \quad = \quad \sum_{j\in\mathcal{J}} C_j \cdot \frac{q_j}{C_j} = \sum_{j\in\mathcal{J}} q_j = \sum_{i\in\mathcal{I}} p_i.$$

Therefore, the fairness of the pricing scheme is automatically guaranteed.

In the following subsection, based on this pricing mechanism, we propose an algorithm that can be deployed over the current Internet without an extensive modification inside the network. All that is required is a modification of the already existing TCP at the end hosts. We demonstrate, using a fluid model, that at an equilibrium of the algorithm the resulting rates are the optimal rates that solve $SYSTEM(U, A, C)$.

B. Algorithm

The algorithm we have proposed in [13] requires that the users[16] explicitly compute the optimal prices based on the current prices per unit flow and use the (p, 1)-proportionally fair algorithm to solve the *NETWORK(A, C; p)* problem with the given prices $p = (p_i, i \in \mathcal{I})$. In other words, the problem is formulated as a discrete model where, given the prices per unit flow from the previous period $n-1$, users solve $USER(U_i; \lambda_i(n-1))$ for period n. Then the (p, 1)-proportionally fair algorithm is used to drive the users to the solution of *NETWORK(A, C; p(n))*. Theorem 1 in [13] proves that there exists a unique equilibrium, where users' choices of prices or target queue sizes remain the same, i.e., no user changes its price given a chance to update its price, and the resulting rates at the equilibrium are the system optimal rates. There are, however, a few implementation issues that need to be addressed with this algorithm. For instance, since the (p, 1)-proportionally fair algorithm converges asymptotically, the users need to know how long they should wait before updating their prices again or how often they should update their prices. Further, the convergence of the algorithm to the optimal rates has been established only for simple single bottleneck networks.

In this subsection we introduce an algorithm that does not require a computation of the optimal prices, but the users' preferences are implicitly reflected in the window size updating rule. For the rest of this paper we assume that the utility functions satisfy the technical assumption of Appendix A. This assumption

[14] We say the cost is implicit because the users do not necessarily have to pay in monetary form.

[15] Our algorithm discussed in the next subsection maintains the queue-based congestion control mechanism nature of (p, 1)-proportionally fair algorithm, and at the same time updates its target queue size p_i based on the utility function $U_i(\cdot)$.

[16] In practice an agent at the end host may carry out the computation on behalf of the user.

is satisfied by the most commonly used utility functions such as $U(x) = a \cdot \log(x+b)$ and $U(x) = c \cdot x^d$, where $a > 0, 0 \leq b \leq 1$, $c > 0$, and $0 < d < 1$ are arbitrary constants. Suppose that the users update their window sizes according to the following system of differential equations:

$$\frac{dw_i(t)}{dt} = -\kappa \cdot M_i(t) \cdot r_i(t) \tag{10}$$

where

$$M_i(t) = \frac{\bar{d}_i + U_i'(x_i(t)) + x_i(t) \cdot U_i''(x_i(t))}{d_i(t)}, \tag{11}$$

$U_i'(\cdot) = \frac{d}{dx_i} U_i(\cdot), U_i''(\cdot) = \frac{d^2}{dx_i^2} U_i(\cdot)$, and

$$\begin{aligned} r_i(t) &= \frac{w_i(t) - x_i(t)\bar{d}_i - x_i(t)U_i'(x_i(t))}{w_i(t)} \\ &= 1 - \frac{x_i(t)(\bar{d}_i + U_i'(x_i(t)))}{w_i(t)}. \end{aligned} \tag{12}$$

Note that user i's utility function appears in both $M_i(t)$ and $r_i(t)$.

The following theorem states that the algorithm given by (10) - (12) converges to the unique stable point, where the resulting rates solve the *SYSTEM(U, A, C)* problem.

Theorem 1: Let $V(w) = \sum_i (r_i(w))^2$. Then V is a Lyapunov function for the system of differential equations (10) - (12). The unique value minimizing V is a stable point of this system, to which all trajectories converge.

Proof: The proof is given in Appendix B. ■

In this section for the purpose of analysis we have assumed that users are delay insensitive. However, if the users are sensitive to delay and the cost due to queueing delay is given by h_i in (9), then the algorithm does not require any pricing mechanism. Recall that the purpose of the pricing mechanism is to impose the system cost due to queueing delay on the users in an incentive-compatible way. In this case when users maximize their utility functions with *explicit* delay cost, the resulting rate allocation at an equilibrium is the optimal rate allocation.

V. Numerical Example

In this section we give a numerical example of a simple network and demonstrate the convergence of users' parameters through simulation. The simulation is run using the Network Simulator (NS) developed at Lawrence Berkeley Laboratory (LBL) and UC Berkeley.

The topology of the simulated network is given in Figure 1. There are 18 users that share the network, and the routes of the users are as listed in Figure 1. The capacities and delays of the links are given next to the links in the figure. Table I has the utility functions, optimal rates, and equilibrium prices (or target queue sizes in packets) of the users.[17] In this simulation each connection measures its rate, computes $M_i(t)$ and $r_i(t)$, and updates its window size according to (10) - (12) once per round trip time. The parameter κ is set to 0.1 for all connections. Packet sizes are fixed at 1,000 bytes. The simulation is run for 20 seconds.

[17] The rates x in $U_i(x)$ are in packets per second, while x_i^*'s are given in Mbps.

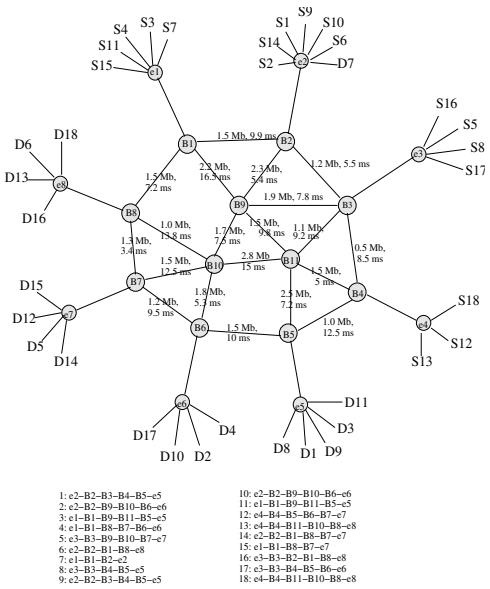

Fig. 1. Topology of the simulated network and the users' routes.

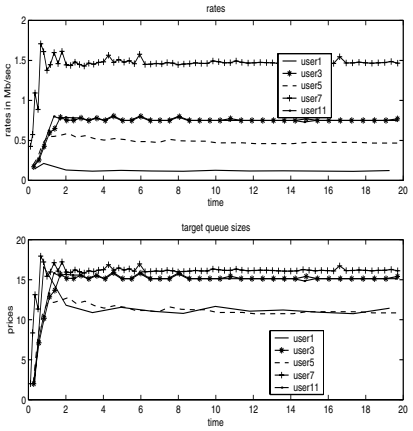

Fig. 2. Convergence of user prices and rates.

The evolution of the rates and prices of users 1, 3, 5, 7, and 11 are plotted in Figure 2. One can see that users' rates converge to a region around the optimal rates and the prices converge to the unique equilibrium. Since the users use instantaneous values of the rates and the queueing delays, their rates oscillate slightly around the optimal rates.

The convergence rate of the algorithm obviously depends on the parameters such as κ. Further, simulation results indicate that the convergence rate of the algorithm decreases with the round trip delays of the users as expected. This is because the window sizes of the users are updated only once per round trip time. Suppose that a user enters a network with many users that is already in an equilibrium with the existing users. Then, the simulation

$user$	$U_i(x)$	x_i^* (Mbps)	p_i^*
1	$2.4x^{0.7}$	0.121	11.25
2	$0.9x^{0.7}$	0.497	11.34
3	$0.9x^{0.7}$	0.750	15.13
4	$1.2x^{0.7}$	0.180	7.43
5	$0.9x^{0.7}$	0.497	11.34
6	$1.5x^{0.7}$	0.380	15.66
7	$0.6x^{0.7}$	1.500	16.38
8	$2.5x^{0.7}$	0.138	12.85
9	$2.4x^{0.7}$	0.121	11.25
10	$2.4x^{0.7}$	0.706	16.11
11	$0.9x^{0.7}$	0.750	15.16
12	$0.9x^{0.7}$	0.500	15.18
13	$1.2x^{0.7}$	0.500	11.39
14	$0.9x^{0.7}$	0.380	15.66
15	$1.5x^{0.7}$	0.180	7.43
16	$0.6x^{0.7}$	0.380	15.66
17	$2.5x^{0.7}$	0.121	11.25
18	$2.4x^{0.7}$	0.500	11.39

TABLE I

UTILITY FUNCTIONS OF THE USERS.

results suggest that with a wide range of round trip delays and a reasonable choice for the value of κ the system takes no more than a few seconds to converge to a small neighborhood of the new equilibrium. This fast convergence can be explained from that the arrival of a new user does not perturb the existing users too much, and this allows the new user to quickly estimate the price per unit flow of its route and reach the equilibrium. Due to the size of the Internet and a large number of users, if the arrivals and departures of the users are reasonably frequent, we expect the system to remain close to an equilibrium at any given time.

VI. CONTROLLING THE BACKLOG AND QUEUEING DELAY IN THE NETWORK

In the previous sections for simplicity of analysis we have assumed that the buffer sizes at the resources are infinite. In the real network, however, the buffer sizes are finite and there may be packet losses due to temporary buffer overflows. In this section we discuss how our pricing scheme and algorithms can be extended to cope with these packet losses.

In our pricing scheme we have assumed that the price per unit flow charged by a resource j is the queueing delay at the resource. Suppose that the price per unit flow at a resource is proportional to the queueing delay, i.e., $g_j = \gamma \cdot \frac{q_j}{C_j}$, where $\gamma > 0$, but not necessarily equal to the queueing delay. In this case the users face the same user optimization problem, except that now the target queue size should be set to its willingness to pay divided by γ. More specifically, $U_i'(x_i(t))$ and $U_i''(x_i(t))$ in (11) and (12) need to be replaced by $\gamma^{-1} \cdot U_i'(x_i(t))$ and $\gamma^{-1} \cdot U_i''(x_i(t))$, respectively. One can show that under this more general pricing scheme and modified algorithms, there exists a unique equilibrium of the algorithms, and at the equilibrium the resulting rates are the optimal rates that solve *SYSTEM(U, A, C)*. In other words, the equilibrium prices and resulting rates are invariant under change of γ. One consequence of this is the following. Because the equilibrium prices are the same for all $\gamma > 0$, the target queue sizes of the users are inversely proportional to γ. Therefore, by increasing γ we can arbitrarily reduce the target queue sizes of the users and, hence, the backlog inside the network. Further, as γ goes to ∞, one can see that the queueing delay goes to 0 for all users because the target queue sizes of the users go to 0. What this means in practice is that γ can be chosen to ensure that packet losses are negligible even when the buffer sizes are finite.

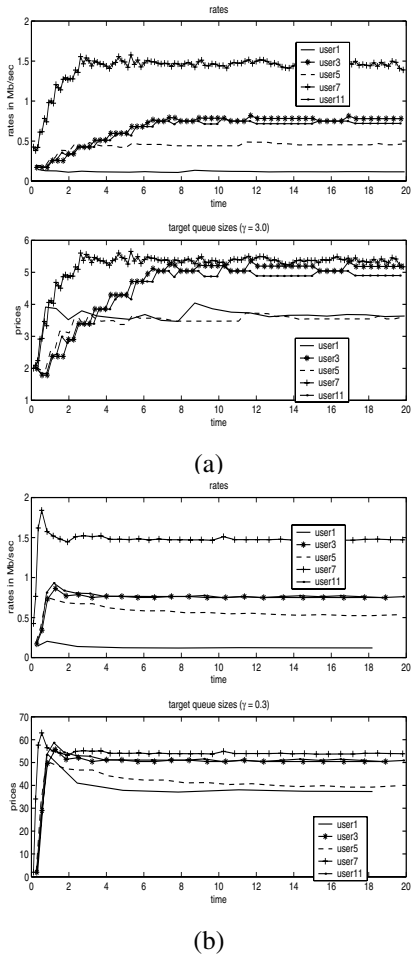

Fig. 3. Convergence of user prices and rates with various γ values. (a) γ = 3.0 (b) γ = 0.3.

We demonstrate this through simulation. Simulations are run with the same network topology and set of users as in section V. We set γ to 3 and 0.3 in the first and second simulation, respectively. The simulations are run for 20 seconds. Figure 3 shows the rates and target queue sizes of the users. One can easily see that users' rates converge to similar values, while the target queue sizes are inversely proportional to γ, i.e., the product of γ and target queue size of each user remains constant. This is consistent with the claim made above.

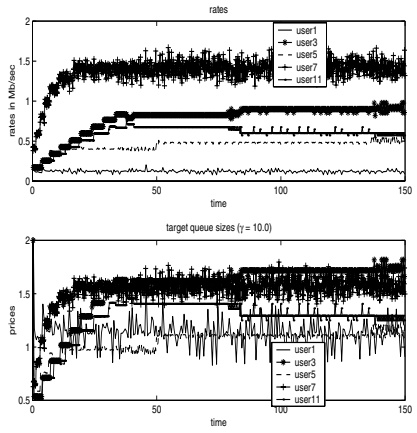

Fig. 4. Increase in measurement noise and decrease in convergence rate.

Maintaining small queue sizes in the network is good from the perspective of the network. Choosing a large value of γ forces users to have small target queue sizes, and hence helps keep small queue sizes in the network. However, since our algorithm uses the queueing delay to estimate the congestion level in the network, the sensitivity of the algorithm to the measurement noise increases with γ because queue sizes are time varying and any fluctuation in queue sizes is magnified by γ. As a result not only the convergence rate of the algorithm decreases with γ, but also the algorithm may not even converge to the optimal rates in a real network if γ is set too large.

This is illustrated in Figure 4. The same network topology and set of users in section V are used, and γ is set to 10. The simulation is run for 150 seconds. The plots clearly show that the relative oscillation in users' target queue sizes increases noticeably from the previous simulations, and the rates of the users do not converge to the same optimal rates as in the previous simulations. This can be explained as follows. In order for the users to get an accurate estimate of the queueing delay, the queue sizes inside the network should be relatively stable and not vary fast compared to the round trip delays of the users. However, as shown in Figure 4 the target queue sizes of the users are no larger than 2 packets. Since packets of all users must traverse through at least 4 routers, on the average users have less than 0.5 packets backlogged at each router. This means that, with a small number of users going through each router, the queue sizes are likely to fluctuate widely at a rate faster than the round trip delays of the users. As a result of this fast fluctuation in queue sizes the users may not be able to obtain an accurate estimation of the congestion level in the network, which is necessary for the convergence of the algorithm. The fact that in the Internet there are a large number of connections going through each router and the relative fluctuation in the queue size during a congestion period may not be large can potentially help the users obtain a better estimate of congestion level even with a large γ. However, choosing an appropriate range of γ values requires further study.

VII. Conclusions

In this paper we have investigated the fundamental problem of the existence of an algorithm that achieves the system optimum in a distributed network without any explicit feedback from the network elements to the end hosts. We have described a pricing scheme and an algorithm that solves the system problem at the unique equilibrium point of the algorithm. We have demonstrated that our algorithm is incentive compatible and does not require any feedback from the network. Moreover, it achieves weighted proportional fairness. It is also important to note that the algorithm is window based. The algorithm is proven to converge to the system optimum in any general network under a mild technical assumption on the utility functions. A numerical example is given to demonstrate the convergence of the algorithm.

In this paper we have assumed that the routes of the users are fixed. However, if the network charges the users based on the congestion level along the route of the connection, the network should ensure that no users get preferential treatment or penalized through the selection of the routes. For instance, the routing has to be done in such a way that no user is routed through a path that has the highest price per unit flow among the available paths, while other connections are routed through the cheapest path.

References

[1] Sanjeewa Athuraliya, Steven Low, and David Lapsley. Random Early Marking. Preprint - available at http://www.ee.mu.oz.au/staff/slow/research, January 2000.

[2] L.S. Brakmo and L.L. Peterson. Tcp vegas: end to end congestion avoidance on a global internet. *IEEE Journal on Selected Areas in Communications*, 13(8):1465–80, October 1995.

[3] P. Dubey. Inefficiency of Nash equilibria. *Mathematics of Operations Research*, 11:1–8, 1986.

[4] R. Edell and P. Varaiya. INFOCOM 1999 Keynote Talk. available at http://www.INDEX.berkeley.edu/reports/99-009S, March 1999.

[5] S. Floyd and V. Jacobson. Connection with multiple Congested Gateways in packet-Switched Networks,Part1: One-way Traffic". *ACM Computer Communication Review*, 21(5):30–47, August 1991.

[6] S. Floyd and V. Jacobson. Random Early Detection Gateways for Congestion Avoidance. *IEEE/ACM Transactions on Networking*, 1(4):397–413, August 1993.

[7] R. J. Gibbens and F. P. Kelly. Resource pricing and the evolution of congestion control. Available at http://www.statslab.cam.ac.uk/˜ frank, June 1998.

[8] V. Jacobson. Congestion avoidance and control . *Computer Communication Review*, 18(4):314–29, August 1988.

[9] F. Kelly. Charging and rate control for elastic traffic. *European Transactions on Telecommunications*, vol.8(1):33–7, January 1997.

[10] F. Kelly, A. Maulloo, and D. Tan. Rate control for communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, vol.49(3):237–52, March 1998.

[11] F.P. Kelly. On tariffs, policing, and admission control for multiservice networks. *Operations Research Letters*, vol.15(1):1–20, Feb. 1994.

[12] Frank Kelly. Lecture Series - Stochastic Network Conference and Workshop. June 19-30 2000.

[13] R. J. La and V. Anantharam. Charge-Sensitive TCP and Rate Control in the Internet. In *Proc. IEEE INFOCOM '00*, March 2000.

[14] T. V. Lakshman and U. Madhow. The performance of TCP/IP for networks with high bandwidth-delay products and random loss. *IEEE/ACM Transactions on Networking*, 5(3):336–50, June 1997.

[15] S. H. Low and D. E. Lapsley. Optimization flow control. *IEEE/ACM Transactions on Networking*, 7(6):861–74, Dec. 1999.

[16] Steven. Low, Larry Peterson, and Limin Wang. Understanding Vegas: Theory and Practice. available at http://www.ee.mu.oz.au/staff/slow/research/projects.html, Feb. 2000.

[17] J.K. Mackie-Mason and H. R. Varian. Pricing congestible network resources. *IEEE Journal on Selected Areas in Communications*, 13(7):1141–9, September 1995.

[18] L. Massoulie and J. Roberts. Bandwidth sharing : objectives and algorithms. In *Proceedings of IEEE INFOCOM '99*, New York, NY, March 1999.

[19] J. Mo, R. J. La, V. Anantharam, and J. Walrand. Analysis and Comparison of TCP Reno and Vegas. In *Proc. INFOCOM '99*, March 1999.
[20] J. Mo and J. Walrand. Fair end-to-end window-based congestion control. Technical Report M98/46, UCB/ERL, July 1998. (To appear in IEEE/ACM Transactions on Network - available at http://www.path.berkeley.edu/~jhmo).
[21] W. Rudin. *Principles of Mathematical Analysis, 3rd Edition*. McGraw-Hill, 1976.
[22] H.R. Varian and J.K. Mackie-Mason. Pricing the internet. *Public Access to the Internet*, 1994.

APPENDIX

I. ASSUMPTION ON THE UTILITY FUNCTIONS

Assumption 1: Utility functions of the users satisfy the following:

$$U_i'(x_i) + x_i \cdot U_i''(x_i) \geq 0,\ x_i \in [0, C^i],$$

where $C^i = \min_{j \in \mathcal{J}_i} C_j$.

II. PROOF OF THEOREM 1

We first define interior and boundary points. An interior point is defined to be a window size vector, around which we can find an open ball such that all the points in the ball have the same set of bottlenecks. A window size vector, for which we cannot find such an open ball is said to be a boundary point. One can show that the boundary regions are linear and divide the window size space into a finite number of convex regions. Hence, if $w(t)$ is an interior point, then for sufficiently small $\epsilon > 0$, $w(t+\epsilon)$ is in the same region as $w(t)$. If $w(t)$ is a boundary point, then for any sufficiently small $\epsilon > 0$ $w(t+\epsilon)$ is in one of the neighboring regions of the boundary point.

Recall that $V(t) = \sum_i (r_i(t))^2$, where

$$r_i(t) = \frac{w_i(t) - x_i(t)(\bar{d}_i + U_i'(x_i(t)))}{w_i(t)}.$$

We show that for small $\epsilon > 0$,

$$V(t+\epsilon) - V(t) = r^T(t) J_r(t)(w(t+\epsilon) - w(t)) + o(\epsilon), \quad (13)$$

where $J_r(t)$ is the partial derivative

$$J_r(t) = \left[\frac{\partial r_i}{\partial w_j}, i, j \in \mathcal{I}\right],$$

which depends on the region in which $w(t+\epsilon)$ is.

We first state a claim that will be used in the proof.

Claim 1: At an interior point w the partial derivative $J_x = \left[\frac{\partial x_i}{\partial w_j}, i, j \in \mathcal{I}\right]$ is given by

$$J_x = \bar{D}^{-1}(I - X A_B (A_B^T X \bar{D}^{-1} A_B)^{-1} A_B^T \bar{D}^{-1}), \quad (14)$$

where $\bar{D} = \text{diag}(d_1(w), \cdots, \bar{d}_i(w))$ and $\bar{d}_i(w) = \bar{d}_i + \sum_{j \in \mathcal{J}_i} \frac{q_j(w)}{C_j}$ and where A_B is the submatrix of A with columns corresponding to the bottlenecks under w.

Proof: We first show that the mapping $\mathcal{W}$ from the window sizes to the rates is differentiable at the interior points. Suppose that w is an interior point and $x = x(w)$. Let $\mathcal{B}$ be the set of bottlenecks at w. We consider two cases.

Case (1) A_B has full rank, i.e., $rank(A_B) = \text{B} = |\mathcal{B}|$. Let

$$g_B(q_B', w) = A_B^T x(q_B', w) - C_B, \quad (15)$$

where

$$x_i(q_B', w) = \frac{w_i}{\bar{d}_i + A_{i\cdot} q_B'} \quad (16)$$

and $A_{i\cdot}$ is the i-th row of A_B. We use q_B' to denote the queueing delays, not queue sizes that are denoted by q_B. We define

$$J_{g_q} = \left[\frac{\partial g_i}{\partial q_j'}, i, j = 1, \cdots, \text{B}\right],$$

where

$$\begin{aligned}
\frac{\partial g_i}{\partial q_j'} &= \frac{\partial}{\partial q_j'}\Big(\sum_m A_{m,i} x_m(q_B', w) - C_i\Big) \\
&= \frac{\partial}{\partial q_j'}\Big(\sum_m A_{m,i} \frac{w_m}{d_m + \sum_k A_{m,k} q_k'}\Big) \\
&= \sum_m A_{m,i}\Big(\frac{\partial}{\partial q_j'} \frac{w_m}{d_m + \sum_k A_{m,k} q_k'}\Big) \\
&= \sum_m A_{m,i}\Big(-\frac{w_m}{(d_m + \sum_k A_{m,k} q_k')^2}\Big) A_{m,j} \\
&= -\sum_m A_{m,i} A_{m,j} \frac{w_m}{(d_m + \sum_k A_{m,k} q_k')^2} \\
&= -\sum_m A_{m,i} A_{m,j} \frac{x_m(q_B', w)}{\bar{d}_m(q_B')},
\end{aligned}$$

where $\bar{d}_m(q_B') = d_m + A_{m\cdot} \cdot q_B'$. This can be written in a matrix form as

$$J_{g_q} = -A_B^T X \bar{D}^{-1} A_B, \quad (17)$$

where $\bar{D} = \text{diag}(\bar{d}_1, \cdots, \bar{d}_I)$.

One can easily show that J_{g_q} is nonsingular because $A_B^T X \bar{D}^{-1} A_B$ is positive definite. Therefore, by the implicit function theorem, $q_B'(w)$ is differentiable [21], and so is $x(q_B', w)$ from (16). Further, from the implicit function theorem,

$$J_q(w) = \left[\frac{\partial q_j'}{\partial w_i}, j = 1, \cdots, \text{B}, i \in \mathcal{I}\right] = -(J_{g_q})^{-1} J_{g_w},$$

where

$$J_{g_w} = \left[\frac{\partial g_j}{\partial w_i}, j = 1, \cdots, \text{B}, i \in \mathcal{I}\right] = A_B^T \bar{D}^{-1} \quad \text{from (15)}.$$

Thus,

$$J_q(w) = (A_B^T X \bar{D}^{-1} A_B)^{-1} A_B^T \bar{D}^{-1}. \quad (18)$$

Case (2) $rank(A_B) = \gamma < \text{B}$. Let $\bar{\mathcal{B}} \subset \mathcal{B}$ such that $rank(A_{\bar{\mathcal{B}}}) = \gamma$ and $|\bar{\mathcal{B}}| = \gamma$. Then, applying the implicit function theorem to $g_{\bar{B}}$ as defined in (15) yields the differentiability

of $x(q'_{\bar{B}}, w)$. One can show that $x(q'_{\bar{B}}, w)$ is the unique solution of g_B as follows. Since we delete dependent rows from A_B^T, if x is a solution of $A_{\bar{B}}^T x = C_{\bar{B}}$, then it is a also a solution of $A_B^T x = C_B$. Since the solution is unique, $x(q'_{\bar{B}}, w)$ is the solution of g_B. Therefore, the differentiability follows.

We proceed to prove the claim with the assumption that A_B has full rank. If not, we can take $A_{\bar{B}}$ as described before and prove the claim. Since A_B has full rank, we know that the queueing delays are well defined from [20]. First, from (6) for all $i \in \mathcal{I}$ we have

$$w_i = x_i(w) \cdot \bar{d}_i + x_i(w) A_{i\cdot} q'_B(w). \tag{19}$$

If we differentiate (19) with respect to $w_j, j \in \mathcal{I}$,

$$\delta_{i,j} = (A_{i\cdot} q'_B(w) + \bar{d}_i)(J_x)_{i,j} + x_i(w)(A_{i\cdot}(J_q)_{\cdot j}).$$

Again, note that J_q is well defined from (18). In a matrix form this is given by

$$I = \bar{D} J_x + X A_B J_q,$$

which yields

$$J_x = \bar{D}^{-1}(I - X A_B J_q).$$

Since $J_q = (A_B^T \bar{D}^{-1} X A_B)^{-1} A_B^T \bar{D}^{-1}$ from (18), we get

$$J_x = \bar{D}^{-1}(I - X A_B (A_B^T \bar{D}^{-1} X A_B)^{-1} A_B^T \bar{D}^{-1}).$$

■

We continue the proof of the theorem. If $w(t)$ is an interior point, then from the algorithm and that $x(w)$ is differentiable at interior points from the above claim, we have

$$\begin{aligned} J_r &= DXW^{-2} - DW^{-1}J_x + XU'W^{-2} \\ &\quad -W^{-1}(U' + XU'')J_x \\ &= DXW^{-2} - DW^{-1}\bar{D}^{-1} \\ &\quad +DW^{-1}\bar{D}^{-1}XA_B(A_B^T X \bar{D}^{-1} A_B)^{-1} A_B^T \bar{D}^{-1} \\ &\quad +XU'W^{-2} - W^{-1}(U' + XU'')\bar{D}^{-1} \\ &\quad +W^{-1}(U' + XU'')\bar{D}^{-1}XA_B(A_B^T X \bar{D}^{-1} A_B)^{-1} \\ &\quad \times A_B^T \bar{D}^{-1} \qquad (20) \\ &= (D + U' + XU'')\bar{D}^{-2} A_B (A_B^T X \bar{D}^{-1} A_B)^{-1} A_B^T \bar{D}^{-1} \\ &\quad -W^{-1}XU''\bar{D}^{-1}, \qquad (21) \end{aligned}$$

where
$W = \text{diag}(w(t))$, $X = \text{diag}(x(t))$, $\bar{D} = \text{diag}(d_1(t), \cdots, d_i(t))$, $D = \text{diag}(d_1, \cdots, \bar{d}_i)$, $U' = \text{diag}(U_1'(x_1(t)), \cdots, U_I'(x_I(t)))$, $U'' = \text{diag}(U_1''(x_1(t)), \cdots, U_I''(x_I(t)))$, and A_B is the submatrix of A with columns that correspond to bottlenecks at $w(t)$[18]. If $w(t)$ is a boundary point, then A_B is the submatrix of A with columns that correspond to bottlenecks at $w(t+\epsilon)$. The second inequality in (20) follows from that $J_x = \bar{D}^{-1}(I - XA_B(A_B^T X \bar{D}^{-1} A_B)^{-1} A_B^T \bar{D}^{-1})$.

[18] For sufficiently small $\epsilon > 0$, $w(t+\epsilon)$ has the same bottlenecks as $w(t)$.

Fix $\epsilon > 0$. Define $\delta w(t, \epsilon) = \frac{w(t+\epsilon) - w(t)}{\epsilon}$. Then, since $J_r(\cdot)$ is well defined in $[t, t+\epsilon]$ if the path $w(t')$ follows the straight line between $w(t+\epsilon)$ and $w(t)$ and $V(\cdot)$ is a continuous function of window sizes, one can see that

$$\begin{aligned} V(t+\epsilon) - V(t) &= \int_t^{t+\epsilon} r^T(s) J_r(s) \delta w(t,\epsilon) ds \\ &= \int_t^{t+\epsilon} (r(t) + \Delta r(s))^T [J_r(t) + \Delta J_r(s)] \delta w(t,\epsilon) ds, \end{aligned}$$

where $\Delta r(s) = r(s) - r(t)$, $\Delta J_r(s) = J_r(s) - J_r(t)$, $s \in [t, t+\epsilon]$, and $J_r(\cdot)$ is defined as described above. Then,

$$\begin{aligned} V(t+\epsilon) - V(t) &= \int_t^{t+\epsilon} r^T(t)\, J_r(t)\, \delta w(t,\epsilon)\, ds \\ &\quad + \int_t^{t+\epsilon} r^T(t)\, \Delta J_r(s)\, \delta w(t,\epsilon)\, ds \\ &\quad + \int_t^{t+\epsilon} \Delta r^T(s)\, J_r(t)\, \delta w(t,\epsilon)\, ds \\ &\quad + \int_t^{t+\epsilon} \Delta r^T(s)\, \Delta J_r(s)\, \delta w(t,\epsilon)\, ds \\ &= r^T(t)\, J_r(t)\, (w(t+\epsilon) - w(t)) \\ &\quad + r^T(t) \cdot \int_t^{t+\epsilon} \Delta J_r(s) ds \cdot \delta w(t,\epsilon) \\ &\quad + \int_t^{t+\epsilon} \Delta r^T(s) ds \cdot J_r(t) \cdot \delta w(t,\epsilon) \\ &\quad + \int_t^{t+\epsilon} \Delta r^T(s)\, \Delta J_r(s) ds \cdot \delta w(t,\epsilon) \\ &= r^T(t)\, J_r(t)\, (w(t+\epsilon) - w(t)) \\ &\quad + o(\epsilon). \qquad (22) \end{aligned}$$

Hence, from (22) if $r(t) \neq 0$,

$$\begin{aligned} &\lim_{\epsilon \downarrow 0} \frac{V(t+\epsilon) - V(t)}{\epsilon} \\ &= \lim_{\epsilon \downarrow 0} \frac{r^T(t) J_r(t) (w(t+\epsilon) - w(t))}{\epsilon} + \lim_{\epsilon \downarrow 0} \frac{o(\epsilon)}{\epsilon} \\ &= r^T(t) J_r(t) \dot{w}(t) = -\kappa r^T(t) J_r(t) M(t) r(t) < 0, \end{aligned}$$

where $M(t) = \bar{D}^{-1}(D + U' + X \cdot U'')$. The last inequality follows from that $J_r(t)M(t)$ is a positive definite matrix for (20). Note that for a boundary point $w(t)$ such that $\dot{w}(t)$ lies along a boundary region $r^T(t) J_r(t) \dot{w}(t)$ is the same for all $J_r(t)$ with different A_B of the neighboring regions. Hence, the limit is well defined. Since (23) is true for all t unless $r_i(t) = 0$ for all $i \in \mathcal{I}$, the convergence follows.