Flow Control in Wireless Networks

Minghua Chen, *Student Member, IEEE,* Avideh Zakhor *Fellow, IEEE,* Ryusuke Fujita

***Abstract*—Flow control, including congestion control for data transmission, and rate control for multimedia streaming, is important for both wireline and wireless networks. Widely accepted flow control methods in wireline networks are TCP for data, and TCP Friendly Rate Control (TFRC) for multimedia. However, TCP and TFRC both assume that packet loss in wireline networks is primarily due to congestion, and as such, are not applicable to wireless networks in which the bulk of packet loss is due to errors at the physical layer. In this paper, we propose a new class of schemes for flow control in wireless networks that control the number of connections within an application. We show that these schemes implicitly solve a convex optimization problem, of which Kelly's wireline analysis is a special case. We show users' rates controlled by proposed schemes converge to a unique equilibrium exponentially. Our approach is end-to-end, and the control law is based on only one bit of information that can be reliably estimated at the application layer. The proposed scheme is different from existing approaches in that it requires no modifications to existing protocols such as TCP, or infrastructure elements such as routers. We characterize performance of the proposed scheme using NS-2 simulations and actual experiments over Verizon Wireless 1xRTT and EVDO data networks.**

I. Introduction

TCP has been widely successful on the wireline Internet for decades [1]. The *key* assumption TCP relies on is that packet loss is a sign of congestion. In wireless networks however, packet loss can also be caused by physical channel errors. As a result, the congestion assumption breaks down, and TCP could seriously underutilize the wireless bandwidth. The same arguments and observations hold for TCP-friendly schemes, such as TCP-Friendly Rate Control (TFRC) commonly used for streaming applications [2]. In particular, our experiments over Verizon Wireless 1xRTT wireless data network have shown that TFRC achieves only 56% of the available wireless bandwidth [3]. The need to solve the problem is becoming urgent as wireless data and streaming services are becoming more popular.

Consequently, there have been a number of efforts to improve the performance of TCP or TFRC over wireless [4]–[9]. These approaches either hide packet loss caused by wireless channel error from hosts, e.g. Snoop [4], or provide end-hosts the ability to distinguish between packet loss caused by congestion, and that caused by wireless channel error, e.g. end-to-end statistics based schemes [6]–[9]. By these means, the flow control over wireless problem can be solved by first translating it into flow control in wireline network problem, and then applying the known scheme, i.e. TCP or TFRC.

An alternative is to use non-loss based rate control schemes. For instance, TCP Vegas [10] and TCP Fast [11], in their congestion avoidance stages, use queueing delay as a measure of congestion, and hence could be designed to be robust to any kind of packet loss. It is also possible to enable the routers with ECN marking capability to control rate using ECN as a measure of congestion [12]. As packet loss no longer corresponds to congestion, an ECN based scheme does not adjust sending rate upon observing packet loss caused by wireless channel error.

While these approaches provide insight on how to improve performance of TCP and TFRC over wireless, they do require major modifications to network infrastructure elements, such as routers and base stations, or the transport layer protocol stack inside operating systems of clients such as computers and end-user devices such as cell phones. These modifications make the above approaches hard to deploy in practice.

In the past decade, there has been a great deal of research activity on decentralized flow control algorithms on wireline networks. A widely recognized setting, introduced by Kelly et. al. [13] and Low and Lapsley [14], views flow control schemes as primal and dual algorithms that pursue optimal solution of a utility maximization problem based on feedback from networks. Refining the model for TCP in [15], Kelly has shown that TCP is in fact a primal like algorithm with packet loss rate as the feedback . The stability of TCP with and without delay, with and without disturbance, are developed in [15].

Similar work has focused on relating different versions of TCP to different utility functions, evaluating network stability with and without delay or noise, and more importantly designing new TCP and queue management algorithms [11], [14], [16]–[19]. Although these algorithms provide new insights and features to existing flow control techniques, they require modifications to either routers or transport layer protocols, making them hard to deploy in practice. In this paper, we develop a general framework for solving arbitrary flow control problems, without requiring modifications to today's network infrastructure, such as router, nor the protocol stack in the operating system.

We assume a wireless link is associated with a fixed bandwidth and a fixed packet loss rate caused by the physical channel errors. In the presence of physical channel error, TCP and TFRC converge to an equilibrium that could cause serious underutilization in the wireless networks[1]. We formulate the flow control in wireless networks problem as a concave optimization problem, of which Kelly's wireline one is a special case. We then propose a new class of schemes, which are end-to-end, and achieve reasonable performance by adjusting the number of application's connections according to

This work was supported by AFOSR contract F49620-00-1-0327.

All authors are with Department of Electrical Engineering and Computer Science at University of California at Berkeley, Berkeley, CA 94720 USA (e-mail: {minghua@eecs., avz@eecs., ryusukefujita@}berkeley.edu).

[1]In [20], [21], we have also shown the similar formulation using the framework in [13].

a properly selected control law. We apply our results to design a practical rate control scheme for data transmission over wireless networks, and characterize its performance using NS-2 simulations, and actual experiments over Verizon Wireless 1xRTT and EVDO data networks.

This paper is organized as follows. Section II includes problem formulation. A new approach addressing the problem is proposed in Section III, together with its optimality and stability analysis. The two timescale framework is also presented in Section III. Section IV shows the design of a practical scheme, resulting from the analysis in Section III. NS-2 simulations and actual experiments over 1xRTT and EVDO wireless data networks are included in Section V. Section VI concludes the paper with discussion and future work.

II. Problem Formulation

A. Overview of The Flow Control Framework and Modeling TCP over Wireless

Consider a network with a set J of *resources*, i.e. links, and let C_j be the finite capacity of resource j, for $j \in J$. Let R be the set of routes, where a *route* r is a non-empty subset of J associated with a positive round trip delay T_r. Set $a_{jr} = 1$ if $j \in r$, and set $a_{jr} = 0$ otherwise. This defines a 0-1 routing matrix $A = (a_{jr}, j \in J, r \in R)$, indicating the connectivity of the network.

Associate a route r with a user, i.e. a pair of sender and receiver, and assume users behave independently; furthermore, endow a user with a sending rate $x_r \geq 0$ and a utility function $U_r(x_r)$, which is assumed to be increasing, strictly concave, and continuously differentiable. For convenience, define x to be a vector of users' sending rates, i.e. $x = [x_1, x_2, \ldots]^T$.

Assume utilities are additive, so that aggregate utility of the entire system is $\sum_{r\in R} U_r(x_r)$. The flow control problem under a deterministic fluid model, first introduced by Kelly et. al. [13] and later refined in [15], is a concave optimization problem maximizing the net utility, i.e. the difference between sum utility and cost of using the links[2]:

$$\max_{x\geq 0} \sum_{r\in R} U_r(x_r) - \sum_{j\in J} \int_0^{\sum_{s:j\in s} x_s} p_j(z)\, dz, \qquad (1)$$

where $\int_0^{\sum_{s:j\in s} x_s} p_j(z)\, dz$ can be viewed as cost incurred at link j; $p_j(z)$ is called the price function and is required to be non-negative, continuous, increasing, and not identically zero. With these assumptions on $p_j(z)$, the objective function in (1) is strictly concave. One common price function used in practice is packet loss rate, which is zero if there is no congestion, and concavely increases otherwise:

$$p_j(z) = \frac{(z - C_j)^+}{z} = \begin{cases} 0, & z \leq C_j; \\ 1 - \frac{C_j}{z}, & z > C_j. \end{cases}, \qquad (2)$$

[2]It is easy to show that this is nothing but a penalty relaxation solving the sum utility maximization with capacity constrains, namely:

$$\begin{aligned} \max_{x\geq 0} \quad & \sum_{r\in R} U_r(x_r) \\ \text{s.t.} \quad & Ax \leq C, \text{ (i.e. } \textstyle\sum_{s:j\in s} x_s \leq C_j, j \in J) \end{aligned}$$

where C_j is the capacity of link j. For ease of use in the remainder of the paper, define the aggregate rate arriving at link j as follows:

$$y_j(t) = \sum_{s:j\in s} x_s(t), \quad j \in J.$$

In the rest of this paper, we assume $p_j(y_j(t))$ is small enough such that the end-to-end packet loss rate for user r, i.e. $1 - \prod_{j\in r}(1 - p_j(y_j(t)))$, is approximately $\sum_{j\in r} p_j(y_j(t))$.

Under these settings, Kelly [15] has shown TCP Reno [3] to be a primal-like algorithm:

$$\dot{x}_r(t) = \frac{1}{2S}\left(\frac{2S^2}{T_r^2} - x_r^2(t)\sum_{j\in r} p_j(y_j(t))\right), \quad r \in R, \quad (3)$$

that solves the optimization problem in (1) with $U_r(x_r)$ being $-\frac{2S^2}{T_r^2 x_r}$ where S is TCP packet size, T_r is end-to-end round trip time, and $p_j(y)$ takes the form in (2). Rewriting (1) with these quantities, the net utility maximization problem becomes:

$$\max_x \left\{ -\sum_{r\in R} \frac{2S^2}{T_r^2 x_r} - \sum_{j\in J} \int_0^{\sum_{s:j\in s} x_s} \frac{(z - C_j)^+}{z}\, dz \right\}, \quad (4)$$

As such, TCP can be viewed as a discrete version of the gradient descent algorithm for the above problem.

Kelly [15] has shown that the system in (3) has a unique equilibrium, to which all trajectories converge:

$$x_r^o = \frac{\sqrt{2}S}{T_r\sqrt{\sum_{j\in r} p_j\left(y_j^o\right)}}, \quad r \in R; \qquad (5)$$

where $y_j^o = \sum_{s:j\in s} x_s^o$. Equation (5) is similar to the well known TCP steady state throughput equation describing by Mahdavi and Floyd in [22]. This equilibrium is also the solution for the optimization problem in (1), associated with the following desirable properties: firstly, all routes are fully utilized, yet there is no congestion collapse, i.e. $\forall r \in R, \exists j \in r$, s.t. $C_j \leq y_j^o < \infty$; secondly, there is roughly α-fairness [23] among users with $\alpha = 2$, stating the users are allocated rates that roughly maximize a sum of utility of the form $x_r^{1-\alpha}/(1-\alpha) = -x_r^{-1}$.

B. TCP and TFRC over Wireless

For wireless networks, we assume links $j \in J$ are associated with not only a fixed capacity but also a packet loss rate caused by physical channel errors, necessarily nonnegative, denoted by $\epsilon_j \geq 0, j \in J$. Consequently, the packet loss rate associated with link j, denoted by $q_j(y_j(t))$, is the sum of ϵ_j and $p_j(y_j(t))$, under the assumption that ϵ_j is small:

$$q_j(y_j(t)) = p_j\left(y_j(t)\right) + \epsilon_j \geq \epsilon_j, \quad j \in J. \qquad (6)$$

When the link is not congested, $q_j(y_j(t)) = \epsilon_j$ since all packet loss is caused by channel error; otherwise, $q_j(y_j(t))$

[3]In the rest of the paper, we use this version of TCP.

gradually increases. With this new price in wireless networks, TCP adjusts $x_r(t)$ as follow:

$$\dot{x}_r(t) = \frac{1}{2}\left(\frac{2S^2}{T_r^2} - x_r^2(t)\sum_{j\in r} q_j(y_j(t))\right), \quad r \in R. \quad (7)$$

Following a similar analysis in [15], one can show all trajectories of the system (7)-(6) still converge, but to a different equilibrium:

$$\bar{x}_r = \frac{\sqrt{2}S}{T_r\sqrt{\sum_{j\in r} q_j\left(\bar{y}_j\right)}} \le \frac{\sqrt{2}S}{T_r\sqrt{\sum_{j\in r}\epsilon_j}}, \quad r \in R, \quad (8)$$

where $\bar{y}_j = \sum_{s:j\in s}\bar{x}_s$. The equilibrium rate $\bar{x} \triangleq [\bar{x}_r, r \in R]$ is strictly less than x^o, and solves the same optimization problem as in (4) but with a different cost function $q_j(z)$. Moreover, user r could suffer underutilization if $\sum_{j\in r}\epsilon_j$ is sufficiently large. For instance, in a network with one user, one bottleneck, and sufficiently large buffers at routers, underutilization happens if and only if $\frac{\sqrt{2}S}{T_r\sqrt{\epsilon}} < C$, where C is the bottleneck bandwidth, and ϵ is the aggregate packet loss rate caused by wireless channel error experienced by the user [3].

To address the underutilization problem of TCP over wireless, one straightforward approach is to provide user r only $\sum_{j\in r} p_j(y_j(t))$, i.e. the packet loss rate caused by congestion, and apply it to the control law of $x_r(t)$. This could be done by either end-to-end estimation with or without cross layer information, or by hiding the wireless loss from users via local retransmissions. By doing so, the new problem, flow control in wireless networks, can be transferred into the old problem, flow control in wireline networks, and known approaches like TCP and TFRC will just work.

In fact, most existing approaches [4]–[7], [24]–[36] follow such philosophy. They typically require modifications either to the transport protocols or to the network infrastructure, making them hard to deploy in practice. Moreover, the effectiveness of estimating $\sum_{j\in r} p_j(y_j(t))$ based on end-to-end statistics depends heavily in the scenarios under consideration [7].

III. Proposed Scheme

A. A New Class of Schemes

Intuitively, TCP operates in two regions. In *capacity-limited* region, e.g. in TCP over wireline networks, TCP keeps increasing its rate until the rate reaches wireline link capacity, and packet loss due to router queue overflow is observed. In this case, TCP's rate is limited by the link capacity, and hence it achieves full utilization, as shown in Fig. 1(a).

In *channel-error-limited* region, e.g. in TCP over wireless networks, TCP halves its rate when packet loss caused by wireless channel error is observed. This might happen far before TCP's rate reaches the link's capacity. In this situation, TCP's rate is limited by wireless channel error, and hence it suffers from underutilization, as shown in Fig. 1(b).

Building on the insights from our previous approach MULTFRC [3], we conjecture that whenever TCP suffers underutilization in channel-error-limited region, opening multiple connections can boost up its aggregate rate, and shift TCP from channel-error-limited region to capacity-limited region, and potentially achieve full utilization.

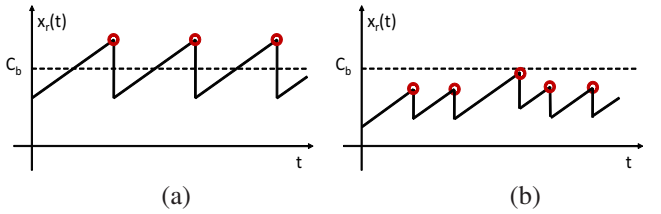

Fig. 1. Conceptual plot of evolution of TCP sending rate: (a) limited by link capacity in wireline scenario; (b) limited by wireless channel error in wireless scenario.

Motivated by this insight, we propose a new approach to flow control by adjusting the number of connections of user r, denoted by $n_r(t)$. The goals of our approach are three fold: first, to stably pursue full utilizations of the bottleneck links; second, to achieve fairness among users; third, to require no modifications to underlying protocol layers and infrastructures.

Specifically, our proposed approach is to dynamically adjust both $x_r(t)$ and $n_r(t)$ as follows:

$$\dot{x}_r(t) = \frac{1}{2Sn_r(t)}\left(\frac{2S^2n_r^2(t)}{T_r^2} - x_r^2(t)\sum_{j\in r} q_j(y_j(t))\right), r \in R \quad (9)$$

$$\dot{n}_r(t) = c_r\left(\frac{1}{n_r(t)} - n_r(t)I_r[\sum_{j\in r} p_j(y_j(t))]\right), \quad r \in R \quad (10)$$

where $c_r, r \in R$ are nonnegative constants, and $I_r[\sum_{j\in r} p_j(y_j(t))]$ is an indicator function implying the congestion status of route r:

$$I_r[\sum_{j\in r} p_j(y_j(t))] = I\left(\sum_{j\in r}\frac{(y_j(t)-C_j)^+}{y_j(t)}\right) \quad (11)$$
$$= \begin{cases} 1, & \text{if route } r \text{ is congested at time } t, \\ & \text{i.e. } \sum_{j\in r}\frac{(y_j(t)-C_j)^+}{y_j(t)} > 0; \\ 0, & \text{otherwise.} \end{cases}$$

The control law of $n_r(t)$ in (10) is Inverse Increase and Multiplicative Decrease (IIMD), which is different from MULTFRC [3]. As we will see later, this modification leads to several desirable properties for the system.

As seen, the dynamics of $x_r(t)$ can be understood as the aggregate dynamics of rates of $n_r(t)$ individual connections, each controlled by TCP. The dynamics of $n_r(t)$ is controlled at the application layer. As such, our approach does not require any modifications to TCP or underlying infrastructure. Moreover, the system pursues full utilization by adjusting $n_r(t)$ according to the conditions in the network. In particular,

- if a route r is underutilized, then $I_r[\sum_{j\in r} p_j(y_j(t))] = 0$; $n_r(t)$ will increase in order to boost up $x_r(t)$, pursuing full utilization on any route r;
- if the route r is fully utilized, i.e. one of its links is congested, then $I_r[\sum_{j\in r} p_j(y_j(t))] = 1$; the proposed scheme will lower $n_r(t)$, and hence $x_r(t)$, to prevent the system from further congestion.

The intuition behind our approach is as follows: when loss rate caused by channel error increases, TCP operates in channel-error-limited region, thus users need to open more connections to improve aggregate throughput, and push TCP towards the capacity-limited region to achieve full utilization. The $I_r(\cdot)$ is the one bit of information required from the end-to-end measurements. In practice, we can estimate $I_r(\cdot)$ by comparing current end-to-end round trip time with the propagation delay to infer existence of queuing delay, and set $I_r(\cdot) = 1$ if queuing delay is detected, and $I_r(\cdot) = 0$ otherwise.

In order to verify that this system actually meets our design goals, we analyze the existence of a unique equilibrium and its stability.

B. Discontinuity Approximation and The Two Time Scale Decomposition

The non-differentiability of functions $I_r[\sum_{j\in r} p_j(y_j(t))]$ and $p_j(y_j(t))$ hinder the analysis of the equilibria. To carry out the analysis, we first approximate these discontinuous functions using continuous functions, in order to generate an approximate continuous system to the original discontinuous system[4]: $\forall j \in J, r \in R$,

$$p_j(y_j(t)) \approx \frac{1}{\beta} \ln\left(1 + e^{\beta \frac{y_j(t) - C_j}{y_j(t)}}\right) \triangleq g_j(y_j(t)), \qquad (12)$$

$$I_r[\sum_{j\in r} p_j(y_j(t))] \approx \frac{e^{\beta \sum_{j\in r} g_j(y_j(t))} - 1}{e^{\beta \sum_{j\in r} g_j(y_j(t))} + 1} \triangleq f(\sum_{j\in r} g_j(y_j(t))), \qquad (13)$$

where β is a nonnegative constant. It should be clear that $f(\sum_{j\in r} g_j(y_j(t)) \to I_r(\sum_{j\in r} g_j(y_j(t))$ and $g_j(y_j(t)) \to p_j(y_j(t))$ as $\beta \to \infty$.

Thus, an approximate continuous version of the original system in (9) and (10) is given by: $\forall r \in R$,

$$\begin{cases} \dot{x}_r(t) = \frac{1}{2Sn_r(t)} \left(\frac{2S^2 n_r^2(t)}{T_r^2} - x_r^2(t) \sum_{j\in r} [\epsilon_j + g_j(y_j(t))] \right), \\ \dot{n}_r(t) = c_r \left(\frac{1}{n_r(t)} - n_r(t) f(\sum_{j\in r} g_j(y_j(t))) \right). \end{cases} \qquad (14)$$

Since the approximate system in (14) is continuous, we can analyze its equilibrium and stability for arbitrary values of β. As $\beta \to \infty$, the system in (14) approaches the original system in (9) and (10). Therefore, the equilibria and the stability results for the approximate system also correspond to the results for the original system in the Filippov sense [37], [38].

The approximate system in (14), though continuous, is still difficult to analyze in general. It is a nonlinear, coupled, multivariate system, and the two equations are not exactly symmetric even though they might appear to be so. As such, we introduce an important two timescale assumption to analyze the approximate system: *The number of connections, $n_r(t)$, changes in a timescale much slower than the source rate, $x_r(t)$.*

This assumption is the key to carrying out the equilibria and stability analysis, as well as to extending the results from TCP to TFRC scenario. It is also very convenient in practical implementation, where the sending rates are expected to change on the order of tens of milliseconds, while the number of connections can be designed to change at a much slower rate, e.g. several or tens of seconds. Consequently, this proposed scheme does not react to instantaneous changes in available bandwidth and users' join/leave dynamics; rather, it targets long-term average performance. As such, this proposed scheme might not be ideal for short-live applications such as web browsing. We also discuss implications of this inter-protocol fairness in Section IV.

Under the above assumption, system (14) fits into the classical singular-perturbation framework [37] [38], and therefore can be decoupled into a fast timescale system and a slow timescale system. The fast timescale system is described by (9) with the corresponding $n_r(t), r \in R$ being constant, namely boundary system [37]: $\forall r \in R$,

$$\begin{cases} \dot{x}_r(t) = \frac{1}{2Sn_r(t)} \left(\frac{2S^2 n_r^2(t)}{T_r^2} - x_r^2(t) \sum_{j\in r} [\epsilon_j + g_j(y_j(t))] \right), \\ n_r(t) = \text{constant}. \end{cases} \qquad (15)$$

The slow timescale system is described by (10) along the manifold defined by the stationary solution of (9), namely reduced system [37]: $\forall r \in R$,

$$\begin{cases} x_r(t) = \frac{n_r(t)\sqrt{2}S}{T_r \sqrt{\sum_{j\in r} [\epsilon_j + g_j(y_j(t))]}}, \\ \dot{n}_r(t) = c_r \left(\frac{1}{n_r(t)} - n_r(t) f(\sum_{j\in r} g_j(y_j(t))) \right). \end{cases} \qquad (16)$$

Under the two timescale assumption, dynamics of the system can be described as follows. On the fast timescale, $n(t)$ can be thought of as being constant since it changes very slowly. The entire system can then be expressed as a boundary system shown in (15). The boundary system is similar to Kelly et. al.'s control system on wireline networks [15] except for the constant $n(t)$, and the price function $p_j(y_j(t))$ being replaced by $\epsilon_j + g_j(y(t)$. Following Kelly's analysis, $x(t) \triangleq [x_r(t), r \in R]$ can be shown to globally exponentially converge to the below equilibrium manifold [15], [17]:

$$x_r(t) = \frac{n_r(t)\sqrt{2}S}{T_r \sqrt{\sum_{j\in r} [g_j(y_j(t)) + \epsilon_j]}}, \quad r \in R. \qquad (17)$$

It can be easily shown that there is a one-to-one mapping between $x(t)$ and $n(t) \triangleq [n_r(t), r \in R]$.

On the slow timescale, $x(t)$ has already converged to the above equilibrium manifold. The system collapses into the reduced system described in (16), and $n_r(t)$ is designed to pursue the desired equilibrium in this slow timescale.

C. Equilibrium of The System: Existence, Uniqueness, Optimality, and Stability

Given the above scenario, important and challenging questions to answer are whether the system in (14) has any equilibria, and if so how many? Second, are these equilibria stable? These two questions are important in the sense that they determine how dynamics of the system evolves, predicting the system's performance in practice. For instance, if the system

[4]We will discuss the relationship between the approximate system and the original system, and performance of the actual implementation later.

in (14) has no equilibrium, the users' sending rates will not converge, making the system undesirable to implement. The consequence would be even worse if the system diverges, and users' rates become arbitrarily large, leading to serious congestion collapse.

The following theorem answers the first question.

Theorem 1: For arbitrary $\beta > 0$, the approximate system in (14) has a unique equilibrium, denoted by (x^*, n^*) as

$$\begin{aligned} n_r^* &= \frac{1}{\sqrt{f(\sum_{j\in r} g_j(y_j^*))}}, \quad r \in R; \\ x_r^* &= \frac{\sqrt{2}S}{f(\sum_{j\in r} g_j(y_j^*))T_r\sqrt{\sum_{j\in r}\left[g_j\left(y_j^*\right)+\epsilon_j\right]}}, \quad r \in R. \end{aligned} \tag{18}$$

Furthermore, this unique equilibrium also solves the following concave optimization problem

$$\max_{x\geq 0} \sum_{r\in R} U_r(x_r) - \sum_{j\in J}\int_0^{y_j} g_j(z)\, dz, \tag{19}$$

with U_r being concave function:

$$U_r(x_r) = \int_0^{x_r} h_r^{-1}\left(\frac{2S^2}{T_r^2\nu^2}\right) d\nu, \quad r \in R, \tag{20}$$

where h_r^{-1} is the inverse of an monotonically increasing function h_r:

$$h_r(z) \triangleq \left(\sum_{j\in r}\epsilon_j + z\right) f(z) = \left(\sum_{j\in r}\epsilon_j + z\right)\frac{e^{\beta z}-1}{e^{\beta z}+1}, \quad r \in R.$$

Proof: Refer to Appendix A in [39]. ■

Two observations can be made from Theorem 1. First, for large β, bottleneck links for every user are close to full utilization at the equilibrium shown in Equation (18). This is because at the equilibrium, $\dot{n}_r^* = 0, r \in R$; from Equation (14), $f(\sum_{j\in r} g_j(y_j^*)) = 1/(n_r^*)^2 > 0$ for $r \in R$ at the equilibrium. Taking into account Equation (13) for large β, $f(\sum_{j\in r} g_j(y_j^*)) > 0$ indicates $\sum_{j\in r} g_j(y_j^*) > 0$, i.e. the packet loss rate caused by congestion is also strictly positive. This implies that at least for one link j along user r's path, the aggregate rate y_j^* passing through link j is close to or lager than the link capacity C_j; hence, bottlenecks are close to full utilization.

Second, the unique equilibrium for the system in Equation (14) in wireless scenario solves a concave optimization problem similar to the one TCP Reno solves in the wired network. They have the same form as Equation (1) but with different users' utility functions $U_r(x_r)$. The only difference is that the $U_r(x_r)$ in the wired case is only a function of x_r, while in wireless scenario it is also a function of $\sum_{j\in r}\epsilon_j$, i.e. the packet loss rate associated with the route r. In fact, if we let $\beta \to \infty$ and $\epsilon_j = 0, \forall j \in J$, i.e. in the wired network scenario, we have $h_r(z) = z$, and the optimization problem in Equation (19) becomes identical to the TCP optimization problem in Equation (4). In this case, $n^* = 1$, and x^* is exactly the same as x^o, implying TCP optimization problem in the wired network is merely a special case of that in Equation (19).

Given the system in (14) has a unique optimal equilibrium, the following two theorems show the equilibrium is in fact locally and globally exponentially stable.

Theorem 2: For arbitrary $\beta > 0$, under the two timescale assumption, the unique equilibrium of the reduced system in (16) is locally exponentially stable. Combining the known fact that the boundary system is locally exponentially stable, with the same arguments used in [37] and [38] for the stability of singular disturbance nonlinear system, the unique equilibrium of the approximate system in (14), (x^*, n^*), is locally exponentially stable.

Proof: Refer to Appendix B in [39]. ■

Theorem 2 implies that if the number of connections $n(t)$ is initially in a small region around the equilibrium n^*, then the system will converge exponentially fast to the equilibrium.

The natural question that arises is how about when $n(t)$ starts far away from the equilibrium n^*? The following theorem states that if n and x are constrained to a compact set, the system is exponentially stable.

Theorem 3: For arbitrary $\beta > 0$, under the two timescale assumption, the unique equilibrium (x^*, n^*) of the system in (14) is semi globally exponentially stable. That is, the system is exponentially stable if n and x are constrained to a compact set that is large enough.

Proof: Refer to Appendix F in [39]. It has been argued in Appendix C in [39] that the "large enough" requirement is trivial to meet in practice. ■

In practice, $n(t)$ and $x(t)$ take finite bounded values and hence are constrained to a compact set. It is also been argued that this compact set is large enough for practical scenarios in Appendix C in [39]. Theorem 3 is sufficient to claim that $(n(t), x(t))$ converges to the optimal equilibrium exponentially fast given arbitrary practical initial conditions.

A conceptual plot on two timescale dynamics of the system in (14) is shown in Fig. 2, and can be described intuitively as follows: $x_r(t)$ first converges in the fast timescale to the equilibrium manifold as in (17) exponentially fast; then in a slow timescale , $n_r(t)$ and $x_r(t)$ follow the control laws of the reduced system in (16) to converge to the optimal equilibrium along the manifold.

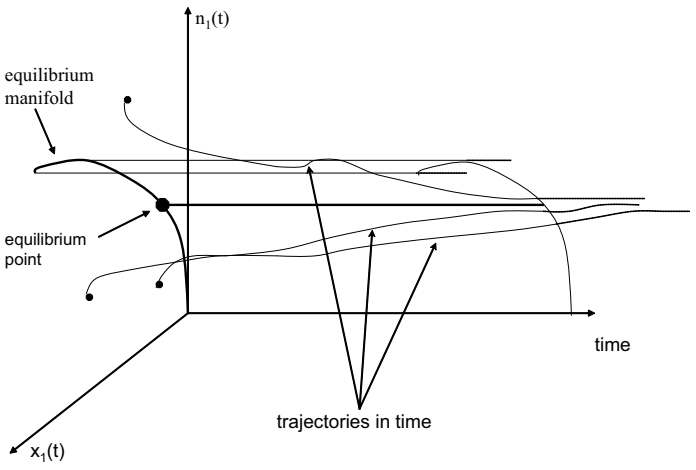

Fig. 2. Time dynamics of a singular perturbation system.

Theorems 1, 2 and 3 state the existence of a unique optimal equilibrium, and ensure its stability for the continuous approximate system in (14), for arbitrary values of β. In

the limit as $\beta \to \infty$, the approximate system approaches the original discontinuous system in (9) and (10). Therefore, for extremely large β, we expect the approximate system to behave quite similarly to the original system, except at the discontinuities $y_j(t) = C_j$. In practice, it is very unlikely for the system to operate at discontinuous points with $y_j = C_j$.

As seen in the next section, in one implementation of the proposed system in (9) and (10), it is necessary to quantize continuous quantities. For instance, controlling $n_r(t)$ is implemented by adjusting the number of connections, which has to be an integer number; controlling $x_r(t)$ is implemented via TCP by adjusting the finite number of packets to be sent out in a given time interval. Therefore, it is very unlikely for the system to operate at discontinuous points with $y_j = C_j$ for some link j. From this point of view, the analysis based on the approximate system is accurate enough to predict and interpret the performance of the actual implementation of the algorithm in practice.

It is worthwhile to point out that the proposed algorithm does not necessarily have to be implemented at application layer - it can be implemented at transport layer as well. This can be accomplished by having one TCP-like protocol mimicking the sending rate of $n_r(t)$ virtual TCP connections. We have already successfully carried out such an implementation for multiple TFRC connections using this idea [39], [40]. In this paper, we choose to implement the proposed algorithm by dynamically adjusting the number of TCP connections, as shown in the next Section, for ease of deployment in practice.

Note that the c_r can be chosen arbitrarily in system (14), as long as the two timescale decomposition holds. Practically, this implies that each user can adjust $n_r(t)$ independently so that a global setting among all users is not necessary. Furthermore, allowing some of the c_r to be equal to zero represents a scenario in which the proposed scheme coexists with TCP. In this situation, all theorems still hold, except for a modification to Theorem 1. More precisely, the utility for users applying TCP should be $U_r(x_r) = 2S^2/(T_r^2 x_r^2) - x_r \sum_{j \in r} \epsilon_j$, rather than the one defined in the theorem. These observations make incremental deployment of our scheme feasible in the current Internet where TCP is dominant. The simulations in Section V partially support this observation.

E-MULTCP maintains the same equilibrium rate as TCP if TCP operates in capacity-limited region, because buffers in the bottlenecks build up, and E-MULTCP observes the queuing delay exceed the threshold. As such, it keeps decreasing the number of connections and ends up with opening only one connection - the minimum number of connections for E-MULTCP. When TCP operates in wireless-error-limited region, E-MULTCP opens multiple TCP connections and has a higher equilibrium rate than TCP. However, this is only to explore the bandwidth other applications based on a single TCP connection cannot utilize. This observation is verified by simulations in Section V. We refer to this as "opportunistic fairness". In either case, a single TCP connection achieves the same long-term throughput as it could have achieved had it coexisted with the same number of TCP connections under the same network setting [39].

Since all of the above analysis and results hold regardless of the values of $\epsilon_j, j \in J$, it is possible to design only one scheme, following (9)-(10), for both wireless and wireline networks, with the latter corresponding to the case where $\epsilon_j = 0, j \in J$.

In summary, the equilibrium x^* of the approximate system in (14), not only solves the optimization problem in (19), but also enjoys several desirable properties: first, all network bottlenecks are close to full utilization for large β, yet the network is free of congestion collapse, for *arbitrary* topology, *arbitrary* wireline and wireless combination, *arbitrary* initial sending rates, and *arbitrary* number of users applying either proposed scheme or TCP; second, users' rates globally exponentially converge to the equilibrium, resulting in no oscillations in stationary state. Hence all of our design objectives are met.

D. A General Two Timescale Framework for Flow Control

In order to converge to a desired equilibrium, our proposed scheme needs two control laws for both the number of connections and sending rate of individual connections. On one hand, sending rate of individual connections is controlled by TCP to converge to an equilibrium manifold exponentially, on a fast timescale. On the other hand, number of TCP connections is controlled to converge to the desired equilibrium exponentially along the above manifold, on a slow timescale. This can be viewed as performing the congestion control in fast timescale, and the utilization control in slow timescale.

An important consequence of this two timescale convergence argument is that, a combination of control law (10) on $n_r(t)$ and any flow control method on $x_r(t)$ resulting in the same equilibrium manifold as TCP Reno, will retain the convergence behavior shown in Theorems 2 and 3. As such we are able to extend all these results to TFRC. This is because TFRC is designed to have the same stationary behavior as TCP Reno [2], although current implementations of TFRC might not achieve this goal in some scenarios [41].

This implies a uniform two timescale framework resulting from our proposed scheme. In this framework, it is sufficient to solve any flow control problem by the following process:

- Based on the desired flow control goals, choose an equilibrium of sending rates that satisfy all goals.
- On a fast timescale, assuming fixed number of connections, design a control law for sending rate of individual connections, to converge to a equilibrium manifold containing the desired equilibrium exponentially fast.
- On a slow timescale, design a control law for the number of connections, to converge to the desired equilibrium exponentially fast, along the equilibrium manifold.

Following the above process, users' sending rates will converge to the desired equilibrium exponentially fast. Theoretically, this framework is an application of the singular perturbation theorem in [38].

Our proposed scheme for problem of flow control in wireless networks, follows this general framework implicitly. On the fast timescale, the proposed scheme applies TCP to control sending rate of individual connections to converge to an equilibrium manifold containing the desired equilibrium. On the slow timescale, the proposed scheme applies IIMD control

law to control the number of TCP connections, in order to converge to the desired equilibrium along the equilibrium manifold. Eventually, the sending rates controlled by the proposed scheme converge to the desired equilibrium exponentially fast, as expected.

By using TCP as the control law for sending rate in fast timescale, and designing new control law for the number of connections in slow timescale, one can potentially address any flow control problem without changing today's transport layer protocol. Further, if the control law for the number of connections utilizes information that today's infrastructure can provide, then the problem is solved without modifying network infrastructure either. In practice, this means our proposed scheme can be deployed without the need for ISP to change their protocol or infrastructure, or for servers to change their operating systems or protocol stack.

One significant advantage of following this two timescale framework is the separation of control laws in two timescales. Control law in each timescale can be designed *independently* of the other one. Hence, it is possible to replace control law in one timescale, without affecting the one in the other, and the general convergence result still holds.

IV. Enhanced MULTCP (E-MULTCP)

In this section, we design a practical scheme for TCP-based data transmission over wireless networks, called E-MULTCP, by following the control law in (10) to adjust the number of TCP connections. Although the analysis in previous section is based on TCP, it can be extended to TFRC, since it has been shown TFRC has the same stationary behavior as TCP [2].

The system framework of our proposed E-MULTCP is shown in Figure 3. As seen, there are two components in the system: RTT Measurer Sub-system (RMS), and Connections Controller Sub-system (CCS).

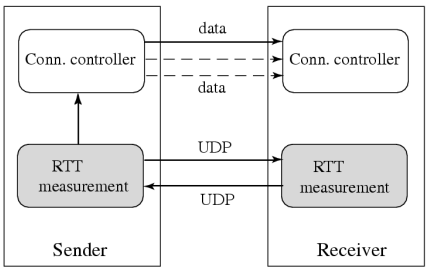

Fig. 3. E-MULTCP system framework.

A. RTT Measurement Sub-system (RMS)

The gray blocks in Fig. 3 represent RMS. They measure round trip time samples between sender and receiver, denoted by rtt_{sample}, by computing difference between the time sender emits a packet, and the time it receives the ACK from receiver. These packets could be sent through the forward and backward UDP connections between sender and receiver. Note these measurement packets can potentially be transmitted through TCP connections as well. Because TCP may retransmit lost measurement packets, RTT estimate based on TCP is likely to be more noisy than that using UDP.

To reduce overhead and to prevent congestion collapse, the rate at which RTT-measurement packets are sent, is set to be the same as that of data packets, which is estimated on sender side by measuring the number of data packets sent in the previous round trip time. RTT measurement packets contain only UDP and IP headers, and a timestamp, a total of 32 bytes. In the case when TCP uses typical packet size of 1500 bytes, the overhead is 2.1% of the total throughput. In 802.11 type of network, these measurement packets might contend with data packets and cause extra access delay to data connections. In this situation, one can lower measuring frequency to reduce the overhead, at a cost of getting more sparse RTT measurements.

After waiting for a fixed interval τ, RMS computes a running average, ave_rtt, of these rtt_{sample}s, and reports it to the CCS. Clearly, $1/\tau$ is the frequency of adjusting the number of connections; τ has to be large enough to ensure the frequency is much lower than that of changing the source rate, which is typically of the order of round trip time.

B. Connection Controller Sub-system (CCS)

The CCS is shown as the white blocks in Figure 3. Its basic functionality is to properly control the number of connections n, following the control law shown in (10). As seen in (10), when route r is underutilized, the indicator function $I_r(\cdot) = 0$, and n_r increases proportional to $1/n_r$. When route r is congested, $I_r(\cdot) = 1$, and n_r will decrease roughly proportional to itself. This indicates CCS at the sender should roughly Inversely Increase and Multiplicatively Decrease (IIMD) the number of connections n, based on route congestion status. Specifically, ave_rtt is reported to CCS by RMS, CCS sets the rtt_min as the minimum over all ave_rtt seen so far, and then adapts the number of connections n as follows:

$$n = \begin{cases} \beta n + \alpha/n, & \text{if } ave\_rtt - rtt\_min > \gamma rtt\_min; \\ n + \alpha/n, & \text{otherwise.} \end{cases} \tag{21}$$

where $\alpha = 1 - \beta < 1$ and γ is a preset parameter. This is nothing but a discrete implementation of the control law in (10), with $c_r = \alpha/\tau$. The value of the indicator function $I_r(\cdot)$ is estimated by comparing the measured queuing delay $ave\_rtt - rtt\_min$ with a dynamic threshold γrtt_min. For a given route, the rtt_min is a constant representing the minimum observed round trip time for that route, approximating physical propagation delay. As such, $ave\_rtt - rtt\_min$ corresponds to current queuing delay, and γrtt_min is a threshold on the queuing delay that E-MULTCP can tolerate before it starts to decrease the number of connections. Therefore, under ideal conditions, E-MULTCP keeps increasing the number of connections to make ave_rtt as close as possible to $(1+\gamma)rtt\_min$ without exceeding it.

C. Discussion

E-MULTCP increases its rate as $\dot{n}(t) = \alpha/(\tau n(t))$ based on (10) whenever $ave\_rtt - rtt\_min > \gamma rtt\_min$. Hence, to increase n from N_2 to N_1, assuming $N_1 > N_2$, it takes

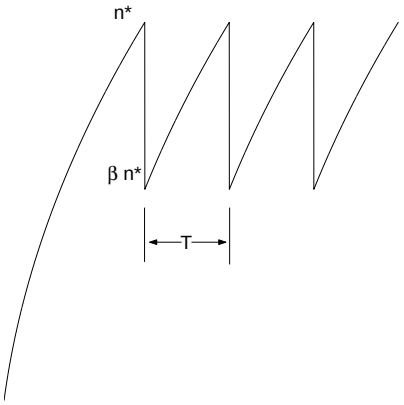

Fig. 4. Demonstration of the change on the number of connections n, controlled by single E-MULTCP over single wireless link.

E-MULTCP about $\frac{\tau}{2\alpha}(N_1^2 - N_2^2)$ seconds. Here, α is defined as the normalized responsiveness, a crucial design parameter representing how responsive TCP is in shifting from channel-error-limited region to capacity-limited region.

In the decreasing stage, E-MULTCP approximately implements $\dot{n}(t) = -\frac{\alpha n(t)}{\tau}$, and hence it takes E-MULTCP $\frac{1}{\alpha}\tau \ln(N_1/N_2)$ seconds to decrease n from N_1 to N_2. Thus, E-MULTCP is conservative in adding connections, but aggressive in closing them.

In a simple topology with one E-MULTCP over one wireless link, we can use the above results to compute bandwidth utilization ratio. At steady state, E-MULTCP periodically increases the number of connections n to the optimal n^* so as to fully utilizes the wireless bandwidth, and proportionally decreases n upon reaching n^*. The approximate continuous version of this process is demonstrated in Fig. 4. In the plot, T is the time for $n(t)$ to increases from βn^* to n^*, and hence is given by $T = (n^*)^2(1-\beta^2)\tau/2\alpha$. The number of connections at time t is given by $n(t) = \sqrt{(t-kT)2\alpha/\tau + (\beta n^*)^2}$ for $kT \le t \le (k+1)T$ and $k \in Z^+$.

Based on the above analysis, the utilization ratio at steady state, is then the ratio between the average number of connections and n^*, as follows:

$$\frac{1}{Tn^*}\int_0^T n(\delta)d\delta = \frac{2}{3}\frac{1+\beta+\beta^2}{1+\beta}. \qquad (22)$$

This ratio is at least 2/3, only depends on β, and is independent of wireless packet loss rate, τ, and n^*. The larger β is, the higher utilization ratio is. There is clearly a trade-off between the utilization ratio and normalized responsiveness $\alpha = 1-\beta$, as shown in Fig. 5. Thus, it is possible to achieve increased responsiveness at the expense of lower utilization ratio and vice versa.

One of our target applications for E-MULTCP is mobile devices on commercial wireless networks. We have empirically found that 2 or 3 parallel connections are sufficient to fully utilize 1xRTT CDMA and EVDO wireless channel, which have the measured available throughput of around 110 kbps and 400 kbps, respectively [3]. As the channel bandwidth is small and the optimal number of connections, n^*, to pursue is small in these scenarios, it is more important to achieve high utilization ratio than being responsiveness to channel conditions. Moreover, since most cell phones are power limited, it would be advantageous to select β in such a way as to make E-MULTCP control procedure computationally efficient.

As such, we have selected $\beta = 0.75$ for E-MULTCP, to obtain a utilization ratio of 0.88, and a normalized responsiveness of 0.25. This means it takes E-MULTCP $2\tau(N_1^2 - N_2^2)$ seconds to increase from N_2 to N_1 connections. The resulting IIMD control law of n requires only binary shift and addition operation, and is therefore computationally efficient.

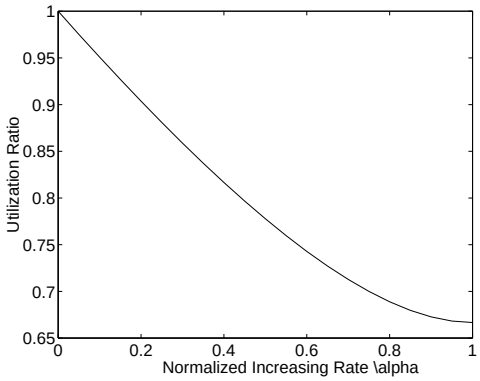

Fig. 5. A trade off between utilization ratio and normalized responsiveness.

When there is a route change either due to change in the wireless base station, or due to route change within the wireline networks, the value of rtt_min changes significantly, affecting the performance of E-MULTCP. Under these conditions, it is conceivable to use route change detection tools such as traceroute [42] at the sender to detect the route change, in order to reset rtt_min to a new value. Detecting route changes in a more elegant and effect way is of practical interests, and is still open. Furthermore, it can be argued that the overall throughput of E-MULTCP does not go to zero, resulting in starvation; this is because E-MULTCP always keeps at least one connection open.

Since the data stream in E-MULTCP is transmitted using multiple connections, the receiver could potentially receive out of order packets. However, reordering application packets from multiple TCP connections using a receiving buffer is a rather mature technology widely used in peer-to-peer applications, such as BitTorrent file sharing, e.g. Kazza [43], and peer-to-peer streaming, e.g. PPlive [44]. We have also implemented and successfully tested a file downloading software on an actual BREW cellular telephone, combining E-MULTCP and a simple packets reordering procedure [45].

On one hand, the number of TCP connections in E-MULTCP decreases slowly when congestion level increases. This can potentially be temporarily unfair to other applications running a single TCP connection during its convergence to the fair equilibrium. On the other hand, E-MULTCP also increases the number of TCP connections conservatively when congestion is released. In such a case, a single TCP connection observes less competition and can use more bandwidth than if E-MULTCP were to adapt in a fast timescale. Therefore, at times of high congestion, E-MULTCP temporarily uses more

bandwidth than the opportunistically fair equilibrium rate, and it temporarily uses less bandwidth at times of low congestion. As such, in long-term, E-MULTCP is fair to other protocols such as single TCP. This observation is similar to the one for slow TCP friendly protocol, e.g. TFRC, in [46]. Even though long-term fairness is not universally acceptable [46], it is generally considered to be sufficient in the networking community [2], [47], [48].

From a more practical perspective, dynamically adjusting the number of connections inherently prefers a slower control timescale as compared to the sending rate of each connection. Opening and closing connections on the same timescale as adjusting sending rate of each connection is expensive and undesirable for practical implementations in end systems, such as resource limited wireless devices.

In summary, in the E-MULTCP system, the number of connections is controlled according to a discrete version of (10) at the sender; the sending rate of each TCP connection is adjusted according to whatever version of TCP that runs at the client. The rate of change of the number of connections is expected to be much slower than that of sending rate, satisfying the two time scale assumption in the previous section. Thus, the optimality and stability analysis in the previous section for the dynamic system in (9) and (10) applies to E-MULTCP, indicating that E-MULTCP results in a stable, yet fully utilized network as shown in (19).

Notice that E-MULTCP is similar to our previously proposed scheme MULTFRC [3] in its design. The differences are two fold. First, E-MULTCP is based on TCP, while MULTFRC is based on TFRC. Second, E-MULTCP applies IIMD as the control law for the number of connections n, while MULTFRC applies Inversely Increase and Additively Decrease (IIAD). It is of course conceivable to design an Enhanced MULTFRC (E-MULTFRC) based on TFRC by using IIMD to control the number of connections [39]. A fundamental and inherent advantage of E-MULTCP and E-MULTFRC over MULTFRC are their provable convergence and stability properties, and faster adaptation to congestion. We have empirically found the performance of E-MULTFRC to be comparable to that of E-MULTCP [39].

V. NS-2 Simulations and 1xRTT/EVDO Wireless Experiments

In this section, we carry out NS-2 [49] simulations and actual experiments over Verizon Wireless 1xRTT and EVDO CDMA data networks to evaluate the performance of E-MULTCP. We use TCP NewReno implementation for TCP protocol in all simulations. We use the same settings in simulations and experiments as our previous work [50], because they well demonstrates effectiveness of proposed scheme.

A. Setup

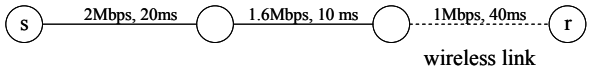

Fig. 6. Simulation topology.

The topology used in simulations is shown in Figure 6. The sender denoted by s, and the receiver denoted by r, both run E-MULTCP. The number of connections, as mentioned in the previous section, is controlled by the sender. For all simulations, the wireless bandwidth B_w is set be 1 Mbps, and is assumed to be the bottleneck. The wireless link is modeled by an exponential error model, and p_w varies from 0.0 to 0.08 in increments of 0.02. DropTail type queue is used for each node. In order to evaluate E-MULTCP's performance in the presence of wireless channel errors, we examine three issues; first, how E-MULTCP performs in terms of average throughput, average round trip time, and packet loss rate, as a function of p_w. Second, whether the number of connections is stable. Third, whether or not a E-MULTCP application can fairly share with an application using one TFRC or one TCP connection. In all the simulations, throughput is measured every second, packet loss rate is measured every 30 seconds, the average round trip time is measured every 100 packets, and the number of connections is sampled whenever there is a change.

For the actual experiments over 1xRTT, we stream from a desktop connected to Internet via 100 Mbps Ethernet in eecs.berkeley.edu domain, to a notebook connected to Internet via Verizon Wireless 1xRTT CDMA data network. Thus it is quite likely that the last 1xRTT CDMA link is the bottleneck for the connection. The connection is live for 30 minutes. As we cannot control p_w in actual experiments, we measure the average throughput, the average number of connections, and packet loss rate.

B. Performance Characterization of E-MULTCP

Following the discussions in Section IV-C, we use the following parameters for E-MULTCP in our simulations and experiments in order to achieve a good balance between utilization ratio and normalized responsiveness: $\alpha = 0.25, \beta = 0.75$, $\gamma = 0.2$, and $\tau = 20$ seconds. Intuitively, larger τ results in a more reliable estimate of round trip time, but at the expense of a lower sampling frequency, resulting in a less responsive system.

We use NS-2 to simulate the E-MULTCP system to send data for 9000 seconds, compute the average throughput and packet loss rate for p_w =0.0, 0.02, 0.04, 0.06 and 0.08[5], and compare them to the optimal, i.e. $B_w(1 - p_w)$ for each p_w. The results for $B_w = 1\ Mbps$ and $RTT_{min} = 168\ ms$ are shown in Figure 7. As seen, the throughput is within 25% of the optimal, and is reasonably close to the predicted one, i.e. the product of the optimal and the utilization ratio computed using (22). The round trip time is within 20% of RTT_{min}, and the packet loss rate is almost identical to the optimal, i.e. a line of slope one as a function of wireless channel error rate. As expected, the average number of connections increases with wireless channel error rate, p_w. [6]

[5]It is well known that TCP's degraded performance is dominated by timeout if wireless packet loss rate p_w is higher than 0.1. As our model does not capture the timeout effect, we are more interested in cases where $p_w < 0.1$.

[6]Note the round trip time for $p_w = 0$ is not shown in Figure 7 because it represents the case when the channel is error free. In this case, E-MULTCP reduces to one TCP connection.

Considering the throughput plot in Figure 7, for some values of p_w, there is a significant gap between the actual and optimal throughput. This is due to the quantization effect in situations where the number of connections is small, i.e. 2 to 4. In these situations, a small oscillation around the optimal number of connections results in a large variation in the observed throughput. One way to alleviate this problem is to increase γ in order to tolerate larger queuing delay and hence absorb throughput fluctuations, at the expense of being less responsive. Another alternative is to use a smaller packet size in order to reduce the "quantization effect" at the expense of (a) larger overhead and hence lower transmission efficiency, and (b) the slower rate of convergence to the optimal number of connections. In the case of TFRC, we have shown that it is possible to combine multiple TFRC connections into one connection, in order to mitigate the quantization effect [40]. It is also possible to combine multiple TCP connections into one UDP connection whose rate emulates the aggregate rate of multiple TCP connections. However, as UDP is fundamentally an unreliable protocol, in this case, the sender and receiver have to apply erasure codes or carry out retransmissions to recover lost packets, if the goal is end to end reliable communication.

One might also notice that the gap between E-MULTCP's throughput and the optimal becomes larger as p_w increases. This is due to increased timeout events as p_w increases. Upon timeout, TCP slow starts again, generating bursty traffic that results in transient queuing delay. Consequently, detecting congestion based on RTT estimate becomes less reliable, decreasing the average throughput. This effect is more pronounced for large p_w and number of connections.

In order to examine E-MULTCP's performance as a function of p_w, as well as the dynamics of E-MULTCP, we use E-MULTCP with p_w initially set at 0.02. Then at 3000^{th} second, p_w is switched to 0.06, and at 6000^{th} second switched back to 0.02. Here, we artificially change p_w to see how E-MULTCP adapts to the change in p_w. The throughput, packet loss rate, round trip time, and the number of opened connections are shown in Figure 8. As seen, the number of connections varies from around 2-3 to around 5 as p_w switches from 0.02 to 0.06. The ramp up and ramp down times from 2 to 5, and 5 to 2 connections are about 660 and 80 seconds, respectively. These are very close to the theoretical values obtained from Section IV-C, i.e. 650 and 73 seconds respectively.

To show E-MULTCP can adapt faster with smaller τ, we repeat the above simulations with $\tau = 5$ seconds instead of 20 seconds. The throughput, packet loss rate, round trip time, and the number of opened connections are shown in Figure 9. As seen, now the ramp up and ramp down times are now about 184 and 16 seconds, respectively. These are again very close to the theoretical values 190 and 19 seconds respectively. As such, E-MULTCP's adaptation rate can be adjusted by varying τ. In practice, since the adaptation is done by opening and closing TCP connections, adaptations might be expensive for end hosts, especially the resource-limited wireless handheld devices.

Similar experiments are carried out on Verizon Wireless 1xRTT CDMA data network. The 1xRTT CDMA data network

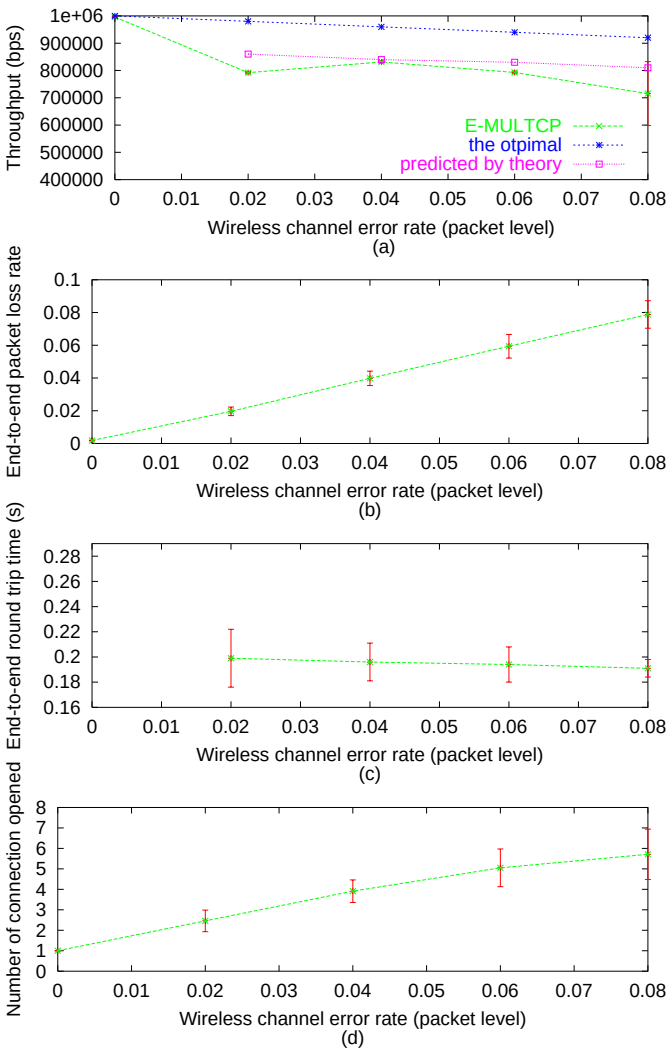

Fig. 7. NS-2 simulations for $B_w = 1\ Mbps$ and $RTT_{min} = 168\ ms$; (a) throughput, (b) end-to-end packet loss rate, (c) end-to-end round trip time, (d) number of connections, all as a function of packet level wireless channel error rate.

is advertised to operate at data speeds of up to 144 kbps for one user. As we explore the available bandwidth for one user using UDP flooding, we find the highest average available bandwidth averaged over 30 minutes to be between 80 kbps to 97 kbps. In our experiments, we stream for 30 minutes from a desktop on wireline network in eecs.berkeley.edu domain to a laptop connected via 1xRTT CDMA modem using E-MULTCP, E-MULTFRC and TCP. The results are shown in Table I for packet size of 1460 bytes. As seen, on average E-MULTCP opens up 1.7 connections, and results in 58% higher throughput, at the expense of a larger round trip time, and higher packet loss rate. Similar observations can be made for E-MULTFRC [39].

We have also carried out experiments over Verizon Wireless EVDO data network, by sending 5 MB files from a desktop on wireline network in eecs.berkeley.edu domain to an EVDO cell phone using E-MULTCP and TCP. A file size of 5 MB is chosen to be typical of a MP3 song. The results are shown in

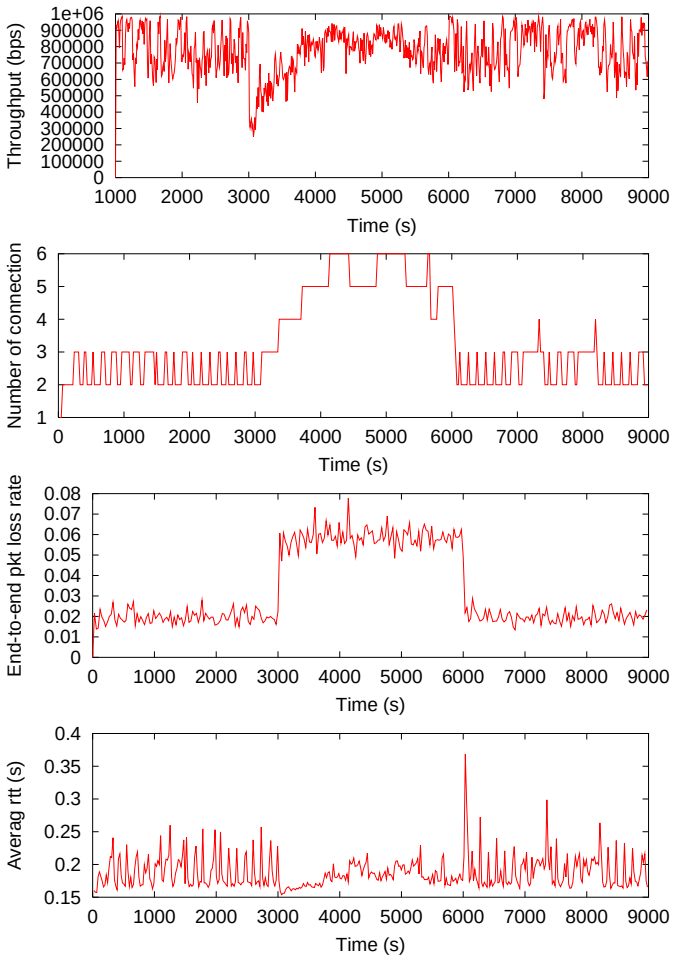

Fig. 8. NS-2 simulation results as p_w changes from 0.02 to 0.06 and back again with $\tau = 20$ seconds; (a) end-to-end round trip time, (b) throughput, (c) number of connections, (d) end-to-end packet loss rate, all as a function of time.

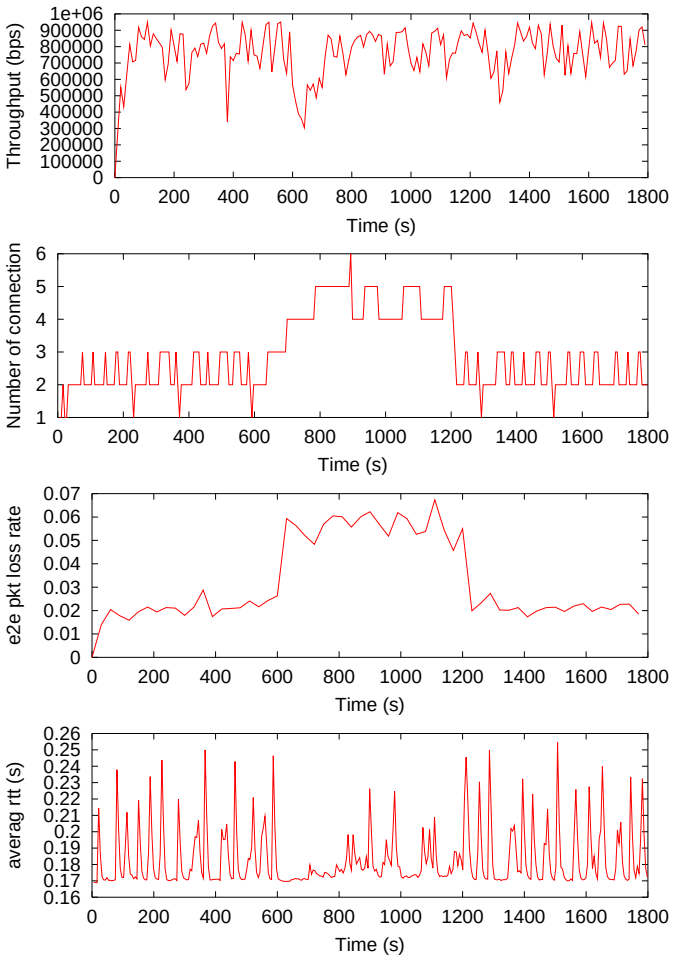

Fig. 9. NS-2 simulation results as p_w changes from 0.02 to 0.06 and back again with $\tau = 5$ seconds; (a) end-to-end round trip time, (b) throughput, (c) number of connections, (d) end-to-end packet loss rate, all as a function of time.

TABLE I
ACTUAL EXPERIMENTAL RESULTS FOR E-MULTCP, E-MULTFRC AND TCP OVER COMMERCIAL 1XRTT CDMA DATA NETWORK.

scheme	throughput (kbps)	rtt (ms)	ave. # of conn.
one TCP	59	1954	N/A
E-MULTCP	93	2447	1.7
E-MULTFRC	89	2767	1.9

Table II for packet size of 1460 bytes. As seen, on average E-MULTCP opens up 1.8 connections, and results in 43% higher throughput, at the expense of a larger round trip time, and higher packet loss rate.

TABLE II
ACTUAL EXPERIMENTAL RESULTS FOR E-MULTCP AND TCP OVER COMMERCIAL EVDO DATA NETWORKS.

scheme	throughput (kbps)	rtt (ms)	ave. # of conn.
one TCP	249	702	N/A
E-MULTCP	355	946	1.8

C. Fairness between E-MULTCP and TCP

To investigate the fairness of E-MULTCP, we carry out NS-2 simulations based on the "dumbbell" topology shown in Fig. 10. Senders are denoted by $si, i = 1, \ldots, 16$, and receivers are denoted by $di, i = 1 \ldots, 16$. We investigate two types of fairness: the inter-protocol fairness between E-MULTCP and TCP, and the intra-protocol fairness within E-MULTCP.

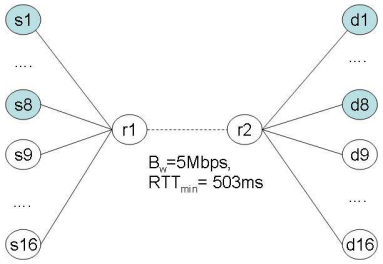

Fig. 10. The simulation topology for E-MULTCP's fairness evaluation.

The intra-protocol fairness is defined as the fairness between E-MULTCP flows. In our NS-2 simulations, we run E-MULTCP on all 16 sender-receiver pairs shown in Fig. 10

for 5000 seconds, and compare their throughput. E-MULTCP is said to be intra-protocol fair if all receivers achieve the same throughput. The fairness ratios for $p_w = 0.01$ and $p_w = 0.04$ are shown in Table III. The fairness ratio is defined as receivers' throughput divided by the average throughput; the closer to one, the more fair the E-MULTCP system is. As seen, the fairness ratio is fairly close to one, indicating E-MULTCP flows are fair to each other, at least in this simulation setting. The bandwidth utilization ratios are 96% for $p_w = 0.01$ and 98% for $p_w = 0.04$.

TABLE III
SIMULATION RESULTS FOR INTRA-PROTOCOL FAIRNESS OF E-MULTCP.

receiver	fairness ratio p_w=0.01	fairness ratio p_w=0.04	receiver	fairness ratio p_w=0.01	fairness ratio p_w=0.04
d1	1.06	1.03	d9	1.07	0.98
d2	1.00	1.01	d10	0.97	0.98
d3	1.06	1.03	d11	1.02	1.02
d4	1.02	1.08	d12	1.02	0.99
d5	0.90	0.98	d13	0.98	0.97
d6	0.93	0.98	d14	0.89	1.00
d7	1.02	1.04	d15	1.01	1.01
d8	0.99	1.01	d16	0.98	0.94

The inter-protocol fairness is defined as the fairness between E-MULTCP and TCP. In our simulations, we run E-MULTCP on the first 8 sender-receiver pairs, i.e. $(si, di), i = 1, \ldots, 8$, and TCP on the remaining 8 sender-receiver pairs shown in Fig. 10; each session lasts 5000 seconds, and we compare their throughput for $p_w = 0.01$ and $p_w = 0.02$. Under the simulation settings, we would expect each E-MULTCP consumes more bandwidth than one TCP under full utilization. This is because the wireless channel error rate is large enough to make the number of virtual connections of each E-MULTCP to be larger than one. Hence, it is meaningless to define the fairness between E-MULTCP and TCP as having the same throughput.[7] As such, in our simulations, we define E-MULTCP to be fair to TCP if it does not result in a significant decrease in TCP's throughput as compared to TCP throughput in the absence of E-MULTCP. Specifically for our simulations, it implies TCP retains the same throughput whether or not it coexists with E-MULTCP under the same network setting. The throughput of E-MULTCP and TCP, as well as the total bandwidth utilization ratios for the setup shown in Fig. 10, are shown in Table IV for two scenarios: (a) 8 E-MULTCP coexisting with 8 TCP connections, (b) 16 TCP connections. Comparing E-MULTCP+TCP with TCP-alone in Table IV, we see the former achieves a much higher utilization of the wireless bandwidth at the expense of slightly lower TCP throughput. A careful examination of the traces reveals that this throughput drop is caused by the higher RTT experienced by TCP connections in the E-MULTCP+TCP scenario as compared with TCP-alone scenario. For example, for $p_w = 0.01$ and $\gamma = 0.2$, The RTT for E-MULTCP+TCP is around 0.56 seconds, while for TCP-alone is 0.5 seconds, i.e. the propagation delay[8]. As TCP's throughput is known to be inversely proportional to RTT, the 12% increase in the RTT of TCP connections roughly explains the 11% decrease in the TCP's throughput shown in first row in Table IV.

TABLE IV
SIMULATION RESULTS FOR FAIRNESS BETWEEN E-MULTCP AND TCP.

settings	8 E-MULTCP + 8 TCP			16 TCP	
	ave. thput. (E-MULTCP) (kbps)	ave. thput. (TCP) (kbps)	utilization (%)	ave. thput. (TCP) (kbps)	utilization (%)
p_w=0.01 γ=0.2	432.1	160.3	96	179.8	58
p_w=0.01 γ=0.1	402.3	170.0	93	179.8	58
p_w=0.02 γ=0.2	468.8	107.8	94	120.4	40
p_w=0.02 γ=0.1	446.1	114.3	92	120.4	40

This increase in the RTT experienced by TCP is, by design, a consequence of E-MULTCP controlling n according to (21). As n is only decreased after the queuing delay exceeds the threshold γrtt_min, round trip time is increased when E-MULTCP increases n to achieve full utilization. One way to address this problem is to use a smaller value for γ, in order to reduce the increase in the RTT, and hence minimize the TCP's throughput drop. However, smaller values of γ also result in lower bandwidth utilization due to increased sensitivity of E-MULTCP to RTT measurements. As shown in Table IV, decreasing γ from 0.2 to 0.1 results in both a smaller drop in the TCP's throughput and lower utilization. Regardless, the stability of the network with mixed TCP and E-MULTCP is guaranteed by Theorem 3.

VI. DISCUSSION AND FUTURE WORK

Our proposed scheme is different from existing schemes such as TCP-Vegas [10], TCP-FAST [11], and VTP [48] that use delay or round trip time variation to infer congestion in several ways: first both TCP-Vegas and VTP are transport layer schemes, and as such could potentially solve the wireless flow control problem only if they were deployed universally by every single computer, cell phone, PDA, and handheld device. However, since there are many flavors of TCP deployed today on a heterogeneous set of devices, requiring all the users to deploy a particular flavor of TCP is impractical, if not impossible. As such, one advantage of our proposed application layer scheme is that it requires absolutely no modifications to the transport layer, nor to the operating systems of the end devices as it can work with all flavors of TCP. It also does not require any changes to the network infrastructure such as routers.

Furthermore, TCP-Vegas and TCP-FAST need to measure precise queueing delay every RTT, which can be as small as several milliseconds. It is difficult to measure delay accurately and reliably in such a short interval, because the large variation in packet processing time at both routers and end-hosts may

[7]Obviously, there are situations in which E-MULTCP ends up with performing similar to one TCP. An example would be E-MULTCP competing for bandwidth with TCP on wireline networks. In that case, however, the fairness between E-MULTCP and TCP is reduced to the fairness between TCP connections themselves, and has been well explored in literature.

[8]In this NS simulation, TCP and E-MULTCP share the same route and hence both have the same round trip time.

dominate the actual value of queuing delay. Consequently, delay-based schemes such as TCP-Vegas can suffer underutilization [51]. In contrast, our proposed scheme is based on only one bit of information to detect existence of queuing delay once in a while, e.g. once every 20 seconds. As such, simple filtering techniques such as averaging can be applied to filter out noise in delay measurements and to obtain the one-bit information reliably.

Even though C_j and ϵ_j are assumed to be constant in our analysis, in some networks such as wireless Local Area Networks (WLAN) and CDMA networks, C_j and ϵ_j might be time varying or even change in a correlated fashion. E-MULTCP adjusts the number of connections in a timescale much slower than that of each connection's sending rate, in order to satisfy the two timescale assumption, as well as to reduce the complexity and overhead of opening/closing connections. As a result, in a scenario where C_j and ϵ_j are time varying, by design, E-MULTCP could not react instantaneously. Hence, it is interesting to quantify how fast E-MULTCP can adapt to the changes in C_j and ϵ_j. The basic insight we gain in this work is that opening multiple connections helps to improve the link utilization when TCP operates in channel-error-limited region. In practice, our experimental results in this paper have verified that our proposed scheme works in an actual CDMA network, where the C_j and ϵ_j are typically not constant. In [52], Chen et. al. studied the flow control problem in wireless ad-hoc networks, where capacities of adjacent links are correlated, but with no wireless error in consideration. Hence, an interesting question is how to perform flow control in wireless ad-hoc networks with wireless error, by combining the ideas in this paper with those in [52].

Currently, E-MULTCP relies on accurate detection of queuing delay along the route, which in turn requires knowledge of the propagation delay. In situations where propagation delay is highly volatile, such as in 3G wireless networks that deploy opportunistic scheduling, E-MULTCP's performance could potentially degrade due to inaccurate estimation of queuing delay. As discussed in the beginning of Section V-B, even though it is possible to choose a large γ to tolerate inaccuracy in estimating queuing delay, it would be desirable to develop a more reliable way to estimate the congestion status of a route, i.e. the one bit of information needed by E-MULTCP.

Our proposed control law for the number of connections currently utilizes only one bit of delay information. The motivation behind this has been robustness to noise and measurement errors. Yet, it might be worthwhile to utilize more precise delay information, and design alternative control laws to achieve a smoother behavior.

There are many other interesting and important directions for future research. Stability in the presence of delay and noisy disturbances are of great interest, from both control and networking points of view. It is also interesting to investigate the fairness properties of the equilibrium, in various scenarios combining wireline and wireless networks.

Our framework captures the impact of wireless channel packet loss on TCP and TFRC performance over wireless. However, it does not model the timeout effect of TCP and TFRC over wireless caused by link layer local retransmissions or heavy wireless loss, potentially degrading the performance. It is desirable to extend our framework in order to take into account the effect of timeout in wireless networks.

ACKNOWLEDGEMENT

The authors would like to express their thanks to Alessandro Abate and Shankar Sastry for helpful discussions and help in preparing the paper.

REFERENCES

[1] V. Jacobson, "Congestion avoidance and control," in *Proc. ACM SIGCOMM*, Stanford, CA, Aug. 1988, pp. 314–329.

[2] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-based congestion control for unicast applications," in *Proc. ACM SIGCOMM*, Stockholm, Sweden, Aug. 2000, pp. 43–56.

[3] M. Chen and A. Zakhor, "Rate control for streaming video over wireless," in *Proc. IEEE INFOCOM*, Hongkong, China, Mar. 2004.

[4] H. Balakrishnan, V. Padmanabhan, S. Seshan, and R. Katz, "A comparison of mechanisms for improving tcp performance over wireless links," *IEEE/ACM Trans. Netw.*, vol. 5, no. 6, pp. 756–769, 1997.

[5] H. Balakrishnan and R. Katz, "Explicit loss notification and wireless web performance," in *Proc. of IEEE Globecom Internet Mini-Conference*, Nov. 1998.

[6] N. Samaraweera, "Non-congestion packet loss detection for tcp error recovery using wireless links," *IEE Proceedings of Communications*, vol. 146, no. 4, p. 222C230, Aug. 1999.

[7] S. Cen, P. Cosman, and G. Voelker, "End-to-end differentiation of congestion and wireless losses," *IEEE/ACM Trans. Netw.*, vol. 11, no. 5, pp. 703–717, 2003.

[8] G. Yang, M. Gerla, and M. Y. Sanadidi, "Adaptive video streaming in presence of wireless errors," in *Proc. ACM MMNS*, San Diego, USA, Jan. 2004.

[9] F. Yang, Q. Zhang, W. Zhu, and Y.-Q. Zhang, "End-to-end TCP-Friendly streaming protocol and bit allocation for scalable video over mobile wireless internet," in *Proc. IEEE INFOCOM*, Hongkong, China, Mar. 2004.

[10] L. S. Brakmo and L. L. Peterson, "TCP Vegas: end-to-end congestion avoidance on a global internet," *IEEE J. Sel. Areas Commun.*, vol. 13, no. 8, pp. 1465–1480, Oct. 1995.

[11] D. X. Wei, C. Jin, S. H. Low, and S. Hegde, "FAST TCP: motivation, architecture, algorithms, performance," *IEEE/ACM Trans. Netw.*, vol. 14, no. 6, pp. 1246–1259, Dec. 2006.

[12] S. Floyd, "TCP and explicit congestion notification," *ACM Computer Communication Review*, vol. 24, pp. 10–23, Oct. 1994.

[13] F. P. Kelly, A. Maulloo, and D. Tan, "Rate control for communication networks: shadow prices, proportional fairness, and stability," *Journal of the Operationl Research Society*, vol. 49, pp. 237–252, 1998.

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

[15] F. P. Kelly, "Fairness and stability of end-to-end congestion control," *European Journal of Control*, vol. 9, pp. 159–176, 2003.

[16] F. Paganini, Z. Wang, J. Doyle, and S. Low, "A new TCP/AQM for stable operation in fast networks," in *Proc. IEEE INFOCOM*, San Francisco, CA, Mar. 2003.

[17] S. Kunniyur and R. Srikant, "A time scale decomposition approach to adaptive ecn marking," *IEEE Trans. Autom. Control*, vol. 47, no. 6, pp. 882–894, Jun. 2002.

[18] G. Vinnicombe, "On the stability of end-to-end congestion control for the internet," University of Cambridge, Cambridge, UK, Tech. Rep. CUED/F-INFENG/TR.398, 2001.

[19] F. Paganini, J. Doyle, and S. Low, "Scalable laws for stable network congestion control," in *Proc. IEEE CDC*, Dec. 2001.

[20] M. Chen, A. Abate, and S. Sastry, "New congestion control schemes over wireless networks: Stability analysis," in *Proceedings of the* 16^{th} *IFAC World Congress*, Prague, 2005.

[21] A. Abate, M. Chen, and S. Sastry, "New congestion control schemes over wireless networks: Delay sensitivity analysis and simulations," in *Proceedings of the* 16^{th} *IFAC World Congress*, Prague, 2005.

[22] J. Mahdavi and S. Floyd. (1997, Jan.) TCP-Friendly unicast rate-based flow control. Technical note sent to end2end-interest mailing list. [Online]. Available: http://www.psc.edu/networking/papers/tcp_friendly.html

[23] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Trans. Netw.*, vol. 8, no. 5, pp. 556 – 567, Oct. 2001.

[24] S. Biaz and N. H. Vaidya, "Distinguishing congestion losses from wireless transmission losses: a negative result," in *Proc. of the Seventh International Conference on Computer Communications and Networks (IC3N)*, New Orleans, USA, Oct. 1998.

[25] ——, "Discriminating congestion loss from wireless losses using inter-arrival times at the receiver," in *Proc. of IEEE Symposium on Application-specific System and Software Engr. and Techn.*, Richardson,TX, USA, Mar. 1999, pp. 10–17.

[26] Y. Tobe, Y. Tamura, A. Molano, S. Ghosh, and H. Tokuda, "Achieving moderate fairness for udp flows by pathstatus classification," in *Proc. of 25th Annual IEEE Conf. on Local Computer Networks*, Tampa,FL, USA, Nov. 2000, p. 252C261.

[27] C. JA and A. P., "Congestion or corruption? a strategy for efficient wireless tcp sessions," in *Proc. of IEEE Symposium on Computers and Communications*, Los Alamitos, CA, USA, 1995, pp. 262–268.

[28] P. Sinha, T. Nandagopal, N. Venkitaraman, R. Sivakumar, and V. Bharghavan, "A wireless transmission control protocol for cdpd," in *Proc. of IEEE Wireless Communications and Networking Conference*, Piscataway, NJ, USA, Jan. 1999, pp. 953–957.

[29] ——, "Wtcp: a reliable transport protocol for wireless wide-area networks," *Wireless Networks*, vol. 8, no. 2-3, pp. 301–316, 2002.

[30] W. Ding and J. A, "A new explicit loss notification with acknowledgment for wireless tcp," in *Proc. of 12th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, Piscataway, NJ, USA, 2001, pp. B–65–9.

[31] C. CF and M. M., "Improving tcp over wireless through adaptive link layer setting," in *Proc. of IEEE Global Telecommunications Conference*, Piscataway, NJ, USA, 2001, pp. 1766–1770.

[32] J.-H. Choi, S.-H. Yoo, and C. Yoo, "A flow control scheme based on buffer state for wireless tcp," in *Proc. of the 4th International Workshop on Mobile and Wireless Communications Network*, Piscataway, NJ, USA, 2002, pp. 592–596.

[33] K. Ratnam and I. Matta, "Wtcp: an efficient mechanism for improving wireless access to tcp services," *International Journal of Communication Systems*, vol. 16, no. 1, pp. 47–62, Feb. 2003.

[34] J. Rendon, F. Casadevall, and J. Carrasco, "Wireless tcp proposals with proxy servers in the gprs network," in *Proc. of 13th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, Piscataway, NJ, USA, 2002, pp. 1156–1160.

[35] Y. Yang, H. Zhang, and K. R., "Channel quality based adaptation of tcp with loss discrimination," in *Proc. of IEEE Global Telecommunications Conference*, Piscataway, NJ, USA, 2001, pp. 2026–2030.

[36] J.-J. Lee, F. Liu, and K. C-CJ, "End-to-end wireless tcp with non-congestion packet loss detection and handling," in *Proc. of the SPIE*, San Jose, USA, Jan. 2003, pp. 104–113.

[37] S. Sastry, *Nonlinear Systems, Analysis, Stability and Control*. New York, NY: Springer Verlag, 1999.

[38] H. Khalil, *Nonlinear Systems (3rd edition)*. Prentice Hall, 2001.

[39] M. Chen, "A general framework for flow control in wireless networks," Ph.D. dissertation, University of California at Berkeley, Berkeley, CA 94706, Dec. 2006. [Online]. Available: http://www-video.eecs.berkeley.edu/m̃inghua/papers/phd.thesis.pdf

[40] M. Chen and A. Zakhor, "AIO-TFRC: A light-weighted rate control scheme for streaming over wireless," in *Proc. of IEEE WirelessCom Symposium on Multimedia over Wireless 2005*, Jun. 2005.

[41] I. Rhee and L. Xu, "Limitations of equation-based congestion control," in *Proc. ACM SIGCOMM*, Philadelphia, Pennsylvania, Aug. 2005, pp. 49–60.

[42] Traceroute. [Online]. Available: http://www.traceroute.org/

[43] Kazza. [Online]. Available: http://www.kazza.com

[44] Pplive - internet peer-to-peer video streaming. [Online]. Available: http://www.pplive.com

[45] Brew - binary runtime environment for wireless. [Online]. Available: http://brew.qualcomm.com/brew/en/

[46] B. Briscoe, "Flow rate fairness: Dismantling a religion," *ACM SIGCOMM Computer Communication Review*, vol. 37, pp. 63–74, Apr. 2007.

[47] S. F. Deepak Bansal, Hari Balakrishnan and S. Shenker, "Dynamic behavior of slowly-responsive congestion control algorithms," in *Proc. ACM SIGCOMM*, San Diego, CA, Aug. 2001.

[48] G. Yang, L.-J. Chen, T. Sun, M. Gerla, and M. Y. Sanadidi, "Smooth and efficient real-time video transport in presence of wireless errors," *ACM Transactions on Multimedia Computing, Communications and Applications*, vol. 2, no. 2, pp. 109 – 126, May 2006.

[49] Network simulation version 2. [Online]. Available: http://www.isi.edu/nsnam/ns/

[50] M. Chen and A. Zakhor, "Flow control over wireless network and application layer implementation," in *Proc. IEEE INFOCOM*, Barcelona, Apr. 2006.

[51] S. Liu, T. Basar, and R. Srikant, "Tcp-illinois: A loss and delay-based congestion control algorithm for high-speed networks," *Special Issue of Performance Evaluations*, 2007, to appear.

[52] L. Chen, S. H. Low, M. Chiang, and J. C. Doyle, "Cross-layer congestion control, routing and scheduling design in ad hoc wireless networks," in *IEEE Infocom*, Barcelona, Spain, Apr. 2006.

Minghua Chen received B. Eng. and M. S. degrees in Electronic Engineering from Tsinghua University, and a Ph.D. degree in Electrical Engineering and Computer Sciences from University of California at Berkeley, in 1999, 2001 and 2006 respectively. He is now with the Communication and Collaboration System group in Microsoft Research. His current research interests are in flow control and routing in wireless networks, utility maximization for Peer-to-Peer applications, digital video and audio processing, erasure coding for distributive storage and wireless communications. He is co-author of the book IPv6 Principle and Practice (People's Posts and Telecommunication Publishing House, 2000). He is the recipient of the 2007 Eli Jury award given by Department of Electrical Engineering and Computer Sciences at University of California at Berkeley.

Avideh Zakhor received a B. S. degree from California Institute of Technology, Pasadena, and S. M. and Ph. D. degrees from Massachusetts Institute of Technology, Cambridge, all in electrical engineering, in 1983, 1985, and 1987 respectively. In 1988, she joined the Faculty at U. C. Berkeley where she is currently Professor in the Department of Electrical Engineering and Computer Sciences. Her research interests are in the general area of image and video processing, multimedia communication, and 3D modeling. Together with her students, She has won a number of best paper awards, including the IEEE Signal Processing Society in 1997, IEEE Circuits and Systems Society in 1997 and 1999, international conference on image processing in 1999, and Packet Video Workshop in 2002. She holds 5 U.S. patents, and is the co-author of the book, "Oversampled A/D Converters" with Soren Hein.

Prof. Zakhor was a General Motors scholar from 1982 to 1983, was a Hertz fellow from 1984 to 1988, received the Presidential Young Investigators (PYI) award, and Office of Naval Research (ONR) young investigator award in 1992. From 1998 to 2001, she was an elected member of IEEE Signal Processing Borad of Governers. In 2001, she was elected as IEEE fellow. She received the Okawa Prize in 2004.

She co-founded OPC technology in 1996, which was later by Mentor Graphics (Nasdaq: MENT) in 1998, Truvideo in 2000, and Urban Scan in 2005.

Ryusuke Fujita is currently an undergraduate student at U.C. Berkeley with Electrical Engineering and Computer Science and Physics major. He is a member of EECS Honor Degree Program, and expected to receive B.S and B.A degree by December 2006. During summer 2005, he worked for TruVideo, founded by Prof. Avideh Zakhor, and developed video/audio streaming software-application for cell phones. He was a participant of Undergraduate Research Opportunity program at U.C. Berkeley in 2005 with adviser Prof. Ruzena Bajcsy, and his project focused on 3-D image reconstruction with moving cameras.