Flow Control over Wireless Network and Application Layer Implementation

Minghua Chen and *Avideh Zakhor*
Department of Electrical Engineering and Computer Sciences
University of California at Berkeley, CA 94720
{minghua, avz}@eecs.berkeley.edu

***Abstract*— Flow control, including congestion control for data transmission, and rate control for multimedia streaming, is an important issue in information transmission in both wireline and wireless networks. Widely accepted flow control methods in wireline networks are TCP [1] for data, and TFRC [2] for multimedia. Kelly [3] [4] has laid down theoretical framework for TCP in wireline networks demonstrating its optimality, fairness, and stability. 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 first show flow control in the wireless networks can be formulated as the same concave optimization problem Kelly defined in the wireline networks. TCP and TFRC in the wireless networks pursue the optimal solution in the presence of inaccurate feedback. All existing approaches to this TCP/TFRC over wireless problem correct the inaccurate feedback by casting modifications to existing protocols, such as TCP, or infrastructure elements such as routers, thereby making them hard to deploy in practice. We then formulate the problem as another concave optimization problem with a different utility function, and propose a new class of solutions. Our approach is end-to-end, and achieves good performance by adjusting the number of connections of a user according to a properly selected control law. The control law is based on only one bit of information, which can be reliably measured at the application layer. We show that the control system has a unique stable equilibrium that solves the concave optimization problem, implying scalability and optimality of the solution. We apply our results to design a practical rate control scheme for streaming over wireless networks, and characterize its performance using NS-2 simulations and actual experiments over Verizon Wireless 1xRTT data network. Analysis and simulation results also indicate our scheme is applicable to both wireline and wireless scenarios.**

I. Introduction

TCP has been widely successful on the wireline Internet since its first implementation by Jacobson [1] in 1988. TCP Reno, the most popular TCP version today, in its congestion avoidance stage, increases its windows size by one if no packet is lost in the previous round trip time, and halves the windows size otherwise. 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 seriously underutilizes the wireless bandwidth. Similar observations hold for TCP-friendly schemes, such as TCP-Friendly Rate Control (TFRC) commonly used for streaming applications [2], as they share the same key assumption as TCP. Particularly, our experiments over Verizon Wireless 1xRTT wireless data network have shown that TFRC achieves only 56% of the available wireless bandwidth [5]. The need to solve the problem is becoming urgent today as wireless data and streaming services are becoming increasingly more popular.

Consequently, There have been a number of efforts to improve the performance of TCP or TFRC over wireless [6]–[11]. These approaches either hide end-hosts from packet loss caused by wireless channel error, e.g. Snoop [6], 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 solutions [8]–[11].

Another alternative is to use non-loss based rate control schemes. For instance, TCP Vegas [12], in its congestion avoidance stage, uses queueing delay as a measure of congestion, and hence could be designed not to be sensitive to any kind of packet loss, including that due to wireless channel error. It is also possible to enable the routers with ECN markings capability to do rate control using ECN as the measure of congestion [13]. As packet loss no longer corresponds to congestion, ECN based rate control does not adjust sending rate upon observing a packet loss.

Our recent work, MULTFRC [5], opens appropriate number of connections to fully utilize the wireless bandwidth. This approach improves TFRC performance in wireless networks by modifying only the application layer, and as such is also applicable to TCP. We have also proposed an improved scheme named AIO-TFRC [14], which eliminates the overhead and performance degradation associated with operating multiple connections.

While the existing approaches provide practical insight on how to improve the performance of TCP and TFRC over wireless, it is unclear whether their performance can easily scale to a network as large as the Internet, and whether they are the only possible way to achieve optimal performance. Hence a general framework for flow control over wireless is needed to address the issues of optimality and stability, and to provide guidelines and performance prediction prior to any implementation. Specifically, the following questions need to be answered:

- How does one define the problem of flow control over wireless network? What is the analysis framework?
- Under the framework, is there any *optimal* and *stable*

application layer scheme that solves the problem?

In the past decade, there has been a great deal of research activity on decentralized end-to-end flow control algorithms on wireline network. A widely recognized setting, introduced by Kelly et. al. [3] and refined for TCP in [4], views flow control schemes as algorithms to compute the optimal solution to a utility maximization problem. Some similar work focusing on relating different versions of TCP to different utility functions, evaluating network stability with and without delay or noise, and designing new TCP and queue management algorithms, are included in [15]–[19]. In one kind of the algorithms Kelly et. al. proposed, namely primal algorithms, the users adapt source rates dynamically based on the prices along the path, and the routers select a static law to determine their prices directly from the arrival rates at the link [3]. Kelly showed TCP is in fact a primal like algorithm with packet loss rate as the associated price function [4]. The stability for the system with and without delay, with and without disturbance, are developed in [4].

In this paper, we assume a wireless link is associated with a fixed bandwidth and a fixed packet loss rate caused by the physical channel errors. We first show in the presence of the packet loss caused by physical channel error, flow control in the wireless network can be formulated as the same concave optimization problem defined by Kelly in the wireline network [4]. TCP and TFRC in the wireless networks pursue the optimal solution in the presence of inaccurate feedback[1]. All existing approaches to this problem correct the inaccurate feedback by casting modifications to existing protocols, such as TCP, or infrastructure elements such as routers, thereby making them hard to deploy in practice.

We then formulate the problem as another concave optimization problem with a different utility function, followed by a new class of solutions. Our approach is end-to-end, and achieves good performance by adjusting the number of users' connections according to a properly selected control law. The control law is based on only one bit of information, which can be reliably measured at the application layer. We show that the resulting control system has a unique stable equilibrium that solves the concave optimization problem, implying scalability and optimality of the solution. We then apply our results to design a practical rate control scheme for streaming over wireless networks, and characterize its performance using NS-2 simulations, and actual experiments over Verizon Wireless 1xRTT data network. Analysis and simulation results indicate our scheme is applicable to both wireline and wireless scenarios.

This paper is structured as follows. Section II includes problem formulation. A new approach addressing the problem is proposed in Section III, together with analysis for its optimality and stability. Section IV shows the design of a practical scheme. NS-2 simulations and actual experiments over 1xRTT wireless data networks are included in Section V. Section VI concludes the paper with discussions and future work.

[1]In [20], [21], we have also shown the similar formulation using the framework in [3].

II. Problem formulation

A. Overview of the flow control framework and TCP modeling

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 to be the set of routes, where a *route* r is a non-empty subset of J associating 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.

Assume utilities are additive, so that the 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. [3] and later refined in [4], is a concave optimization problem maximizing the net utility[2]:

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

where $\int_0^{\sum_{s:j\in s} x_s} p_j(z)\, dz$ can be considered as the 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), the sum of users' utility minus the costs associated with utilizing the links, is strictly concave. One common price function used in practice is the packet loss rate, which is zero if there is no congestion, and concavely increases otherwise:

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

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

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

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. $1 - \prod_{j\in r} p_j(y_j(t))$, is approximately $\sum_{j\in r} p_j(y_j(t))$.

Under these settings, Kelly [4] has shown TCP Reno [3] to be a primal-like algorithm, as follows:

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

[2]It is not hard to realize this is nothing but a penalty relaxation solving the sum utility maximization with capacity constrains, namely:

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

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

solving the optimization problem in (1) with $U_r(x_r) = -\frac{2S^2}{T_r^2 x_r}$ where S is the TCP packet size, T_r is the end-to-end round trip time, and $p_j(y)$ takes the form in (2). Rewriting (1) with these quantities, we obtain:

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

Thus, TCP can be viewed as a discrete version of the steepest gradient descent algorithm which solves the optimization problem in (4).

Equation (3) describes the time evolution of sending rate $x_r(t)$, whereby the user exploits only the aggregate packet loss information along its path. We assume the same flow $x_r(t)$ is presented to all links $j \in r$, even though the flow in downstream links shrinks due to losses at upstream links. This is a direct implication of our previous assumption that the packet loss rate on link j, i.e. $p_j(\sum_{s:j\in s} x_s)$, is small.

Kelly [4] showed the system in (3) has a unique equilibrium, to which all trajectories converge, as follows:

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

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 as described by Mahdavi and Floyd in [22]. This equilibrium is also the finite 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. $C_j \le y_j^o < \infty$; secondly, there is roughly α-fairness [23] among users with $\alpha = 2$, implying the users are allocated rates that roughly maximize a sum of utility of the form $x_r^{-\alpha} = x_r^{-2}$.

B. TCP and TFRC over wireless

For wireless networks, we assume the links $j \in J$ are associated with not only a fixed capacity but also a packet loss rate caused by the physical channel errors, necessarily nonnegative, denoted by $\epsilon_j \ge 0, j \in J$. Nevertheless, TCP over wireless is still associated with the same concave optimization problem as with the wireline networks, shown in (4). This is because neither the utility function of users in (4), i.e. $U_r(x_r)$, nor the cost associated with using network resources, i.e. $\int_0^{y_j} p_j(z)\,dz$, is a function of $\epsilon_j \ge 0, j \in J$. In effect, ϵ_j results in the price functions fed back to users to be inaccurate, as we will see in following paragraphs. Hence TCP algorithms still aim to address the same optimization problem shown in (4), but with inaccurate prices fed back from network.

The inaccurate price function, denoted by $q_j(y_j(t))$, is a the sum of ϵ_j and $p_j(y_j(t))$, under the assumption that ϵ_j is small:

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

When the link is not congested, $q_j(y_j(t)) = \epsilon_j$ since all packet losses are caused by channel error; $q_j(y_j(t))$ gradually increases otherwise. With this inaccurate price, TCP now adjusts the sending rates as:

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

Following a similar analysis in [4], one can show the system (7)-(6) has a new unique equilibrium, to which all trajectories converge, as follows:

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

where $\bar{y}_j = \sum_{s:j\in s}\bar{x}_s$. Although it can be shown that there is roughly α-fairness among users with $\alpha = 2$, $\bar{x} \triangleq [\bar{x}_r, r \in R]$ is a suboptimal solution as it is different from the unique optimal one x^o. Furthermore, user r could suffer underutilization if $\sum_{j\in r}\epsilon_j$ is sufficiently large. For instance, in the one user one bottleneck network, underutilization happens if and only if $\frac{\sqrt{2}S}{T_r\sqrt{\epsilon}} < C$, where C is the bottleneck bandwidth and ϵ represents the aggregate packet loss rate caused by wireless channel error experienced by the user. This sufficient and necessary condition has been verified in [5]. Hence the main problem with TCP over wireless is underutilization; in fact, similar analysis shows that it is also the main problem with any flow control method that uses packet loss rate as price function, such as TFRC.

One straightforward solution is to provide user r with the accurate price $\sum_{j\in r} p_j(y_j(t))$, 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 retransmission. In fact, most existing approaches belong to this class of solutions, thus requiring modifications either to the protocols or to the network infrastructure, whereby making them hard to deploy in practice.

In essence, the main challenge of TCP optimization problem shown in (4) for the wireless network setting is to accurately estimate price $\sum_{j\in r} p_j(y_j(t))$ from the noisy measurements $\sum_{j\in r} q_j(y_j(t))$. One way to overcome this problem is to specifically choose a slightly different utility maximization problem, resulting in a new solution that requires measurements that are easy to obtain in a practical networking setup. In the next section, we will show that by selecting a different utility to maximize, there exists a new, stable and optimal solution, which requires only one bit of end-to-end measurement.

III. Proposed solution

A. A new class of solutions

Motivated by our previous approach MULTFRC [5], we now propose a new approach to flow control based on gradually adjusting the number of connections for user r, denoted by $n_r(t)$. This is an end-to-end application layer based scheme, and requires no modification to the network infrastructure or the protocol stack. The goals of our approach are three fold. First, to stably pursue full utilizations of the bottleneck links; second, to optimally solve a net utilities maximization problem; third, to achieve fairness among users.

In MULTFRC [5], the sending rate of individual connections is controlled by TFRC itself; the number of connections $n_r(t)$ is inversely increased if there is no congestion, and is additively decreased otherwise, commonly referred as an IIAD control. The NS-2 simulations and actual experiments

in [5] show that the performance of MULTFRC is reasonably good even though there is no theoretical framework justifying its stability and salability. Motivated by MULTFRC [5], our proposed approach here dynamically adjusts both $x_r(t)$ and $n_r(t)$ as follows:

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

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

where c is a constant, and $I_r(\sum_{j\in r} p_j(y_j(t)))$ is an indicator function implying the congestion status of route r:

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

It is easy to see that the control law of $n_r(t)$ in (10) is Inverse Increase and Multiplicative Decrease (IIMD), which is different from MULTFRC [5]. As we will see later, this modification leads to several desirable properties for the system.

As shown in (9), the control law of the aggregate rates for user r, i.e. $x_r(t)$, can be understood as the sum of rates from $n_r(t)$ individual connections, with each connection being controlled using the standard TCP Reno algorithm. Therefore, our approach does not require any modifications to the TCP protocol. On the other hand, as seen in (10), the system tries to achieve full utilization by adjusting the number of connections $n_r(t)$ accordingly. In particular,

- if a route r is underutilized, then $I_r[\sum_{j\in r} p_j(y_j(t))] = 0$; this implies that the number of connections $n_r(t)$ will increase in order to boost the user's rate $x_r(t)$, pursuing full utilization on any route r;
- if the route r is fully utilized, i.e. one of its links is congested, then $I_r[\sum_{j\in r} p_j(y_j(t))] = 1$, lowering $n_r(t)$, 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, individual connection's sending rate is lowered, thus users need to open more connections to increase the aggregate throughput. The $I_r(\cdot)$ is the one bit of information required from the end-to-end measurements. In practice, we can estimate $I_r(\cdot)$ using end-to-end round trip time measurements. In particular, we estimate the queuing delay by comparing current round trip time with the propagation delay, and set $I_r(\cdot) = 1$ if the queuing delay is positive and $I_r(\cdot) = 0$ otherwise.

The system in Eqns. (9) and (10) is a coupled, discontinuous, nonlinear system. In order to verify that this system actually meets our design goals, we need to analyze the existence of a unique equilibria, its stability and its optimality in the sense of solving a utility maximization problem.

B. Discontinuity approximation and the two time scale decomposition

The discontinuities 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]: $\forall j \in J, r \in R$,

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

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

where β is a nonnegative constant. It should be clear that $f_r(\sum_{j\in r} g_j(y_j(t)) \to I_r(\sum_{j\in r} g_j(y_j(t))$ 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: $\forall r \in R$,

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

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 [24], [25].

The approximate system in (14), though continuous, is difficult to analyze in general. It is a nonlinear, coupled, multivariable system, and the two equations are not exactly symmetric even though they might appear to be so. Hence, we introduce a two time scale 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. It is also reasonable in practice, where the sending rates are expected to change on the order of tens of milliseconds, while number of connections is expected to change at a much slower rate, e.g. tens of seconds.

Under the above assumption, system (14) fits into the classical singular-perturbation framework [24] [25], and therefore can be decoupled into a short-scale and a long-scale system. The short-scale system is described by (9) with the corresponding $n_r(t), r \in R$ being constant, namely boundary system: $\forall r \in R$,

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

The long-scale system is described by (10) along the manifold defined by the stationary solution of (9), namely reduced

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

system: $\forall r \in R$,

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

Under the two times scale setting, the behavior of the system can be described as follows. On the fast timescale, $n(t)$ can be thought of as being constant since its dynamics happens at a slow timescale. The entire system can then be expressed as a boundary system shown in (15). As the boundary system is similar to Kelly et. al.'s control system on wireline networks except for the constant $n(t)$, and the price function $p_j(y_j(t))$ being replaced by $\epsilon_j + g_j(y(t)$, the behavior of the boundary system can be easily inferred from the known results for the system in (3). Specifically, at the fast timescale, $x(t) \triangleq [x_r(t), r \in R]$ globally exponentially converges to the equilibrium manifold defined as follows [4], [17]:

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

This relation between $x(t)$ and $n(t) \triangleq [n_r(t), r \in R]$ can be shown to be a one-to-one mapping between $n(t)$ and $x(t)$ [26]. On the slow timescale, $x(t)$ has already converged to the above equilibrium manifold, and now the system collapses into the reduced system described in (16), whose behavior determines how the approximate system evolves at the slow timescale. Therefore, together with boundary system, it fully characterizes behavior of the system for all possible timescales. For illustration purpose, we have handdrawn the time dynamics of singular perturbation system comprising of a single user with sending rate $x_1(t)$, and $n_1(t)$ connections in Figure 1. As shown, given any initial condition, the system first converges rapidly onto the equilibrium manifold, representing the fast timescale indicated by its boundary system. On the manifold, the system's behavior follows its reduced system; if the reduced system has the equilibrium point as the globally asymptotically stable equilibrium, then the trajectories, along the manifold, will converge to the equilibrium as time goes to infinity.

In the next subsection, we present the results on the existence of a unique optimal equilibrium of the system and its stability.

C. The existence of a unique optimal equilibrium and its stability

Given the system in (14), the first question to answer is whether it has any equilibria, and if so how many? The second question is the local and global stability of these equilibria. These two questions are important in the sense that they describe the behavior of the system as time evolves, predicting the system's performance in actual implementations in practice. For instance, if the system in (14) has no equilibrium, the users' sending rates would not converge, making the system undesirable to implement. The consequence would be even worse if the system has one globally stable equilibrium at infinity, implying the users' rates become arbitrarily large, resulting in serious congestion collapse.

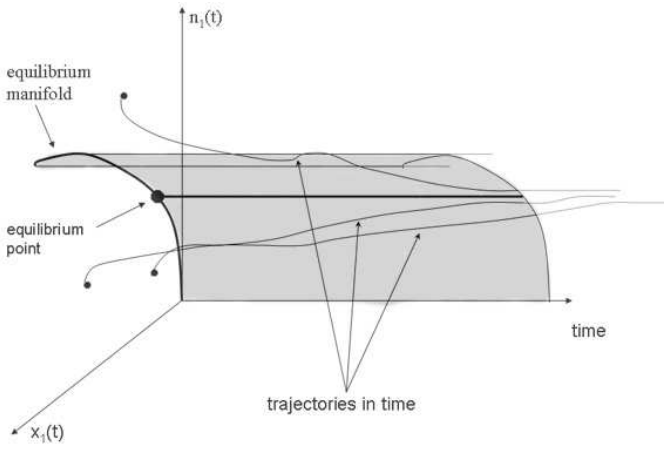

Fig. 1. Time dynamics of a singular perturbation system.

We now show that the system in (14) has only one unique equilibrium, which is locally exponentially stable, and globally asymptotically stable under certain conditions. Further, the unique equilibrium solves a concave optimization problem. We show this through the following theorem.

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

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

Further, this unique equilibrium 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:

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

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

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

Proof: Proof utilizes the two timescale decomposition, and is included in [26]. ■

One observation to be made from Theorem 1 is that the unique equilibrium for the system in (14) in wireless scenario solves a concave optimization problem similar to the one TCP Reno solves in the wireline network. They have the same form as (1) but with different users' utility functions $U_r(x_r)$. The only difference is that the $U_r(x_r)$ in the wireline 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 wireline network scenario, we have $h_r(z) = z$,

and the optimization problem in (19) becomes identical to the TCP optimization problem in (4). In this case, the equilibrium (x^*, n^*) is exactly the same as x^o, implying TCP optimization problem in the wireline network is merely a special case of that in (19).

Given the system in (14) has a unique optimal equilibrium, an important question to answer is that whether it is stable, i.e. will the users' rates converge to it asymptotically. The following two theorems explore the answer to this question.

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 globally exponentially stable, with the same arguments used in [24] and [25] for the stability of singular disturbance non-linear system, the unique equilibrium of the approximate system in (14), (x^*, n^*), is locally exponentially stable.

Proof: Proof utilizes the two timescale decomposition, and is included in [26]. ■

Theorem 2 implies that if the number of connections $n(t)$ is initially in a small ball around the equilibrium n^*, then the entire system will converge exponentially fast to the equilibrium. As no convergence can be faster than exponential, this is the best result one can expect in a local region around the equilibrium.

How about when $n(t)$ starts far away from the equilibrium n^*? To answer this question, we need to explore the global stability of the equilibrium. The following theorem states that if there is only one bottleneck link for all the users in the network, for example the dumb-bell topology, then all users' rates converge globally to the optimal equilibrium.

Theorem 3: For arbitrary $\beta > 0$, under the two timescale assumption, if all the f_r are the same, then the unique equilibrium of the reduced system in (16) is globally asymptotically stable; hence the unique equilibrium of the approximate system in (14), (x^*, n^*), is globally asymptotically stable.

Proof: Proof utilizes the two timescale decomposition, and is included in [26]. ■

The intuition behind both the local and global convergence of the entire system is as follows: $x_r(t)$ first converges in the fast timescale to the equilibrium of the boundary system (15), defining the equilibrium manifold as in (17); 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. An important consequence of this 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. Therefore, it is possible to extend all these results to TFRC since it has been shown in [2] that TFRC has the same stationary behavior as TCP Reno.

Remarks. 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 $y_j(t) = C_j$.

As seen in the next section, in actual implementation of the proposed system in (9) and (10), it is necessary to discretize 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 by TCP to adjust the finite number of packets to be sent out in a time interval. Therefore, it is highly unlikely for the system to operate at discontinuous points. 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.

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 routes are fully utilized, yet there is no congestion collapse; second, the system is locally exponentially stable, and globally asymptotically stable under certain conditions. Hence all of our design objectives are met.

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 practical scheme, following (9)-(10), for both wireless and wireline networks, with the latter corresponding to the case where $\epsilon_j = 0, j \in J$.

IV. Enhanced MULTFRC (E-MULTFRC)

Although the analysis in previous section is based on TCP model, it can be extended to TFRC, since it has been shown TFRC has the same stationary behavior as TCP [2]. As we are interested in rate control for streaming over wireless, we design a practical scheme for that, based on our analysis and control law in (10), by adjusting the number of TFRC connections.

The framework of our proposed E-MULTFRC is shown in Figure 2. As seen, there are two components in the system: rtt measurement sub-system (RMS), and connections controller sub-system (CCS).

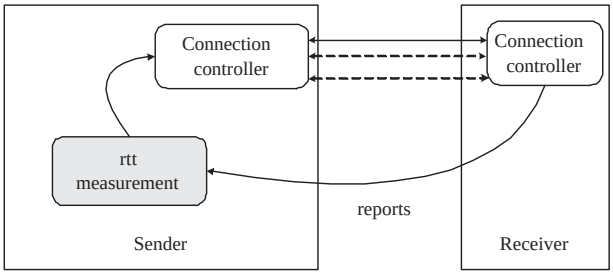

Fig. 2. E-MULTFRC system framework.

A. rtt Measurement Sub-system (RMS)

The gray block in Fig. 2 represents RMS that resides at the sender; it receives reports from receiver every round trip time, containing average round trip time rtt_{sample} measured in the past round trip time window. After waiting for a fixed interval, denoted by τ, RMS computes the running average, ave_rtt, of

these rtt_{sample}s, and reports it to the CCS. Here, $1/\tau$ defines the frequency of adjusting number of connections; it 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. Currently, we choose τ to be 20 seconds.

B. Connection Controller Sub-system (CCS)

The CCS is shown as the white blocks in Figure 2. Its basic functionality is to properly control the number of connections n, following the control law designed in (10). As seen in (10), when route r is underutilized, the indicator function $I_r(\cdot) = 0$, and n_r is inversely increased, i.e. proportional to $1/n_r$. When route r is congested, $I_r(\cdot) = 1$, and the decrease on n_r will be roughly proportional to n_r. This indicates CCS at the sender should roughly Inversely Increase and Multiplicatively Decrease (IIMD(α, β)) the number of connections n, based on route congestion status. Specifically, CCS is reported ave_rtt from 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:

$$n = \begin{cases} \beta n + \alpha/n, & \text{if } ave\_rtt - rtt\_min > \gamma rtt\_min; \\ n + \alpha/n, & \text{otherwise.} \end{cases} \tag{20}$$

where $\alpha = 1 - \beta < 1$ and γ is a preset parameter. This is nothing but a discrete implementation of the control law (10). 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-MULTFRC can tolerate before it starts to decrease the number of connections. Therefore, under ideal conditions, E-MULTFRC keeps increasing the number of connections to make ave_rtt as close as possible to $(1+\gamma)rtt\_min$ without exceeding it.

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-MULTFRC. Under these conditions, it is conceivable to use route change detection tools such as traceroute [27] at the sender to detect the route change, in order to reset rtt_min to a new value. Furthermore, it can be argued that the overall throughput of E-MULTFRC will not go to zero, resulting in starvation; this is because E-MULTFRC always keeps at least one connection open.

Since the stream is transmitted using multiple connections, the receiver could potentially receive out of order packets. If the receiver uses a buffer to reorder the arriving packets, the out of order arrived packets can be reordered according to their sequence numbers.

In summary, in the E-MULTFRC system, the number of connections are controlled according to a discrete version of (10) at the sender; the sending rate of each TFRC connection is controlled by TFRC itself, which has the same stationary behavior as TCP in (9). The rate of change of number of connections is expected to be much slower than that of sending rate; this meets 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-MULTFRC, indicating that E-MULTFRC results in a stable, yet fully utilized network as shown in (19).

Notice that E-MULTFRC is similar to our previously proposed scheme MULTFRC [5] in its design, except for the control law for number of connections n. E-MULTFRC applies IIMD, while MULTFRC applies IIAD. An inherent advantage of E-MULTFRC over MULTFRC is its provable convergence and stability properties as shown in the previous section.

V. NS-2 SIMULATIONS AND 1XRTT WIRELESS EXPERIMENTS

In this section, we carry out NS-2 [28] simulations and actual experiments over Verizon Wireless 1xRTT CDMA data network to evaluate the performance of E-MULTFRC.

A. Setup

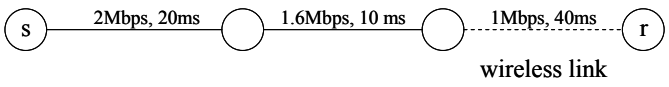

Fig. 3. Simulation topology.

The topology used in simulations is shown in Figure 3. The sender denoted by s, and the receiver denoted by r, both run E-MULTFRC at the application layer. 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 modelled 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-MULTFRC's performance in the presence of wireless channel errors. We examine three issues; first, how E-MULTFRC 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-MULTFRC 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 in it.

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 streaming connection. The streaming takes 30 minutes. As we cannot control p_w in actual experiments, we measure the average throughput, average number of connections, and packet loss rate.

B. Performance Characterization of E-MULTFRC

We have empirically found the following parameters to result in reasonable performance: $\alpha = 0.25, \beta = 0.75$, $\gamma = 0.2$, and $\tau = 20$ seconds. Intuitively, larger τ results in more reliable estimates of round trip time, but at the expense of a lower sampling frequency, resulting in a less responsive system. Larger γ results in a system that is more robust to round trip time estimates, but at the expense of longer queues in routers, and hence a longer queueing delay. Larger α results in a faster convergence rate to full utilization, but at the expense of a slower response to congestion. We have determined these values empirically through simulations and experiments.

We simulate the E-MULTFRC system to stream for 9000 seconds, and compute the average throughput and packet loss rate for p_w =0.0, 0.02, 0.04, 0.06 and 0.08, 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 4. As seen, the throughput is within 25% of the optimal, the round trip time is within 120% 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. [5]

Considering the throughput plot in Figure 4, we notice that for some values of p_w, there is a significant difference 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 large variation in 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 smaller packet size in order to reduce the "quantization effect" at the expense of (a) lower transmission efficiency and (b) the slower rate of convergence to the optimal number of connections. In the next subsection, we propose an Enhanced All-In-One TFRC algorithm to alleviate this quantization effect.

The comparison between the performance of E-MULTFRC and MULTFRC in Figure 4 also shows the difference caused by applying different control laws in these two schemes. When p_w is small, e.g. 0.02, the optimal number of connections is small. In this case, E-MULTFRC outperforms MULTFRC since when decreasing the number of connections n, E-MULTFRC only decreases it proportionally whereas MULTFRC decrements it. On the other hand, when p_w is large, e.g. 0.08, the optimal number of connections is large. In this case, the proportional decrease in n is more significant than decrementing it. Therefore, as the behavior of E-MULTFRC and MULTFRC are similar when increasing the number of connections, the one with more significant decrease in n will have lower average throughput.

In order to examine E-MULTFRC's performance as a function of p_w, as well as the dynamics of E-MULTFRC, we use E-MULTFRC 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-MULTFRC 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 5. As seen, the number of connections varies from around 2-3 to around 5 as p_w switches from 0.02 to 0.06.

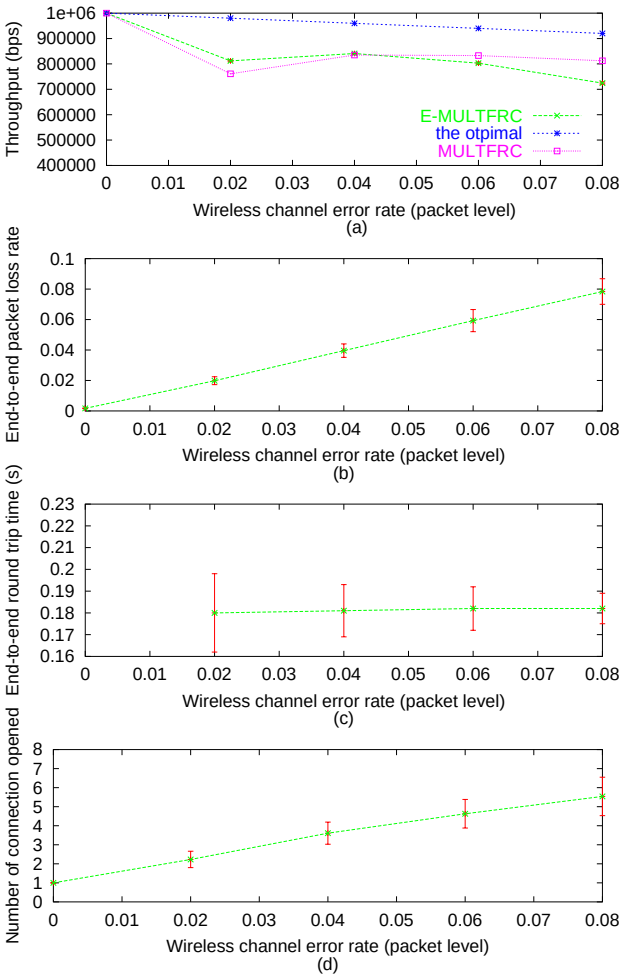

Fig. 4. NS-2 simulations for $B_w = 1\ Mbps$ and $RTT_{min} = 168\ ms$; (a) throughput, (b) end-to-end packet loss rate, (c) end-to-end round trip time, (d) number of connections, all as a function of packet level wireless channel error rate.

Similar experiments are carried out on Verizon Wireless 1xRTT CDMA data network. The 1xRTT CDMA data network 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-MULTFRC and TFRC. The results are shown in Table I for packet size of 1460 bytes. As seen, on average E-MULTFRC opens up 1.9

[5]Note the round trip time for $p_w = 0$ is not shown in Figure 4 because it represents the channel error free case in which E-MULTFRC reduces to one TFRC connection.

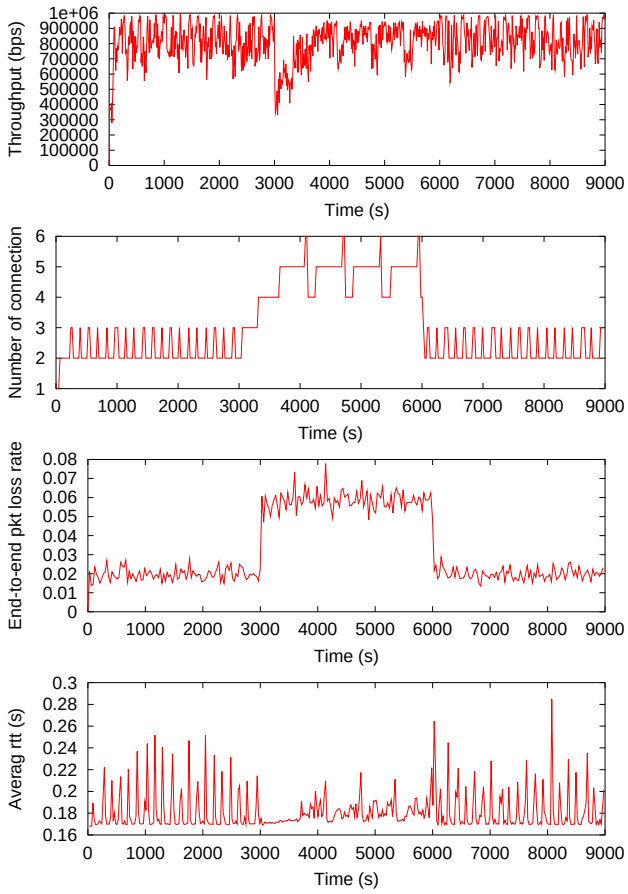

Fig. 5. NS-2 simulation results as p_w changes from 0.02 to 0.06 and back again; (a) end-to-end round trip time, (b) throughput, (c) numbers of connections, (d) end-to-end packet loss rate, all as a function of time.

connections, and results in 65% higher throughput, at the expense of a larger round trip time, and higher packet loss rate.

Table II shows packet loss details of E-MULTFRC for one of the 30 minutes long experiments over 1xRTT. As expected, both the packet loss rate and burstiness of the loss increase as the number of connections increases.

TABLE I

ACTUAL EXPERIMENTAL RESULTS FOR A E-MULTFRC SYSTEM OVER 1XRTT CDMA.

scheme	throughput (kbps)	rtt (ms)	packet loss rate	ave. # of conn.
one TFRC	54	1624	0.031	N/A
E-MULTFRC	89	2767	0.041	1.9

C. Enhanced All-In-One TFRC (EAIO-TFRC)

There are two drawbacks associated with E-MULTFRC as seen from the simulations and actual experiments in the previous section. First drawback has to do with bandwidth underutilization, and the second one with implementation complexity. We will begin with utilization drawback. NS-2 simulations show that although E-MULTFRC performs reasonably well,

TABLE II

PACKET LOSS DETAILS OF E-MULTFRC

# of conn. opened	% of time	pkt loss rate	avg. burst error length	std. dev.	max. burst length
one	24.1	0.016	2.78	3.26	7
two	61.1	0.045	2.43	3.33	10
three	14.8	0.076	3.45	8.93	11

there is still some gap between its throughput and the optimal. Specifically, E-MULTFRC achieves only 81% utilization for $p_w = 0.02$. This suboptimal performance has two causes. First one is the control behavior described in (20): as described, n is multiplicatively decreased when the full utilization of bottlenecks is detected, and is inversely increased until the next full utilization is detected. During this period, bottlenecks stay underutilized, resulting in suboptimal average throughput. It is impossible to remove this sub-optimality resulting from the control law without changing the law itself. The second reason for bandwidth underutilization is the "quantization effect" in E-MULTFRC whereby in practice the number of connections is forced to be an integer. This loss of granularity typically results in bandwidth underutilization. For example, if the optimal number of connections has been determined to be 1.5, then n is forced to take fractional values between 1 and 3, e.g. 1, 1.25, 1.45, 2.14, 1.6, ..., as dictated by (20). E-MULTFRC then quantizes n to the closest integer to oscillate between one and two, resulting in loss of throughput granularity. This effect can be eliminated by avoiding quantization.

The second drawback of E-MULTFRC is of a more practical nature. Operating multiple connections in one application could potentially consume too much system resources. For example, each TFRC connection uses a different port to send out data packets, carries out individual feedback process, and updates the loss event rate and RTT even though they are highly correlated for these TFRC connections. Clearly, there is unnecessary overhead associated with operating multiple connections, in terms of computation, processing power, memory, and ports, particularly for today's low power, resource-limited handheld devices.

We can design an alternative to E-MULTFRC, namely EAIO-TFRC, in order to address the two drawbacks of E-MULTFRC, while retaining the same control law for n as in E-MULTFRC [14]. EAIO-TFRC achieves these goals by creating one connection whose throughput is equivalent to that of the optimal number of TFRC connections even though the optimal number could be non-integer. It does so by measuring round trip times, adjusting the number of virtual connections n based on the measurements according to (20), and then controlling the sending rate of only one connection to be n times that of one TFRC's, using the Bandwidth Filtered Loss Detection (BFLD) technique from [29]. NS-2 simulations show that EAIO-TFRC achieves similar throughput as E-MULTFRC in high packet loss rate situation, but better throughput than E-MULTFRC at the low packet loss rate scenarios, as shown in Fig. 6. For example, when $p_w = 0.02$, EAIO-TFRC achieves 92% utilization of the wireless bandwidth, while E-MULTFRC's utilization is only 81%. Therefore, by avoiding the "quantization effect", EAIO-TFRC achieves better

throughput performance than E-MULTFRC.

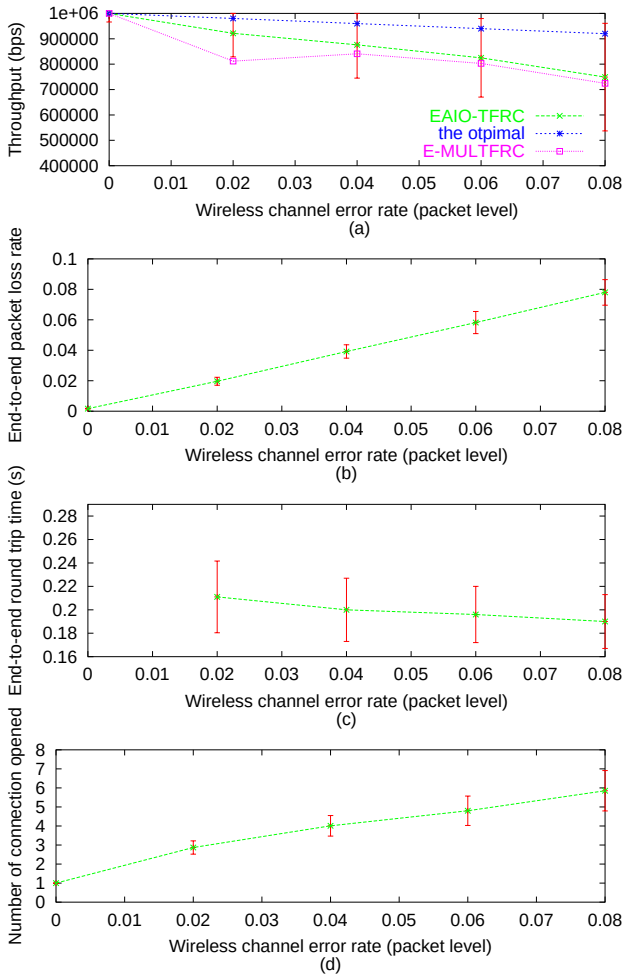

Fig. 6. NS-2 simulations for Bw = 1 Mbps and RTTmin = 168 ms; (a) throughput, (b) end-to-end packet loss rate, (c) end-to-end RTT, (d) number of connections, all as a function of packet error rate on the wireless channel.

EAIO-TFRC is fundamentally similar to E-MULTFRC as they share the same theoretical analysis and design insights but only differ in implementation details. Implementation details of EAIO-TFRC are similar to AIO-TFRC shown in [14].

D. Fairness between E-MULTFRC and TCP

To investigate the fairness of E-MULTFRC, we carry out NS-2 simulations based on the "dumbbell" topology shown in Fig. 7. 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-MULTFRC and TCP, and the intra-protocol fairness within E-MULTFRC.

The intra-protocol fairness is defined as the fairness between E-MULTFRC flows. In our simulations, we run E-MULTFRC on all 16 sender-receiver pairs shown in Fig. 7 for 5000 seconds, and compare their throughput. E-MULTFRC 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

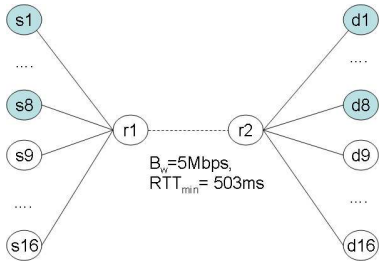

Fig. 7. The simulation topology for E-MULTFRC's fairness evaluation.

as receivers' throughput divided by the average throughput; the closer to one, the more fair the E-MULTFRC system is. As seen, the fairness ratio is fairly close to one, indicating E-MULTFRC 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-MULTFRC.

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	1.08	1.03	d9	1.04	0.98
d2	0.99	1.01	d10	0.98	0.98
d3	1.05	1.03	d11	1.03	1.02
d4	1.01	1.08	d12	1.03	0.99
d5	0.91	0.98	d13	1.00	0.97
d6	0.92	0.98	d14	0.90	1.00
d7	1.02	1.04	d15	1.02	1.01
d8	0.98	1.01	d16	1.00	0.94

The inter-protocol fairness is defined as the fairness between E-MULTFRC and TCP. In our simulations, we run E-MULTFRC 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. 7; each session lasts 5000 seconds, and we compare their throughput for $p_w = 0.01$ and $p_w = 0.02$. Under the simulation settings, each E-MULTFRC consumes more bandwidth than one TCP under full utilization. This is because in this case, the wireless channel error rate is large enough to make the number of virtual connections of each E-MULTFRC to be larger than one. Hence, it is meaningless to define the fairness between E-MULTFRC and TCP as having the same throughput.[6] As such, in our simulations, we define E-MULTFRC to be fair to TCP if it does not result in a decrease in TCP's throughput. Specifically for our simulations, it implies TCP retains the same throughput whether or not it coexists with E-MULTFRC under the same network setting. The throughput of E-MULTFRC and TCP, as well as the total bandwidth utilization ratios for the setup shown in Fig. 7, are shown in Table IV for two scenarios: (a) 8 E-MULTFRC coexisting with 8 TCP connections, (b) 16 TCP connections. Figs. 8 and 9 also show the dynamics of throughput, packet loss rate, RTT, and the number of virtual connections n.

[6]Obviously, there are situations in which E-MULTFRC ends up with performing similar to one TFRC. An example would be E-MULTFRC competing for bandwidth with TCP on wireline networks. In that case, however, the fairness between E-MULTFRC and TCP is reduced to the fairness between TFRC and TCP, and has been well explored in [2].

Comparing E-MULTFRC+TCP with TCP-alone in Table IV, we see the former achieves a much higher utilization of the wireless bandwidth at the expense of lower TCP throughput. A careful examination of Fig. 8 reveals that this throughput drop is caused by the higher RTT experienced by TCP connections in the E-MULTFRC+TCP scenario as compared with TCP-alone scenario. For example, for $p_w = 0.01$ and $\gamma = 0.2$, E-MULTFRC+TCP experiences around 0.55 seconds RTT, while TCP-alone only experiences 0.5 seconds RTT, i.e. the propagation delay[7]. As TCP's throughput is known to be inversely proportional to RTT, the 10% increase in the RTT of TCP connections roughly explains the 13% decrease in the TCP's throughput shown in first row in Table IV.

TABLE IV

SIMULATION RESULTS FOR FAIRNESS BETWEEN E-MULTFRC AND TCP.

settings	8 E-MULTFRC + 8 TCP			16 TCP	
	ave. thput. (E-MULTFRC) (kbps)	ave. thput. (TCP) (kbps)	utilization (%)	ave. thput. (TCP) (kbps)	utilization (%)
p_w=0.01 γ=0.2	436.59	176.67	99	200.168	65
p_w=0.01 γ=0.1	408.84	188.40	97	200.168	65
p_w=0.02 γ=0.2	472.52	127.81	98	139.674	46
p_w=0.02 γ=0.1	444.90	134.64	93	139.674	46

This increase in the RTT experienced by TCP is, by design, a consequence of E-MULTFRC controlling n according to (20). As n is only decreased after the queuing delay exceeds the threshold γrtt_min, round trip time is increased when E-MULTFRC 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 results in lower bandwidth utilization due to increased sensitivity of E-MULTFRC to RTT measurements. As shown in Table IV, $\gamma = 0.1$ results in a smaller drop in the TCP's throughput than $\gamma = 0.2$.

VI. DISCUSSION AND FUTURE WORK

In this paper we have shown that flow control in the wireless network can be formulated as the same concave optimization problem defined by Kelly et. al. in the wireline network, with the exception of an inaccurate feedback price, resulting from packet loss caused by physical channel error. All existing approaches to this problem correct the inaccurate feedback by casting modifications to existing protocols, such as TCP, or infrastructure elements such as routers, thereby making them hard to deploy in practice.

We have formulated the problem as another concave optimization problem, of which Kelly's optimization problem can be shown to be a special case. This reformulation results in a new class of end-to-end based solutions, in which an appropriate number of connections are opened at the application layer by the sender. The solutions require only one bit information on whether or not the route is congested, making it easy to estimate accurately at the application layer. Hence no modification to either existing protocols, e.g. TCP, or infrastructure, e.g. router, is expected. We have shown that our proposed scheme has a unique stable equilibrium that solves the concave optimization problem, implying the scalability and optimality of the solution. Specifically, the unique optimal equilibrium is locally exponentially stable in general, and is globally asymptotically stable in the case there is only one bottleneck in the network.

To demonstrate the power of our proposed solution, we have developed a practical scheme called E-MULTFRC for streaming over the wireless network. Its efficient performance, and fairness to both itself and TCP are characterized and evaluated using both NS-2 simulations and 1xRTT wireless experiments. Analysis and simulation results also indicate our scheme in fact works in both wireline and wireless scenarios. An alternative scheme EAIO-TFRC, which overcomes the undesirable consequence of quantization of the number of connections is also investigated and evaluated by NS-2 simulations. This scheme is shown to only differ from E-MULTFRC in implementation details.

E-MULTFRC is different from existing schemes such as [8]

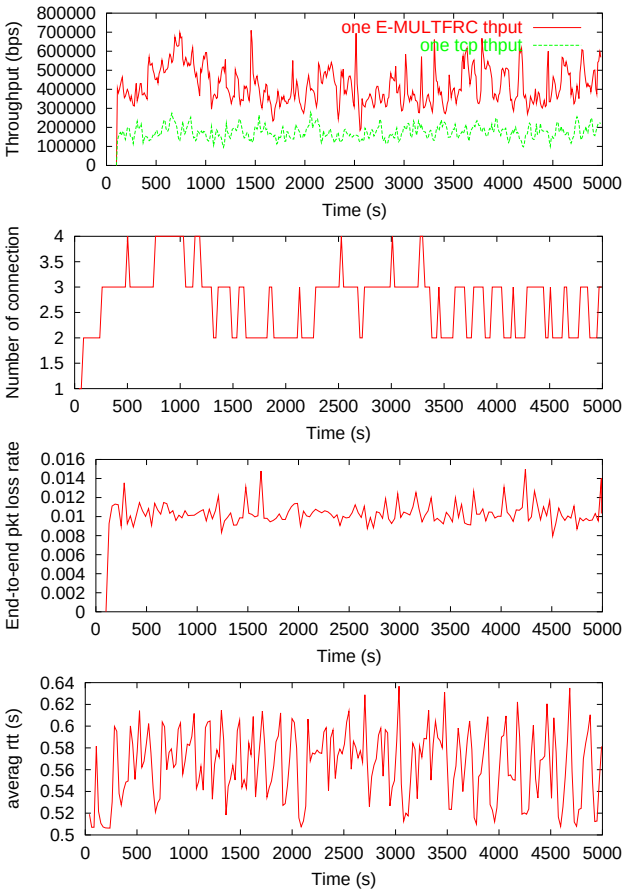

Fig. 8. NS-2 simulation results for the case $p_w = 0.01, \gamma = 0.2$ for E-MULTFRC+TCP scenario: the dynamics of (a)throughput, (b) number of connections , (c) end-to-end packet loss rate, (d) end-to-end RTT, all as a function of time.

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

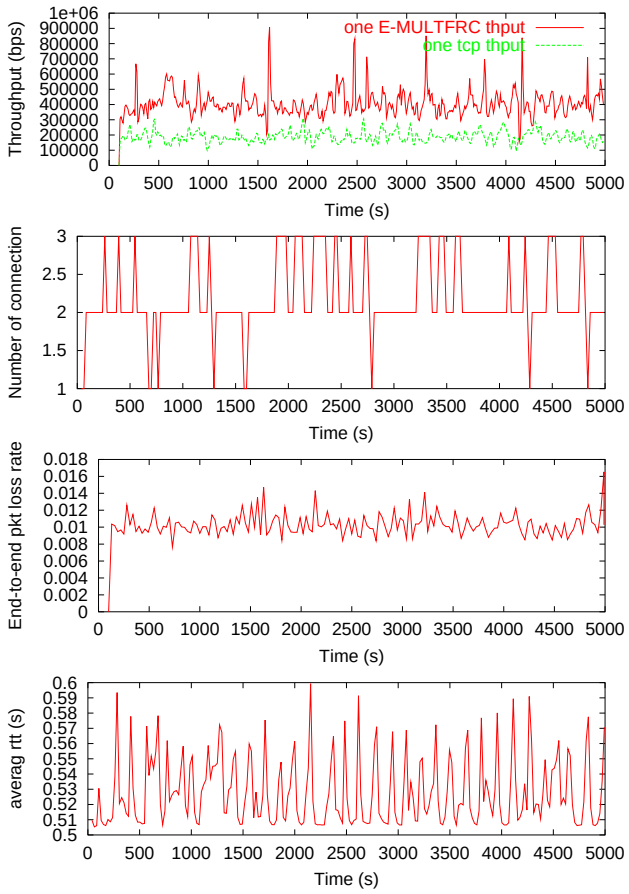

Fig. 9. NS-2 simulation results for the case $p_w = 0.01, \gamma = 0.1$ for E-MULTFRC+TCP scenario: (a)throughput, (b) number of connections , (c) end-to-end packet loss rate, (d) end-to-end RTT, all as a function of time.

that use delay or round trip time variation to infer congestions in several ways.: first E-MULTFRC measures the round trip time variations over a large time window, while the others measure the round trip time variations instantaneously, resulting in a more noisy measurement; second, E-MULTFRC uses the measured variation to adapt the number of opened connections in order to adapt the rate, while the existing schemes use it to differentiate between congestion loss and wireless loss.

There are many interesting and important directions for future research. Global stability, stability in the presence of delay and noisy disturbance are of great interest, from both control and the networking points of view. It is also possible to design a practical scheme, similar to E-MULTFRC, in order to improve data transmission over wireless using TCP. Last but not the least, it is interesting to examine whether it is possible to use a different utility maximization problem that leads to some other fundamentally different solution for the TCP/TFRC over wireless network problem.

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] 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*, pp. 237–252, 1998.

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

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

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

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

[8] 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.

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

[10] 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.

[11] 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.

[12] L. S. Brakmo and L. L. Peterson, "Tcp vegas: end-to-end congestion avoidance on a global internet," *IEEE J. Select. Areas Commun.*, no. 8, pp. 1465–1480, Oct. 1995.

[13] S. Floyd, "Tcp and explicit congestion notification," *ACM Computer Communication Review*, pp. 10–23, Oct. 1994.

[14] 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*, June 2005.

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

[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. Automat. Contr.*, no. 6, pp. 882–894, June 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. Networking*, no. 5, pp. 556 – 567, Oct. 2001.

[24] S. Sastry, *Non Linear Systems, Analysis, Stability and Control*. New York, NY: Springer Verlag, 1999.

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

[26] M. Chen and A. Zakhor. (2005, June) Proofs of the three theorems. Technical report of EECS Department, University of California at Berkeley. [Online]. Available: http://www-video.eecs.berkeley.edu/∼minghua/papers/proof.infocom06.pdf

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

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

[29] D. E. Ott, T. Sparks, and K. Mayer-Patel, "Aggregate congestion control for distributed multimedia applications," in *Proc. IEEE INFOCOM*, Hongkong, China, Mar. 2004.