Congestion Control in Wireless Networks

Technical Report

by

Hemant Kumar Rath
Roll No: 04407001

under the guidance of

Prof Anirudha Sahoo

Department of Electrical Engineering
Indian Institute of Technology, Bombay
Mumbai - 400076
January 2005

Acknowledgment

I would like to express my deep sense of gratitude to Prof. Anirudha Sahoo for giving me an opportunity to work in this area and his motivation and fresh ideas he provides during the course of this project. Also, I thank Prof. S A Soman, Mr. Ashutosh Gore for their inputs during the compilation of this technical report.

Hemant Kumar Rath
Roll No.: 04407001

Abstract

In this technical report we discuss a cross layer congestion control technique of TCP Reno-2 in wireless networks, in which TCP layer and MAC layer jointly control congestion. The MAC layer changes transmission power as per the channel condition, whereas the TCP layer controls congestion using Reno-2 windowing flow control. We formulate congestion control mechanism of TCP Reno-2 in terms of control system equations with a feedback system model. We use transmission power of wireless nodes as a function of cost and derive a utility function. Using optimization techniques, we determine the maximum aggregate utility (and hence, throughput) of all the sources, subject to link capacity constraints and maximum transmission power of wireless nodes. Our simulations show that power control using the optimization technique provides performance improvement in terms of throughput and window size variations.

Contents

List of Figures

Chapter 1

Introduction

Wireless networks [1] are inherently limited by battery power and bandwidth constraints. They are characterized by mobility, random changes in connectivity, fluctuations in channel and interference due to neighboring nodes etc. Due to these factors, packet loss of a wireless network is much more than that of a wired network, in which packet loss occurs mainly due to congestion in the network.

Congestion in a network is characterized by delay and packet loss in the network. Transport Control Protocol (TCP) is used as a reliable transport layer protocol in the traditional best effort (wired) network and deals with congestion effectively. The congestion control mechanism of various versions of TCP provides better throughput in an wired network, where the packet loss is mainly due to congestion at various nodes and routers. However, this mechanism may not be suitable in a wireless network, where packet loss due to time-varying nature of channel and interference of other nodes are considerably high.

Hence, instead of usual congestion control technique, we propose a cross layer technique involving TCP and MAC (Medium Access Control) layer. TCP layer performs the windowing flow control and MAC layer varies transmission power of wireless nodes depending on the channel condition and interference. Our approach consists of:

1. Formulation of TCP congestion control mechanism in terms of control system equations.
2. Use of transmission power of wireless nodes as a function of cost in an optimization equation.
3. Use of optimization techniques to determine the maximum aggregate utility of all the sources, subject to capacity constraints and maximum transmission power [2] of wireless nodes.

1.1 Organization of the Report

In this technical report, we apply our technique to TCP Reno-2 [3]. This technical report is organized as follows. In Chapter 2, we discuss various congestion control techniques of TCP for a wired network. We discuss the difference between TCP Reno and Reno-2 congestion control techniques and the need of TCP Reno-2 for congestion control in a wireless network. In Section 2.3, we discuss various related works on congestion control in wired and wireless network and the need of our cross layer approach for TCP Reno-2 in a wireless network. In Chapter 3, we discuss the system model based on control system equations for a wired network, its modification for a wireless network and the utility function for TCP Reno-2 traffic. Further, we discuss our Cross Layer implementation frame work, pricing in Reno-2 and implementation algorithm in Chapter 4. Our Simulation and Results are discussed in Chapter 5. Finally, we present the future work and weakness of our cross layer algorithm in Chapter 6.

Chapter 2

Congestion Control

When the aggregate demand for resources (e.g., bandwidth) exceeds the capacity of the link, congestion results. Congestion is characterized by delay and loss of packets in delivery. In TCP, congestion is said to have occurred when the sender receives three duplicate acknowledgments (*dupacks*) or when a timeout (packet loss) occurs, resulting in wastage of resources. *Congestion Control* and *Congestion Avoidance* are two known solutions which address the above problem. In congestion control [4], system controls the network parameters after realizing congestion (reactive); whereas, in congestion avoidance, system controls the network parameters before congestion (proactive).

2.1 Congestion Control in Wired Network

Most of the traffic (around 80%) in the Internet are TCP traffic [4]. TCP's congestion control in wired network is based on *Adaptive Window Management* technique. In this technique, congestion window (*cwnd*) increases or decreases based on packet drops and *dupacks*. Different versions of TCP uses different method to increase/decrease *cwnd* during congestion and are discussed below. (i) *TCP Tahoe*: Slow Start and Congestion Avoidance, (ii) *TCP Reno and Reno-2*: Fast Retransmit and Fast Recovery, and (iii) *TCP Vegas*.

2.1.1 TCP Tahoe

TCP Tahoe [5] assumes losses due to packet corruption are much less probable than losses due to buffer overflows in the network resulting congestion. It uses tripple *dupacks* or timeouts to detect congestion or packet loss in the network. It decreases *cwnd* size to one (from the current *cwnd* size) after detecting a congestion and then increases *cwnd* size from one to *ssthresh* (which acts as limit point for exponential increase and is set to half of the *cwnd* size before experiencing a congestion) an exponential manner in each Round Trip Time (RTT). After reaching *ssthresh* it increases *cwnd* linearly in each RTT till next congestion occurs.

2.1.2 TCP Reno and Reno-2

Unlike TCP Tahoe, TCP Reno [6] distinguishes between triple *dupacks* and packet loss (timeout). On packet loss it works similar to TCP Tahoe. But, for a triple *dupacks*, instead of declaring it as a packet loss and entering the slow start process, it follows fast recovery technique and decreases the *cwnd* value by half of the current *cwnd* and then increases linearly till experiencing a congestion. Though it is better than TCP Tahoe for dealing with single packet loss and *dupacks*, it is not good when multiple packets are lost within one RTT. This problem is solved in its newer version called TCP Reno-2.

In TCP Reno-2 [3], the *cwnd* value is not decremented for every packet loss, rather is decremented in an intelligent manner. *cwnd* is decremented by half only when one or more than one packet loss occurs in an RTT. In this technical report, we study congestion control in TCP Reno-2 in wireless network.

2.1.3 TCP Vegas

In TCP Vegas [7], *cwnd* size is increased or decreased depending on the difference of ratio of current window size, propagation delay and queuing delay. This congestion control mechanism is similar to a sliding window protocol.

2.2 Congestion Control in Wireless Network

The adaptive window based congestion control mechanism used by TCP for wired network may not be appropriate for wireless network. This is due to the time varying nature of an wireless channel and interference due to other nodes causing packet loss, which is different from packet loss due to congestion. But, TCP's congestion control mechanism does not discriminate packet loss due to congestion and that due to bad channel or interference; rather apply the same congestion control mechanism for both. So, there must be some mechanism to tackle the problem of packet losses due to bad channel condition and interference, such that we can extend the congestion control mechanism of wired network to wireless network.

2.3 Related Work

TCP's congestion control mechanism has evolved as a reliable mechanism over the years. In 1988, Van Jacobson proposed TCP Tahoe [5] as a congestion control mechanism. After a couple of years again Van Jacobson and his group proposed another congestion control mechanism known as TCP Reno [6]. Though TCP Reno is an improved version of TCP Tahoe and is basically applicable for high speed networks and it has certain drawbacks like multiple packet loss in one RTT. In the year 1994, Lawrence et al. [7] had proposed another congestion control mechanism called TCP Vegas. This is similar to a sliding window protocol. Further, in year 1996, Mathis et al. had proposed another technique based on selective acknowledgment techniques for congestion control. Though, these are the basic congestion control protocols used, various modifications were proposed by several authors over the years.

Several authors like Kelly [2] have modeled the TCP traffic as elastic traffic and discussed the charging mechanism based on aggregate throughput using the concept of utility function. They used several optimization techniques to find out the maximum aggregate throughput. Further, they have assumed the utility function ($U(x)$) of elastic traffic is concave, which was proved by Shenker et al. [8].

Seungwan et al. [4], discussed the advances in Internet congestion control using control-theoretic analysis, pricing based approach and optimization based approach. Steven Low et al. have published several papers on control-theoretic analysis of congestion control. They have discussed various utility functions in [9], [10] for various versions of TCP and analyzed those in terms of stability, feedback control, maximization of utility function etc.

Efforts were made by various authors to use the proven congestion control techniques of TCP in a wireless network. Chiang and Man [11] used power control along with congestion control in a wireless network and analyzed it for TCP Vegas. This is one of the new approaches for congestion control in wireless network scenario.

2.4 Contribution

We have extended the original approach of [11] for TCP Reno-2 and used maximum transmission power as a constraint in the optimization phase. Further, we have derived the utility function for Reno-2 in a wireless network with some practical assumptions and have simplified the cost function for wireless network for TCP Reno-2 by an $M/M/1/B$ queue model. The congestion control, which is a window based one, is modelled as rate based optimization problem, for simplicity. Ours is a cross layer approach and is similar to that used in [11]. In cross layer approach, the TCP layer and MAC layer jointly control congestion. The MAC layer changes transmission power as per the channel condition and interference received from the neighboring nodes, whereas the TCP layer controls congestion using Reno-2 windowing flow control. Simulations are used to verify the cross layer congestion control technique for TCP Reno-2 in a wireless network.

Chapter 3

Problem Formulation

We have formulated our problem statement initially for a wired network, similar to [9] [10] and then extended it to a wireless network. Our system model is based on fluid-flow abstraction model. We consider long TCP connections ("elephants") instead of short ones ("mices") in our model, as the former generate most of the TCP traffic in the Internet.

3.1 System Model

We consider a feedback system model for analyzing TCP congestion, shown in Fig. 3.1, which consists of a Source Controller and a Link Controller. The Source Controller controls the transmission rate (x_i) of the sources based on the changes in link prices (λ_i) due to congestion and interference caused in a wireless network. The Link Controllers controls the link prices based on the changes in the aggregate traffic (y_i) in a link. Source Controller acts on the basis of feedback it receives from the Link Controller, whereas Link Controller acts on the basis of the feedback it receives from the Source Controller in a closed loop. This is explained in Fig. 3.1.

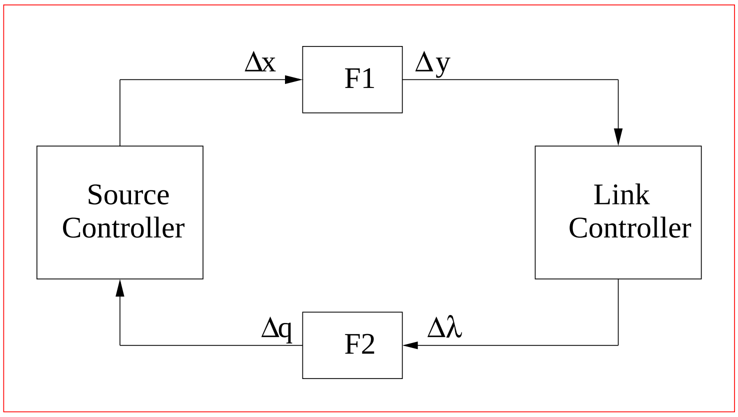

Figure 3.1: Feedback System Model

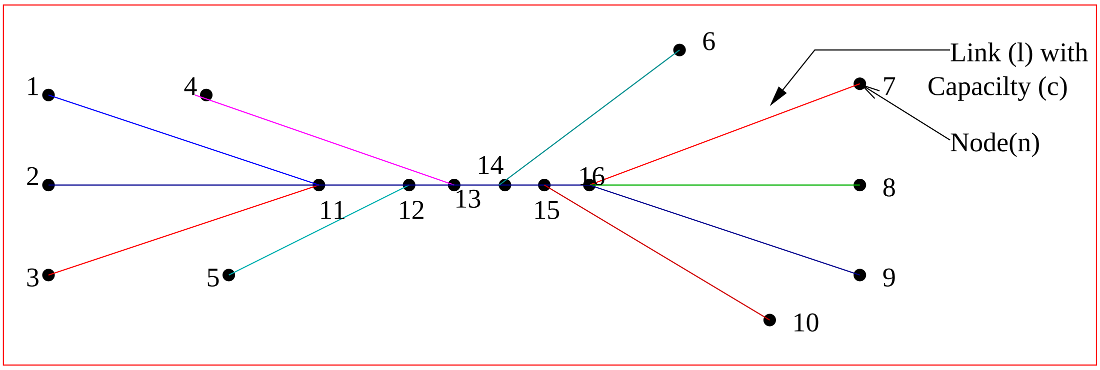

Figure 3.2: System Model

The network in our discussion consists of N communicating nodes connected by L communicating links, similar to [10]. The communicating links are indexed by $l \in L$, and have finite capacity c_l packets/sec. These communicating links are shared by a set of source-sink pairs indexed by $i \in I$, is the set of all source-sink combinations (Fig. 3.2). Now, we write the routing matrix R as:

$$R_{li} = \begin{cases} 1 & \text{if, source-sink pair } i \text{ uses the link } l, \\ 0 & \text{otherwise.} \end{cases} \tag{3.1}$$

Each source-sink pair i has an associated transmission rate $x_i(t)$, i.e., the transmission rate of an individual TCP flow for a source-sink pair (packets/sec). The aggregate flow $y_l(t)$ (packets/sec) at each link is defined as:

$$y_l(t) = \sum_i R_{li} x_i(t - \tau_{li}^f), \tag{3.2}$$

where τ_{li}^f is the forward delay between the source and the link (between the Source and Link Controllers through F1 in Fig. 3.1). The Source Controller controls the transmission rate x_i based on the change in price (due to change in aggregate losses ∇q at F2 in Fig. 3.1). The time scale of change of routing is higher than the time scale of our analysis, which is a multiple of RTT (τ_i). Hence, we assume stationary routing.

If each link l is associated with price $\lambda_l(t)$, as a congestion measure, then the aggregate price[1‡] of all links in the route is:

$$q_i(t) = \sum_l R_{li} \lambda_l(t - \tau_{li}^b), \tag{3.3}$$

where τ_{li}^b is the backward delay in the feedback path (of the acknowledgment packets between the link and the source) between the Link and Source Controllers through F2 in Fig. 3.1. The Link Controller controls the individual link prices ($\lambda_l(t)$) based on the change

[1‡] Sources are assumed to have an access to the aggregate price of all links in their route

in aggregate traffic (∇y) in the link at F1 in Fig. 3.1. We write the RTT as $\tau_i = \tau_{li}^f + \tau_{li}^b$. In vector form we write Eqns. (3.2) and (3.3) as:

$$y = Rx; \quad q = R^T \lambda \tag{3.4}$$

3.1.1 Use of Control System Equations

We use control system equations as in [10] to determine the equilibrium point (x^*, y^*, q^*) for our system with the following assumptions: (i) resource allocation is characterized by a demand curve $x_i^* = f_i(q_i^*)$, (ii) aggregate flow is less than or equal to the capacity of the link, i.e., $y_l^* \leq c_l$, and (iii) the equilibrium queues are either empty or small enough to avoid the queuing delays.

The demand curve specified above is a decreasing function of price, which is equivalent to use of an utility function $U_i(x_i)$ defined by Shenker et al. [8]. This utility function is a concave and continuously differentiable function of x_i, in the range of $x_i \geq 0$, i.e. $f_i = \left(U_i^{'}\right)^{-1}$ [8]. At equilibrium, a source chooses its maximum profit by choosing local parameters as:

$$\max_{x_i^*} \left[U_i(x_i^*) - q_i^* x_i^* \right], \tag{3.5}$$

where, $q^* = R^T \lambda^*$, and x^*, y^*, q^* are the optimal fixed parameters. Since at equilibrium, each source tries to maximize its profit (individual optimality) by choosing an appropriate rate, the individual optimality of Eqn. 3.5 is re-written as social optimality equation [12] as follows:

$$\begin{aligned} \max_{x>0} \sum_i U_i(x_i&), \\ \text{s. t.,} \quad RX \leq C&; \\ X = \{x_i\} \quad \text{and} \quad C = \{c_l&\} \end{aligned} \tag{3.6}$$

Using Karush-Kuhn-Tucker (KKT) condition [13] on the social optimal equation (Eqn. 3.6) as in [9], constrained global maxima point[2‡] $X^* = \{x_i^*\}$ is expressed as:

$$\nabla\big(\sum_i U(x_i)\big) - \sum_l \lambda_l \nabla(RX) = 0; \quad \forall \lambda_l > 0, \qquad l \in L, \tag{3.7}$$

where, λ_l are the Lagrangian Multipliers and are used as link prices (shadow prices) as discussed by Kelly [2] and Low et al. [14]. Hence, Eqn.3.7 is reduced to $U_i^{'}(x_i) = q_i^*$ at the global maximum point.

[2‡] Since the utility function $U(x_i)$ is concave and increasing, KKT conditions are necessary and sufficient for a global maximum

Different variants of TCP have different utility functions. The primal-dual distributed algorithm of the maximization equation (Eqn. 3.6) signifies that the price is updated as a congestion parameter and is a dual variable. We discuss the utility function of Reno-2 in Section 3.2.

3.1.2 Modification for Wireless Network Case

As discussed in [11], power transmitted by a source is considered as a function of price and by appropriate power control of all nodes maximum aggregate throughput is attained. However, the power increase in one node has a direct effect on the data rates of other nodes due to interference. Hence, we modify the maximization Eqn 3.6 as follows:

$$\begin{aligned} \max_{X \geq 0} \sum_i U_i(x_i), \\ \text{s.t.,} \quad RX \leq C(P); \quad P = \{P_l\}, \\ P_l \leq P_{l_{Max}}, \quad \forall l, \\ P, X \geq 0, \end{aligned} \tag{3.8}$$

where P_l is the transmission power of a node in l^{th} link. Here, the link capacity is a function of transmitting power P_l of the nodes. Since, power control significantly increases throughput in both CDMA based and non-CDMA based wireless networks [15, 16, 17], we use power control mechanism to achieve congestion control by the above optimization techniques. This is a cross layer [18, 19] mechanism and hence is a joint power control and congestion control mechanism.

3.2 Utility Function for Reno-2

In TCP Reno-2, the *cwnd* is halved if there is one or more mark in one RTT. We assume that the marking probability as a measure of congestion [9].

If, τ_i is the equilibrium RTT ($\tau_i = \tau_{li}^f + \tau_{li}^b$, as defined before), then, transmission rate $x_i(t)$ (in packets/second) is expressed as:

$$x_i(t) = \frac{w_i(t)}{\tau_i} \tag{3.9}$$

If there is no mark in one RTT (τ_i), then, $w_i(t+\tau_i) = w_i(t)+1$, else, $w_i(t+\tau_i) = w_i(t)/2$, for one or more mark in one RTT.

Let, t, a period for our analysis for Reno-2 and be a multiple of few RTTs. Then, the increase in window size is $1/\tau_i$ with probability $1 - \hat{q}_i(t)$ and the decrease is by $\frac{2w_i(t)}{3\tau_i}$ [3‡] with probability $\hat{q}_i(t)$, where $\hat{q}_i(t)$ is the end-to-end probability that at least one packet is

[3‡] For a single Reno-2 flow, the window oscillates between $\frac{4w_i(t)}{3\tau_i}$ and $\frac{2w_i(t)}{3\tau_i}$ with an average of $w_i(t)$ [9]

marked out of all packets of an window, in period t for the source-sink pair i. We have $\hat{q}_i(t)$ is approximated as $\hat{q}_i(t) = w_i(t)q_i(t)$, where $q_i(t)$ is the aggregate price of all links in the route defined in Eqn. 3.3.

So, the average change in window size[4‡] in the period t is:

$$\begin{aligned} &\frac{1}{\tau_i}\left(1 - \hat{q}_i(t)\right) - \frac{2}{3}\frac{w_i(t)}{\tau_i}\hat{q}_i(t) \\ &= \frac{1 - w_i(t)q_i(t)}{\tau_i} - \frac{2}{3}q_i(t)x_i(t)w_i(t) \end{aligned} \tag{3.10}$$

From the above equation (Eqn. 3.10) we determine the equilibrium parameters as follows:

$$\dot{x} = \frac{x_i(t)}{\tau_i} = \frac{1 - w_i(t)q_i(t)}{\tau_i^2} - \frac{2}{3\tau_i}q_i(t)x_i(t)w_i(t) \tag{3.11}$$

At equilibrium, setting $\dot{x} = 0$, $x_i(t) = x_i^*$ and $q_i(t) = q_i^*$ in Eqn. 3.11) we determine q_i^* as follows:

$$\begin{aligned} &\frac{1 - w_i(t)q_i(t)}{\tau_i^2} - \frac{2}{3\tau_i}q_i(t)x_i(t)w_i(t) = 0 \\ &\implies q_i^* = \frac{3}{x_i(t)\tau_i(3 + 2x_i(t)\tau_i)} \end{aligned} \tag{3.12}$$

Now, using the fact that $U_i'(x_i) = q_i^*$, (resulted from Eqn. 3.7) the utility function is expressed as:

$$U_i(x_i) = \int q_i^* dx_i = \frac{1}{\tau_i}\log\left[\frac{x_i\tau_i}{2x_i\tau_i + 3}\right] \tag{3.13}$$

Since the nature of the utility function of TCP Reno-2, derived above, is concave for $x_i, \tau_i \geq 0$, our problem formulation in Eqn. (3.5) holds good. The utility function we derived here is for an average model of TCP as described in Eqn. (3.10). This is not an attempt to impose equilibrium on the detailed evolution of TCP windowing.

[4‡] The change in window size due to time-out is neglected here

Chapter 4

Implementation

For implementation, we considered a CDMA-based wireless network. Let, $SINR_l$ be the signal-to-interference-to-noise ratio of a time varying link l, $l \in L$.

$$SINR_l = \frac{P_l G_{ll}}{\sum_{k \neq l} P_k G_{lk} + n_l}, \tag{4.1}$$

where P_l is the transmission power on the link l, by the source, G_{ll} is the path gain of the link l, G_{lk} is the path gain of a node on link k to another node on link l, and, n_l is the thermal noise on the link l. Using Shannon's capacity theorem, we determine the maximum capacity attainable in link l as:

$$c_l = \frac{1}{T} \log(1 + M.SINR_l) \qquad \text{packets/sec}, \tag{4.2}$$

where T is the symbol period and M is a constant that depends on the modulation scheme used by the node for a successful transmission.

4.1 Implementation Framework

We followed the algorithms derived in [9] and [11] for our implementation in Reno-2. We associate a Lagrangian Multiplier λ_l for the first constraint in Eqn. 3.8. Then, we solve Eqn. 3.8 using KKT [13] optimality conditions by solving the complementary slackness conditions at equilibrium. Then we determine the stationary points of the Lagrangian as:

$$\phi_{system}(X, P, \lambda) = \sum_i U_i(x_i) - \sum_l \lambda_l \Big(\sum_i R_{li} x_i - c_l(P) \Big) \tag{4.3}$$

Maximization of $\phi_{system}(X, P, \lambda)$ in Eqn. 4.3 is decomposed as in [11] and a distributed solution[1‡] is obtained as:

[1‡] Distributed solution is possible as along as there is an interaction between the two decomposed equations through some information passing (message passing in our case)

$$
\begin{aligned}
\max \quad I(X,\lambda) = \sum_i U_i(x_i) - \sum_l \lambda_l \sum_i R_{li} x_i, \\
\max \quad I(P,\lambda) = \sum_l \lambda_l c_l(P), \\
\text{s.t.,} \quad X \geq 0; \quad 0 \leq P_l \leq P_{l_{Max}}
\end{aligned}
\tag{4.4}
$$

The first maximization equation involving $I(X,\lambda)$ in Eqn. 4.4 is a direct consequence of congestion control of TCP. It is solved by the congestion control mechanism of TCP by increasing/decreasing the window size ($cwnd$) (and hence the individual data rates) for each flow. This results in individual optimality.

The second maximization equation involving $I(P,\lambda)$ in Eqn. 4.4 is solved by choosing appropriate transmission power of wireless nodes. Both $I(X,\lambda)$ and $I(P,\lambda)$ are related by a common variable λ, which plays a significant role in determining the equilibrium window size (hence the data rate) and transmission power. Any change in λ results in change in throughput and transmission power.

Without loss of generality, we assume that the symbol period T and the modulation index M of Eqn. 4.2 as unity and re-write $I(P,\lambda)$ as follows:

$$
I(P,\lambda) = \sum_l \lambda_l \log(SINR_l(P)) \tag{4.5}
$$

$I(P,\lambda)$ is a concave function of a logarithmic transformed power vector and is simplified further as:

$$
\begin{aligned}
I(P,\lambda) &= \sum_l \lambda_l \log(SINR_l(P)) \\
&= \sum_l \lambda_l \log\left(\frac{P_l G_{ll}}{\sum_{k\neq l} P_k G_{lk} + n_l}\right) \\
&= \sum_l \lambda_l \left[\log(P_l) + \log(G_{ll}) - \log\left(\sum_{k\neq l} P_k G_{lk} + n_l\right)\right]
\end{aligned}
\tag{4.6}
$$

Now, by differentiating $I(P,\lambda)$ with respect to P_l, we evaluate the l^{th} component of the gradient $\nabla I(P,\lambda)$.

$$
\nabla I(P,\lambda) = \frac{\lambda_l(t)}{P_l} - \sum_{j\neq l} \frac{\lambda_j(t) G_{jl}}{\sum_{k\neq j} P_k G_{jk} + n_j} \tag{4.7}
$$

Now, we use the Steepest Descent method as in [11] [13] to solve the maximization problem as:

$$
\begin{aligned}
P_l(t+1) &= P_l(t) + \Delta \nabla I(P,\lambda) \\
&= P_l(t) + \Delta \left(\frac{\lambda_l(t)}{P_l(t)} - \sum_{j \neq l} \frac{\lambda_j(t) G_{jl}}{\sum_{k \neq j} P_k(t) G_{jk} + n_j} \right) \\
&= P_l(t) + \Delta \frac{\lambda_l(t)}{P_l(t)} - \Delta \sum_{j \neq l} \frac{\lambda_j(t) G_{jl} SINR_j(t)}{p_j(t) G_{jj}} \\
&= P_l(t) + \Delta \frac{\lambda_l(t)}{P_l(t)} - \Delta \sum_{j \neq l} m_j(t) G_{lj},
\end{aligned} \tag{4.8}
$$

where Δ is a constant, called the step size in the direction of the gradient and $m_j(t)$ is the interference received from node j (similar to message passing in [11]) to the link l and is defined as:

$$
m_j(t) = \frac{\lambda_j(t) SINR_j(t)}{P_j(t) G_{jj}} \tag{4.9}
$$

From Eqn. 4.8, it is evident that the transmission power of a node in the next time slot $P_l(t+1)$ in a link l depends on three parameters, viz., (i) transmission power in the present time slot $P_l(t)$, (ii) shadow price $\lambda(t)$, and (iii) the weighted sum of interference received from all neighboring nodes. The third factor is responsible for decreasing the transmission power of the concerned node in the next time slot, i.e., the transmission power of the concerned node should be such that the interference caused by other nodes are below some threshold. This is known as the co-operation principle in power control of wireless network. $\lambda(t)$ is responsible for increasing power in the next time slot. Intuitively, more the shadow loss, more the congestion, more the transmission power in the next time slot. i.e., transmission power increases with respect to shadow loss. However, this increase is not linear.

In power control techniques, each wireless node needs to advertise its $SINR_l$ requirement either on a separate channel [17] or on the same channel [20]. These nodes update their path gains, noise levels, interference causes by other nodes etc., either after receiving the advertised signal or in periodic manner.

4.2 Shadow Prices in Reno-2

As discussed in Section 4.1, Lagrangian Multipliers λ_l are the shadow prices on a link. The concept of pricing is different for different schemes of TCP, viz., queuing delay in TCP Vegas and loss probability in TCP Reno and Reno-2. We focus on the pricing scheme of TCP Reno-2.

When the aggregate traffic is more than the link capacity, packets are dropped at the link when buffer overflows. This dropping phenomena is modeled as a $M/M/1/B$ queue [21], where, B is the buffer size at the link. In this kind of queuing model, y_l and c_l

act as average arrival rate and average service rate [2‡] respectively. Here, the states are restricted and don't grow to infinity [22]. The loss probability (p_l) for this kind of queue at equilibrium is defined as:

$$p_l(c_l, y_l) = \frac{(1-\rho)\rho^B}{(1-\rho^{B+1})}, \tag{4.10}$$

where $\rho := \frac{y_l}{c_l}$. When the arrival rate and buffer capacity is scaled by a factor K as in a many-sources large-deviation scaling and by taking $K \to \infty$ we find out the loss probability as in [21]:

$$\begin{aligned}
\lim_{K\to\infty} p_l(c_l, y_l) &= \lim_{K\to\infty} \frac{(1-\rho)\rho^{KB}}{1-\rho^{KB+1}} \\
&= \lim_{K\to\infty} \left[1 - \frac{1-\rho^{KB}}{1-\rho KB+1}\right] \\
&= \frac{(y_l - c_l)^+}{y_l} = \frac{\max\big(0, (y_l - c_l)\big)}{y_l},
\end{aligned} \tag{4.11}$$

This is valid only for $y_l > 0$. When $y_l = 0$, $p_l(c_l, y_l) = 0$. Since the loss probability in TCP Reno-2 is considered as the price, the price λ_l is expressed as:

$$\lambda_l(t) = p_l(c_l, y_l) = \begin{cases} \frac{\max\big(0, (y_l(t) - c_l(t))\big)}{y_l(t)} & \text{if } y_l > 0, \\ 0 & \text{if } y_l = 0. \end{cases} \tag{4.12}$$

As discussed in [21], price in our discussion is equivalent to the fraction of fluid lost when the arrival rate exceeds the capacity of a link. This leads to a resulting equilibrium which is maximized as follows:

$$\max_{x>0} \sum_i U_i(x_i) - \sum_i \int_0^{y_l} \lambda_l dy, \tag{4.13}$$

which is similar to Eqn. 3.6 as shown in [21, 23].

4.3 Algorithm

Our cross layer congestion control algorithm based on joint power control in MAC layer and congestion control in TCP layer for Reno-2 is explained in Algorithm 1.

In our algorithm, the initial data rate $x_{initial}$, initial power $P_{l_{Min}}$, minimum $SINR_l$ and δ are configuration parameters. The frequency of updating of $SINR_l$ is also a configuration parameter. Usually the updation is carried out at each nodes periodically. Data rate x_i is calculated from Eqn. 3.9, whereas w_i and τ_i are decided using Reno-2 congestion control principle.

[2‡] The maximum capacity of the link is c_l and hence the service rate

Algorithm 1 : Our Cross Layer Congestion Control Algorithm for Reno-2

```
1: Set initial transmission rate x_i = x_initial
2: Initialize P_l = P_{l_Min}
3: Advertise the minimum SINR_l required
4: Update G_lj and G_jj periodically or after receiving the advertised signals
5: Determine maximum capacity of the link using Eqn. 4.2
6: Determine λ_l(t) using Eqn. 4.12
7: Determine m_j(t) using Eqn. 4.9
8: Calculate P_l(t+1) using Eqn. 4.8
9: if |P_l(t+1) − P_l(t)| ≤ δ then
10:    Continue transmission at P_l(t)
11: else
12:    Transmit at min(P_l(t+1), P_{l_Max})
13: end if
14: Change w_i according to the congestion control algorithm of Reno-2 (which changes x_i
    according to Eqn. 3.9
15: Update SINR_l at each node Eqn. 4.1 and go to Step 3
```

Chapter 5

Numerical Results

5.1 Simulation

We considered a simple topology with 6 wireless nodes and two pairs of TCP transmitters and receivers as shown in Fig. 5.1 for our simulations. The two pairs of transmitters and receivers in our simulation are (1-5) and (2-6). All nodes are TCP Reno-2 agents and their transmitted powers are updated in a time frame depending on the loss probabilities. Since, the RTTs are very small in our case (nodes placed are very near to each other), we have used the TCP timeout as a multiple of four RTTs. We update RTTs using following exponential averaging technique.

$$RTT = \alpha RTT_{estimated} + (1 - \alpha) RTT_{measured} \tag{5.1}$$

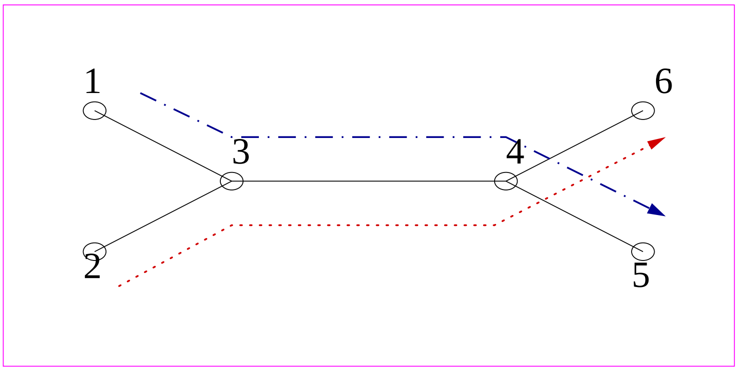

Figure 5.1: Topology for Simulation

The value of α is taken as 0.85. Also, for simplicity, we assumed that the time required for transmission in each of the segments 1-3, 2-3, 3-4, 4-5 and 4-6 are same and both forward and reverse channel characteristics are same. The channel gains are randomly chosen with a log-normal shadowing of $\sigma = 8dB$. The path loss factor γ is assumed to be 4. We used Matlab software [24] for our simulations.

5.2 Results

We have simulated TCP Reno-2 congestion control mechanism using both power control based on the $SINR$ values and without power control. In the latter case, the transmission power of nodes are fixed at some maximum value. But, in the former case, the transmission power of nodes are limited to some minimum required power level depending on the channel condition and interference. Fig. 5.2 shows the *cwnd* variation of joint power and congestion control mechanism, whereas Fig. 5.3 shows *cwnd* variation without power control. We observe that the fluctuations in *cwnd* with power control mechanism is lower as compared to the fluctuations in *cwnd* without power control mechanism. Also, the average window size of joint power and congestion control scheme is larger than that of congestion control without power control. Hence, power control provides stabilized throughput. Intuitively, this occurs because the maximization of utility function with power control Eqn. 3.8 is done over a larger set of constraints than without power control Eqn. 3.6.

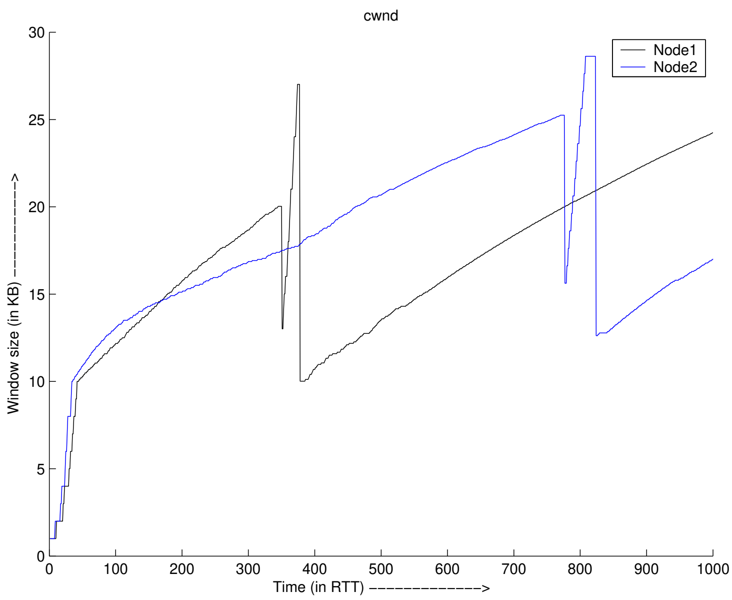

Figure 5.2: Variation of Congestion window - with Power Control

The transmission power of all Reno-2 agents are shown in Fig. 5.6 (with power control). We observe that the transmission power is a function of packet loss and hence is a function of $SINR$. The power consumption of nodes in our cross layer scheme are lower as compared to the fixed power scheme. Further, we have analyzed the pricing mechanism for both fixed and power control schemes in Fig. 5.4 and 5.5. The price in Reno-2 is a function of packet loss and hence is a function of congestion window. Price rises at the point of congestion (e.g., in Fig. 5.3 and Fig. 5.5 at $t = 220, 250, 400, 550, 650, 900$ and 950, *cwnd* is at the peak and hence the price is also at the peak, whereas immediately after the peaks, the *cwnd* decreases by half of previous *cwnd* and hence the price also decreases) and falls after congestion control (by decreasing *cwnd*).

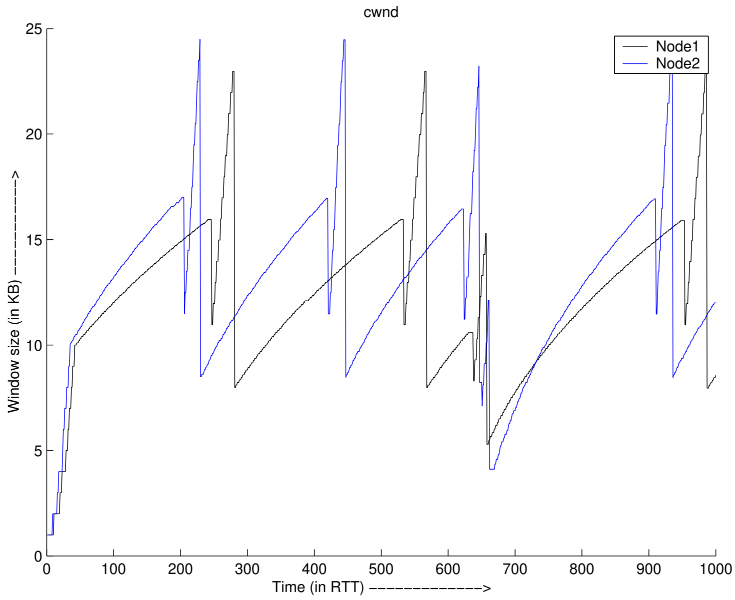

Figure 5.3: Variation of Congestion window - without Power Control

5.2.1 Convergence Criteria

In TCP Reno-2, probability of marking is considered as a measure of shadow price [25] and is controlled by power control in the wireless channel. Also, as defined in Eqn. 4.4, the transmitted power is bounded by minimum and maximum power levels. So, use of Steepest Descent method of optimization should converge for a small value of step size Δ.

We perform simulations with three different values of Δ (0.1, 0.2 and 0.5) and observe the number of iterations to converge to an optimum power transmission value in all three cases at different $SINR$ level. For, $\Delta = 0.1$, it takes about 150 iterations to converge, (but, converges to a stable transmission power level), which becomes constant over time for a particular $SINR$. For higher value of Δ though it takes fewer number of iterations to converge initially, but does not converge to a stable transmission power level.

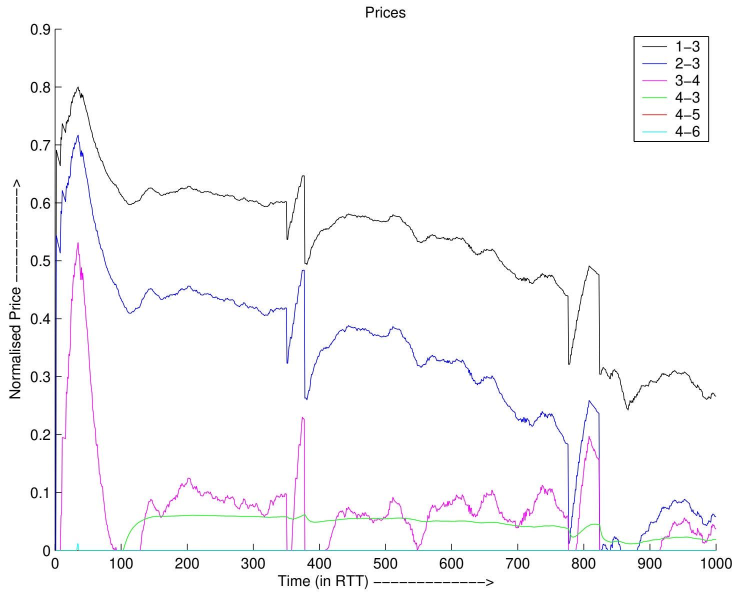

Figure 5.4: Variation of Price - with Power Control

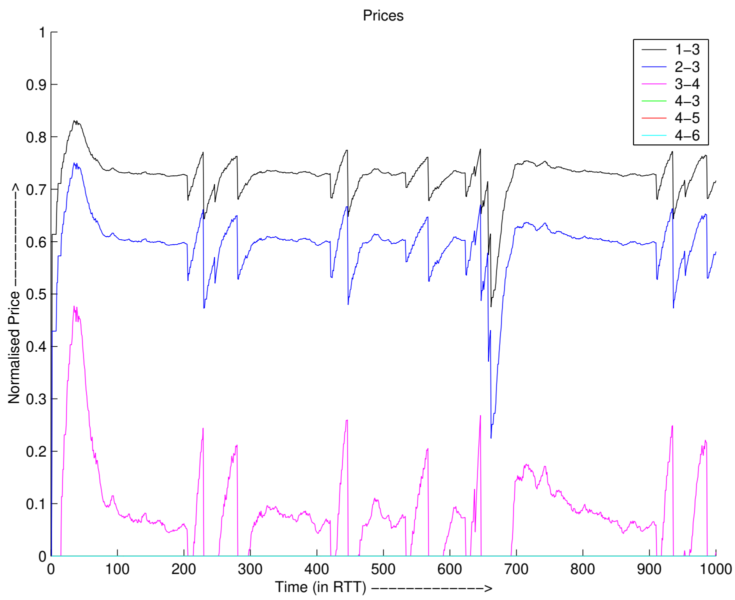

Figure 5.5: Variation of Price - without Power Control

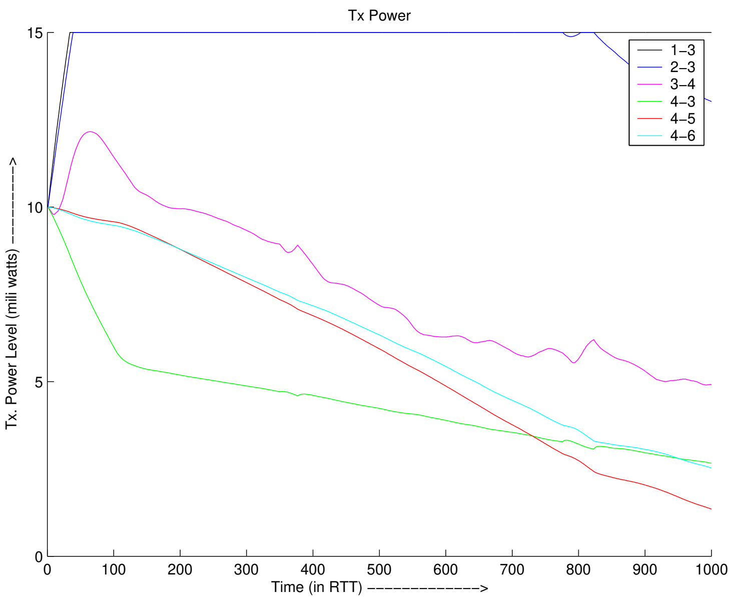

Figure 5.6: Transmission Power - with Power Control

Chapter 6

Discussion and Conclusion

Our cross layer congestion control technique for Reno-2 converges very fast. This is due to the small step size (0.1) of Steepest Descent method in Eqn. 4.8, and fixed maximum and minimum transmitting powers for each node. Our simulations verify the theoretical models we have discussed, which is a maximization problem of an utility function. As expected, the cross layer congestion control technique provides stabilized throughput at low power transmission.

But, if the channel conditions are very bad, then there would be more losses due to bad channel resulting a significant increase in λ, which in turn results in an increase in power transmission. In that case, our power control algorithm does not converge. This is a drawback of the joint power and congestion control algorithm. This algorithm holds good as long as the minimum $SINR$ is maintained at the nodes.

We have considered a simple topology for our simulation. A complex topology can be used to study other issues. Also, use of joint power and congestion control algorithm in bad channel condition needs some modification in the definition of packet loss and congestion. This modification can significantly increase the throughput.

Bibliography

[1] J. H. Schiller, *Mobile Communications*. Addison-Wesley, 2000.

[2] F. P. Kelly, "Charging and rate control for elastic traffic," *European Transactions on Telecommunications*, vol. 8, pp. 33–37, January 1997. Corrected version of the original paper is available at http://www.statslab.cam.ac.uk/∼frank/elastic.ps.

[3] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "TCP Selective Acknowledgment Options," vol. RFC-2018, April 1996.

[4] S. Ryu, C. Rump, and C. Qiao, "Advances in Internet congestion control," *IEEE Communications Surveys and Tutorials*, vol. 3, pp. 28–39, 2003.

[5] V. Jacobson, "Congestion avoidance and control," in *ACM SIGCOMM '88*, (Stanford, CA), pp. 314–329, August 1988.

[6] R. Braden, V. Jacobson, and L. Zhang, "TCP extensions for highspeed paths," vol. RFC-1185, October 1990.

[7] L. S. Brakmo, S. W. O'Malley, and L. L. Peterson, "TCP vegas: New techniques for congestion detection and avoidance," in *Proc. of IEEE SIGCOMM*, pp. 24–35, 1994.

[8] S. Shenker, "Fundamental design issues for the future internet," *IEEE Journal on Selected Areas in Communication*, vol. 13, September 1995.

[9] S. H. Low, "A duality model of TCP and queue management algorithms," *IEEE/ACM Transactions on Networking*, vol. 4, pp. 526–536, August 2003.

[10] S. H. Low, F. Paganini, and J. C. Doyle, "Internet Congestion Control," *IEEE Control Systems Magazine*, pp. 28–43, February 2002.

[11] M. Chiang and R. Man, "Jointly optimal congestion control and power control in wireless multihop networks," in *Proc. of GLOBECOMM*, 2003.

[12] C. Courcoubetis and R. Weber, *Pricing Communication Networks: Economics, Technology and Modelling*. West Sussex, England: John Wiley and Sons, 2003.

[13] D. P. Bertsekas, *Nonlinear Programming*. Athena Scientific, 2nd ed., 1999.

[14] F. Paganini, Z. Wang, J. C. Doyle, and S. H. Low, "Congestion control for high performance, stability and fairness in general networks." http://www.ee.ucla.edu/∼paganini/PDF/Papers/fast-jour.pdf.

[15] J. Monks, V. Bharghavan, and W. Hwu, "A power controlled multiple access protocol for wireless packet networks," in *Proc. of IEEE INFOCOM*, pp. 219–228, 2001.

[16] M. Singh, P. Sahni, J. Vyas, K. Paul, and A. Karandikar, "Interference Based Multiple Access (IBMA) protocol for CDMA based ad hoc networks," in *Proc. of GLOBECOM, 2003.*

[17] A. Muqattash and M. Krunz, "Cdma-based mac protocol for wireless ad hoc networks," in *MobiHoc'03*, (Annapolis, Maryland, USA), pp. 153–162, June 2003.

[18] S. Shakkottai, T. S. Rappaport, and P. C. Karlsson, "Cross-layer design for wireless networks," *IEEE Communications Magazine*, vol. 41, pp. 74–80, October 2003.

[19] A. Butala and L. Tong, "Cross layer design for Medium Access Control in CDMA ad-hoc Networks," *Eursip Journal on Applied Signal Processing*, December 2004. To appear.

[20] A. Muqattash and M. Krunz, "A single-channel solution for transmission power control in wireless ad hoc networks," in *MobiHoc'04*, (Roppongi, Japan), pp. 210–222, May 2004.

[21] S. Kunniyur and R. Srikant, "End-to-end congestion control schemes: Utility functions, random losses and ecn marks," *IEEE/ACM Transactions on Networking*, vol. 11, October 2003.

[22] S. K. Bose, *An Introduction to Queing Systems*. 233, Spring Street, New York: Kluwer Academic / Plenum Publishers, 2002.

[23] S. H. Low and D. E. Lapsley, "Optimization flow control-i: Basic algorithm and convergence," *IEEE/ACM Transactions on Networking*, vol. 7, November 1999.

[24] "Matlab version-6." http://www.mathworks.com.

[25] S. H. Low, F. Paganini, J. Wang, S. Adlakha, and J. C. Doyle, "Dynamics of tcp/red and a scalable control," in *Proc. of IEEE INFOCOM*, vol. 1, pp. 239–248, June 2002.