CLAMP: A System to Enhance the Performance of Wireless Access Networks

Lachlan L. H. Andrew, Stephen V. Hanly and Rami G. Mukhtar
ARC Special research Centre for Ultra-Broadband Information Networks (CUBIN)
Department of Electrical and Electronic Engineering
University of Melbourne, Victoria 3010, Australia
{lha,hanly,rgmukht}@ee.mu.oz.au

***Abstract*— This paper presents an improved version of CLAMP, a system that controls the behavior of TCP to enhance the performance of wireless access points. It only requires modifications to be made to the access network, and is totally compatible with TCP senders. We demonstrate its performance by simulation, and provide insight into the stability of the algorithm via analysis.**

I. Introduction

The past decade has seen a revolution in wireless technology for communications. There has been an explosion in the popularity of wireless access data networks as a means to access the Internet. This has been facilitated by the improvements in wireless technology, the ongoing deployment of high speed packet data cellular networks, and wireless access points becoming commodity equipment. Recognition that certain Internet traffic classes demand low packet loss rates has lead to the development of wireless link layers that substantially reduce packet loss rates over the wireless channel. This has been made possible by the emergence of link layer radio bearer service differentiation [1], which enables different radio transmission policies to be applied to each traffic class. For example, when using interactive applications like the World Wide Web, it is desirable to trade variable packet transmission latency for very low wireless packet loss rates. Additionally, smart MAC layer scheduling algorithms now take advantage of channel fluctuations to increase transmission capacity [2,3], and to satisfy quality of service requirements [4].

Combined, all of these link and network layer technologies, have the potential to greatly enhance the untethered Internet experience. However, their effectiveness can be significantly compromised by the behavior of the higher layer transport algorithm. For instance, an "optimised" Medium Access Control (MAC) scheduler will only be optimal if it has a packet available, when it wishes to service the channel.

Currently, the most commonly used transport algorithm on the Internet is TCP [5,6]. Last decade, a great deal of research studied the effect of wireless packet loss and link layer retransmissions on TCP performance [7–9] and proposed techniques for improvement [10–12]. This is not the concern of the present paper. Since the time of those papers, wireless technologies have improved a great deal, and packet loss due to wireless errors is no longer the major concern. The major problems with using TCP over wireless networks today are:

1) TCP does not provide any explicit control over the tradeoff between queueing delay and wireless link utilisation.
2) The cyclic probing of TCP's congestion avoidance algorithm [6] interferes with lower layer mechanisms [13] (e.g. MAC layer packet schedulers).
3) When using TCP to control multiple flows sharing the same bottleneck link, the rate obtained by each flow varies inversely with its propagation delay [14]. This interferes with differentiated service mechanisms that may be operating at the lower network layer.

The focus of the present paper is an enhanced version of CLAMP [15,16] that provides better stability characteristics. CLAMP is a software mechanism that can be installed in wireless access points and receivers. By controlling the TCP receiver's advertised window (AWND), CLAMP tames TCP to mitigate its inherent deficiencies by meeting the following key objectives:

1) **Maintain a fixed mean queue at the wireless access point,** and reduce fluctuations from TCP's congestion avoidance algorithm. This ensures that the lower layer MAC scheduler can actually schedule packets for transmissions when it wants to, *e.g.*, when channel conditions are favourable. It also enables us to explicitly control the tradeoff between queueing latency at the wireless access point and link utilisation.
2) **Differentiate capacity amongst competing flows.** This enables a user to prioritise between flows in the same service class, independent of their respective propagation delays.

CLAMP was introduced in [15,16], but in the present paper we enhance it in a way that removes instabilities that were present in the initial proposal. We also introduce a "receiver-side" slow start that greatly improves transient behavior.

Active Queue Management algorithms (AQM) [17] can also maintain relatively stable buffers at routers [18]. However, this is only true when there are a very large number of TCP flows multiplexed on a single link. This is realistic in the core, but unrealistic in a wireless access point when there may be times when only a single flow is active per radio bearer service. The

0-7803-7975-6/03/$17.00 (C) 2003

only ways to maintain a steady queue in this situation are either to modify the TCP sender, or to control AWND. CLAMP is a distributed, and efficient algorithm for computing the AWND size.

II. Related Work

A great deal of research has been devoted to improving the performance of the existing TCP protocol over wireless links. Many recent techniques stem from ideas that were reviewed in [19]. Techniques can be grouped into three categories: sender-side modifications, receiver-side modifications, or intermediate Performance Enhancing Proxies (PEP). Sender-side modifications require wide spread changes to the Internet, and hence are unlikely to see widespread deployment in the near term. PEP based schemes [20] attempt to hide wireless-related losses from the end hosts by locally retransmitting packets lost over the air interface from an intermediate node, which is located within the network. However, PEP's have been shown to be undesirable since they impose significant limitations to end-to-end connectivity [20]. For these reasons, the present section focuses on reviewing proposals in the area of receiver-side modifications. CLAMP falls within this category.

All of the proposals presented in [21–25] implement receiver side flow control by exploiting the legacy AWND feature in TCP to enhance its performance.

In [24] the AWND value is statically set to the mean bandwidth delay product (mean RTT $\times$ maximum wireless channel rate). This is sensitive to fluctuations in RTT and the number of active flows.

Spring *et al.* [23] set the AWND to maintain a target queue size at the access router according to a service differentiation policy. This requires estimating the RTT, devoid of any queueing delays, the average access link capacity and the number of active flows, quantities that are difficult to obtain accurately, especially when the system is itself attempting to maintain a target queue.

In [21,25,22] it is proposed that routing nodes insert an AWND value into packets or returning acknowledgments, with a value proportional to the available buffer size. These proposals all suffer from limitations; they either assume all flows have common RTTs, require per flow information, or require changes be made to the entire network.

Unlike other schemes [24,23], CLAMP does not rely on accurately measuring the propagation delay, and the receivers do not need to know the bottleneck link rate. This is important, as TCP does not provide sufficient information for receivers to estimate these quantities accurately. Receivers must either use explicit probes, or approximate values.

Unlike [21,25,22], CLAMP does not bias flows based on their RTTs. Its implementation is distributed; the algorithm does not need any knowledge about how many flows are active, nor does it need to maintain any per flow information. Its implementation is completely confined to the access network.

Like most receiver window control schemes, CLAMP also works in conjunction with TCP and other AQM schemes.

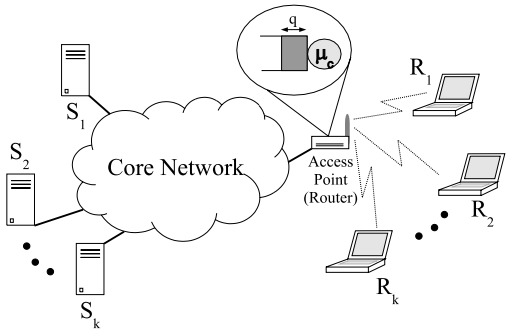

Fig. 1. Model of k flows sharing a single bottleneck access link.

TCP and AQM schemes manage congestion in the core, while CLAMP enhances the performance of the access network.

The goals of CLAMP are similar to those outlined in [26], which proposes the eXplicit Control Protocol (XCP). However, unlike CLAMP, XCP requires modifications to the sender and routers, and estimates of the RTT. Since RTT estimates are a key part of the operation of XCP, it would not be possible to apply it to a receiver side implementation. XCP is more suited to a future internet when widespread changes are made to all clients and routing architectures.

III. System Topology and Design Constraints

A. Design Constraints

The design constraints of CLAMP are that it [16]:

- only requires modifications to the access network,
- relies on aggregate (as opposed to flow-by-flow) feedback from the access point,
- does not rely on precise knowledge of propagation delays,
- is compatible with existing TCP sender implementations,
- has a simple and distributed implementation.

B. Topology and Notation

The access network topology of interest is illustrated in Fig. 1. Consider k flows of data packets that share a single bottleneck link from an access router, with mean output rate μ_c. Referring to Fig. 1, each flow, i, has a sending node, S_i, and a receiving node, R_i. Let d_i denote the total transmission delay, including all propagation and queueing delays, but excluding the queueing delay at the access router.

Each sender implements TCP control. CLAMP is an algorithm that will select values for AWND, $w_i(t)$, in a decentralized way, such that each flow obtains a proportional share of the channel rate, and the equilibrium buffer occupancy of the access router, $q(t)$, can be controlled as discussed below. CLAMP is an enhancement of an algorithm presented by the authors in [15], [16]. It has been modified here to provide stability independent of propagation delay and operate with non-greedy sources. We have also added a receiver-side "slow-start" to the algorithm.

IV. The CLAMP Protocol

CLAMP is a pair of software agents that reside in the access router and the client, as described below.

A. Access Router Agent

The software agent in the access router samples the queue length, q, at regular intervals. A convex monotonic increasing function of q, $p(q)$, is evaluated, and the value passed to each receiver. This may be achieved either by inserting the value into the header (*e.g.* as an IP option) of some or all packets leaving the access router, or alternatively by each receiver explicitly requesting the value from the access router as required. The simulations in this paper use

$$p(q) = \begin{cases} (bq - a)/\mu_c & \text{if } q > a/b \\ 0 & \text{otherwise,} \end{cases} \quad (1)$$

where b determines how sensitive the bottleneck queue size is to the number of flows. The parameters a and b control the equilibrium mean queue size, q^*, as will be seen in Section VI. They can be statically set or dynamically tuned to obtain a desired tradeoff between queueing delay and link utilisation. An algorithm to set these parameters dynamically is out of the scope of the present paper.

B. Client Side Agent

The second modification required is the installation of a software agent located at the receiver. The agent will either replace the existing TCP implementation, or sit between the TCP stack and the network interface (*e.g.*, as a wrapper to the network interface driver).

The agent's function will be to implement CLAMP (as described below), which will appropriately set the receiver's TCP AWND value, hence clamping the sender's TCP congestion window.

C. Window Update Algorithm

For simplicity, the algorithm will be described for flow i in the case of equal-length packets, however, it can be easily generalized. Let t_k denote the time instant when the kth packet is received by the receiving client for flow i. CLAMP is given as:

$$w(t_k) = \begin{cases} w(t_{k-1}) - 1 & \text{if } \Delta w(t_k) < -1 \\ w(t_{k-1}) + \Delta w(t_k) & \text{if } -1 \le \Delta w(t_k) \le \bar{\Delta} \\ w(t_{k-1}) + \bar{\Delta} & \text{if } \Delta w(t_k) > \bar{\Delta} \end{cases} \quad (2)$$

where

$$\Delta w(t_k) = \left[\frac{\phi_i \tau - p(q(t_k))\hat{\mu}(t_k)}{\hat{d}_i(t_k)} \right] (t_k - t_{k-1}) \quad (3)$$

and $\phi_i > 0$ is a positive constant, $\tau > 0$ (packets/sec) is a constant, $\hat{\mu}$ (packets/sec) is an estimate of the received rate of the flow, and $\hat{d}_i(t_k)$ is given below.

The current received rate, $\hat{\mu}$, is estimated using a sliding window averaging function,

$$\hat{\mu}(t_k) = \frac{\alpha}{t_k - t_{k-\alpha}}, \quad (4)$$

and

$$\hat{d}_i(t_{k+1}) = (1 - \beta)\hat{d}_i(t_k) + \beta \frac{w(t_k)}{\hat{\mu}(t_k)}, \quad (5)$$

where the integers α and β are smoothing factors. These estimators were chosen for their simplicity, and other estimators may prove to be more effective. Note that in equilibrium, $\hat{d}_i$ will be the RTT of flow i, *i.e.*, propagation plus queueing delay. However, to compute $\hat{d}_i(t_k)$ does not require an explicit measurement of such delay, and the algorithm achieves its objectives even if it is not an accurate estimate of the RTT.

CLAMP was initially proposed in [15,16]. A new and important ingredient introduced in the present paper is the division by $\hat{d}_i(t_k)$ in (3), which improves stability of the algorithm, as demonstrated in Sections V–VI of the present paper.

The AWND value advertised to the sender is then

$$\text{AWND} = \min(w(t_k), \text{AWND}_i), \quad (6)$$

where AWND_i is the AWND value received by the client side agent from the receiver's operating system.

The maximum window increase in (2) is limited to a constant, $\bar{\Delta} > 0$. This prevents large $t_k - t_{k-1}$ from causing large changes in w when packets arrive infrequently, such as when a source becomes idle for an extended period of time. Furthermore, the limit on the window decrease in (2) is to satisfy the requirements specified in [5, p. 42].

The flow control algorithm can provide non-uniform sharing of the bottleneck bandwidth by appropriately setting the constants ϕ_i. Section VI shows that, under certain conditions, flow i will obtain the proportion $\phi_i / \sum_{j=1}^{k} \phi_j$ of the bottleneck capacity.

Finally, it is important to note that the algorithm as stated does not prevent the window size falling to zero. Since window updates only occur on receiving a packet, this would cause the window to remain at zero indefinitely. There are several techniques that can be used to deal with this special case. However, the simplest solution is to limit the minimum window size to a constant, $w_{\min}$. The simulations presented here take $w_{\min} = 1$.

D. Slow Start

When a flow is initiated it is usually desirable to minimize the initial transient period. The TCP *slow start* algorithm [6] is specifically designed to meet this requirement.

TCP slow start ultimately terminates in buffer overflow (an undesirable side effect). However, it is very effective at reducing a flow's initial transient period. In order to retain this feature, we need to ensure that CLAMP does not inhibit slow start. Hence, we extend CLAMP to implement a receiver side slow start, defined as follows.

CLAMP starts in the slow start phase. For each packet received, the receiver sets

$$w(t_k) \leftarrow w(t_k) + 1(\text{segment}),$$

doubling w once per RTT. Slow start terminates when congestion is detected, *i.e.*, $\Delta w(t_k) < 0$ in (3) or three duplicate acknowledgments are received, indicating packet loss. The receiver then sets $w(t_k) \leftarrow w(t_k)/2 = w(t_k - \text{RTT})$. The

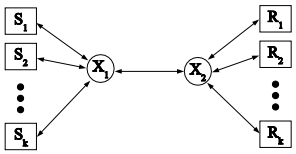

Fig. 2. Simulation network topology.

TABLE I
LINK CONFIGURATION.

Node 1	Node 2	Rate	Delay
S_i	X_1	10 Mbps	$d_i/2$
X_1	R_i	random (0.15–1.5 Mbps)	11 msec

reason for this is that the observations at time t_k which indicate congestions reflect the AWND size one round trip time earlier.

Although buffer dimensioning is beyond the scope of the present paper, note that if the bottleneck buffer size is large enough relative to the equilibrium queue size then the condition $\Delta w(t_k) < 0$ will be true before a buffer overflow occurs, and the slow start algorithm will terminate without causing a buffer overflow (see Section V). In effect, by taking advantage of the additional control information from the access router, CLAMP receiver side slow start tames the behavior of TCP sender side slow start.

V. PERFORMANCE EVALUATION

The aim of this section is to demonstrate that the modified CLAMP protocol retains the characteristics demonstrated in [16]. The network topology shown in Fig. 2 is simulated under various conditions.

Referring to Fig. 2, X_1 is the bottleneck access router and contains the CLAMP router software agent, it monitors the queue size, computes $p(q)$ and inserts the value into the TCP header options field. $S_1, S_2, \ldots, S_k$ are TCP sending clients. $R_1, R_2, \ldots, R_k$ are TCP receiving clients that have been modified to intercept the CLAMP information contained in the TCP headers of incoming packets, and compute a new AWND value which is inserted into the header of outgoing acknowledgments. Node X_2 plays no explicit role in CLAMP; it illustrates that the connections X_1–S_i share the radio medium. (Alternatively, X_2 could be an actual LAN router.) All links are bidirectional, with characteristics as shown in Table I. Other simulation parameters were $\tau = 500$ Bytes, $\bar{\Delta} = 10\,000$ Bytes, $b = 1\,\mathrm{s}^{-1}$, $a = 2000$ Bytes/s. Packets were all 500 Bytes long.

To model a fading wireless channel, the rate of the link between X_1 and X_2 oscillated randomly between two possible transmission states. The time spent in either state was determined by the two state continuous time Markov chain shown in Fig. 3. The link rate in the "good" state (G) was $r_g = 1.5$ Mbit/s and in the "bad" state (B) was 0.15 Mbit/s. We set $\lambda = 1\,\mathrm{s}^{-1}$, and $\mu = 10\,\mathrm{s}^{-1}$. To illustrate the behavior of CLAMP, the received rates per flow were plotted for $k = 4$,

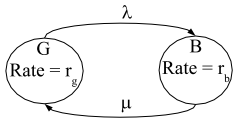

Fig. 3. Markov chain model of variable rate link.

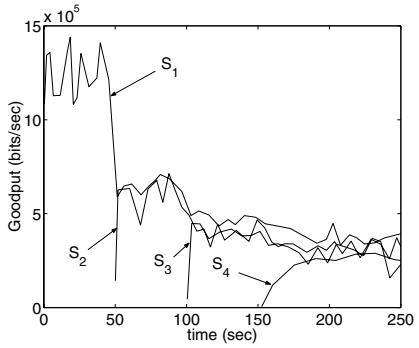

Fig. 4. Typical goodput per flow for the topology of Fig. 2.

with $d_1 = 1.6122$s, $d_2 = 0.2943$s, $d_3 = 0.0756$s, and $d_4 = 0.00458$s. (Note that the algorithm of [15,16] would have been unstable for $bd_1 = 1.6122 > \pi/2$.) The flows from S_1, S_2, S_3 and S_4 were started at 0, 50, 100 and 150 seconds respectively.

The goodputs for each flow, averaged over a sliding window of eight RTT periods, are illustrated in Fig. 4. This demonstrates that each flow obtains an almost equal share of the bottleneck capacity, indicating that the modified CLAMP algorithm retains the fairness characteristic of the original algorithm [15,16].

We next compared the goodput for the same system running CLAMP in conjunction with TCP against a system just running TCP. The results of the simulations are shown in Fig. 5, illustrating the performance enhancement obtained by using CLAMP. It is clear that CLAMP provides a distinct improvement on throughput, especially for small amounts of queueing. With a queueing delay of 40 ms, CLAMP achieves 90% utilisation (1.25 Mbit/s), while TCP obtains a utilisation of 75%. For a utilisation of 90%, TCP requires a queueing delay of 200 ms.

The explanation is that CLAMP controls the queueing at the bottleneck router. It attempts to maintain a queue at the bottleneck router, so that packets are immediately available for transmission when the bandwidth increases. The size of the queue can be controlled by the choice of algorithm parameters. In contrast, the only way to limit the queueing delay under pure TCP is to limit the capacity of the buffer, whose occupancy then fluctuates wildly and is often zero. A thorough investigation of how CLAMP's parameters (a and b) should be set to handle time varying channel capacity is beyond the scope of the present paper.

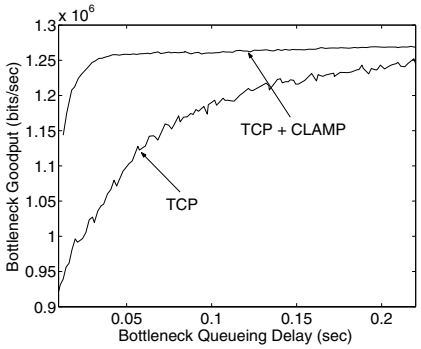

Fig. 5. Bottleneck Goodput vs. buffering at bottleneck link for TCP and TCP with CLAMP

VI. A Fluid Model for Convergence Analysis

A fluid model was proposed in [15] to study the stability of the version of CLAMP proposed in that paper. It was found that the stability can be characterized by the parameters bd_i, $i = 1, 2, \ldots, k$ where b is the parameter of the algorithm, as in (1), and the d_is are the propagation delays. It was shown that, if an upper bound on d_i is known, then b can be configured to ensure stability. In [15,16] it was demonstrated that the fluid model analysis was able to characterise the stability of the real system.

The following is an analysis of the fluid model for the modified version of CLAMP presented in this paper. Retaining the notation, and making the same assumptions as [16], the modified fluid model for the case when $\hat{d}_i = d_i$ is:

$$\frac{dB_i}{dt} = \left.\frac{dw_i}{dt}\right|_{t-d_i} + \frac{B_i(t-d_i)}{q(t-d_i)}\mu_c - \frac{B_i(t)}{q(t)}\mu_c. \tag{7a}$$

where

$$\frac{dw_i}{dt} = \begin{cases} -\frac{B_i(t)}{q(t)}\mu_c & \text{if } g_i(t) \le -\frac{B_i(t)}{q(t)}\mu_c \\ g_i(t) & \text{if } -\mu_c < \frac{q(t)}{B_i(t)}g_i(t) \le \mu_c\bar{\Delta} \\ \frac{B_i(t)}{q(t)}\mu_c\bar{\Delta}+\epsilon & \text{otherwise} \end{cases} \tag{7b}$$

and

$$g_i(t) = \frac{1}{d_i}\left[\phi_i\tau - p(q(t))\frac{B_i(t)}{q(t)}\mu_c\right]. \tag{7c}$$

Here ϵ is a small constant that serves the same purpose as $w_{\min}$, and $B_i(t)$ is the proportion of $q(t)$ corresponding to flow i.

The change in the total queue size is obtained by summing (7a) over i:

$$\frac{dq}{dt} = \sum_{i=1}^{k}\left.\frac{dw_i}{dt}\right|_{t-d_i} + \sum_{i=1}^{k}\frac{B_i(t-d_i)}{q(t-d_i)}\mu_c - \rho(t)\mu_c, \tag{8}$$

where $\rho(t)$ is the utilisation of the bottleneck channel at time t.

The only new ingredient in the fluid model is the division by d_i in (7c), and this has no impact on the existence or uniqueness of fixed points. Thus, the following result from [16] still holds:

Theorem 1: There is a unique equilibrium point defined by

$$B_i^* = \frac{\phi_i}{\sum_{j=1}^{k}\phi_j}p^{-1}\left(\frac{T}{\mu_c}\right). \tag{9}$$

for all i, where

$$T = \tau\sum_{j=1}^{k}\phi_j \tag{10}$$

Theorem 1 says that under the assumptions that all sources are greedy and the system converges, flow i will obtain the proportion $\phi_i/\sum_{j=1}^{k}\phi_j$ of the bottleneck link capacity. The key issue then is stability, and in this regard it must be noted that the system (7) is nonlinear and hence difficult to analyze. As in [16], we attempt to obtain insight through analysing the equal delay case.

In the special case of equal propagation delays, and ignoring the boundary conditions in (7b), equation (8) takes the simple, linear form

$$\frac{dq}{dt} = \frac{1}{d}[T + a - bq(t-d)] \tag{11}$$

where $d_i = d$ for all i is assumed.

Theorem 2: Let $d_i = d$ for all i, $B_i(t) \ge 0$ for all $t < 0$, and $p(\cdot)$ be given by (1). Then the necessary and sufficient condition for the total queue size, $q(t)$, to converge under (11) is:

$$b < \frac{\pi}{2}. \tag{12}$$

This theorem follows from the corresponding result in [16] by rescaling time by a factor of d. It can also be proved directly by standard methods, such as [27, p 26].

If $q(t)$ converges to a constant, the equations (7) decouple, and each is stable if and only if (12) holds. Thus (12) is a necessary and sufficient condition for the asymptotic stability of (7) for the case of equal delays.

The question of stability with distinct propagation delays remains. To investigate this, we simulated tens of thousands of instances of the system (7) with random combinations of delays chosen from $(0, 2]$, between two and twenty flows, and random initial conditions. We never found the system to be unstable when $b < \pi/2$. The worst case appears to be the case of equal delays, as is illustrated for the case of two flows in Figure 6, and in that case instability occurs only for $b > \pi/2$.

VII. Conclusions

We have enhanced the CLAMP system to be stable for any propagation delay. CLAMP provides differentiated proportional allocation of the capacity of a bottleneck link and simultaneously improves performance. A CLAMP deployment only requires modifications to the access network; it works in conjunction with current TCP sender implementations and AQM routers in the core network.

Acknowledgement

This work was supported by the Australian Research Council.

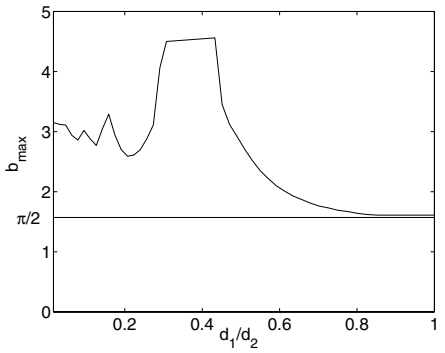

Fig. 6. Maximum value of b for which (7) is stable, as a function of d_1/d_2 for a two-flow system.

REFERENCES

[1] 3GPP, "Quality of service (QoS) concept and architechture," Technical Standard 3G TS 23.107 v5.9.0, 3G Partnership Project, July 2003.

[2] X. Liu, E. Chong, and N. Shroff, "Opportunistic transmission scheduling with resource-sharing constraints in wireless networks," *IEEE J. Select. Areas Commun.*, vol. 19, pp. 2053 –2064, Oct. 2001.

[3] P. Bender, P. Black, M. Grob, N. Padovani, R. Sindhushyana, and A. Viterbi, "CDMA/HDR: a bandwidth efficient high speed wireless data service for nomadic users," *IEEE Communications Magazine*, vol. 38, pp. 70–77, July 2000.

[4] K. Chang and Y. Han, "QoS-based adaptive scheduling for a mixed service in HDR system," in *Proc. 13th IEEE Int. Symp. Personal, Indoor Mobile Radio Commun.*, vol. 4, pp. 1914 –1918, 2002.

[5] Information Sciences Institute University of Southern California, "RFC 793: Transmission control protocol," RFC 793, IETF, 1981.

[6] W. Stevens, "TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms," RFC 2001, IETF, 1997.

[7] Y. Bai, A. Ogielski, and G. Wu, "Interactions of TCP and radio link ARQ protocol," vol. 3, pp. 1710–1714, 1999.

[8] G. Bao, "Performance evaluation of TCP/RLP protocol stack over CDMA wireless link," *Wireless Networks*, vol. 2, no. 3, pp. 229–237, 1996.

[9] A. Chockalingham, M. Zorzi, and R. R. Rao, "Performance of TCP on wireless fading links with memory," pp. 595–600, 1998.

[10] H. Balakrishnan, S. Seshan, and R. H. Katz, "Improving reliable transport and handoff performance in cellular wireless networks," *ACM Wireless Networks*, vol. 1, no. 4, 1995.

[11] K. Brown and S. Singh, "M-TCP: TCP for mobile cellular networks," *ACM Computer Communication Review*, vol. 27, no. 5, 1997.

[12] Z. Haas and P. Agrawal, "Mobile-TCP: An asymmetric transport protocol design for mobile systems," (Montreal, Canada), IEEE, 1997.

[13] M. Malkowski and S. Heier, "Interaction between UMTS MAC scheduling and TCP flow control mechanisms," in *Proc. International Conference on Communication Technology*, pp. 1373–1376, 2003.

[14] S. Floyd, "Connections with multiple congested gateways in packet-switched networks, part I: One-way traffic," *ACM Comp. Commun. Rev.*, vol. 21, pp. 30–47, Oct. 1991.

[15] L. L. H. Andrew, S. V. Hanly, and R. G. Mukhtar, "Analysis of rate adjustment by managing inflows," in *Proc. 4th Asian Control Conference*, (Singapore), pp. 47–52, 2002.

[16] L. L. H. Andrew, S. V. Hanly, and R. G. Mukhtar, "CLAMP: Differentiated capacity allocation in access networks," in *Proc. IEEE Int. Performance Computing and Communications Conf.*, (Phoenix), pp. 451–458, Apr. 2003.

[17] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Networking*, vol. 1, no. 4, pp. 397–413, 1993.

[18] S. Athuraliya, V. H. Li, S. H. Low, and Q. Yin, "REM: active queue management," *IEEE Network*, vol. 15, no. 3, pp. 48–53, 2001.

[19] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz, "A comparison of mechanisms for improving TCP performance over wireless links," *IEEE/ACM Trans. Networking*, 1997.

[20] J. Border, "Performance enhancing proxies intended to mitigate link-related degradations," RFC 3135, IETF, 2001.

[21] M. Gerla, R. L. Cigno, S. Mascolo, and W. Weng, "Generalized window advertising for TCP congestion control," Technical Report 990012, University of California Los Angeles Computer Science Department, 1999.

[22] J. Aweya, M. Ouellette, D. Y. Montuno, and Z. Yao, "Enhancing network performance with TCP rate control," in *Proc. IEEE Globecom*, (San Francisco, CA), pp. 1712–1718, 2000.

[23] N. T. Spring, M. Chesire, M. Berryman, V. Sahasranaman, T. Anderson, and B. N. Bershad, "Receiver based management of low bandwidth access links," in *Proc. IEEE INFOCOM*, pp. 245–254, 2000.

[24] R. Mukhtar, S. Hanly, M. V. Ivanovich, H. Vu, and P. G. Fitzpatrick, "Analysis of TCP performance over hybrid fast fixed-to-slow wireless links," in *Proc. IEEE Globecom 2001*, (San Antonio, TX), pp. 1816–1820, 2001.

[25] L. Kalampoukas, A. Varma, and K. K. Ramakrishnan, "Explicit window adaption: A method to enahance TCP performance," *IEEE/ACM Trans. Networking*, vol. 10, pp. 338–350, June 2002.

[26] D. Katabi, M. Handley, and C. Rohrs, "Internet congestion control for future high bandwidth-delay product environments," in *ACM Sigcomm 2002*, August, 2002.

[27] J. E. Marshall, H. Górecki, K. Walton, and A. Korytowski, *Time-Delay Systems: Stability and Performance Criteria with Applications*. New York, NY: Ellis Horwood, 1992.