

# A Case for Delay-based Flow Control in CDMA 2.5G Networks

Ranjith S. Jayaram, Injong Rhee

**Abstract—**This paper presents an early experience with a CDMA 2.5G wireless network commercially deployed in South Korea. It finds that there are high signal losses and latency commonly present in packet delivery causing TCP to under-utilize the available network bandwidth significantly. In this environment, there is inherent limitation in using packet losses as congestion indicators because of lack of correlation between congestion and packet losses. To remedy this problem, a new flow control protocol that uses delay hysteresis as a congestion indicator is presented. The protocol actively manages delays to keep them within a certain bound by throttling its transmission rate when network delays tend to increase and also probing for more bandwidth when network delays tend to decrease. A variant of the protocol is currently incorporated in a cellular-phone based video-on-demand system as the main transport protocol for video file download and also for streaming. Our experiment results suggest that the protocol achieves higher and more consistent throughput than TCP, and exhibits some degree of fairness to its own flows and TCP.

## I. INTRODUCTION

Next generation wireless networks are being deployed in many parts of world. Asia, especially South Korea, has taken the lead in deploying CDMA 1XRTT 2.5G networks (144kbps) from year 2000 and now moving into CDMA EV-DO (2Mbps) and WCDMA as Europe deploys GPRS networks. The US carriers are also following the suit. These networks will move voice-centric networks to data packet-switched networks.

Cell-phones and PDAs are now beyond social status symbols and became the necessity. For instance, South Korea (with the total population of 50 million) hosting 28-30 million cell phone users are now touting more than 10% of their subscribers being daily users of wireless data networks. Spurred by ubiquity and high availability of the CDMA networks, high popularity of color handsets, and wide use of Internet, their consumers enjoy a variety of data applications ranging from ring tone downloads to mu-

sic and video downloads and streaming. Many new applications such as mobile lottery, karaoke, and video games are daily introduced. This fast consumer adoption of wireless data applications has created a new fertile revenue source for carriers over the already saturated voice market. While this motivates more investment into network infrastructures, development of enhanced data transport protocols as a more cost-effective solution invites attention from research community.

TCP is the most popular transport protocol used by (wireless) application developers. However, the performance of TCP does not catch up to the speed of infrastructure. This is a well-known problem, namely TCP's misinterpretation of packet losses as congestion signal [1], [11], [10], [2]. Errors on wireless links tend to be frequent and occur due to a variety of reasons including not only congestion, but also path loss, fading, and interference. In this environment, TCP tends to under-utilize the network bandwidth since losses may trigger spurious timeouts and unnecessary rate reduction in TCP. Furthermore, network delays and jitters are significantly increased as the wireless networks employ a very large buffer. Buffers may cause delay when channel conditions deteriorate to cause more link-level retransmission or when competing flows in the same cell may take away allocated channels from a flow. High RTT delays have known to create long and wrong timeouts for TCP [9].

In a CDMA network commercially deployed, we commonly observe 3-5% random packet losses and also frequently over 7-10% losses. The typical round-trip delays observed are in the range of 500ms to 800ms. However, delays as high as 3 to 15 seconds are not uncommon. TCP running in this environment has a very limited, erratic throughput ranging from 0kbps to 80kbps, which forces many TCP applications running on handsets frequently to suffer from loss of connections and services.

Among motivation to study a different protocol than TCP is the real-time requirement of some applications such as video streaming that cannot be met using the currently available transport protocols. As it is obviously true for video conferencing applications or online game applications, even less time-constraint applications such as video-on-demand applications require lower delays and

This paper is an extended abstract, significantly reduced from the full paper to meet the space limitation of the workshop submission.

I. Rhee is with the Department of Computer Science, North Carolina State University, Raleigh, NC 27695 Email: rhee@cs.ncsu.edu, and R. Jayaram is with Togabi Technologies, INC, 6815 Flanders Drive, Suite 150, San Diego, CA 92121. Email: ranjith@togabi.com.

delay jitters because of lack of buffer space in cellular phones. Thus, although these applications can adapt to bandwidth variations by some simple media scaling techniques, high delay jitters in the range of 10 seconds result in frequent play stalls or buffering.

In this paper, a new flow control protocol is proposed that can give a reasonably high performance in this extremely “hazardous” environment. The protocol uses a mechanism called *delay hysteresis* to detect congestion which helps discerning congestion using delay samples at the presence of noise in the samples. This technique allows the protocol to control its transmission rate at the decrease and increase of network delays. By managing the delays directly, it keeps the system within a safe region of network delays, preventing network thrashing. Our experiment shows that the protocol achieves higher and more predictable throughput than TCP while still exhibiting some degree of fairness to TCP. We incorporated the protocol in a commercially available wireless VOD system. The network infrastructure where we base our work is a CDMA 1xRTT network commercially deployed in South Korea. Since it is one type of such systems and different vendors may provide different implementations and different standards, our observation may not be of representation for all CDMA networks. Our work can be viewed as a preliminary study of an early system of 2.5G networks in its evolutionary path to high speed reliable wireless networks.

## II. RELATION WORK

There has been much work in improving TCP performance in wireless cellular networks. Most work focuses either on adding link-level or router mechanisms to reduce packet losses at wireless RF ends through retransmission or forward error correction [1], [2], [11], or modifying TCP to improve its reaction to packet losses [5], [7]. Some work related to handling high rate and delay variations in the wireless networks also proposes router or transport level additions and modifications to ease ack compression caused by the variations [4], [6]. However, most of the proposed techniques require changes in infrastructure (i.e., changes in link level or routers)—not so cost effective, or present some deployment problem as they require changes in both TCP sender and receiver. Not much work has been done in proposing a new flow control that handles delay variations and achieves high performance without changes in infrastructure or deployment problems. Delays have been used as congestion control [3], [13], [8]. However, these are proposed for the environments where losses are still primary congestion indicators or for more deterministic network environments.

## III. EXPERIMENTAL SETUP

We omit the description of CDMA 2.5G architecture for lack of space.

To minimize chances of packet losses in the wired networks, we run a server at an ISP’s Internet Data Center (IDC) which is a managed network by an ISP with 45Mbps to 144Mbps connections to other ISPs including the wireless carrier’s networks. The server runs on Microsoft Windows 2000 Server while the clients embedded in a cellular phone running BREW version 1.1 which is Qualcomm’s run time environment for wireless applications. During the experiment, all cellular phones are stationary in a location that is not so busy while providing consistent connectivity to wireless networks throughout day. The server transmits dummy data to the client through TCP or UDP sockets on top of PPP connections. When sending UDP packets, each packet contains a unique sequence number with a timestamp. When receiving a UDP packet, the client sends back acknowledgment containing the sequence number. The server computes the round trip time from the timestamp of the packet and its acknowledgement and records the RTT. In our experiment, we choose 512 bytes as the packet size for UDP. The size of acknowledgment (payload) is 12 bytes containing the sequence number of the packet it acknowledges and other book-keeping information.

## IV. NETWORK CHARACTERISTICS

In this section, we present the packet losses and delay characteristics of the network we study.

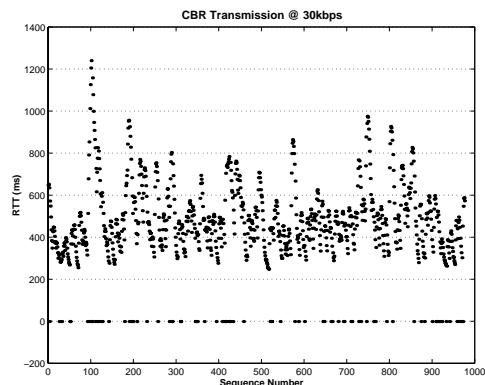


Fig. 1. Round-trip delays of UDP packets transmitted at a constant bit rate of 30kbps

Figures 1 and 2 show the observed round-trip times over transmitted UDP packets (shown in packet sequence number) with fixed sending rates of 30kbps and 60kbps respectively. The packets with zero RTTs are lost packets. In this experiment, we observe that the network conditions are good and the RTTs are between 500-800ms most of the time. Such high RTTs arise primarily due to the high

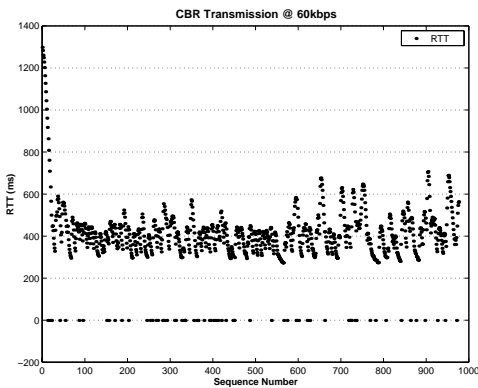


Fig. 2. Round-trip delays of UDP packets transmitted at a constant bit rate of 60kbps

propagation latency of wireless links. However, even under good conditions, the loss rate of around 3-5% is common which is still very high compared to that of wired networks. The losses are highly unpredictable and random; we did not observe much correlation between losses and delays. Similar loss and delay patterns were also observed for lower sending rates. It is hard to distinguish which packet losses are attributed to congestion. Since these random losses were present in most of our experiments, we conjecture these losses are likely due to signal degradation rather than congestion.

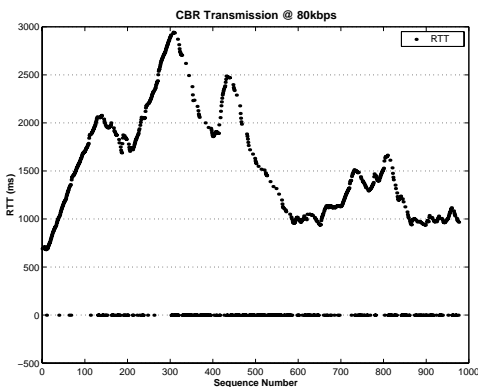


Fig. 3. Round-trip delays of UDP packets transmitted at a constant bit rate of 80kbps

We observe that the sending rate increases result in a dramatic increase in the loss rates and end-to-end delays. Figure 3 shows the results for a run with a constant sending rate of 80kbps. We find that the RTTs are above 1 second most of the time. The packet loss rate increases beyond 10% with longer and more frequent burst losses. These losses are likely due to congestion as we observe these loss rates and high latency under high transmission rates. However, again it is hard to distinguish which losses are attributed to congestion.

Sending at a low constant rate (e.g., well below 80kbps) does not necessarily guarantee the lower delays and loss

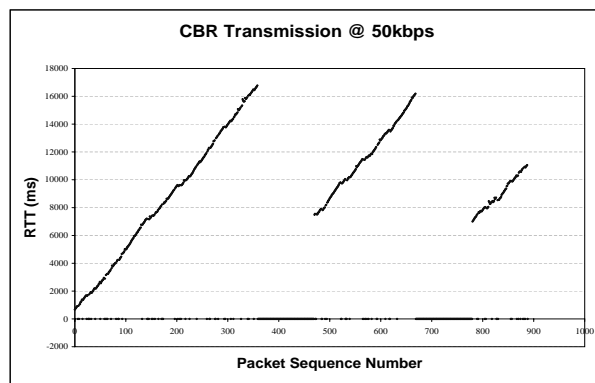


Fig. 4. Round-trip delays of UDP packets transmitted at a constant bit rate of 50kbps

rates. Figure 4 shows the case where even if the sending rate is set to 50Kbps, the RTTs increase continually up to 17 seconds and then the network finally thrashes with more than 100 packets being lost in a row. The network delays are affected by a number of things including the channel conditions and channel scheduling. For instance, when channel conditions are bad, the link layer retransmission will cause delays as more packets are kept in the buffer longer. When there are more users in a cell, channels allocated to one flow can be moved to others. The channel allocation algorithm is proprietary and varies from one vendor to another. When a channel is de-allocated from a handset, the handset's bandwidth decreases accordingly which causes high delays for the data in transit. When the delays are not actively managed by the transmission source, the delays can shoot up to the point where network thrashing occurs. Flow control must be able to detect this type of congestion early enough to avoid network thrashing by reducing transmission rates.

## V. DELAY BASED PROTOCOL (DBP)

### A. Protocol sketch

Instead of using packet loss as congestion indication, we use the round-trip time as an indication of network congestion. We call this protocol *Delay Based Protocol* (DBP). In the protocol, the sender measures RTTs by taking an exponentially weighted sample of RTTs (similar to TCP's RTT estimation). When it finds that the round-trip delays are increasing beyond a certain limit, it invokes the rate decrease. The sender responds by cutting down its sending rate and this results in a lowering of the RTTs. At first we experimented with a single RTT threshold, i.e., whenever the RTT falls below beyond this threshold, the rate grows, and when the RTT increases beyond the threshold, the rate shrinks. This rudimentary scheme results in high rate oscillations because of its sensitivity to noise in delay estimation and thus the network makes rapid back and forth transitions between the regions that lie on either side of the

threshold. To overcome the oscillatory behavior, we introduce the notion of *delay hysteresis*. Hysteresis is a common phenomenon in physical systems and it represents the history dependence of physical systems. The term is most commonly applied to magnetic systems and is the operating principle behind magnetic tape storage - because the magnetization lags behind the field from the tape head, when the field drops to zero, the tape stays magnetized, thereby storing data.

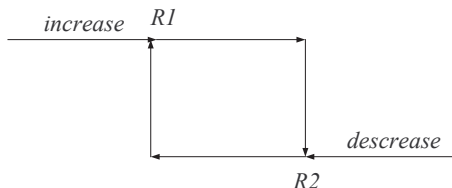


Fig. 5. Delay hysteresis loop

Hysteresis loops happen when a system is repeatedly wiggled back and forth between “safe” and “unsafe” regions of RTTs. Figure 5 shows the delay hysteresis loop we use to dampen its sensitivity to noise in sampled delay estimation. The idea is to introduce a buffer between the safe and unsafe regions in order to absorb the noise signals that might occur during transitions between the two. This buffer region helps avoiding high rate oscillations. When the RTT is in the region to the left of  $R1$ , DBP’s response is to increase its rate and when the RTT is in the region to the right of  $R2$ , DBP responds by reducing its rate. Finally, when the RTT is in the hysteresis region between  $R1$  and  $R2$ , DBP’s response depends on the history of the network. If the current RTT was reached in the process of a rate increase, DBP increases the rate further. If the current RTT was reached in the process of a rate decrease, DBP decreases the rate further. The response signal (rate increase) lags behind the congestion signal (the round-trip time) when approaching from the direction of  $R2$  and hence we say that the loop exhibits hysteresis. For our implementation we have used fixed values for  $R1$  and  $R2$ , with  $R1 = 800\text{ms}$  and  $R2 = 1200\text{ms}$ .

Each rate adjustment happens only at one RTT period after the previous adjustment. The amount of increase and decrease in the rate adjustment is determined by the history of rate fluctuation. We use the rate smoothing similar to the one used in TEAR protocol [12]. The details of this smoothing is omitted for lack of space.

We note that the current scheme does not yield enough flexibility to seamlessly migrate between various networks which have different delay characteristics. The packet latency on wireless networks depends on a whole range of factors ranging from link-level retransmission mechanisms

to transport-level data interleaving. Thus it can be expected to vary widely between different networks. Modifying the DBP scheme to adapt to varying network conditions and heterogeneous networks and have dynamically varying hysteresis thresholds is work for the future.

## B. Performance Evaluation

We implemented DBP on top of reliable UDP which is implemented using a form of selective acknowledgment. The receiver tells the sender of missing packets, and the sender retransmits the lost packets until the receiver acknowledges the reception of the lost packets. We don’t give much detail on this functionality due to space.

We compare the performance of DBP and TCP since TCP is the most common protocol used in wireless applications (surprisingly even in streaming application because of its convenience and reliability) and DBP can be used for many applications including file transfer as well as streaming. All the measurements below are taken at the receiver (i.e., client).

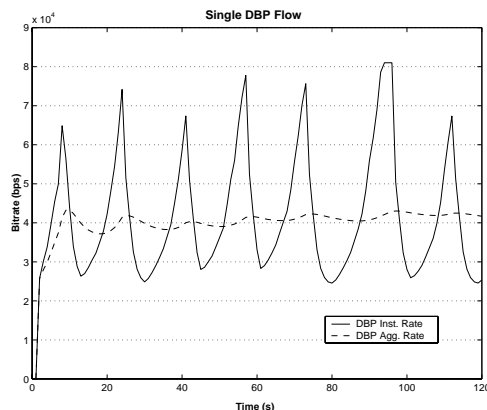


Fig. 6. The instantaneous and aggregate rate fluctuations of DBP

**Short-term characteristics:** We first observe whether DBP does not overrun the system so that it operates within a bound of oscillation. Figure 6 shows the instantaneous and aggregate rate variations of a single DBP flow. DBP keeps increasing its sending rate until the measured round-trip times exceed the threshold  $R2$ . Thereafter, it decreases its rate to bring the network back into the safe region below the threshold  $R1$  and then it starts probing for bandwidth again by increasing the rate gradually. The process keeps repeating and results in a number of triangular regions formed by successive increase and decrease runs. This property allows DBP to contain the delays within a certain bound by actively managing the delays through rate adjustment at the increase or decrease of delays. Although not shown in the figure, our measurement indicated that DBP contains the delays well within the bounds of  $R1$  and

R2. The average throughput DBP reaches is about 40kbps.

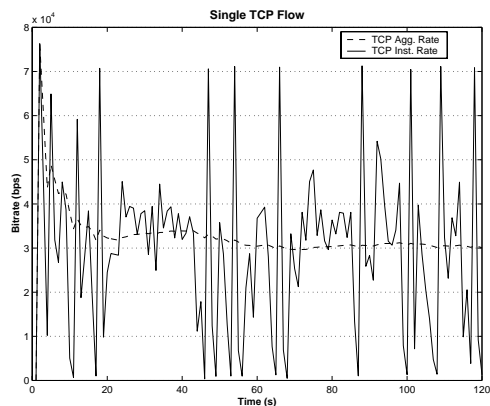


Fig. 7. The instantaneous and aggregate rate fluctuations of TCP

In comparison with DBP, Figure 7 shows the receiving rates of one TCP flow from the server to the client. It shows both instantaneous and aggregate rates (arithmetic mean over the transmission period). As expected, TCP's instantaneous rates fluctuate quite severely over the wide operating rates between zero and 80kbps. It shows high sensitivity of TCP to packet losses in the network. The average throughput reached by TCP is about 30kbps. There is no way to ascertain whether this is the actual available bandwidth, but it is highly likely that because of the problems discussed before, TCP ends up under-utilizing the network.

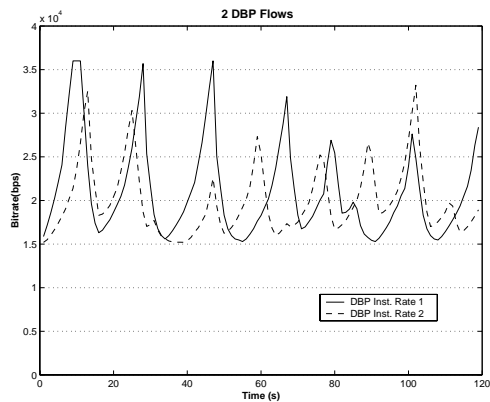


Fig. 8. Two flows of DBP competing

**Fairness and TCP-friendliness:** We find out how DBA flows compete with each other. Figure 8 shows the instantaneous rate variations of 2 DBP flows running together and sharing the network. The rate variations of both the flows are similar and consist of a sequence of triangular regions. The aggregate throughput obtained by the flows is just over 20kbps, which can be considered to be the fair-share because a single DBP flow running alone achieves a maximum throughput of 40kbps.

It is also important that DBP does not starve TCP flows

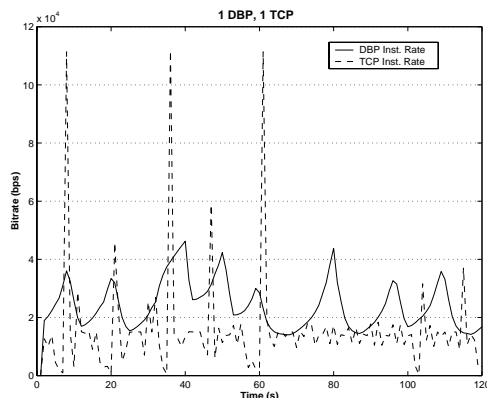


Fig. 9. One TCP and DBP flows competing

when competing against TCP although DBP achieves higher throughput. Figure 9 shows the instantaneous rate variations of one DBP flow and one TCP flow running together. The variations in TCP's rate are drastic, sometimes the rate shoots up to 100kbps and many times it falls to zero. Most of the time, however, the rate falls below 20kbps and the throughput achieved by TCP is about 16kbps. Note that TCP's rate is around half of the single flow experiment. This suggests that DBP does not necessarily take away the bandwidth of TCP, but rather use the bandwidth left unused by TCP. More tests, however, are required to confirm this behavior. Although this is much lower than the 25kbps that DBP is able to get from the network, it is important to note that DBP does not drive TCP to ground and is still able to share the bandwidth. This sharing is due to the correlation between losses and delays when the network is congested as observed earlier with UDP experiment.

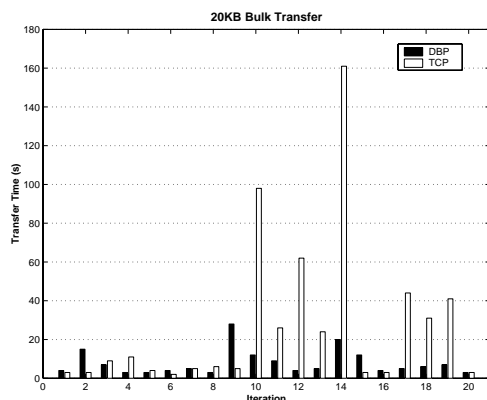


Fig. 10. Bulk transfer of 20KB file using TCP and DBP taken at different times of day from 10AM (left) to 8PM (right)

**Bulk transfer:** We compare the throughputs achieved by TCP and DBP protocols and their consistency with which they achieve their respective throughputs. Figures 10 and 11 show the transfer time samples of files sized 20KB and 100KB, respectively. The tests are taken at regular inter-

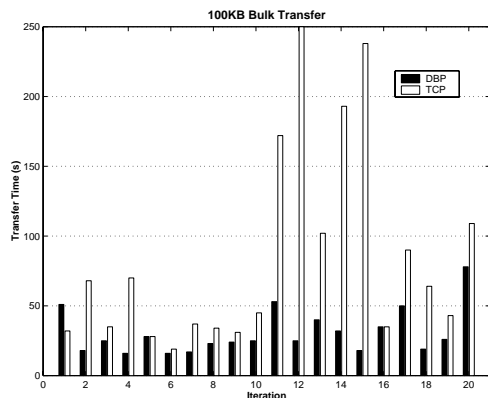


Fig. 11. Bulk transfer of 100KB file using TCP and DBP taken at different times of day from 10AM (left) to 8PM (right)

vals from 10AM (leftmost) to 8PM (rightmost). Each TCP transmission test is immediately followed by a DBP test so that TCP and DBP can run under a reasonably similar network condition.

At 20KB, TCP's throughput is comparable with that of DBP. When the network conditions are good, TCP is sometimes superior because of its more aggressive bandwidth probing. But even for short transfers TCP cannot be trusted to provide this performance consistently. There are cases of severe degradation in transfers throughput - these can be seen as spikes in Figure 10. For 100KB transfer, the performance difference between DBP and TCP becomes wider. TCP's throughput often goes below 15kbps as the transfer size increases. Although DBP is not immune from such performance degradations either, it is more consistent and predictable than TCP in achieving higher throughputs under varying network conditions and during different times of the day.

## VI. CONCLUSION

Wireless networks have different characteristics from the wired Internet and provide different challenges. One of the main problems protocols face on wireless networks is the inability to distinguish packet losses caused by lossy wireless links from those caused by network congestion. This severely degrades the performance of protocols like TCP. We described how a delay based protocol that uses round-trip delays instead of packet losses as an indication of congestion can be adopted in wireless networks. We presented and analyzed our preliminary results from running DBP over commercially deployed wireless networks in South Korea. We showed that DBP achieves superior throughput compared to TCP. It also exhibits lower fluctuations in its sending rate and its behavior is more predictable than TCP. DBP is friendly to TCP over wireless networks in that it does not drive TCP to starvation. Fi-

nally we presented our results comparing the throughputs for bulk transfer achieved by TCP and a reliable protocol we implemented over DBP. Much work needs to be done to improve the performance of DBP over wireless networks. Modifying the simple hysteresis based approach of reacting to congestion to a more generic delay-gradient as in [8] based approach is worth considering.

## REFERENCES

- [1] A. Bakre and B. R. Badrinath, "Handoff and System Support for Indirect TCP/IP" *Proceedings of the 2nd Usenix Symposium on Mobile and Location-Independent Computing*, April 1995.
- [2] H. Balakrishnan, R. Katz, V. Padmanabhan, and S. Seshan, "Improving Performance of TCP over Wireless Networks," *Proceedings of ACM Mobicomm*, Nov. 1995.
- [3] L. Brakmo, S. O'Malley, and L. Peterson, "TCP Vegas: New techniques for congestion detection and avoidance," *Proceedings of the SIGCOMM*, Aug. 1994, pp. 24-35.
- [4] M. Chan, and R. Ramjee, "TCP/IP Performance over 3G Wireless Links with Rate and Delay Variation," *Proceedings of ACM Mobicomm*, 2002.
- [5] H. Inamura, *et al.*, "TCP over 2.5G and 3G Wireless Networks," Internet Draft, draft-ietf-pilc-2.5g3g-07.txt, Aug. 2000.
- [6] S. Karandikar, *et al.*, "TCP Rate Control," *ACM Computer Communications Review*, Jan. 2000.
- [7] F. Khafizov and M. Yauvz, "TCP over CDMA2000 Networks," Internet Draft, draft-khafizov-pilc-cdma2000-00.txt.
- [8] R. Jain, "A Delay-Based Approach for Congestion Avoidance in Interconnected Heterogeneous Computer Networks," *Computer Communication Review*, V.19 N.5, October 1989, pp. 56-71.
- [9] R. Ludwig, "A Case for Flow-Adaptive Wireless Links," Technical Report CSD-99-1053, University of California, Berkeley, 1999.
- [10] R. Ludwig, "Eliminating Inefficient Cross-Layer Interactions in Wireless Networking," PhD Thesis, Aachen University of Technology, April 2000.
- [11] S. Paul, *et al.*, "An Asymmetric Link-Layer Protocol for Digital Cellular Communications," *Proceedings of INFOCOM*, 1995.
- [12] I. Rhee, V. Ozdemir, and Y. Yi, "TEAR: TCP Emulation At the Receivers - Flow Control for Multimedia Streaming," Technical Report, Department of Computer Science, North Carolina State University, 1999.
- [13] D. Sisalem, and H. Schulzrinne, "The Loss-Delay Adjustment Algorithm: A TCP-friendly Adaptation Scheme," *Workshop on Network and Operating System Support for Digital Audio and Video*, July 1998.