Bandwidth Sharing Strategy between Concurrent TCP Connection in Wireless Mobile Network

JaeGyu Jung [1,2], BangHun Chun [1], YoungJoo Kim [1], Chuck Yoo [2]
[1] Software Center, Samsung Electronics, Kangnamgu, Seoul, Korea
[2] Dep. of Computer Science & Engineering, Korea University, Sungbukgu, Seoul, Korea

Abstract—**This paper presents a mechanism that maintains the queue length at bottleneck link can be minimal and stable while achieving full utilization of the link bandwidth without changing the TCP sender. With this scheme, a user can achieve fair bandwidth allocation between competing flows within mobile equipment.**

I. INTRODUCTION

Wireless link have relatively low bandwidth in comparison with the backbone as before. Since almost services on wireless network are sent from the server to the client, moreover, wireless link is often the network bottleneck. Managing contention between incoming traffic flow at the receiver's wireless link is thus becoming a critical issue. Because TCP accumulates all available buffer space allocated to it and wireless link is the bottleneck, concurrent TCP connections cannot divide the wireless link bandwidth fairly. Since the first TCP connection occupies almost of base station buffer allocated to that mobile equipment, the later connections must bear the brunt of excess queueing delays [1]. The fundamental idea behind our solution is to control the usable window size of sender for each connection by manipulating the receiver's advertise window size. We applied TCP Vegas [2] similar mechanisms at the receiver. We used a cellular phone itself to measure the performance on a nationally deployed commercial CDMA2000-1x network.

II. THE ALGORITHMS

We measured packet inter-arrival time at the TCP receiver and defined two types of packet inter-arrival time such as *long_time* and *short_time*. We assumed *long_time* is the time suspended by transmission between a TCP sender and a TCP receiver and *short_time* is the time suspended by propagation between a base station and a TCP receiver. A packet arrived within *short_time* boundary can be considered as a queued packet at a base station.

Since the sender transfers second and third packets simultaneously after receiving an ack for first packet, the initial value of *shorttime_base* and *longtime_base* can be set when second and third packets are arrived.

$$shorttime\_base = 2^{nd}\ packet\_arrivaltime - 1^{st}\ packet\_arrivaltime$$
$$longtime\_base = 3^{rd}\ packet\_arrivaltime - 2^{nd}\ packet\_arrivaltime$$
$$s\_variance = \tfrac{1}{10} \times shorttime\_base$$

Whenever a new packet is arrived, the TCP receiver determines this packet inter-arrival time is included in *shorttime_boundary* or not. If it is included in that boundary, a new *shorttime_base* and a new *shorttime_boundary* will be calculated with this equation.

$$shorttime\_base = \tfrac{7}{8} \times shorttime\_base + \tfrac{1}{8} \times interarrival\_time$$
$$s\_variance = \tfrac{3}{4} \times s\_variance + \tfrac{1}{4} \times |s\_variance - shorttime\_base|$$
$$shorttime\_boundary = [0, shorttime\_base + s\_variance]$$

If the inter-arrival time of a new packet is over *shorttime_boundary*, new *longtime_base* can be calculated with similar equation.

$$longtime\_base = \tfrac{7}{8} \times longtime\_base + \tfrac{1}{8} \times interarrival\_time$$

We divide *longtime_base* with *shorttime_base* and use the quotient that we call β as an indicator that the queue of base station is filled sufficiently. We also achieve α through halving β.

$$\alpha = \frac{\beta}{2}, \qquad \beta = \frac{longtime\_base}{shorttime\_base}$$

With these factors, we can choose the value for advertise window according to Fig. 1. When a new connection is established, the *shorttime_base* must be recalculated due to packets for the new connection. The TCP receiver can easily detect about the new connection, because it requests or receives the request of a new connection. Therefore the receiver resets the *shorttime_base* and recalculats these values suitable for new situation. Both α and β will also be changed.

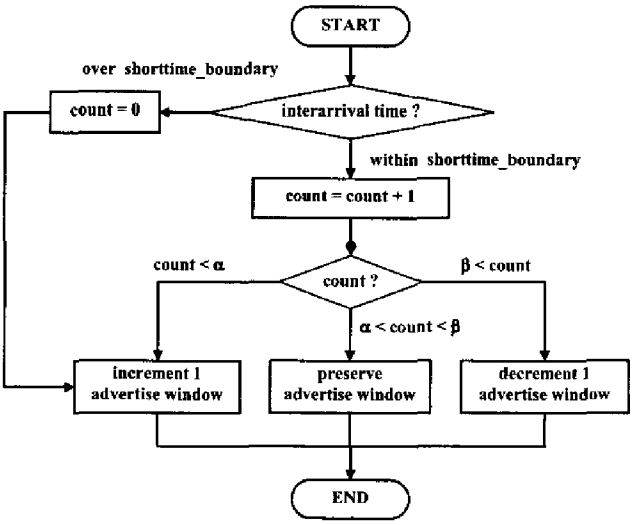


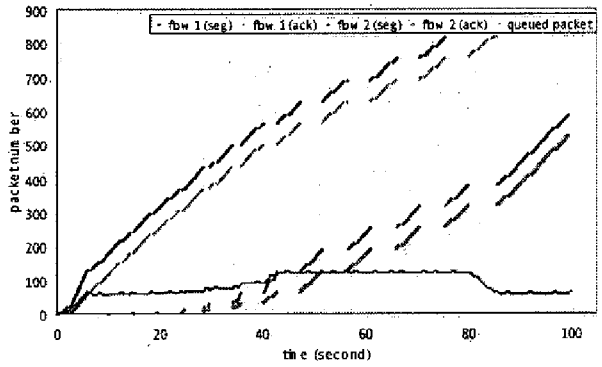
Fig. 1 Flow chart for advertise window calculation

Since we can adjust the size of the advertise window in

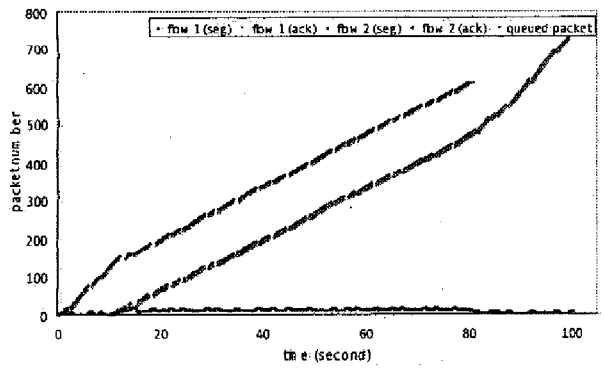
each acknowledgement, we can quickly respond to changes in workloads. We believe our scheme has fewer barriers for acceptance because it requires no modifications to the network, server or application software and requires no support from a service provider.

III. PERFORMANCE EVALUATION

The measurements were all performed over nationally deployed commercial CDMA2000-1X network in South Korea. We used a cellular phone itself. The host is equipped with Pentium 4 1.4GHz processors, 256 Mbytes of memory, and 100Mbps ethernet card. It ran Windows XP professional version. We used windump to observe sending and receiving packets at the host. Measurements have been repeated at many different time zones. In all cases, the mobile terminal was stationary even though several locations were used.



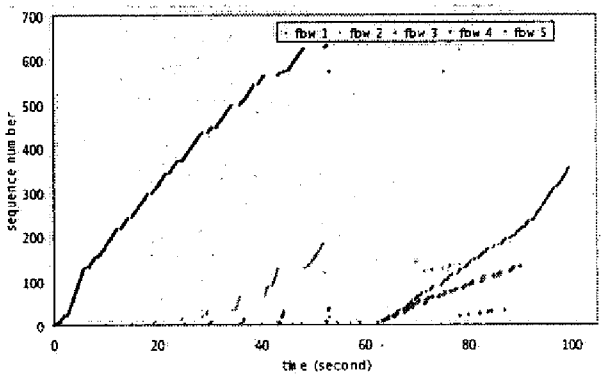
(a) TCP Reno



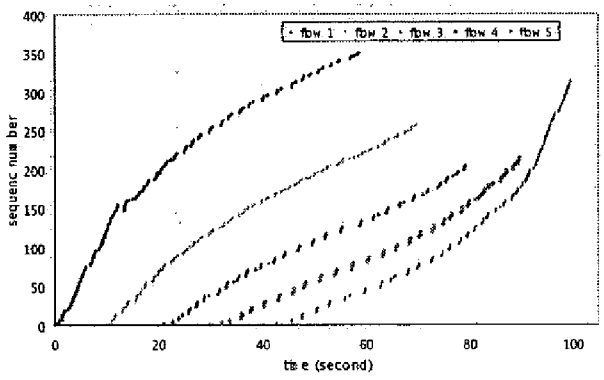
(b) Receiver Based Bandwidth Sharing

Fig. 2 Time sequence plots of two concurrent flows

Fig. 2 shows a file transfer flow 2 initiated 10 seconds after transfer flow 1 with TCP has 64 window size, recommended for improving TCP performance in wireless environment by RFC 2757, long thin network. With TCP-Reno, packets of flow 2 are queued at the bottleneck link behind a large number of flow 1 packets. As a result, flow 2 bears the full brunt of excess queueing delays due to flow 1. Until flow 1 canceled the operation at 80 seconds, flow 2 cannot achieve enough bandwidth. But, with our receiver based bandwidth sharing strategy, flow 2 can occupy equivalent bandwidth to flow 1 without losing total throughput.



(a) TCP Reno



(b) Receiver Based Bandwidth Sharing

Fig. 3 Time sequence plots of five concurrent flows

Fig. 3 shows the superiority of our proposed scheme with more concurrent flows. With five concurrent flows, TCP Reno could not support any operation except first connection due to overflow at bottleneck link. However, our strategy can provide fairness between flows and does not incur any packet loss since it maintains the queueing length minimal and stable.

IV. CONCLUSION

We proposed receiver based bandwidth sharing strategy. Our mechanism maintains the queue length at bottleneck link can be minimal and stable while achieving fully utilization of the link bandwidth without changing the TCP sender. It is impractical to change the protocol stacks of all stationary hosts merely to accommodate mobile hosts. With our approach, concurrent TCP connections can divide the wireless link bandwidth fairly. So a user can download files and check e-mails during surfing the WWW with his mobile equipment..

EXAMPLES OF REFERENCE STYLES

[1] R. Chakravorty and I. Pratt, "Performance Issues with General Packet Radio Service (GPRS)," *Journal of Communications and Networks (JCN)*. 266-281, Vol. 4, No. 2, December 2002

[2] L. S. Brakmo, S. W. O'Malley and L. L. Peterson, "TCP Vegas : New techniques for congestion detection and avoidance", in *Proc. Of the ACM SIGCOMM'94*, 24-35, August 1994