Freeze TCPv2: An Enhancement of Freeze TCP for Efficient Handoff in Heterogeneous Networks

Minu Park[1], Jaehyung Lee[1], Jahwan Koo[2], and Hyunseung Choo[1,*]

[1] School of Information and Communication Engineering, Sungkyunkwan University, Korea
{minupark,jhyunglee,choo}@skku.edu
[2] Computer Sciences Department, University of Wisconsin-Madison, USA
jhkoo@cs.wisc.edu

Abstract. The advancement of mobile communications for the last few years has accompanied an increasingly extensive and diverse array of environments for TCP applications, which has led to a host of wireless TCP approaches that may provide more stability and ensure higher performance. In a heterogeneous network, as opposed to conventional wireless networks, a mobile node performs a handoff to another network cells with a different bandwidth and latency. Conventional wireless TCP schemes, however, are not capable of properly addressing sudden changes in round trip time (RTT) and packet loss resulting from handoffs within a heterogeneous network and do experience such problems as the waste of available bandwidth and frequent packet loss. As a solution to these problems, this paper proposes Freeze TCP version 2 (v2), an enhancement of Freeze TCP designed to dynamically obtain the available bandwidth in a new network cell. Comprehensive simulation study is conducted by using a network simulator, ns-2, to compare the proposed scheme to a few other schemes, such as Freeze TCP, DEMO-Vegas, and TCP Vegas, in a heterogeneous network environment with vertical handoffs in terms of throughput per speed of the mobile node and per bit error rate of the wireless link.

Keywords: TCP, Congestion control, Heterogeneous Networks, Vertical Handoff.

1 Introduction

Transmission control protocol (TCP) is one of the most widely used Internet protocols for web browsing, email, file transfer and other applications [1], [2]. The advancement of mobile communications for the last few years has accompanied an increasingly extensive and diverse array of environments for TCP applications, which has led to a host of wireless TCP schemes that may provide more stability and ensure higher performance [3] such as TCP Westwood [4], TCP Jersey [5] and TCP New Jersey [6]. As these schemes, however, are based on homogeneous wireless networks, each of which consists of a single wireless access network, they do not deal with the performance degradation occurring in heterogeneous networks with different link

* Corresponding author.

M.J. Smith and G. Salvendy (Eds.): Human Interface, Part II, HCII 2009, LNCS 5618, pp. 448–457, 2009.

characteristics [7], [8]. A classic example of a heterogeneous network would be a GPRS-WLAN network consisting of a general packet radio service network (GPRS) [9] and a wireless local area network (WLAN) [10]. As a mobile node changes its connection point to the network from time to time in these heterogeneous and homogeneous environments, sustained connection requires handoffs. A handoff in a heterogeneous wireless network, in particular, spans heterogeneous network cells and is thus called a 'Vertical Handoff' as opposed to a 'Horizontal Handoff', which refers to a handoff of a more conventional type [11], [12].

Conventional wireless TCP schemes, however, experience serious performance degradation in vertical handoffs due to two primary reasons: The first one is an abrupt change in RTT within a heterogeneous network. To ensure reliable transfer, TCP is designed to control the transfer rate by repeatedly sending a chunk of data and receiving an acknowledgement of it. In a heterogeneous network, however, each network cell has distinct bandwidth and latency attributes, causing the sender to receive acknowledgements in different arrival patterns than in conventional networks, which could translate into sudden changes in RTT. Consequently, the sender may transfer a series of packets before recognizing and responding to changes in RTT and obtaining the available bandwidth in the new network cell. The second one is that the conventional wireless TCP schemes do not take into account the packet loss that may occur during handoffs. Moving from one access router to another, a mobile node (MN) must undergo the discovery of a new access router, registration, and authentication, resulting in handoff latency. This in turn produces packet loss resulting from retransmission timeout (RTO), causing the TCP sender to assume congestion in the network and make unwarranted cuts in the transfer rate. In consequence, this phenomenon implies an issue of inefficiency in the use of network resources.

In this respect, this paper proposes Freeze TCPv2, an RTT-based congestion control scheme designed to make efficient use of the resources of a new network and provide an appropriate transfer rate according to the network bandwidth within a heterogeneous wireless network. The proposed scheme measures *RTT* values prior to and after the occurrence of handoffs, and monitors trends in changes. It then uses buffer status checking mechanism [13] to find out how many packets sent by the sender to the receiver (MN) remain in the network. As the count of remaining packets is determined in this manner, the MN controls the transfer rate according to the condition of traffic existing in the link when it resumes communication in the new network cell, effectively minimizing the probability of network congestion and ensuring a higher transfer rate. Performance evaluation using ns-2 establishes the superiority of Freeze TCPv2 working in a heterogeneous network and demonstrates that the proposed scheme is capable of obtaining the network bandwidth in a more efficient manner and sustaining a higher transfer rate.

This paper is organized as follows: Chapter 2 introduces recent studies on TCP schemes working in wireless and cellular network environments while Chapter 3 describes the proposed algorithm. Chapter 4 is a comparative analysis of the proposed algorithm in terms of performance evaluation through simulation. Finally, Chapter 5 concludes this paper.

2 Related Work

2.1 Freeze TCP

In contrast to conventional TCP schemes, Freeze TCP [14] prevents unnecessary packet transmission during a handoff by modifying the MN that serves as a receiver in the network. If the strength of the signal received by the MN from the base station falls below a certain level, the MN predicts the occurrence of a handoff and then sends a zero window advertisement (*ZWA*) to the sender. Having received the *ZWA*, the sender sets *cwnd* maintained by itself to 0 and fixes values of all other variables as well so that it could temporarily cease transmission during a handoff. Upon the completion of the handoff, the MN sends a positive acknowledgement (*PACK*) to the sender to inform that communication is available, and the sender receives this and then restores the values of *cwnd* and other variables to their original values that it had prior to the occurrence of the handoff before finally resuming data transmission.

This process allows Freeze TCP to prevent unnecessary packet transmission and resultant performance degradation and thereby cope with handoff situations in a more proper manner than TCP Reno. The assumptions made during the development process, however, took into consideration a homogeneous network consisting of network cells with an identical bandwidth only, which lead to a difficulty in obtaining the available bandwidth after a handoff to a new network cell in a heterogeneous network. Furthermore, this scheme has another downside that a *ZWA* or *PACK* missing in the middle of a link, if any, leaves the sender unable to detect the start and end of a handoff, resulting in severe performance degradation.

2.2 DEMO-Vegas

As a MN enters a new network cell within a cellular network environment, it performs a handoff to maintain communication. A TCP Vegas' sender at this point invokes a routing optimization algorithm to resume communication and applies the RTT_{base} value as measured prior to the handoff without modifying it. In the above case, however, the new network cell has different propagation delay and bandwidth values, which implies an accuracy issue with the RTT_{base} as measured prior to the handoff. Consequently, the Δ value based on the inaccurate RTT_{base} results in the inability of TCP Vegas to properly dealing with handoff situations.

DEMO-Vegas [15], therefore, handles a change in the care-of-address (CoA) of a MN in a mobile IP environment by using a reserved bit in the header of the acknowledgement packet sent by the MN to inform the sender of the handoff. Having received an acknowledgement with the reserved bit named 'SIG', the sender is informed of the occurrence of the handoff and temporarily ceases transmission. Upon the completion of the handoff, the MN resets the reserved bit in the header. Having received this, the sender updates its RTT_{base} value and resumes transmission. DEMO-Vegas informs the sender of the occurrence of a handoff in this manner and thereby prevent unnecessary transmission, providing higher performance than TCP Vegas.

This scheme, however, has a flaw that the congestion control algorithm obtains the bandwidth available in a new network cell in the same manner as with TCP Vegas, resulting in less than better performance when the network bandwidth undergoes a significant change.

3 Proposed Scheme

3.1 Dynamic Available Bandwidth Estimation (*DABE*)

As two different network cells in a heterogeneous network are likely to provide different network bandwidths, a sender operating in the same manner as in a homogeneous network would exhibit unsatisfactory performance. For example, if the bandwidth provided by a new network cell is larger than that by the previous cell, the available bandwidth between the sender and the receiver increases. In other words, this implies that the sender makes use of more network resources and then RTT, which takes to transfer a packet, becomes shorter. In this situation, however, the sender keeps up with the same transfer rate as prior to the occurrence of the handoff because of ignoring a change of bandwidth. It in turn leads to inefficiency of the available bandwidth as well as the inability to produce a higher transfer rate. Furthermore, if the bandwidth provided by a new network cell is smaller than that by the previous cell, the sender applies the larger bandwidth, which exceeds the available bandwidth, to the new network cell as obtained prior to the occurrence of the handoff.

It means that a considerable amount of time is required in order to accurately adjust the transfer rate to the available bandwidth. It generates serious network congestion and packet loss as well as a relatively longer *RTT*. Consequently, it does not produce a higher transfer rate because of the sender's inability to properly obtain the available bandwidth within a heterogeneous environment consisting of two neighboring network cells with different bandwidth. To address this issue, Freeze TCPv2 compares the *RTT* values measured prior to and after the occurrence of a handoff to predict a change in the available bandwidth and utilizes the *ZWA* and *PACK* of Freeze TCP [16].

As in the case with Freeze TCP, if signal strength falls down to a certain level due to the mobility of the MN (○,4 in Fig. 1) while communication is on going (○,1 - ○,2 in Fig. 1), the MN assumes the occurrence of a handoff and then generates a *ZWA* before sending it to the sender (○,5 - ○,8 in Fig. 1). Having received the *ZWA* message, the sender sets its *cwnd* to 0 maintained by itself and fixes the *RTO* and other variables used to calculate the *RTO* value. This process allows the sender to halt the transmission temporarily at the occurrence of a handoff, as in the case with Freeze TCP. At this point, the sender stores the *RTT* value for the *ZWA* to RTT_{prev} in order to determine the bandwidth provided by the previous network (○,9 in Fig. 1). Upon ompletion of the handoff (○,10 in Fig. 1), the MN generates and sends a *PACK* message to inform the sender of the availability of communication (○,11 - ○,14 in Fig. 1), which enables the sender to resume packet transmission. In addition, the sender, having

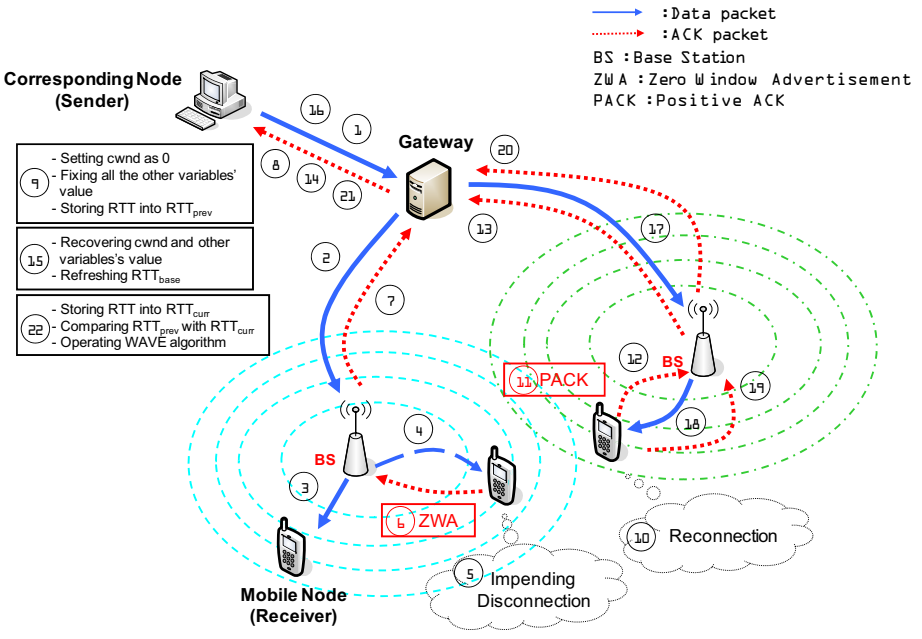


Fig. 1. DABE mechanism in Freeze TCPv2

received the *PACK*, restores the values of the *cwnd* and other variables to their original values stored before the handoff (○,15 in Fig. 1).

The sender stores the *RTT* value for the first packet transmitted after the handoff (○,16 - ○,21 in Fig. 1) as RTT_{curr} in order to monitor the changes occurring during the handoff (○,22 in Fig. 1). If the RTT_{prev} is smaller than the RTT_{curr}, the sender ascribes the increase of *RTT* to the decreased available bandwidth in the new network cell as well as to certain difficulties in packet transmission. The sender linearly increases the window size saved prior to the handoff. If the RTT_{prev} is larger than the RTT_{curr}, however, the bandwidth provided by the new network cell is increased and there is no problem in packet transmission with the decreased RTT value. The sender thereafter adjusts the transfer rate as in Eq. (1), according to the change of *RTT* and the size of the packet that has been sent.

$$cwnd + = \frac{PacketSize}{|RTT_{prev} - RTT_{curr}|} \tag{1}$$

As shown above, Freeze TCPv2 is capable of accurately adjusting the transfer rate according to the change in bandwidth while monitoring the change of *RTT* during a handoff. Even in a homogeneous network having a constant bandwidth, detecting any change in the bandwidth provided by a new network cell makes it easy to determine the available bandwidth.

3.2 Window Adjustments Based on Vegas Estimator (*WAVE*)

Since the sender and the receiver, MN, cannot be connected with each other during the handoff, the data packets, which are sent before the handoff, remain in the network links. As a result, packets buffered in the network links decline the accuracy of RTT_{curr} calculated after the handoff. Freeze TCPv2 adjusts the transmission rates and checks the network congestion status based on the buffer checking algorithm of TCP Vegas to solve this problem.

Freeze TCPv2 keeps two variables, the RTT_{base} measured as a minimum *RTT*, and the current *RTT* measured from the last transmitted packet. These two values are used to deduce the Expected and Actual using the following Eq. (2). The Expected indicates the transmission rate when the network status is stable, and the Actual implies the sending rate in the current network status. The difference between the two values is represented as Δ, and is calculated by the following Eq. (3) which indicates the amount of packets buffered in network elements. In other word, Δ implies the congestion status in the network. Moreover, Freeze TCPv2 defines α and β, which are the minimum and the maximum numbers of the network buffers, respectively. Using these variables, Freeze TCPv2 determines the network condition and remaining packets in the network links, subsequently adjusting its transmission rates based on the network congestion status after the handoff by calculating α, β, and Δ.

$$Expected = \frac{WindowSize}{RTT_{base}} \qquad Actual = \frac{WindowSize}{RTT} \tag{2}$$

$$\Delta = (\frac{WindowSize}{RTT_{base}} - \frac{WindowSize}{RTT}) \times RTT_{base} \tag{3}$$

When the sender receives a *PACK*, it refreshes the RTT_{base} as RTT_{prev} so as to check the number of remain packets buffered; this algorithm is adapted from DEMO-Vegas. Since the communication between the sender and the MN is connected during the handoff, the RTT_{base} means the minimum *RTT* from the previous network cell. Here, the RTT_{base} cannot be applied for the variable in the new network because it causes the difficulty for the sender to determine the available bandwidth well. As a result, for checking the network condition through TCP Vegas, the proposed scheme refreshes RTT_{base} as soon as receiving the *PACK*. After refreshing the RTT_{base}, the sender calculates Δ and compares it to α and β. If Δ is less than α, the sender considers the network status to be stable because the network buffer is empty and it activates its *DABE* algorithm. If Δ is larger than β, the sender assumes that there are many packets remaining in the intermediate routers, so it counts the number of ACK packets which the sender received during one window transmission time. This algorithm is based on TCP Westwood. The reason for this is that TCP Westwood is considered to be a good algorithm for ameliorating the network congestion in wireless networks. As a result, Freeze TCPv2 uses the ABE algorithm from TCP Westwood to take care of the congestion in network links. If Δ is between α and β, it just initiates the original TCP Vegas (line 26 in Fig. 4). In the case that the sender receives a normal ACK without *ZWA* or *PACK*, it follows the congestion control algorithm of TCP Reno as Freeze TCP.

4 Performance Evaluation

4.1 Simulation Setup

We evaluate the performance of Freeze TCPv2 with metrics such as the throughput, and fairness, using the ns-2 network simulator [16]. We examine the performance of the proposed scheme in a heterogeneous network topology with TCP Vegas, Freeze TCP, and DEMO-Vegas. The simulation topology is described in Fig. 2.

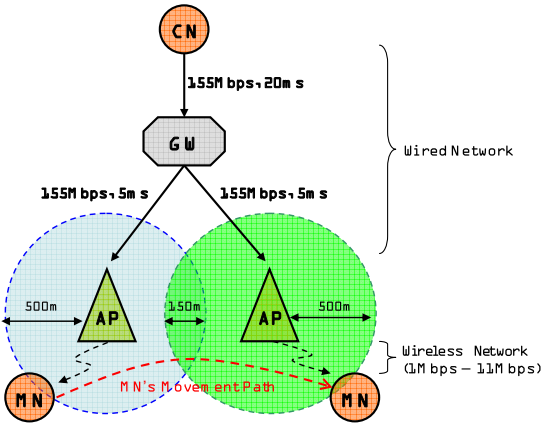


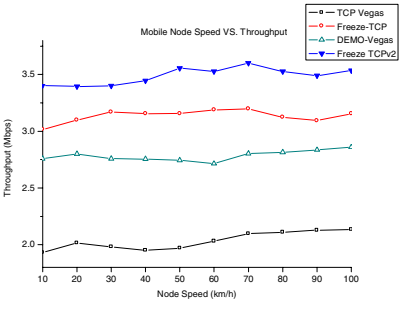
Fig. 2. Simulation topology

All simulations assume that a handoff occurs only once, and in an overlapping area of network cells. The corresponding node (CN) as a sender transmits the data to the MN though a wired connection from the CN to the access point (AP). The radio coverage from the AP is 500 m, and the MN moves from the old network cell to the new network cell with various speeds. At this time, the AP forwards the packets from the gateway (GW) to the MN through wireless links. In this scenario, we evaluate the throughput under the speed of the MN and the wireless link error rates. The queue sizes for all of the nodes are set to 20, and a single TCP connection running a long-live FTP application delivers the data from the sender to the receiver. The two buffer thresholds of Freeze TCPv2, α and β, are set to 14 and 16, as done in [17].

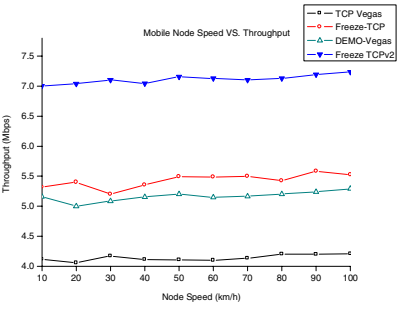
4.2 Throughput under Various Speeds of the MN in Heterogeneous Networks

In this simulation, we evaluate the performance in terms of the speed of the MN in heterogeneous networks. These scenarios are chosen to describe two cases where the MN moves from a lower bandwidth network cell to a higher bandwidth network cell (6Mbps to 11Mbps), and from a higher bandwidth cell to a lower bandwidth cell (6Mbps to 1Mbps). We measure and compare the throughput during the movement of the MN, where the speed of the MN speeds up from 10 km/h to 100 km/h. Fig. 3 depicts the results of a simulation lasting over 180 seconds, an average of 10 times.

Comparing with TCP Vegas, Freeze TCP, and DEMO-Vegas, Freeze TCPv2 outperforms by 40%, 8% and 14%, respectively, when the MN moves from a previous network cell with 6 Mbps to a new network cell with 1 Mbps (Fig. 3). Moreover, in the case where the MN moves from an old network cell with 6 Mbps to a new network cell with 11 Mbps, Freeze TCPv2 shows a performance improvement of 55% over TCP Vegas, 13% over Freeze TCP, and 17% over DEMO-Vegas. As for two plottings, the shorter handoff durations they experience, the more throughput of all schemes increases. In this situation, Freeze TCPv2 recognizes the change of network bandwidth, and subsequently obtains the available bandwidth of the new network cell by activating the *DABE* algorithm.



(a) When MN moves from 6 Mbps network cell to 1 Mbps network cell



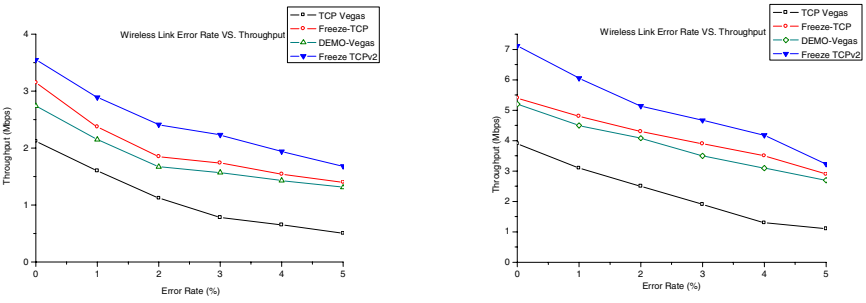
(b) When MN moves from 6 Mbps network cell to 11 Mbps network cell.

Fig. 3. Throughput under various speeds of the MN

4.3 Throughput under Various Wireless Link Error Rates in Heterogeneous Networks

In this simulation, we investigate the throughput of both the proposed scheme and the others in terms of wireless link error rates in heterogeneous networks. With an altered network bandwidth, from 6 Mbps to 1 Mbps and from 6 Mbps to 11 Mbps, we evaluate the throughput of Freeze TCPv2, Freeze TCP, DEMO-Vegas and TCP Vegas with various wireless link error rates, between 1% and 5%. In this simulation, the speed of the MN is fixed to 50 km/h. The simulation results show that Freeze TCPv2 outperforms the other TCP schemes, achieving 20% - 50% improvements in goodput as shown in Fig. 4. As the wireless link error rates increases, the throughput of all of the schemes also decreases. In particular, when the MN moves from a 6 Mbps network to a 1 Mbps network, Freeze TCPv2 outperforms Freeze TCP by 7%, DEMO-Vegas by 9%, and TCP Vegas by 48%, respectively. In addition, Freeze TCPv2 shows a better performance than TCP Vegas, Freeze TCP and DEMO-Vegas by 58% , 12% and 11% when the MN moves from a previous network cell with 6 Mbps to a new network cell with 11 Mbps.

These trends are owing to the *WAVE* algorithm of Freeze TCPv2, because the sender controls the network congestion through investigating the remaining packets in the network. Consequently, it is possible for the sender to obtain an appropriate transmission rate sustaining a higher performance.



(a) When MN moves from 6 Mbps network cell to 1 Mbps network cell.

(b) When MN moves from 6 Mbps network cell to 11 Mbps network cell.

Fig. 4. Throughput in terms of wireless link error rates

5 Conclusion

In this paper, we propose Freeze TCPv2 using *DABE* and *WAVE* algorithms in heterogeneous networks and homogeneous networks. In the *DABE* algorithm, the sender checks the differentiation between RTT_{pre}, prior to handoff, and RTT_{curr}, after handoff. Afterwards, the sender adjusts its sending rates according to the estimated bandwidth changes. Ordinarily, the sender and the MN cannot maintain their communication during the handoff. To solve this, Freeze TCPv2 using the *WAVE* algorithm enables the sender to keep on the stable transmission by investigating the remaining packets in the network links. As a result, the sender quickly adjusts to the appropriate available bandwidth without any network congestion. A simulation result using the ns-2 simulator demonstrates that Freeze TCPv2 shows more better performance over TCP Vegas, Freeze TCP, and DEMO-Vegas, regardless of the network characteristics. Finally, Freeze TCPv2 satisfies the criteria of the Fairness index in terms of sharing the network bandwidth equally.

Acknowledgments. This research was supported by the MKE(Ministry of Knowledge Economy), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA(Institute of Information Technology Advancement) (IITA-2009-(C1090-0902-0046)), and This work was supported by the Korea Research Foundation Grant funded by the Korean Government (KRF-2008-314-D00296).

References

1. Postel, J.: Transmission Control Protocol. RFC 793 (1981)
2. Allman, M., Paxson, V., Stevens, W.: TCP Congestion Control. RFC 2581 (1999)
3. Tian, Y., Xu, K., Ansari, N.: TCP in Wireless Environments: Problems and Solutions. IEEE Radio Communications 43(3), S27–S32 (2005)
4. Casetti, C., Gerla, M., Mascolo, S., Sanadidi, M.Y., Wang, R.: TCP Westwood: Bandwidth Estimation for Enhanced Transport over Wireless Links. ACM/IEEE MobiCom, 287–297 (July 2001)

5. Xu, K., Tian, Y., Ansari, N.: TCP-Jersey for Wireless IP Communications. IEEE Journal of Selected Areas in Communications 22(4), 747–756 (2004)
6. Xu, K., Tian, Y., Ansari, N.: Improving TCP Performance in Integrated Wireless Communications. Networks Computer Networks 47, 219–237 (2005)
7. Hansmann, W., Frank, M., Wolf, M.: Performance Analysis of TCP Handover in a Wireless/Mobile Multi-Radio Environment. In: Proc. of IEEE LCN 2002, November 2002 (2002)
8. Chakravorty, R., Vidales, P., Subramanian, K., Pratt, I., Crowcroft, J.: Practical Experiences with Wireless Integration using Mobile IPv6. ACM Mobile Computing and Communication Review 7(4) (October 2003)
9. Ghribi, B., Logrippo, L.: Understanding GPRS: the GSM Packet Radio Service. Computer Network 34(5) (November 2000)
10. Crow, B., Widjaja, I., Kim, J., Sakai, P.: IEEE 802.11 Wireless Local Area Networks. IEEE Communication Magazine (September 1997)
11. Liao, W., Kao, C., Chien, C.: Improving TCP Performance in Mobile Networks. IEEE Transactions on Communications 53(4), 569–571 (2005)
12. Wu, X., Chan, M., Ananda, A.: TCP HandOff: A Practical TCP Enhancement for Heterogeneous Mobile Environments. IEEE ICC, 6043–6048 (June 2007)
13. Brakmo, L., Peterson, L.: TCP Vegas: End to End Congestion Avoidance on a Global Internet. IEEE Journal of Selected Areas in Communications 13(8), 1465–1480 (1999)
14. Goff, T., Moronski, J., Phatak, D., Gupta, V.: Freeze-TCP: A True End-to-End TCP Enhancement Mechanism for Mobile Environments. In: Proc. of the IEEE INFOCOM 2000, vol. 3, pp. 1537–1545 (2000)
15. Ho, C., Chan, Y., Chen, Y.: An Efficient Mechanism of TCP-Vegas on Mobile IP Networks. IEEE INFOCOM, 2776–2780 (March 2005)
16. UCB/LBNL/VINT Network Simulator, `http://www.isi.edu/nsnam/ns`
17. Jain, R., Chiu, D., Hawe, W.: A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Computer Systems. DEC Research Report TR-301 (September 1984)