Computer Networks 52 (2008) 3062–3074

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Improving TCP performance in heterogeneous mobile environments by exploiting the explicit cooperation between server and mobile host ☆

Xiuchao Wu *, Mun Choon Chan, A.L. Ananda

Communication and Internet Research Lab, School of Computing, National University of Singapore, Computing 1, Law Link, Singapore 117590, Singapore

ARTICLE INFO

Article history:
Received 2 September 2007
Received in revised form 27 February 2008
Accepted 12 July 2008
Available online 5 August 2008

Responsible Editor: G. Morabito

Keywords:
TCP
Handoff
Heterogeneous wireless networks

ABSTRACT

In recent years, many different kinds of wireless access networks have been deployed and become inseparable parts of the Internet. But TCP, the most widely used transport protocol of the Internet, was designed for stationery hosts. It faces severe challenges when user moves around in these networks and handoff occurs frequently. In this paper, we investigate the potential benefits of bringing explicit cooperation between TCP server and mobile host.

For this purpose, TCP HandOff (TCP-HO), a practical end-to-end mechanism, is designed for improving TCP performance in heterogeneous mobile environments. TCP-HO assumes that a mobile host is able to detect the completion of handoff immediately and has a coarse estimation of new wireless link's bandwidth. When a mobile host detects handoff completion, it will immediately notify the server through two duplicate ACKs, whose TCP option also carries the bandwidth of new wireless link. After receiving this notification, the server begins to transmit immediately and keeps updating *ssthresh* according to the bandwidth from mobile host and its new RTT samples. This update will end after four RTT samples or after congestion is detected.

TCP-HO has been implemented in FreeBSD 5.4. Experimental results indicate that in heterogeneous mobile environments, TCP-HO can improve TCP performance a lot without adversely affecting cross traffic even when mobile host only has a coarse estimation of new wireless link's bandwidth. Considering that more and more users are accessing the Internet through heterogeneous wireless networks and mobile host could have a coarse estimation of wireless link's bandwidth, it should be worthwhile to change both server and mobile host for improving TCP performance.

1. Introduction

In recent years, many kinds of wireless networks, such as cellular network (WCDMA [1], GPRS [3], etc.), Wireless LAN (Wi-Fi [4–7], HiperLAN [8], etc.), and Wireless MAN (WiMax [9]), have been deployed and have become integral parts of the Internet. These wireless networks complement each other in terms of coverage, bandwidth, latency, etc. and form a heterogeneous mobile environment. These networks along with portable and affordable computing devices, such as laptops, PDAs, and smart phones, enable the wide spread and affordable mobile Internet access. But the lossy wireless links and mobile hosts violate the assumptions of TCP [15], the most widely used transport protocol of the Internet. And TCP performs very bad when users move around in these wireless networks.

TCP was designed for reliable links. When packets are transmitted on lossy wireless link, they are corrupted frequently. These corrupted packets are wrongly regarded as congestion signals and TCP sender reduces its sending

☆ A preliminary version of this paper was presented in the IEEE ICC 2007 conference, Glasgow, Scotland.

* Corresponding author. Tel.: +65 97682558.
E-mail addresses: wuxiucha@comp.nus.edu.sg (X. Wu), chanmc@comp.nus.edu.sg (M.C. Chan), ananda@comp.nus.edu.sg (A.L. Ananda).

1389-1286/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.anpedi.2008.07.002

rate unnecessarily. Hence, TCP can not fully utilize the precious bandwidth of wireless link and provide very bad performance to applications. Many mechanisms, such as I-TCP [28], Snoop [17], TCP Veno [18], TCP Hack [19], and TCP ELN [20–22], had been proposed for enhancing TCP performance over lossy wireless links. Based on the above investigations, new wireless networks normally adopt FEC (forward error correction) and/or ARQ (automatic retransmission request) in link layer with the aim of hiding lossy characteristic of wireless link. For example, RLC [2], the link layer protocol of GPRS/UMTS, adopts link layer ARQ. Link layer ARQ is also used by the MAC layer of IEEE 802.11 [4]. Hence, the current wireless networks have already been reliable wireless networks.

In reliable wireless networks, link layer ARQ may bring large bandwidth and delay variation. This problem has been solved by regulating ACK and TCP window field at base station [23,24].

The challenges brought by lossy wireless link had been well studied. But TCP was also designed for stationery hosts. When a user moves around in these heterogeneous wireless networks, TCP performance is very poor due to the challenges brought by handoff. In this paper, we will investigate how to enhance TCP performance when users move around in the current heterogeneous mobile environments.

Within the current heterogeneous mobile environments, many kinds of handoff may occur and they can be classified in many different ways. For example, in [25], handoff is classified into vertical handoff and horizontal handoff according to the techniques used by the wireless networks. In this paper, we classify handoff from TCP point of view. More specifically, since the BDP (bandwidth-delay product) of a network path affects TCP performance significantly, we classify handoff according to BDP of old network path (before handoff) and that of new network path (after handoff).

Handoff is first classified into HH (horizontal handoff) and VH (vertical handoff) based on whether BDP changes significantly during handoff. Within HH, BDP of new and old paths do not vary appreciably. For VH, the difference in BDP is large. According to the value of BDP, HH is further divided into L-HH and H-HH. Within L-HH, BDPs of both paths are low. Within H-HH, BDPs of both paths are high. According to the direction of BDP change, VH is further divided into D-VH (downward-VH) and U-VH (upward-VH) [26]. During all kinds of handoff, there is normally a long disconnection time, which means a period of zero BDP. Fig. 1 shows BDP fingerprint of our four kinds of handoff.

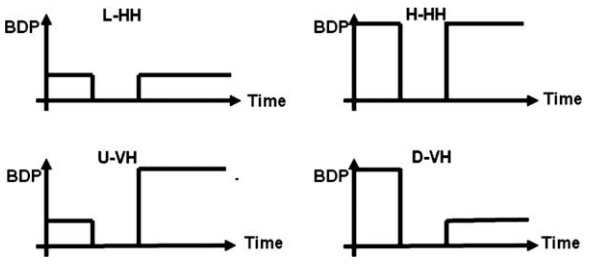


Fig. 1. BDP fingerprint of different kinds of handoff.

Although TCP reacts to changes of network capacity through congestion control, abrupt BDP changes during handoff can still bring severe challenges. In this paper, we study the potential benefits of bringing explicit cooperation between server and mobile host. TCP-HO, a practical end-to-end mechanism based on explicit cooperation, is proposed for solving the challenges brought by all kinds of handoff with the aim of improving TCP performance in heterogeneous mobile environments.

The rest of this paper is organized as follows. In Section 2, the challenges faced by TCP during each kind of handoff are analyzed. Section 3 presents related work, and the details of our proposal, TCP handoff, are presented in Section 4. The testbed and experimental results are described in Section 5, and the issues related with wireless link bandwidth estimation are discussed in Section 6. This paper is concluded in Section 7.

2. TCP during handoff

TCP, the most widely used transport protocol, uses congestion control to probe network capacity and keep network stability. Within TCP congestion control [27], TCP sender maintains two variables, *cwnd* (which determines the current sending rate) and *ssthresh* (which is a coarse estimation of the network path's BDP).

When *cwnd* is less than *ssthresh*, the sender is in slow start state. For each new acknowledgement, *cwnd* is increased by one segment (exponential increase) so that TCP sender can quickly probe network capacity. When *cwnd* is larger than *ssthresh*, the sender is in congestion avoidance state. And for each RTT (round trip time), *cwnd* is increased by one segment (linear increase) so that the sender can still probe network capacity and avoid frequent network congestion.

TCP regards segment loss as the signal of network congestion. When congestion is detected, *ssthresh* is reduced to half of the current *cwnd*. *cwnd* is reduced to one if congestion is detected by timeout of retransmission timer. If congestion is detected by 3*DupACK* (three duplicate acknowledgement), *cwnd* is reduced to half of the current *cwnd*. When a lost segment is detected by retransmission timer and has to be retransmitted more than once, TCP retransmission timer adopts an exponential back-off algorithm with a maximal *timeout* value (64 s).

With TCP congestion control and its ACK-based self-clock, TCP sender can react to slow changes of network capacity and achieve good throughput. However, during handoff, BDP changes abruptly and long disconnection destroys TCP's self-clock. In the following paragraphs, we will analyze the problems faced by TCP during all kinds of handoff.

First, unless seamless handoff is implemented, *many segments are lost during handoff*. For heterogeneous wireless data access networks, availability of seamless handoff is unlikely due to its complexity. As a result, when handoff occurs, segments will be discarded at the base station of previous wireless link.

Second, *after handoff completion, TCP sender does not know this fact and still sillily waits for timeout*. Due to TCP's exponential retransmission timer back-off and long disconnection time of handoff, the sender may wait for quite a long time and precious bandwidth of new wireless link can not be utilized.

Third, *after handoff completion, the sending rate is quite slow for a long time*. During handoff, timeout occurs at TCP sender, *ssthresh* is reduced to half of the *cwnd*, and *cwnd* is reduced to one. TCP sender needs some time to fully utilize the bandwidth of new link. Especially when *ssthresh* is much less than BDP of new path, TCP sender will enter into congestion avoidance state too early. This problem is worse in H-HH than in L-HH, and it is much worse in U-VH because the BDP of new path is much larger than *ssthresh*, a coarse estimation of old path's BDP.

Fourth, *TCP sender may over-shoot the new wireless link after completion of D-VH*. Within D-VH, the BDP of new path is much less than *ssthresh*. The exponential *cwnd* increase in slow start state may cause multiple segment loss, trigger timeout, and adversely affect TCP throughput.

Table 1 summarizes the problems faced by TCP during our four kinds of handoff that may occur in the current heterogeneous mobile environments.

3. Related work

The root of TCP's poor performance during handoff is that congestion control is carried out by the server and handoff is only known to mobile host. Based on this understanding, we classify the related works according to where adaptations take place.

(1) Network-centric approaches: I-TCP [28] and MTCP [29] split a connection between a server and a mobile host at the base station. Handoff is entirely handled by mobile host and base stations. TCP server continues to send segments to the old base station during a handoff. These segments will be buffered at the old base station and forwarded to the new base station after handoff is completed. In this kind of approaches, handoff is hidden from TCP servers. But within heterogeneous mobile environments, wireless networks may use different techniques and belong to different administrative domains. These proposals may not be practical again. In addition, base stations need to maintain large buffers within high-speed wireless networks. M-TCP [30] only works at the base station. Base station holds back the ACK of the last byte. When it detects handoff occurrence, it sends back the last byte's ACK with *zero window*, which will force TCP sender to enter into persist mode, thus the values of *cwnd* and *ssthresh* are frozen. When a new base station detects handoff completion, it immediately notifies TCP sender to resume transmission with the frozen *cwnd* and *ssthresh*. M-TCP is a good network-centric solution. But the overhead of base station is too large and increases with the speed of wireless networks. More severely, M-TCP, I-TCP, and MTCP cannot work with IPSEC [31].

(2) Receiver-centric approaches: RCP [32], which moves congestion control to mobile host, is the most radical proposal of this category. Since mobile host knows handoff and carries out congestion control, it can solve all problems brought by handoff. But it is too dangerous to let mobile users decide the sending rate of fixed servers, especially in today's Internet. Freeze-TCP [33] and TCP ACK-Pacing [26] are two other receiver-centric proposals. They change mobile host's behaviors during handoff with the aim of driving TCP server's congestion control to probe network capacity well. Freeze-TCP and TCP ACK-Pacing are end-to-end proposals and only mobile hosts need to be changed. Hence, they can work with IPSEC and have good deployability. But in the current heterogeneous mobile environments, Freeze-TCP and TCP ACK-Pacing also have their own problems. Freeze-TCP shifts the tasks of M-TCP's base station to mobile host and avoids holding the ACK of the last byte. Freeze-TCP assumes accurate handoff prediction so that *zero window* can arrive the server and old wireless link will not be under-utilized. But it is very hard to accurately predict handoff in heterogeneous wireless networks. In addition, after handoff completion, the server begins to transmit with previous frozen *cwnd* and *ssthresh*. This mechanism may work well during L-HH. But it violates the congestion control principle that data sender should re-probe network capacity after a long idle time [36]. Considering the faster and faster wireless networks, it may generate large data burst and harm cross traffics. In addition, Freeze-TCP does not consider the challenges brought by D-VH and U-VH too. TCP ACK-Pacing considers the challenges brought by U-VH and D-VH. It assumes that a mobile host knows RTT of the new path and the bandwidth of the new wireless link. Hence, it knows the BDP of the new path when the wireless link is the bottleneck. But mobile host normally acts as TCP receiver and does not keep measuring RTT. The ACK-generation algorithm of the mobile host is changed in order to drive the server's *cwnd* converge to new path's BDP quickly. However, less ACKs after D-VH may cause TCP sender to generate large data burst and break TCP's self clock. More ACKs after U-VH may consume precious uplink bandwidth of asymmetric wireless link. In addition, TCP ACK-Pacing cannot work at all if TCP byte-counting [34] is used by the server. When the server and network allow TCP ACK-Pacing, it can be exploited by misbehavior users to get unfairly high throughput.

Table 1
Problems brought by different kinds of handoff

	Segment loss	Silly Waiting	Slow start	Over-shooting
L-HH	Yes	Yes	A little	No
H-HH	Yes	Yes	Severe	No
U-VH	Yes	Yes	Very severe	No
D-VH	Yes	Yes	Very little	Yes

Since the root of TCP's poor performance during handoff is that congestion control is carried out by the server and handoff is only known to mobile host, it should be promising to introduce explicit cooperation between server and mobile host. In the following section, TCP HandOff (TCP-HO), a practical end-to-end TCP enhancement based on explicit cooperation, is designed for heterogeneous mobile environments. Within TCP-HO, a mobile host reports handoff information to the server. And the server is responsible to well utilize wireless links based on mobile host's feedback. Hence, TCP-HO is a *sender + receiver approach* and both TCP end-points need to be changed.

4. TCP HandOff mechanism

During handoff, the ideal solution is to let TCP sender stop transmitting one RTT before handoff occurrence and immediately begin to transmit ($cwnd = BDP$ of new path) after handoff completion. By this way, TCP sender can fully utilize the old and new wireless links and avoid any segment loss. However, this is not a practical solution. Below is the design principles and mechanism details of TCP-HO.

4.1. Design principles

(1) *Assumptions made must be reasonable*. Since it is very hard to accurately predict handoff in heterogeneous wireless networks, mobile host of TCP-HO does not predict handoff and notify the server. Hence, TCP-HO can not avoid segment loss.
(2) *The mechanism should be deployable on the current Internet*. In order to avoid complicating network infrastructure and work with IPSEC, TCP-HO should be a truly end-to-end mechanism. In addition, it should be easy to implement TCP-HO in the existing TCP codes.
(3) *The mechanism should not bring new security problems*. Especially, when TCP sender accepts the notification of new link's bandwidth, it must use this information scrupulously and discard it quickly.
(4) *The mechanism should be friendly to cross traffic*. With the increase of mobile users and the increase of wireless link's bandwidth, TCP-HO must consider its effects on network stability and cross traffic. After handoff completion, instead of sending with full speed, TCP-HO must probe network capacity as same as normal TCP. In order to accelerate capacity probing, *ssthresh* can be set according to BDP of new path.
(5) *The server should perform most of the work*. Considering that mobile host is usually resource constraint, most work of TCP-HO should be carried out by the server.

4.2. Details of TCP-HO

There are only two assumptions in TCP-HO. Firstly, a mobile host can immediately know the completion of handoff. This knowledge can be acquired easily by some cross-layer implementations. Secondly, a mobile host has a *coarse* estimation of new wireless link's bandwidth. This assumption is reasonable since mobile host can estimate bandwidth based on the type of wireless interface and the corresponding bandwidth estimation mechanisms, such as [40,41]. These bandwidth estimation mechanisms will be discussed further in Section 6 when the effects of bandwidth estimation error are evaluated. Based on previous studies [38,39], the knowledge of bottleneck link's bandwidth is helpful for improving TCP performance. By exploiting this knowledge carefully, the performance improvement of TCP-HO should be better than that of *receiver-centric* approaches.

TCP-HO extends TCP by including two new TCP options (see Fig. 2). TCP capability option is an enabling option used in SYN segments. Each bit of its 16-bits capability mask can be used to negotiate one TCP capability. TCP-HO is negotiated through the last bit. TCP capability option is designed to save precious option space of SYN segment. As for TCP bandwidth option, it is used by mobile host to notify the server about the completion of handoff and the bandwidth of new wireless link. The unit of bandwidth is Kbps.

Within TCP-HO, when a mobile host gets to know the completion of handoff, it immediately sends out two duplicate ACKs with TCP bandwidth option which carries BWD_{new} (the bandwidth of new wireless link). Two duplicate ACKs make the notification more robust in lossy networks. If more than two duplicate ACKs are sent out, they may trigger TCP sender's fast retransmit and fast recovery [27].

When the server receives one ACK with TCP bandwidth option, it sets $ssthresh = BWD_{new} * SRTT_{old}$ and enters into HO-ADJUST state (see Fig. 3). Here, the server only responds to TCP bandwidth options received just after a *timeout* event with the aim of avoiding misbehavior users to exploit this option for acquiring higher throughput. In HO-ADJUST state, for each new RTT sample, the server keeps updating *ssthresh* according to BWD_{new} and the smooth average of new RTT samples.

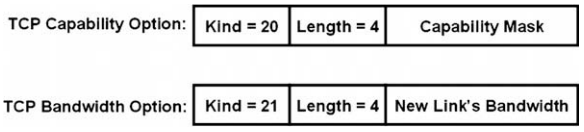


Fig. 2. TCP options for TCP-HO.

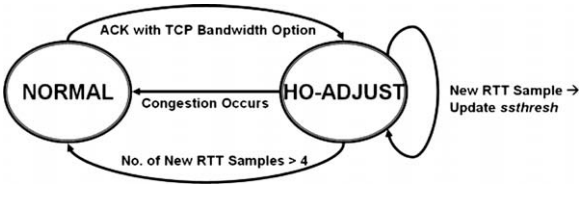


Fig. 3. State transition diagram of TCP-HO sender.

After congestion is detected, the server returns to normal state and works as normal TCP. According to source codes of FreeBSD 5.4, four RTT samples can give a quite accurate estimation of average RTT. In addition, the bandwidth of wireless link may change with time and CPU is consumed in HO-ADJUST state for updating *ssthresh*. Hence, TCP-HO server also returns to normal state after four new RTT samples. Fig. 3 shows the state transition diagram used by TCP-HO sender.

Fig. 4 shows the time vs. sequence no. graph of TCP Newreno [35] and TCP-HO during our four kinds of handoff which occur in the same mobile scenario. For better comparison, we implemented TCP-HO in NS2 [10] and generated the two graphs. This figure shows that TCP Newreno does suffer the problems pointed out by us in Table 1. It also indicates that TCP-HO solves all problems except segment loss and achieves much higher throughput than TCP Newreno.

Within [37], a similar idea was presented and evaluated by simulation. But its network scenario is different with ours. The authors assume that the mobile host is the sender of data.

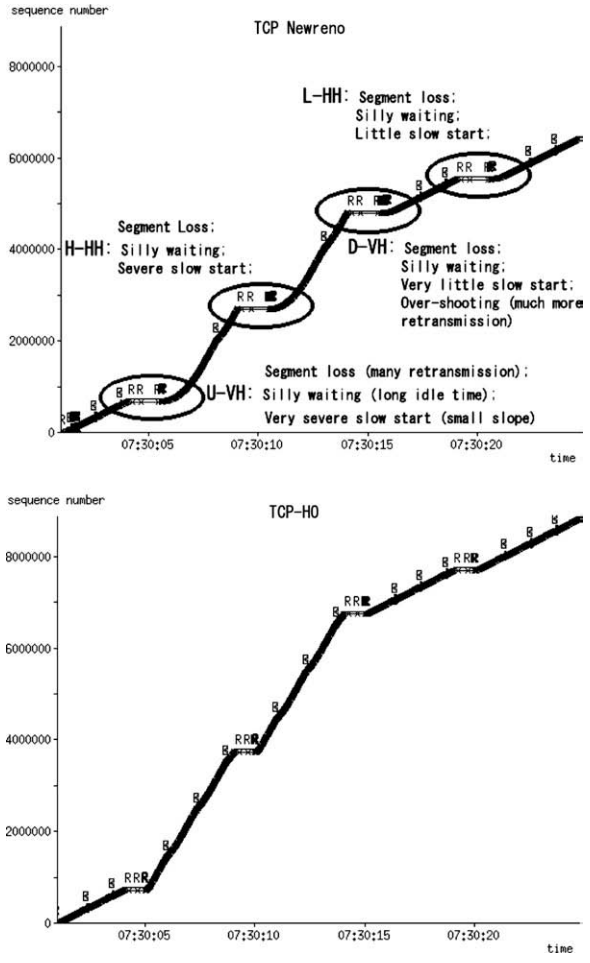


Fig. 4. Time vs. sequence No. graph of TCP Newreno and TCP-HO under four kinds of handoff occurred in the same mobile scenario.

5. Performance evaluation

With *immediate transmission* [14], TCP-HO can reduce the disconnection time of data flow. This metric is very important to interactive applications, such as Telnet and WWW. The improvement of TCP-HO can be analyzed as follows.

Since TCP's retransmission timer adopts binary exponential back-off algorithm with a maximal *timeout* value (64 s), when link disconnection time is so long that the *timeout* reaches 64 s, the improvement is between 0 and 64 s. Otherwise, the improvement can be presented by Fig. 5. If we denote the time between handoff occurrence and the $N-1_{\text{th}}$ timeout as T, the time between handoff occurrence and the N_{th} timeout is $2T$. Handoff may be completed at any time between T and $2T$. If we denote link disconnection time during handoff as x, the improvement on data flow disconnection time can be approximately expressed as $\frac{2T-x}{x}$. And the average can be calculated by the following equation:

$$\frac{1}{T}*\int_T^{2T}\frac{2T-x}{x}\,dx=\frac{[2T*(\ln(x)|_T^{2T})-(x|_T^{2T})]}{T}$$
$$=\frac{2T*[\ln(2T)-\ln(T)]-(2T-T)}{T}$$
$$=\frac{2T*\ln(2)-T}{T}=2\ln(2)-1\approx 0.3863.$$

Hence, the improvement is about 0.3863 of the link disconnection time. Freeze-TCP can achieve the same improvement in this metric.

In this paper, we focus on evaluating and comparing these proposals in another important metric, the throughput of a long-lived TCP connection. For the purpose of performance evaluation, TCP-HO is implemented in FreeBSD 5.4, a controlled network environment is set up, and many experiments are carried out.

5.1. Testbed setup

Within the testbed shown in Fig. 6, a mobile host is connected with handoff emulator by cross-cable and all other computers are connected by two virtual LANs of a 3COM Super Stack II Switch 3900.

FreeBSD 5.4 is installed on all of these computers. The connection between server and mobile host is the connection to be investigated. Cross traffic is generated between cross traffic generator and cross traffic sink in order to investigate the friendliness of TCP-HO. Iperf [11] is used

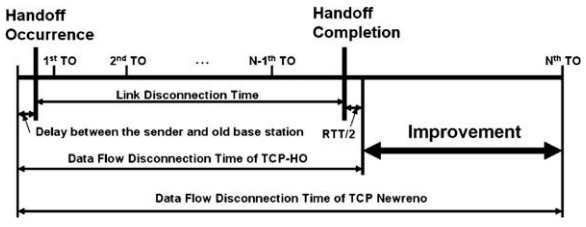


Fig. 5. Improvement of TCP-HO on data flow disconnection time.

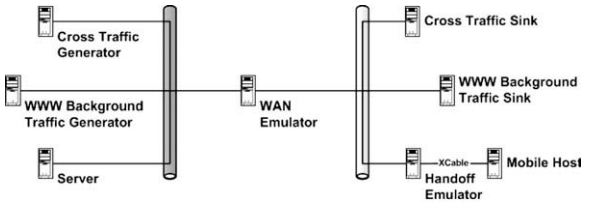


Fig. 6. Testbed.

by server and cross traffic generator for generating traffics and measuring their throughput. WWW background traffic generator and WWW background traffic sink are responsible to generate some Internet-like background traffic. Apache is installed on WWW background traffic generator and surge [12] is used to emulate WWW traffics generated by 10 users.

Dummynet [13] is used by WAN emulator to emulate the delay and bandwidth of a wide area network. The bandwidth of WAN is set to 10 Mbps. Considering the small number of connections in our experiments, the queue at WAN emulator is set to 20 packets. As for the delay of WAN, it is changed in different experiments in order to emulate that mobile host accesses servers that locate at different places.

In order to better emulate the bandwidth and delay of different wireless links, instead of mobile host, a separate computer (handoff emulator) is used and its kernel is rebuilt with finer timer resolution (1 ms). Dummynet is used for emulating mobile scenarios and a special ICMP message is used by handoff emulator to notify the mobile host about handoff completion and the bandwidth of new wireless link. This ICMP message emulates handoff completion detection and bandwidth estimation functions of mobile host.

5.2. Experimental results

In order to evaluate TCP-HO, we first generate several mobile scenarios (handoff sequences) that may occur in different mobile environments of the real world (WLAN and WCDMA). For each mobile scenario, the delay of WAN are set to different values (10, 20, 50, and 100 ms) to create various network scenarios.

For each network scenario, the connection between server and mobile host may use TCP Newreno, Freeze-TCP, and TCP-HO. In order to analyze the effects of different parts of TCP-HO mechanism, two additional TCP variants, TCP-HO-Immediate and TCP-HO-Aggressive, are also investigated. Within TCP-HO-Immediate, the sender immediately begins to transmit after receiving notification. But the bandwidth of new wireless link is ignored. In TCP-HO-Aggressive, the sender sets both *cwnd* and *ssthresh* to $BWD_{new} * SRTT_{old}$. Hence, for each network scenario, five experiments will be carried out.

Within each experiment, the cross traffic connection uses TCP Newreno and runs simultaneously with the connection between server and mobile host. WWW background traffics are also there. In order to get accurate results, each experiment is carried out for ten times. The average and 95% confidence interval will be computed and presented. In the following parts, we will present our experimental results under three emulated mobile scenarios.

5.2.1. WCDMA scenario

In this scenario, we emulate that a user drives a car (speed: 60 km/h) in a rural area covered by WCDMA network (cell coverage: 2 km) for half an hour. The average connection time is 50 s and the disconnection time is 3 s. We assume that ten users share one 2 Mbps WCDMA channel and the bandwidth of each user is about 200 Kbps.

Fig. 7 shows the throughput between server and mobile host when different TCP mechanisms are used. This figure shows that Freeze-TCP, TCP-HO and the variants of TCP-HO do improve TCP performance. The difference among these mechanisms is very small. It is reasonable since L-HH dominates the WCDMA mobile scenario and *silly waiting* is the main problem. These results accord with Freeze-TCP's claim that *immediate transmission* dominates Freeze-TCP's performance enhancement.

Fig. 7 also shows that Freeze-TCP performs better than TCP-HO sometimes. It is reasonable because that after the completion of handoff, the sending rate of Freeze-TCP is higher than that of TCP-HO. As for TCP-HO-Aggressive, its throughput is lower than TCP-HO even though it is more aggressive than Freeze-TCP. It may be too aggressive and harms itself due to the large data burst after handoff completion.

In Fig. 7, we also find that the throughput does not response to the change of WAN delay well. The reason is that RTT is dominated by the slow WCDMA link whose base station has a large buffer (20 packets).

Fig. 8 shows the throughput of cross traffic flow under WCDMA mobile scenario. It indicates that under WCDMA scenario, all TCP mechanisms are friendly to cross traffic flow. Duo to the low bandwidth of WCDMA link, the flow between server and mobile host can not significantly affect cross traffic flow no matter which TCP mechanism is used.

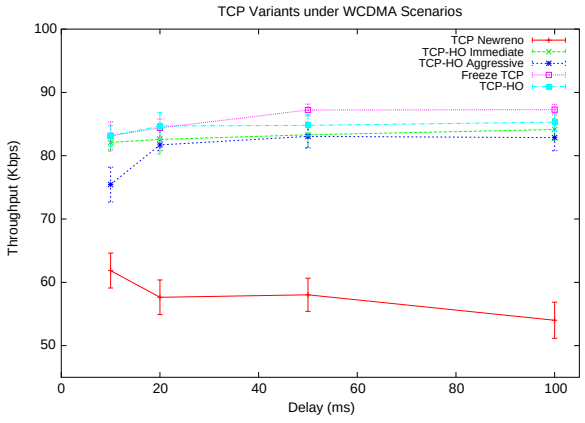


Fig. 7. Average and 95% confidence interval of the throughput received by the flow between server and mobile host under WCDMA mobile scenario.

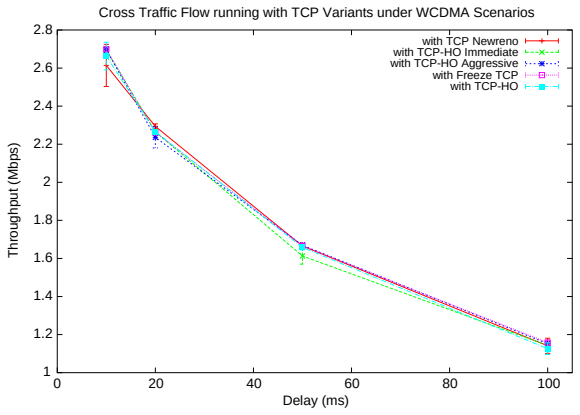


Fig. 8. Average and 95% confidence interval of cross traffic flow's throughput when it runs with different TCP variants under WCDMA mobile scenario.

5.2.2. WLAN scenario

In this scenario, we emulate that a user drives a car (speed: 30 km/h) around a campus with some Wi-Fi hot-spots (cell coverage: 100 m) for 15 min. The average dwelling time is 12 s and the disconnection time is 10 s (bad coverage). Per-cell user number is either large (9–10) or very small (1–2). These users share 7 Mbps channel bandwidth (effective channel bandwidth of IEEE 802.11b [6]). Available bandwidth for each user depends on the number of users per-cell and the value can vary significantly.

Fig. 9 shows the throughput between server and mobile host in WLAN mobile scenario. It shows that TCP-HO achieves the highest throughput in all cases. It is reasonable since TCP-HO has been designed to solve the challenges brought by H-HH, U-VH and D-VH. Fig. 9 also shows that TCP-HO-Immediate achieves trivial improvement and TCP-HO-Aggressive is even worse than Newreno. This is reasonable since H-HH, U-VH and D-VH dominate this mobile scenario. TCP-HO-Immediate does not solve severe "*slow start*" problem of H-HH and U-VH. In case of TCP-HO-Aggressive, during each kind of handoff occurred in this high-speed wireless network, the large data burst generated by itself makes it immediately suffer multiple segment loss, which will trigger timeout and reduce its own throughput.

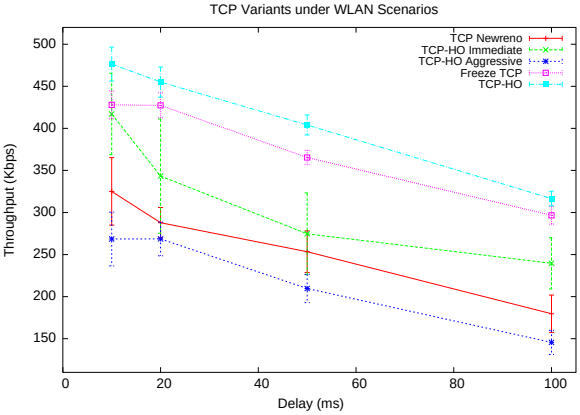


Fig. 9. Average and 95% confidence interval of the throughput received by the flow between server and mobile host under WLAN mobile scenario.

Fig. 9 also shows that Freeze-TCP can acquire quite high throughput too. But Freeze-TCP is too aggressive under WLAN mobile scenario. Fig. 10 shows the throughput of cross traffic flow under WLAN mobile scenario. It indicates that under WLAN scenario, Freeze-TCP obviously harms cross traffic flow in all cases. The reason may be that after handoff completion, Freeze-TCP is too aggressive and the segments of cross traffic flow are dropped due to the large data burst of Freeze-TCP. As for TCP-HO-Aggressive, it may be too aggressive, harm itself too much, and cross traffic flow can acquire more bandwidth when flow with TCP-HO-Aggressive is idle.

5.2.3. WCDMA and WLAN scenario

In this scenario, we emulate a user who drives a car (speed: 45 km/h) around a city with WCDMA coverage and sporadic Wi-Fi hot-spots for 20 min. If Wi-Fi hot-spot exists, the user always switches to Wi-Fi. Otherwise, WCDMA is used. The probability of handoff between two networks and the disconnection time follow the numbers in Table 2. Here, we assume that two Wi-Fi hot-spots exist in one WCDMA cell.

The average dwelling time of WCDMA cell is 20 s and per-user bandwidth is 200 Kbps. The average dwelling time of Wi-Fi is 8 s and per-cell user number follows the same distribution used by WLAN scenario.

Fig. 11 shows the throughput between Server and Mobile Host in WCDMA and WLAN mobile scenario. It indicates that TCP-HO can achieve the highest throughput. Freeze-TCP and TCP-HO-Immediate are also quite good.

Fig. 10. Average and 95% confidence interval of cross traffic flow's throughput when it runs with different TCP variants under WLAN mobile scenario.

Table 2
Handoff probability and disconnection time

Probability/disconnection time	WCDMA (s)	WLAN (s)
WCDMA	0.33/3	0.67/5
WLAN	0.9/5	0.1/10

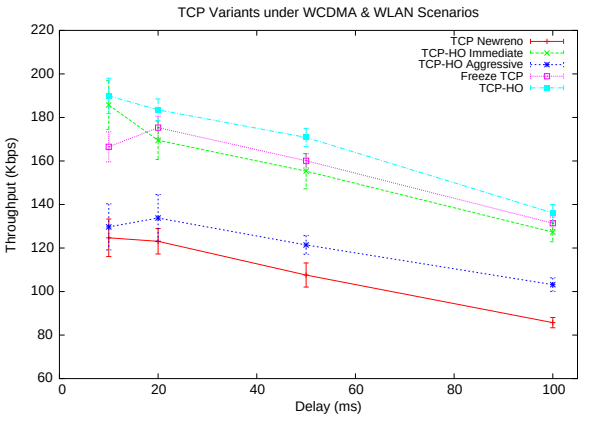


Fig. 11. Average and 95% confidence interval of the throughput received by the flow between server and mobile host under WCDMA and WLAN scenario.

TCP-HO-Aggressive is only a little better than Newreno. Fig. 11 looks like a mixture of the results of WCDMA scenario and WLAN scenario. It is reasonable since L-HH, H-HH, U-VH, and D-VH all occur in this scenario.

Fig. 11 also indicates that TCP-HO does help WCDMA users to utilize Wi-Fi hot-spots. If Newreno is used, a user who uses WLAN and WCDMA interfaces alternately can not acquire much higher throughput than a user who only uses the WCDMA interface.

Fig. 12 shows the throughput of cross traffic flow under WCDMA and WLAN mobile scenario. It indicates that Freeze-TCP is still too aggressive and harms cross traffic flow in some cases.

According to the above results, TCP-HO is the best mechanism for improving mobile host's throughput in most of our experiments without adversely affecting cross traffic flow. Although Freeze-TCP can improve mobile host's throughput quite well, it is too aggressive and even one Freeze-TCP flow can obviously harm cross traffic flow in some cases, especially under WLAN mobile scenario.

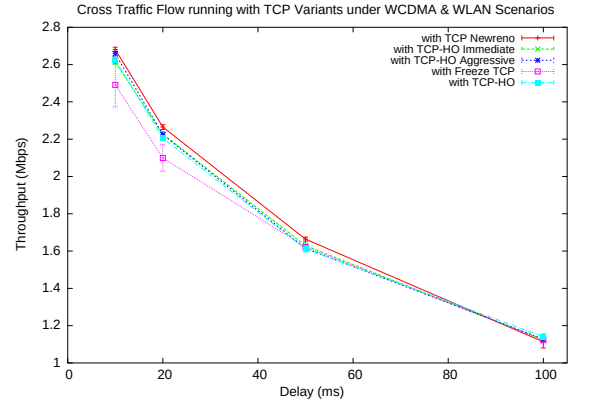


Fig. 12. Average and 95% confidence interval of cross traffic flow's throughput when it runs with different TCP variants under WCDMA and WLAN mobile scenario width, it should be worthwhile to deploy TCP-HO by changing both TCP end-points.

Considering that more and more users will access the Internet through wireless network and the bandwidth of wireless network is increasing quickly [5,7], Freeze-TCP is not a safe solution again. With the considerations that more and more users are accessing the Internet through wireless networks and wireless link is normally the last link with the smallest bandwidth, it should be worthwhile to deploy TCP-HO by changing both TCP end-points.

6. TCP-HO and wireless link bandwidth estimation

Bandwidth estimation error can be regarded as the reason that TCP suffers *slow start* and *over-shooting* during handoff. The server regards the bandwidth probed on old wireless link as the bandwidth of the new wireless link. During handoff, especially during vertical handoff, compared with the server, mobile host normally has better opportunity to estimate the new wireless link's bandwidth timely and accurately. This fact is the main motivation of TCP-HO.

In this section, we discuss wireless bandwidth estimation at mobile host, analyze the effects of its bandwidth estimation error on TCP-HO, and present TCP-HO performance under the achievable bandwidth estimation accuracy.

6.1. Wireless link bandwidth estimation

The estimation of a wireless link's bandwidth had been used by base station selection, routing protocol, and other cross-layer optimizations [44–46], and a lot of mechanisms had also been proposed to estimate the bandwidth of a wireless link [40–47]. These mechanisms estimate the bandwidth based on the model of layer 2 protocol with the parameters of signal strength and contention state that are acquired through passively monitoring the wireless link or actively transmitting several probing frames.

In the real world, we can learn the type of wireless link based on which network interface card (NIC) is used. Hence, layer 2 protocol model can be determined. Signal strength and data rate used for coding have also been provided by tools from NIC manufacturers. If the wireless link is a dedicated channel or time slot, its available bandwidth can be estimated without any further support. If the wireless link is shared, such as IEEE 802.11-based WLAN, contention state could be acquired by modifying the firmware and drivers of NIC, and this method had been verified in [47]. Hence, these bandwidth estimation mechanisms should be implementable in the real systems.

Existing simulation and field trial results show that mobile host with these mechanisms can get an accurate estimation of the bandwidth in a short time. It is reported that, even in the complex 802.11 network, bandwidth estimation relative error can be less than 10% [47].

In the case of TCP-HO, before sending out the first TCP segment through the new wireless link, mobile host need associate to the new wireless network, acquire IP address from it, and complete layer 3 handoff [48]. Quite a few packets will be exchanged in these procedures, and these procedures can take several seconds in some networks

[49]. During this period, mobile host with these mechanisms is able to learn wireless link's quality and its contention state. Hence, a quite accurate bandwidth estimation is feasible for mobile host when TCP Bandwidth Option needs to be sent out.

6.2. Effects of mobile host's bandwidth estimation error on TCP-HO

In this part, we numerically analyze TCP-HO performance when wireless link bandwidth estimated by mobile host is not accurate. In this analysis, $\hat{b}$ represents the real BDP after handoff, $\hat{x}$ represents the product of bandwidth estimated by mobile host and RTT estimated by the server, and $\hat{c}$ is the *ssthresh* maintained by the server before handoff.

When b is small, bandwidth estimation accuracy has a much less significant effect on the performance of TCP-HO. Firstly, when $\hat{x}$ is smaller than the small $\hat{b}$, since TCP-HO still increases *cwnd* after it has been exponentially increased to $\hat{x}$, TCP-HO can fully utilize the new wireless link very soon. Secondly, even when $\hat{x}$ is larger than the small $\hat{b}$, queue at the new base station still can accommodate the extra segments. Hence, in this analysis, we only consider U-VH and H-HH during which $\hat{b}$ is quite large.

For simplicity, we reasonably assume that $\hat{x}$ follows a normal distribution (mean: $\hat{b}$, variance: σ^2) and σ will vary when we analyze TCP-HO sensitivity to bandwidth estimation error. We also constrain $\hat{x}$ to be within $(0, 2\hat{b}]$. This constraint is made so that when $\hat{b} < \hat{x} < 2\hat{b}$, we can assume that queue at the new base station can accommodate these extra segments. Hence, we can avoid to analyze the effects of segment loss. Considering the achievable bandwidth estimation accuracy, this range should be large enough.

With the above assumptions, we first model TCP-HO performance in two cases, $\hat{b} > \hat{c}$ and $\hat{b} < \hat{c}$. We then analyze its sensitivity to bandwidth estimation error by fixing $\hat{b}$ and $\hat{c}$ according to several typical handoff scenarios and varying the value of σ.

We use the difference between N_{tcpho} and N_{tcp} for comparing the performance of TCP-HO and TCP and analyzing TCP-HO sensitivity to bandwidth estimation error. Here, N_{tcpho} and N_{tcp} are the number of segments, that are transmitted by TCP-HO and TCP before the new wireless link can be well utilized by both protocols. Since we are analyzing the effects of bandwidth estimation error, we ignore the effects of TCP-HO's *immediate transmission* and assume that TCP-HO and TCP begin to transmit at the same time. This will result similar performance for TCP-HO and TCP when the difference between $\hat{b}$ and $\hat{c}$ is small. Nevertheless, our analysis will show that TCP-HO performance will not degrade much even when there is substantial bandwidth estimation error.

6.2.1. Model for $\hat{b} > \hat{c}$

According to the approximate *cwnd* vs. time graphes shown in Fig. 13, without considering delayed ACK [16], we can deduce N_{tcp} and N_{tcp} in the following three scenarios.narios.

(1) $\hat{x} < \hat{c} < \hat{b}$: As shown in Fig. 13, in the unit of RTT, $T_1 = \log_2\hat{x}$, $T_2 = \log_2\hat{c} - \log_2\hat{x}$, $T_3 = \hat{b} - \hat{c}$, and $T_4 = (\hat{b} - \hat{x}) - T_2 - T_3 = \hat{c} - \hat{x} + \log_2\hat{x} - \log_2\hat{c}$. Hence,

$$N_{\text{tcp}} = A_{T_1+T_2} + A_{T_3} + A_{T_4} = \hat{c} + \frac{(\hat{b}^2 - \hat{c}^2)}{2} + \hat{b}(\hat{c} - \hat{x} + \log_2\hat{x} - \log_2\hat{c}),$$

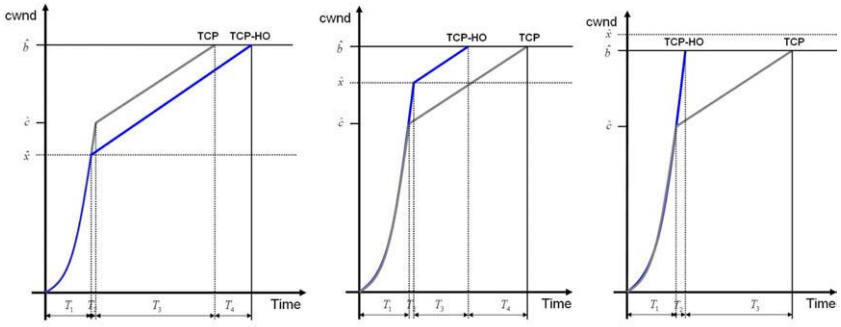
$$N_{\text{tcpho}} = A_{T_1} + A_{T_2+T_3+T_4} = \hat{x} + \frac{(\hat{b}^2 - \hat{x}^2)}{2},$$

$$\triangle_1(\hat{x}) = N_{\text{tcpho}} - N_{\text{tcp}} = -\frac{\hat{x}^2}{2} + (\hat{b}+1)\hat{x} - \hat{b}\log_2\hat{x} + \frac{\hat{c}^2}{2} - (\hat{b}+1)\hat{c} + \hat{b}\log_2\hat{c}.$$

(2) $\hat{c} < \hat{x} < \hat{b}$: As shown in Fig. 13, $T_1 = \log_2\hat{c}$, $T_2 = \log_2\hat{x} - \log_2\hat{c}$, $T_3 = \hat{b} - \hat{x}$, and $T_4 = (\hat{b} - \hat{c}) - T_2 - T_3 = \hat{x} - \hat{c} + \log_2\hat{c} - \log_2\hat{x}$. Hence,

$$N_{\text{tcp}} = A_{T_1} + A_{T_2+T_3+T_4} = \hat{c} + \frac{(\hat{b}^2 - \hat{c}^2)}{2},$$

$$N_{\text{tcpho}} = A_{T_1+T_2} + A_{T_3} + A_{T_4} = \hat{x} + \frac{(\hat{b}^2 - \hat{x}^2)}{2} + \hat{b}(\hat{x} - \hat{c} + \log_2\hat{c} - \log_2\hat{x}),\ \triangle_2(\hat{x}) = N_{\text{tcpho}} - N_{\text{tcp}} = -\frac{\hat{x}^2}{2} + (\hat{b}+1)\hat{x} - \hat{b}\log_2\hat{x} + \frac{\hat{c}^2}{2} - (\hat{b}+1)\hat{c} + \hat{b}\log_2\hat{c}.$$



(a) $\hat{x} < \hat{c} < \hat{b}$ (b) $\hat{c} < \hat{x} < \hat{b}$ (c) $\hat{c} < \hat{b} < \hat{x}$

Fig. 13. *cwnd* vs. time graphs of TCP and TCP-HO when $\hat{b} > \hat{c}$.

(3) $\hat{c} < \hat{b} < \hat{x}$: As shown in Fig. 13, $T_1 = \log_2\hat{c}$, $T_2 = \log_2\hat{b} - \log_2\hat{c}$, and $T_3 = (\hat{b} - \hat{c}) - T_2 = \hat{b} - \hat{c} + \log_2\hat{c} - \log_2\hat{b}$. Hence,

$$N_{\text{tcp}} = A_{T_1} + A_{T_2+T_3} = \hat{c} + \frac{(\hat{b}^2 - \hat{c}^2)}{2},$$

$$N_{\text{tcpho}} = A_{T_1+T_2} + A_{T_3} = \hat{b} + \hat{b}(\hat{b} - \hat{c} + \log_2\hat{c} - \log_2\hat{b}),$$

$$\triangle_3(\hat{x}) = N_{\text{tcpho}} - N_{\text{tcp}} = \frac{\hat{b}^2 + \hat{c}^2}{2} + \hat{b} - \hat{c} - \hat{b}\hat{c} + \hat{b}\log_2\hat{c} - \hat{b}\log_2\hat{b}.$$

Finally, we can calculate the expected performance improvement of TCP-HO by the following equation. Here, $p(\hat{x}) = \frac{1}{\sigma\sqrt{2\pi}}\exp(-\frac{(\hat{x}-\hat{b})^2}{2\sigma^2})$

$$\triangle(\hat{b},\hat{c},\sigma) = \frac{\int_0^{\hat{c}} \triangle_1(\hat{x})p(\hat{x}) + \int_{\hat{c}}^{\hat{b}} \triangle_2(\hat{x})p(\hat{x}) + \int_{\hat{b}}^{2\hat{b}} \triangle_3(\hat{x})p(\hat{x})}{\int_0^{2\hat{b}} p(\hat{x})}. \quad (1)$$

6.2.2. Model for $\hat{b} < \hat{c}$

Since only U-VH and H-HH are considered, for simplicity, we assume that $\hat{c} < 2\hat{b}$. When $\hat{c} < 2\hat{b}$, we can assume that TCP will not cause segment loss. Hence, the analysis can be simplified and the analyzed TCP-HO performance improvement is worse than its improvement in the real world. Following the methods used in the above case, we get the following results.

(1) $\hat{x} < \hat{b} < \hat{c}$: As shown in Fig. 14, in the unit of RTT, $T_1 = \log_2\hat{x}$, $T_2 = \log_2\hat{b} - \log_2\hat{x}$, and $T_3 = (\hat{b} - \hat{x}) - T_2 = \hat{b} - \hat{x} + \log_2\hat{x} - \log_2\hat{b}$. Hence,

$$N_{\text{tcp}} = A_{T_1+T_2} + A_{T_3} = \hat{b} + \hat{b}(\hat{b} - \hat{x} + \log_2\hat{x} - \log_2\hat{b}),$$

$$N_{\text{tcpho}} = A_{T_1} + A_{T_2+T_3} = \hat{x} + \frac{(\hat{b}^2 - \hat{x}^2)}{2},$$

$$\nabla_1(\hat{x}) = N_{\text{tcpho}} - N_{\text{tcp}} = -\frac{\hat{x}^2}{2} + (\hat{b}+1)\hat{x} - \hat{b}\log_2\hat{x} - \frac{\hat{b}^2}{2} - \hat{b} + \hat{b}\log_2\hat{b}.$$

(2) $\hat{b} < \hat{x} < \hat{c}$: As shown in Fig. 14, the curves of TCP and TCP-HO are totally overlapped. Hence, $\nabla_2(\hat{x}) = 0$.

(3) $\hat{b} < \hat{c} < \hat{x}$: As shown in Fig. 14, $\nabla_3(\hat{x})$ also equals to 0.

Finally, we can calculate the expected performance improvement of TCP-HO by the following equation. Here, $p(\hat{x}) = \frac{1}{\sigma\sqrt{2\pi}}exp(-\frac{(\hat{x}-\hat{b})^2}{2\sigma^2})$

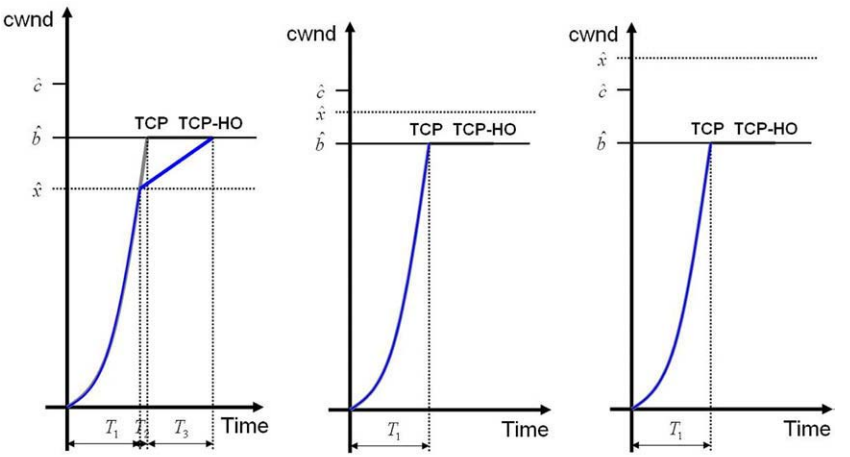
$$\nabla(\hat{b},\hat{c},\sigma) = \frac{\int_0^{\hat{b}} \nabla_1(\hat{x})p(\hat{x})}{\int_0^{2\hat{b}} p(\hat{x})}. \quad (2)$$

6.2.3. Bandwidth estimation error sensitivity analysis

In this part, we carry out bandwidth estimation error sensitivity analysis by fixing b and c according to several typical handoff scenarios and varying the value of σ from 5% to 50% of b. The following four handoff scenarios are used.

(1) U-VH Scenario ($\hat{b} = 117, \hat{c} = 8$): In this scenario, we investigate the effects of bandwidth estimation error when mobile host switches from a WCDMA link (bandwidth = 200 Kbps, RTT = 500 ms, and $\hat{c} \approx 8$ segments) to a WLAN channel (bandwidth = 7 Mbps, RTT = 200 ms, and $\hat{b} \approx 117$ segments). We substitute $\hat{b}$ and $\hat{c}$ in Eq. 1 with these values and calculate $\triangle$ with different σ through MATLAB.

(2) U-VH Scenario ($\hat{b} = 58, \hat{c} = 8$): This is another U-VH Scenario whose BDP difference is smaller than the above one.



(a) $\hat{x} < \hat{b} < \hat{c}$ (b) $\hat{b} < \hat{x} < \hat{c}$ (c) $\hat{b} < \hat{c} < \hat{x}$

Fig. 14. *cwnd* vs time graphs of TCP and TCP-HO when $\hat{b} \leqslant \hat{c}$.

(3) H-HH Scenario ($\hat{b} = 58, \hat{c} = 50$): In this scenario, we investigate the effects of bandwidth estimation error when mobile host switches between two different WLAN channels and the new channel has a little more available bandwidth. And the same method used by U-VH scenarios is adopted.
(4) H-HH Scenario ($\hat{b} = 58, \hat{c} = 66$): In this scenario, we investigate the effects of bandwidth estimation error when mobile host switches between two different WLAN channels and the new channel has a little less available bandwidth. We substitute $\hat{b}$ and $\hat{c}$ in Eq. 2 with these values and calculate ∇ with different values of σ.

Fig. 15 shows these results of sensitivity analysis. It indicates that TCP-HO performance during U-VH and H-HH is not sensitive to the accuracy of bandwidth estimation. During U-VH, due to the large difference in BDP, TCP-HO still can improve TCP performance a lot even when mobile host only has a coarse bandwidth estimation. During H-HH, the performance of TCP-HO is just a little worse than TCP when mobile host has a significant bandwidth estimation error.

6.3. TCP-HO under achievable bandwidth estimation accuracy

In the experiments of Section 5, we assume that mobile host can estimate the bandwidth of new wireless link very accurately. In this subsection, we evaluate TCP-HO when mobile host only has an achievable accurate estimation of wireless link's bandwidth with the aim of evaluating the effects of bandwidth estimation error.

We re-generate WCDMA, WLAN, and WCDMA and WLAN mobile scenarios, and the seeds of random number generator are different with Section 5.2. The bandwidth sent to mobile host is different with the bandwidth emulated by handoff emulator, and the simulated bandwidth estimation error is within 15% of the new wireless link's emulated bandwidth. We re-run the experiments of Section 5.2 with these new mobile scenarios, and Figs. 16 and 17 show the new experiment results. These results are similar to the results in Section 5.2. They indicate that TCP-HO still can improve TCP performance a lot without adversely affecting cross traffic when mobile host only have a coarse estimation of wireless link's bandwidth.

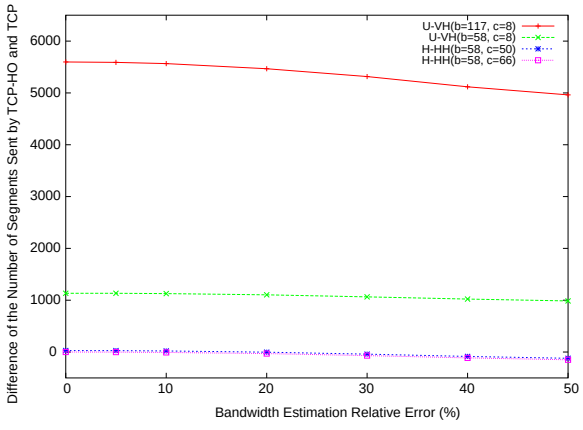
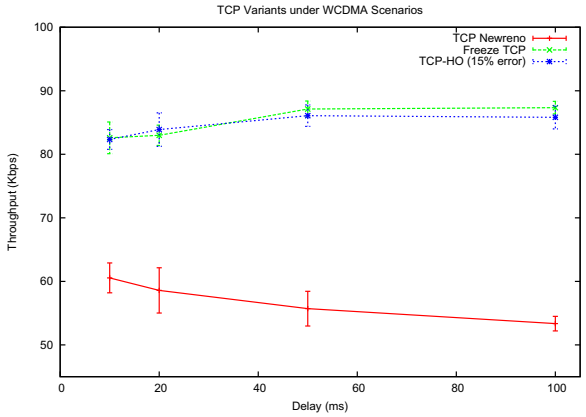
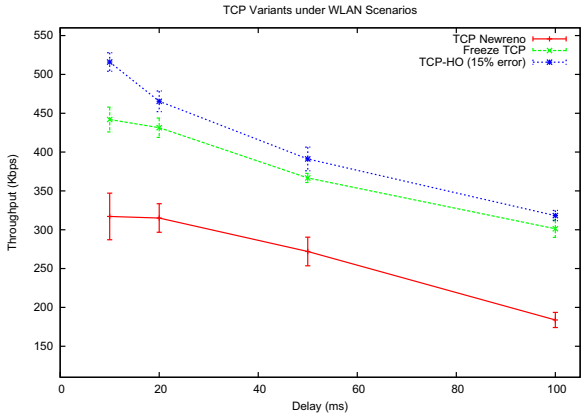


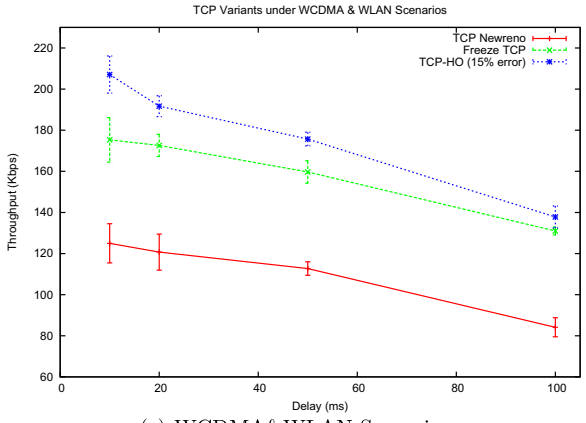
Fig. 15. Bandwidth estimation error sensitivity analysis.



(a) WCDMA Scenario

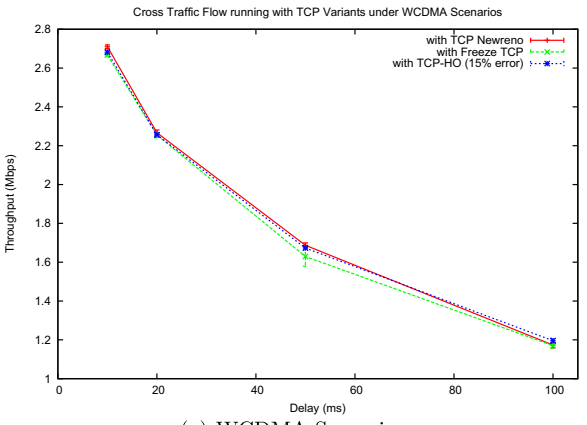


(b) WLAN Scenario

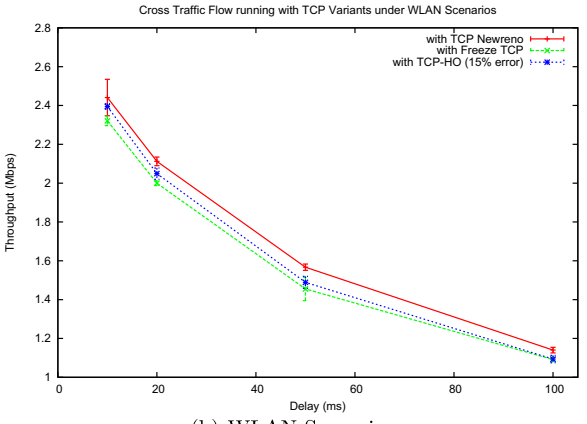


(c) WCDMA&WLAN Scenario

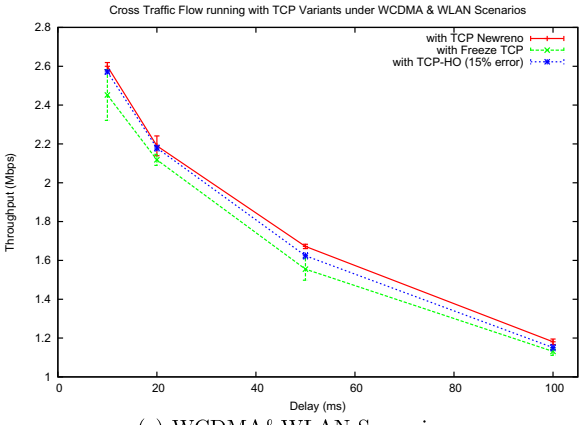
Fig. 16. Average throughput received by the flow between server and mobile host with 15% bandwidth estimation error.



(a) WCDMA Scenario



(b) WLAN Scenario



(c) WCDMA&WLAN Scenario

Fig. 17. Average Throughput of Cross Traffic Flow with 15% Bandwidth Estimation Error.

7. Conclusion

In this paper, TCP-HO, the first practical end-to-end TCP enhancement based on explicit cooperation, is proposed to solve the challenges brought by all kinds of handoff that may occur in heterogeneous mobile environments. Experimental results indicate that in heterogeneous mobile environments, TCP-HO improves TCP performance a lot without adversely affecting cross traffic. Considering that more and more users are accessing the Internet through heterogeneous wireless networks and mobile host could have a coarse estimation of wireless link's bandwidth, it should be worthwhile to change both server and mobile host for improving TCP performance.

Acknowledgements

This work is supported in part by University Research Committee, National University of Singapore under Grants R-252-000-203-112 and R-252-000-313-112. The authors would like to thank the anonymous reviewers for their insightful comments.

References

[1] 3gpp, Available from: <http://www.3gpp.org>.
[2] Rlc Protocol Specification, 3G TS 25.322, V3.2.0, March 2000.
[3] General Packet Radio Service (GPRS) Service Description, version 7.1.0, European Standard 301 344, GSM 03.60, ETSI Std., August 1999.
[4] Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE P802.11, IEEE Std., 1999.
[5] Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications—High-Speed Physical Layer in the 5 GHz Band, IEEE P802.11, IEEE Std., 1999.
[6] Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications—Higher-Speed Physical Layer Extension in the 2.4 GHz Band, IEEE P802.11, IEEE Std., 1999.
[7] Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications—Amendment 4: Further Higher Data Rate Extension in the 2.4 GHz Band, IEEE P802.11, IEEE Std., 1999.
[8] High Performance Radio Local Area Network, Type 2, Requirements and Architectures for Wireless Broadband Access, TR 101 031 V2.2.1, ETSI Std., January 1999.
[9] Wimax, Available from: <http://www.wimaxforum.org>.
[10] Ns2 Network Simulator, <http://www.isi.edu/nsnam/ns/>.
[11] Iperf, Available from: <http://dast.nlanr.net/Projects/Iperf/>.
[12] P. Barford, M. Crovella, Generating representative web workloads for network and server performance evaluation, in: ACM SIGCOMM, 1998.
[13] L. Rizzo, Dummynet: a simple approach to the evaluation of network protocols, ACM CCR 27 (1) (1997) 31–41.
[14] R. Caceres, L. Iftode, Improving the performance of reliable transport protocols in mobile computing environments, IEEE JSAC Special Issue on Mobile Computing Network (1994).
[15] J. Postel, Transmission Control Protocol – Darpa Internet Program Protocol Specification, RFC 793, DARPA, September 1981.
[16] D.D. Clark, Window and Acknowledgement Stratedgy in TCP, RFC 813, July 1982.
[17] H. Balakrishnan, S. Seshan, E. Amir, R.H. Katz, Improving TCP/IP performance over wireless networks, in: ACM MobiCOM, 1995.
[18] Cheng Peng Fu, Soung C. Liew, TCP veno: TCP enhancement for transmission over wireless access networks, IEEE JSAC 21 (2003).
[19] R.K. Balan, B.P. Lee, K.R.R. Kumar, L. Jacob, W.K.G. Seah, A.L. Ananda, TCP hack: a mechanism to improve performance over lossy links, Computer Networks 39 (2002) 347–361.
[20] Hari Balakrishnan, Randy H. Katz, Explicit loss notification and wireless web performance, in: IEEE Globecom, 1998.
[21] Rajesh Krishnan, J.P.G. Sterbenz, W.M. Eddy, C. Partridge, M. Allman, Explicit transport error notification (ETEN) for error-prone wireless and satellite networks, Computer Networks 46 (2004).
[22] Xiuchao Wu, Indradeep Biswas, MunChoon Chan, A.L. Ananda, Utilizing characteristics of last link to improve TCP performance, IEEE IPCCC (2005).
[23] MunChoon Chan, Ram Ramjee, Tcp/ip performance over 3 g wireless links with rate and delay variation, in: ACM MobiCOM, 2002.
[24] MunChoon Chan, Ram Ramjee, Improving TCP/IP performance over third generation wireless networks, in: IEEE INFOCOM, 2004.
[25] S. Mark, Vertical handoffs in wireless overlay networks, Technical Report CSD-96-903, UCB, Technical Report, 1996.

[26] Y. Matsushita, T. Matsuda, M. Yamamoto, Tcp congestion control with ACK-pacing for vertical handoff, in: IEEE WCNC, 2005.
[27] V. Jacobson, Congestion avoidance and control, in: ACM SIGCOMM, 1988.
[28] A. Bakre, I-TCP: indirect TCP for mobile hosts, in: ICDCS, 1995.
[29] R. Yavatkar, N. Bhagawat, Improving end-to-end performance of TCP over mobile internetworks, in: IEEE Workshop on Mobile Computing Systems and Applications, 1994.
[30] K. Brown, S. Singh, M-TCP: TCP for mobile cellular networks, in: ACM CCR, 1997.
[31] S. Kent, R. Atkinson, Security Architecture for the Internet Protocol, RFC2401, November 1998.
[32] H. Hsieh, K. Kim, Y. Zhu, R. Sivakumar, A receiver-centric transport protocol for mobile hosts with heterogeneous wireless interfaces, in: ACM MOBICOM, September 2003.
[33] T. Goff, J. Moronski, D.S. Phatak, V. Gupta, Freeze-TCP: a true end-to-end TCP enhancement mechanism for mobile environments, in: IEEE INFOCOM, 2000.
[34] M. Allman, TCP Congestion Control with Appropriate Byte Counting (ABC), RFC 3465, February 2003.
[35] S. Floyd, T. Henderson, A. Gurtov, The Newreno Modification to TCP's Fast Recovery Algorithm, RFC 3782, April 2004.
[36] M. Handley, J. Padhye, S. Floyd, TCP Congestion Window Validation, RFC 2861, June 2000.
[37] K. Tsukamoto, Y. Fukuda, Y. Hori, Y. Oie, New flow control schemes of TCP for multimodal mobile hosts, in: IEEE VTC-Spring, 2003.
[38] L. Kalampoukas, A. Varma, K.K. Ramakrishnan, Explicit window adaptation: a method to enhance TCP performance, IEEE/ACM TON 10 (2002) 338–350.
[39] A. Karnik, A. Kumar, Performance of TCP congestion control with explicit rate feedback, IEEE/ACM TON 13 (2005) 108–112.
[40] X. Wu, A.L. Ananda, Link characteristics estimation for IEEE 802.11 DCF based WLAN, in: IEEE LCN, 2004.
[41] Jian Zhang, Liang Cheng, Ivan Marsic, Models for non-intrusive estimation of wireless link bandwidth, in: Personal Wireless Communication Conference, 2003.
[42] M. Deziel, L. Lamont, Implementation of an IEEE 802.11 link available bandwidth algorithm to allow cross-layering, in: IEEE WiMobapos, 2005.
[43] K. Lakshminarayanan, V.N. Padmanabhan, J. Padhye, bandwidth estimation in broadband access networks, in: IMC, 2004.
[44] S. Vasudevany, K. Papagiannakiz, C. Diotz, J. Kurosey, D. Towsley, Facilitating access point selection in IEEE 802.11 wireless networks, in: IMC, 2005.
[45] S. Lee, S. Banerjee, B. Bhattacharjee, The case for a multi-hop wireless local area network, in: IEEE Infocom, 2004.
[46] H.K. Lee, V. Hall, K.H. Yum, K.I. Kim, E.J. Kim, Bandwidth estimation in wireless LANs for multimedia streaming, in: IEEE ICME, 2006.
[47] G. Wu, T. Chiueh, Passive and accurate traffic load estimation for infrastructure-mode wireless LAN, in: ACM MSWIM, 2007.
[48] C. Perkins, IP Mobility Support for IPv4, RFC 3344, August 2002.
[49] V. Bychkovsky, B. Hull, A. Miu, H. Balakrishnan, S. Madden, A measurement study of vehicular internet access using in situ WiFi networks, in: ACM MobiCOM, 2006.

Xiuchao Wu is currently a Ph.D. candidate in computer science at National University of Singapore (NUS). He received his B.E. in computer science at University of Science and Technology of China in 1999. In 2004, he received his M.Sc. in computer science at NUS. Before he enrolled in NUS, he had worked as R&D Engineer on wireless communication and security for near four years. His research interests are transport protocols, wireless network, sensor network, and security, etc.

Mun Choon Chan received the B.S. degree from Purdue University, West Lafayette, IN, in 1990 and the M.S. and Ph.D. degrees from Columbia University, NY, in 1993 and 1997, respectively, all in Electrical Engineering. From 1991 to 1997, he was a member of the COMET Research Group, working on ATM control and management. From 1997 to 2003, he was a Member of the Technical Staff at the Networking Research Lab, Bell Laboratories, Lucent Technologies, Holmdel, NJ. Currently, he is an Assistant Professor in the Department of Computer Science, National University of Singapore. Dr. Chan has published more than 40 technical papers and holds five patents. His current research interest includes heterogeneous wireless network and sensor networking. He is a member of ACM and IEEE.

Akkihebbal L. Ananda is an Associate Professor in the Computer Science Department of the School of Computing at the National University of Singapore. His research areas of interest include high-speed computer networks, sensor networks, transport protocols, and distributed systems. He is a member of the IEEE Computer and Communications Societies. Ananda obtained his M.Tech degree in Electrical Engineering from the Indian Institute of Technology, Kanpur in 1973, and Ph.D. degree in computer science from the University of Manchester, UK, in 1983.