2003 Workshop on High Performance Switching and Routing, Torino, Italy - June 24-28, 2003, pp. 171-176

Improving End-to-End Throughput of Mobile IP using SCTP

Shaojian Fu and Mohammed Atiquzzaman
Telecommunications and Networks Research Lab
School of Computer Science
University of Oklahoma,
Norman, OK 73019-6151, USA.
Email addresses: {sfu,atiq}@ou.edu

Abstract— **Mobile IP is the IETF-proposed standard to offer seamless mobile computing. A new transport layer protocol, called Stream Control Transmission Protocol (SCTP), has recently been accepted by IETF as a proposed standard to address a number of TCP limitations. Most of the previous research on end-to-end throughput over Mobile IP has been carried out on TCP-Reno. Both TCP and SCTP can use Selective Acknowledgment (SACK) for error recovery. In this paper, the effect of SCTP on the improvement of throughput during handovers in Mobile IP is analyzed. We show that SCTP has a better performance than TCP-Reno and TCP-SACK due to its support for unlimited number of SACK blocks. We conclude that SCTP can be used to improve the end-to-end throughput when the bottleneck link bandwidth is low.**

Keywords: Mobile IP, Routing during handovers, SCTP, TCP, SACK.

I. Introduction

Mobile IP [1] is the standard proposed by IETF to offer seamless mobile computing. For example, it enables a TCP connection to keep alive and re-route packets when the mobile host moves from one point of attachment to another. During the handover from the home agent (HA) to foreign agent, a mobile host will need to perform registration with the foreign agent, wait for the allocation of channels, and update its location in the HA database. Meanwhile, the signal power may fall far below the successful receiving power threshold [2]. Therefore, within this period, there is a very high probability of packet losses in wireless channels, which may eventually result in the transport protocol backing off in terms of data transmission rate.

TCP is the dominant transport layer protocol in the IP protocol suite which was not initially designed for wireless networks. A considerable amount of research has been carried out in the area of TCP throughput over Mobile IP environments [3] [4] [5] [6]. The basic approach in [3] to eliminate data loss is to multicast data from the HA to all the nearby FAs on anticipation of handover. This method can effectively reduce packet losses and handover delay, but incurs some traffic overhead. In order to improve TCP throughput, the authors in [4] proposed to recover packets dropped during the handover by buffering at the base stations. They have shown that even with packet buffering, the packet loss can't be eliminated entirely. In [5], the performance of TCP under micro-mobility scenarios are discussed, and a protocol called Multicast for Mobility Protocol (MMP) is used to improve TCP's throughput when multiple handovers occur in a relatively short time interval. In [6], the authors modeled the TCP throughput during the handover analytically by using a Finite-State Markov Channel (FSMC) model, and showed quantitatively the impairment of Mobile IP handover on TCP throughput. Most of the previous research on end-to-end throughput over Mobile IP have been carried out on TCP-Reno. The TCP Selective Acknowledgment option (SACK) [7] can be used to recover packet losses efficiently, especially when there are multiple packet losses in a single window of data, which is a typical scenario during Mobile IP handovers. In this paper, we take into account the impact of SACK on transport layer throughput during Mobile IP handovers.

A number of limitations of TCP resulted in the IETF's effort on standardizing a new transport layer protocol, called Stream Control Transmission Protocol (SCTP) [8]. It is a reliable network-friendly protocol which could co-exist with TCP in a shared network. In addition to the standard TCP features, it incorporates a number of enhanced features which could make it suitable for data transfer over mobile networks. Previous research on SCTP mainly focused on the co-existence of SCTP and TCP in the Internet [9] [10], or some SCTP features such as multi-streaming and multi-homing to reduce latency and fault tolerance of data transmission in high loss environments or over satellite links [10] [11] [12].

SACK can recover from multiple packet losses in a single window of data. We have shown (see Sec. II-B) that *non-consecutive multiple packet losses in a single window* of data is common during Mobile IP handovers. Compared to TCP-SACK, SCTP supports unlimited number of SACK blocks in one SACK chunk (see Sec. III-B). The *objective* of this paper is to investigate the effectiveness of unlimited number of SCTP SACK blocks on the throughput of transport protocols during mobile IP handovers. The *contributions* of the paper are as follows:

- We compared the performance and congestion control mechanisms of TCP Reno, TCP-SACK, and SCTP during handovers under different network configurations.
- We showed that the unlimited number of SACK blocks permitted by SCTP results in a higher throughput than that of TCP during handovers.

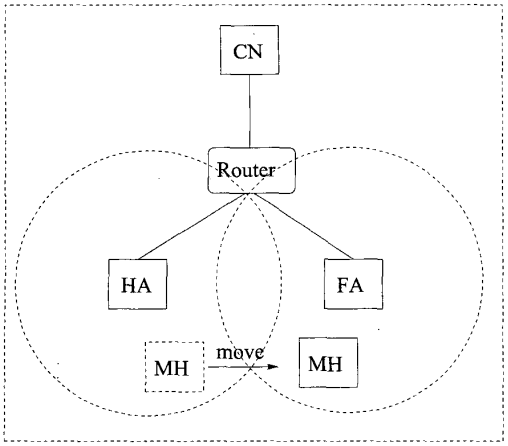

Fig. 1. Mobile IP handover caused by MH movement.

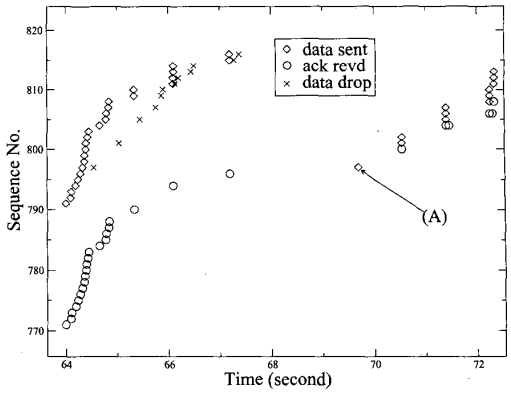

Fig. 2. Packet Losses during Mobile IP handover (with timeout).

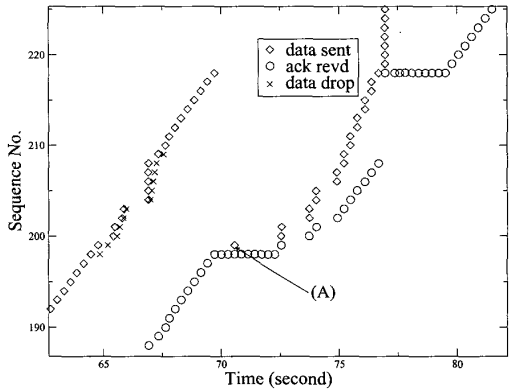

Fig. 3. Packet Losses during Mobile IP handover (no timeout).

- We demonstrated that the effect of SACK is prominent in improving the throughput when the bottleneck link bandwidth (wireline or wireless) is low (under 200Kbps).

II. Packet Losses during Handovers in Mobile IP

We assume the reader is familiar with Mobile IP terminologies and registration operations [1] [13]. A typical scenario of Mobile IP is shown in Fig. 1. In this figure, CN means correspondent node, HA is the home agent, FA is the foreign agent, and MH is the mobile host. TCP sender and receiver are located at MH and CN respectively. This represents a *interdomain* handover scenario where HA and FA belong to different subnets.

A. Instances of packet loss and recovery methods

There will be packet losses during the handover process due to signaling delay and signal fading as shown in Fig. 2 which shows the TCP sequence number versus time of segment sent from the sender (MH). It also shows the dropped data segments in the wireless link and ACKs received from the receiver (CN). Because the handover delay was larger than the sender's RTO estimation, there was a timeout near time 69.5 second at the sender which retransmitted lost segment No. 797 (point Fig. 2-A). The slow start algorithm was then used to retransmit the remaining lost segments.

The timeout, however, doesn't always happen during the handover. If the bottleneck link bandwidth (wireless or wireline) is small enough, the sender will have a large RTO estimation (due to congestion and/or low link speed) and prevent it from timeout. The segment plot shown in Fig. 3 is an example of this kind of scenario. The sender will perform a fast retransmission at point Fig. 3-A due to the receipt of the duplicate ACKs.

In the first case (Fig. 2), the sender will perform a slow start after the handover to recover the lost packets. Even if the TCP is not equipped with SACK, it can cover these packets relatively fast. While in the latter case (Fig. 3) SACK can offer significant help during the fast recovery process, as will be shown in Section VI.

B. Frequency of packet losses

Mobile IP handovers result in frequent multiple non-consecutive packet losses. By "non-consecutive", we mean that packet losses (or burst of losses) are separated from each other. We observed the loss pattern during a 200 second TCP session with Mobile IP for the simulation scenario given in Fig. 1. The MH moves back and forth between the HA and the FA. We correlated the error rate with the receiving power at the MH, and generated errors randomly with a Pareto distribution. We show the losses as a function of time in Fig. 4, where the crosses ($\times$) represent packet losses and the boxes represent time corresponding to one window. The handover duration and the time during which the MH is connected to the HA and FA are also marked in the figure. From the figure, we can observe that during the normal operation (MH with either HA or FA), even though there are packet losses due to wireless link errors, they are generally one loss within a window of data. During the handover, however, we can see multiple non-consecutive losses in a single window.

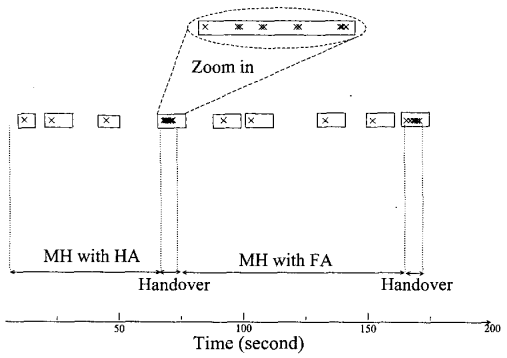

Fig. 4. Packet Loss pattern during a TCP session with Mobile IP

TABLE I
TCP SACK OPTION FORMAT.

0	15	23	31
	Kind=5	Chunk length	
Left Edge of 1^{st} Block			
Right Edge of 1^{st} Block			
.....			
Left Edge of n^{th} Block			
Right Edge of n^{th} Block			

III. Difference between congestion control of TCP and SCTP

The congestion control principles of TCP such as slow start, congestion avoidance, fast retransmit and fast recovery are described in [14]. These algorithms are designed to prevent congestion collapse that is possible in a shared network. SCTP has very similar slow start and congestion avoidance mechanism as those of TCP. Due to the mandatory use of Selective Acknowledgment in SCTP, there are two major differences between TCP and SCTP in terms of Fast Retransmit, Fast Recovery and Selective Acknowledgment as summarized below:

A. Fast Retransmit and Fast Recovery

SCTP incorporates a fast retransmit algorithm based on SACK gap reports similar to that described in [15]. This mechanism speeds up the loss detection, therefore increase the bandwidth utilization. One of the major differences between SCTP and TCP is that SCTP doesn't have a explicit fast-recovery phase, but achieves this automatically with the use of SACK [8].

B. Selective Acknowledgment

The TCP SACK option is defined in [7]. The format of the TCP SACK option is shown in Table I, and the format of SCTP SACK chunk [8] is shown in Table II. The use of SACK is mandatory in SCTP, which allows more robust reaction in the case of multiple losses from a single window of data. This avoids a time-consuming slow start stage after multiple segment losses, thus saving bandwidth and increasing throughput.

TABLE II
SCTP SACK CHUNK FORMAT.

0	7	15	31
Type=3	Chunk Flags	Chunk length	
Cumulative TSN Ack			
Advertised Receiver Window Credit			
Number of GapAck Block		Number of Dup TSN	
Gap Ack Block #1 Start		Gap Ack Block #1 End	
.....			
Gap Ack Block #N Start		Gap Ack Block #N End	
Duplicate TSN 1			
.....			
Duplicate TSN X			

For TCP, the length of the *Options* field is limited to 40 bytes, while a SACK option specifying n blocks will have a length of $8 \times n + 2$ bytes. Therefore, the maximum number of SACK blocks that the TCP SACK option can have is limited to four. If the SACK is used together with time-stamp option (requiring 12 bytes), the maximum SACK blocks allowed would be three.

In contrast to TCP, SCTP allows a large number of blocks in its SACK chunk. The total available chunk space is determined by the *"Chunk Length"* field which is 2^{16} bytes. Subtracting the first 16 bytes required for description of a SACK chunk (see the first four rows in Table II), the maximum length of space for gap blocks is $2^{16} - 16$. Every block needs 4 bytes; therefore the total number of blocks allowed is 16380, we can nearly regard it as an unlimited one. *When there are multiple non-consecutive segment losses in a single window, the number of available SACK blocks in TCP may not be sufficient for reporting all the segment losses.* The large number of SACK blocks makes SCTP more robust in case of multiple losses.

IV. Impact of SACK on throughput during handover

In this section, we describes the behavior of TCP-Reno, TCP-SACK, and SCTP during the Mobile IP handover based on *ns-2* [16] simulation results using the simulation topology shown in Fig. 5. A a router connects the CN to a HA and FA, and a MH moves back and forth between the HA and FA. The coverage of the HA and FA are shown by the dotted circles, where we assume overlapping coverage between the HA and FA. Other parameters are listed in Tables III and IV. Without the loss of generality, the bandwidth between CN-Router and MH-HA/FA are set to 28.8Kb and 384Kb respectively.

A. Behavior of TCP Reno

The congestion control algorithms of TCP Reno is explained in [14]. TCP Reno can not handle multiple segment losses that occur in the same window efficiently [15]. The behavior of TCP Reno during the handover is shown in Fig. 6. In this figure, the segment sequence number is the modulo of actual sequence number by 100 to make it clearer. During the handover, six segments was lost in a window of data (segment No. 7, 11, 13, 14, 15, 16 as shown by $\times$ in the figure). The sender received multiple DupAcks between time 172-176 seconds. After receiving the first 3 DupAcks, the sender re-transmitted segment No. 7

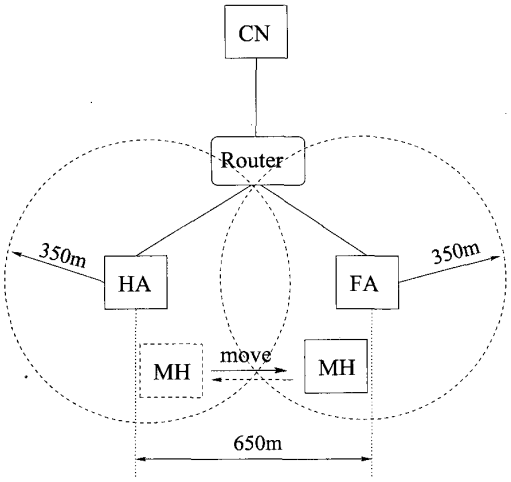

Fig. 5. Simulation topology with Mobile IP handover.

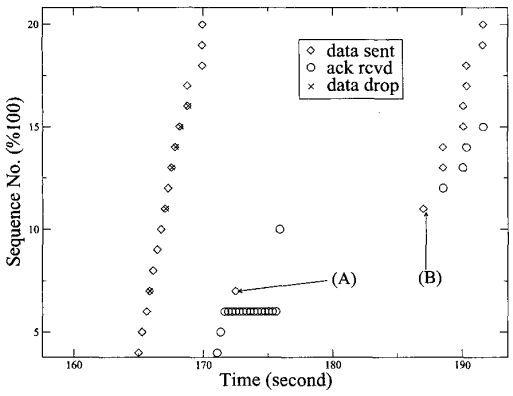

Fig. 6. Segment plot of TCP Reno without SACK during handover

using Fast Retransmit (point Fig. 6-A), but due to the restriction of available receiver window, it must wait until another timeout to retransmit the remaining lost segments (point Fig. 6-B).

B. Behavior of TCP-SACK

The congestion control algorithm for TCP-SACK is given in [15]. In this paper, we assume that the TCP header can only provide three SACK blocks, as discussed in Sec. III-B. The behavior of TCP-SACK during the handover is shown in Fig. 7. In this figure, there are also six segments lost during the handover(segment No. 1, 3, 7, 9, 12, 13 as shown by $\times$ in the figure). With SACK, the TCP sender needn't to wait for a timeout during the handover and can retransmit the lost packets (segment No. 3, 7, 9) earlier than in TCP Reno. However, due to the limitation of three SACK blocks, the sender can't reduce the flight size any more after it retransmit segment No. 9 (point Fig. 7-A) and have to wait for new acknowledgment coming (point Fig. 7-B) before it can retransmit segment No. 12 and 13.

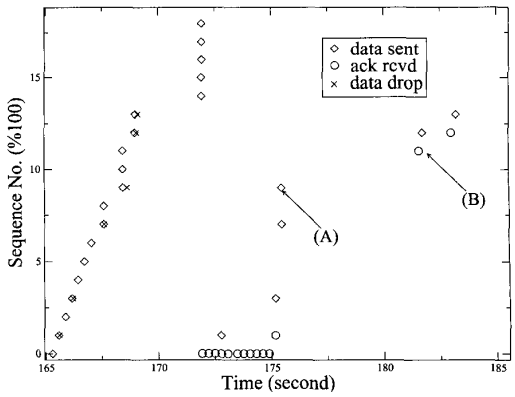

Fig. 7. Segment plot of TCP-SACK during the handover

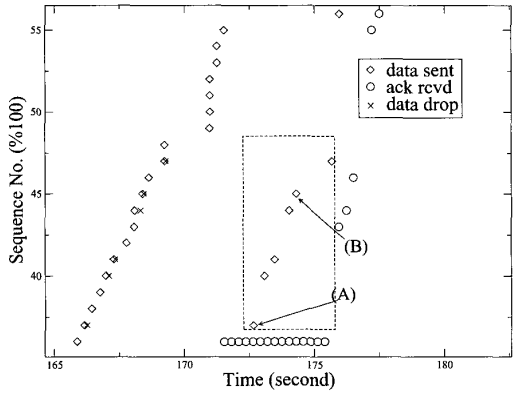

Fig. 8. Segment plot of SCTP during the handover

C. Behavior of SCTP

As discussed in Sec. III-B, the SCTP header format leaves enough space for large number of SACK blocks, which is beneficial in lossy environments. The behavior of SCTP during the handover is shown in Fig. 8. Again, there are six non-consecutive segment losses in a window of data during the handover (segment No. 37, 40, 41, 44, 45, 47 as shown by $\times$ in the figure). Each of the DupAcks received beginning at time 171.8 second (which acknowledges segment No. 36) have four SACK blocks. Starting from time 172.5 (point Fig. 8-A), the six lost segments are retransmitted within a short interval as shown in the area encircled by the dashed-line box. Compared with TCP-SACK, SCTP is not limited by the number of available SACK blocks, the sender can reduce the flight size more accurately to reflect the actual segments that have left the network, and the sender's window (minimum of *cwnd* and *rwnd*) can be opened faster than in TCP-SACK. Therefore, it can retransmit segment No. 45 and No. 47 without waiting for another acknowledgment (point Fig. 8-B). The time needed for SCTP to retransmit the six lost segments are about 3.5 seconds (time 172.5 - 176 secs.), while for TCP it required about 11 seconds (time 172.5 - 183.5 secs.). We can see that the SCTP sender can retransmit lost packets faster based on the information provided by the

TABLE III
SIMULATION PARAMETER SUMMARY—LINKS

Links	Link Bw (Kbps)	Prop. delay (ms)
CN-Router	28.8-1500	200
HA-Router	500	200
FA-Router	500	200
MH-HA/FA	150-2000	vary by distance

SACK. This will reduce the waiting time by several seconds, and greatly improve the throughput and efficiency.

V. SIMULATION SETUP

The network simulator, *ns-2*, was used to perform the simulation and obtain the results reported in this paper. The *ns-2* SCTP module [17] from University of Delaware, and the Mobile IP extension [18] from UC Berkeley have been used. The simulation topology for all the three protocols (TCP-Reno, TCP-SACK, or SCTP) studied in this paper is shown in Fig. 5.

Depending on the protocol (TCP-Reno, TCP-SACK, or SCTP) used to measure the performance for multiple non-consecutive packet losses in a single window, different transport protocol agents are attached to MH and CN. For example, if we are using TCP Reno, Agent/TCP/Reno will be attached to MH, and Agent/TCPSink will be attached to CN. The wireless transmission error was generated randomly by modifying the *ns-2* physical layer implementation. Because this change of physical layer code is independent of transport layer protocols, it has a fair effect on all three protocols we investigated.

The simulation time was 200 seconds. During this period, the MH will start moving from HA to FA at time 50 sec, then move back to HA beginning at time 150sec. Other parameters used in the simulation are shown in Table III and IV. In Table III, we varied the bandwidth between the CN and router for different simulation runs to study the effect of different amount of network bandwidth available on the Internet. The link bandwidth between MH and HA/FA was varied between 150Kbps-2Mbps to simulate the actual bandwidth obtained by MH due to the competition from other mobile hosts at the MAC layer. The propagation delay between the MH and HA/FA varies as the MH moves away or closer to the HA or FA.

Table IV shows the values of various protocol parameters used in our simulation. To ensure fair comparison, the values of the parameters were the same for all the three protocols (TCP-Reno, TCP-SACK, or SCTP) used in this study. As an example, since SCTP's default initial *cwnd* is 2 segments, while TCP and TCP-SACK's default initial *cwnd* is 1 segment, we set the initial *cwnd* for all the three protocols to two segments. We also set the *rwnd* limit and initial *ssthresh* for all the three protocols to 20 segments, which is the default initial value of *ssthresh* for TCP.

VI. RESULTS

A. Throughput Performance Measurements

We measured the number of data segments delivered from MH to CN during the 200 second simulation period. The number of segments received by CN as a function of various wireline bandwidths and wireless bandwidths are shown in Tables V to VII for TCP-Reno, TCP-SACK, and SCTP respectively. In these tables, "Wireless Bw." means the bandwidth between MH and HA or FA, and "Wireline Bw." represents the bandwidth between Router and CN.

In each of these tables, the throughput usually increases with an increase of the wireline and/or wireless bandwidth. However, in higher bandwidths, the throughput increase tends to saturate. Comparing Table V with Table VI, we can observe that the throughput improvement from SACK is more prominent at low bandwidths, as shown in bold numbers which correspond to scenarios with either (or both) the wireline or wireless bandwidth being low. The same rule applies to the comparison of Table VI with Table VII, where the throughput improvement comes from the unlimited number of SACK blocks. This is due to the fact that in low bandwidth networks, the packet losses are more frequently recovered by fast retransmission rather than by timeouts, as seen in Sec. II. In this kind of scenario, the SACK is more important in recovering lost packets.

TABLE IV
PROTOCOLS PARAMETERS SUMMARY—AGENTS

Header size (Bytes):	40 (Reno) 52 (TCP-SACK with timestamp) 52 SCTP
Payload size:	948 bytes
rwnd limit	20 segments
Initial *cwnd*	2 segments
Initial *ssthresh*	20 segments
Mac Layer	802.11

TABLE V
TCP-RENO THROUGHPUT

Wireline Bw. (Kbps)	Wireless Bw.(Kbps)				
	150	*200*	*384*	*1000*	*2000*
28.8	**287**	**316**	**451**	**544**	566
54	**283**	**489**	**774**	**971**	1010
200	**373**	**487**	**1205**	2148	2376
500	**300**	**616**	1399	2492	2723
1000	**419**	680	1626	2278	2695
1500	519	717	1224	2660	2715

TABLE VI
TCP-SACK THROUGHPUT

Wireline Bw. (Kbps)	Wireless Bw.(Kbps)				
	150	*200*	*384*	*1000*	*2000*
28.8	**323**	**391**	**455**	**563**	598
54	**431**	**560**	**934**	**973**	1046
200	**478**	**717**	**1262**	2528	2501
500	**430**	**793**	1472	2614	2826
1000	**468**	625	1643	2672	2957
1500	567	727	1816	2750	2796

TABLE VII
SCTP THROUGHPUT

Wireline Bw. (Kbps)	Wireless Bw.(Kbps)				
	150	*200*	*384*	*1000*	*2000*
28.8	**423**	**432**	**545**	**609**	589
54	**544**	**796**	**955**	**1028**	1077
200	**716**	**1485**	**1446**	2629	2681
500	**623**	**806**	1502	2657	2812
1000	**605**	1030	1672	2676	2963
1500	685	1168	1892	2768	2793

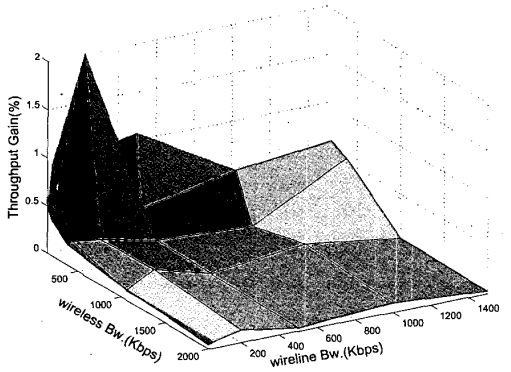

Fig. 9. Throughput gain of SCTP over TCP-Reno

B. Performance Gain of SCTP over TCP-Reno and TCP-SACK

In Fig. 9 and 10, we plot the throughput performance gain percentage of SCTP over TCP-Reno and TCP-SACK, i.e.

$$T(SCTP) - T(Reno)/T(Reno)$$

and

$$T(SCTP) - T(SACK)/T(SACK)$$

where $T(SCTP)$, $T(Reno)$ and $T(SACK)$ are the throughput of SCTP, TCP-Reno, and TCP-SACK during the 200 second simulation period respectively. From these two figures, we can observe clearly the advantage of SCTP in the low-bandwidth zone. Also, the throughput gain is more sensitive to wireless bandwidth changes than wireline bandwidth change. For example, if the bottleneck bandwidth is in wireline part, there is small gain when wireless bandwidth get over 500Kbps; whereas,if the bottleneck bandwidth is in wireless part, there is still considerable gain when wireline bandwidth is 1.5Mbps. This is because when the available bandwidth to application is controlled by wireless part, more SACK blocks available in SCTP will have more impact on the throughput gain.

VII. CONCLUSION

SCTP, a new transport layer protocol from IETF, allows a large number of SACK blocks. We have investigated the effect of a large number of SACK blocks in improving the end-to-end throughput during mobile IP handovers. We have compared the performance of TCP Reno, TCP-SACK, and SCTP

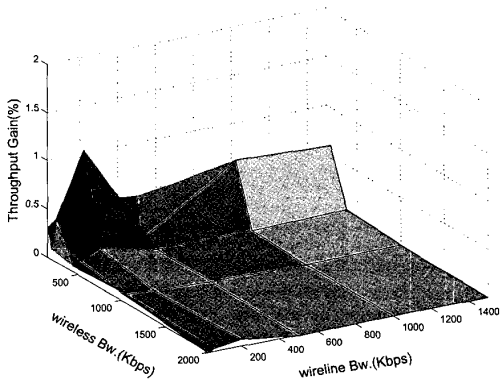

Fig. 10. Throughput gain of SCTP over TCP-SACK

in the presence of handovers, and the related congestion control mechanisms under different network scenarios. It's shown that SCTP benefits from its provision for a large number of SACK blocks, and obtains a higher throughput than TCP-Reno and TCP-SACK. We conclude that, when the bottleneck link bandwidth is low, SCTP can be used to improve the end to end throughput of mobile applications.

REFERENCES

[1] C. Perkins et. al., "IP Mobility Support." IETF RFC 2002, October 1996.
[2] T. S. Rappaport, *Wireless Communications Principles and Practice*, Prentice Hall, 1996.
[3] S. Seshan H. Balakrishna and R. H. Katz, "Handoffs in cellular wireless networks: The daedalus implementation and experience," *Wireless Personal Communications*, vol. 4, pp. 141–162, 1997.
[4] D. S. Eom, H. Lee, and M. Sugano et. al., "Improving TCP handoff performance in mobile IP based networks," *Computer Communications*, vol. 25, no. 7, pp. 635–646, May 2002.
[5] A. Martin A. Mihailovic N. Georganopoulos and A. Aghvami, "Adaptation of transport protocols for an IP-micromobility scheme," *IEEE International Conference on Communications, 2001. ICC 2001*, San Antonio, Texas, USA, pp. 2462 –2466, November 2001.
[6] F. Hu and N. Sharma, "The quantitative analysis of TCP congestion control algorithm in third-generation cellular networks based on FSMC loss model and its performance enhancement," *Proceedings of IEEE INFOCOM 2002*, New York, USA, pp. 407 –416, June 2002.
[7] M. Mathis et. al., "TCP selective acknowledgment options." IETF RFC 2018, October 1996.
[8] R. Stewart and Q. Xie et. al., "Stream control transmission protocol." IETF RFC 2960, October 2000.
[9] *Stream Control Transmission Protocol (SCTP): A Reference Guide.* Addison Wesley, 1 ed., 2001.
[10] A. Jungmaier, "Performance evaluation of the stream control transmission protocol," *Proceedings of the IEEE Conference 2000 on High Perfomance Switching and Routing*, Heidelberg, Germany, pp. 141–148, June 2000.
[11] M. Atiquzzaman and W. Ivancic, "Evaluation of SCTP multistreaming over satellite links." NASA Technical Report, 2002.
[12] P.T. Conrad Conrad G.J. Heinz A.L. Caro et. al., "SCTP in battlefield networks," *IEEE Military Communications Conference (MILCOM)*, McLean, VA, pp. 289–295, October 2001.
[13] C. E. Perkins, "Mobile Networking Through Mobile IP," *IEEE Internet Computing Online*, vol. 2, no. 1, January/February 1998.
[14] M. Allman, V. Paxson, and W. Stevens, "TCP Congestion Control." IETF RFC 2581, April 1999.
[15] K. Fall and S. Floyd, "Simulation-based Comparisons of Tahoe, Reno, and SACK TCP," *ACM Computer Communications Review*, vol. 26, no. 3, pp. 5–21, July 1996.
[16] *The Network Simulator - ns-2.* http://www.isi.edu/nsnam/ns/.
[17] *NS-2 SCTP Module Home Page.* http://pel.cis.udel.edu.
[18] *Mobile IP Extensions to the ns Network Simulator.* www.icsi.berkeley.edu/ widmer/mnav/ns-extension/.