Hindawi Publishing Corporation
EURASIP Journal on Wireless Communications and Networking
Volume 2011, Article ID 103027, 13 pages
doi:10.1155/2011/103027

Research Article

Improving SCTP Performance by Jitter-Based Congestion Control over Wired-Wireless Networks

Jyh-Ming Chen, Ching-Hsiang Chu, Eric Hsiao-Kuang Wu, Meng-Feng Tsai, and Jian-Ren Wang

Department of Computer Science and Information Engineering, National Central University, Jhongli 32001, Taiwan

Correspondence should be addressed to Eric Hsiao-Kuang Wu, hsiao@csie.ncu.edu.tw

Received 22 August 2010; Revised 8 January 2011; Accepted 20 January 2011

Academic Editor: Fabrizio Granelli

With the advances of wireless communication technologies, wireless networks gradually become the most adopted communication networks in the new generation Internet. Computing devices and mobile devices may be equipped with multiple wired and/or wireless network interfaces. Stream Control Transmission Protocol (SCTP) has been proposed for reliable data transport and its multihoming feature makes use of network interfaces effectively to improve performance and reliability. However, like TCP, SCTP suffers unnecessary performance degradation over wired-wireless heterogeneous networks. The main reason is that the original congestion control scheme of SCTP cannot differentiate loss events so that SCTP reduces the congestion window inappropriately. In order to solve this problem and improve performance, we propose a jitter-based congestion control scheme with end-to-end semantics over wired-wireless networks. Besides, we solved ineffective jitter ratio problem which may cause original jitter-based congestion control scheme to misjudge congestion loss as wireless loss. Available bandwidth estimation scheme will be integrated into our congestion control mechanism to make the bottleneck more stabilized. Simulation experiments reveal that our scheme (JSCTP) gives prominence to improve performance effectively over wired-wireless networks.

1. Introduction

Recently, wireless networks [1] play important roles in the next generation communication Internet. More and more novel services in business, entertainment, and social networking applications are widely offered over ubiquitous wireless networks by virtue of its characteristic of seamless mobility [2]. Since the demand of mobile users grows rapidly, the integration of wired and wireless networks is widely deployed. In such ALL-IP wired-wireless heterogeneous networks, the current trend of last mile deployment is towards wireless access networks. Due to the hybrid network topology, transport layer protocols should carry end-to-end semantics and perform well for communication services [3].

With the diversification of wireless access technologies, hosts equipped with wired or wireless network interfaces could access networked data and service anywhere. However, common transport layer protocols such as Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) assume to access one simple network path while transmitting data. It may be a waste of other network paths which could provide alternative to parallel transmission or reliability. Thus, many researchers aim at this multihoming issue and try to provide a new solution which exploits multiple network devices effectively. A general-purpose transport layer protocol which can cope with multihoming feature has been proposed by Internet Engineering Task Force (IETF), called Stream Control Transmission Protocol (SCTP) [4–6].

Since SCTP is originally designed to carry telephony signaling over IP-based networks. It can also be adopted as transport layer protocols like TCP and UDP. SCTP provides reliable and error-free data transmission which makes data delivery service more robust. It adopts selective acknowledgement (SACK) [7] of TCP enhancement. Besides, SCTP overcomes several security deficiencies of TCP by using four-way handshake and cookie mechanism. The major differences between TCP and SCTP are multihoming and multistreaming features [8]. SCTP multihomed hosts can establish an association with other SCTP hosts through its multiple network interfaces with individual IP addresses. An established SCTP connection may be constructed over several different paths experiencing distinct network conditions. As the primary path gets severely congested or experiences link failure, data traffic will be transferred to other alternative paths to increase the probability for reaching the receiver. Nevertheless, standard SCTP only uses multihoming feature for retransmission and link failure. Load sharing and load balancing are not supported yet. In [9] it demonstrates that SCTP which exploits multihoming feature can provide better performance than TCP over wireless scenarios. Another novel feature of SCTP is multistreaming. The stream of SCTP which delivers unidirectional data independently can avoid Head-of-Line blocking and benefit data delivering in time.

SCTP congestion control mechanism follows a minor modification from TCP [10]. TCP slow start and congestion avoidance phases are still adopted in SCTP, but there is no explicit fast-recovery phase. SACK provides packet delivery information that makes SCTP transmit new packets continuously. However, these problems which TCP met before over wireless networks should be solved in SCTP. TCP-like congestion control scheme cannot work well in wireless networks because of its inability to differentiate wireless loss from congestion loss [11]. Sender has no information to distinct loss events, so it treats all packets lost by bottleneck buffer overflowing during network congestion. But sometimes these losses may occur by fading or mobility. If wireless losses are treated as congestion losses, congestion window will be reduced to half unnecessarily. SCTP cannot utilize the network resource effectively.

Up to now, a great number of researches have been proposed for solving congestion control problem of TCP in the wired-wireless networks [12–14]. However, very few proposals address this issue and improve performance ineffectively over SCTP. In this paper, we present a new SCTP enhancement which considers the following characteristics. Our scheme should contain better loss differentiation scheme to alleviate misjudgments of loss events and effective congestion control mechanism to utilize better throughput and avoid causing network congestion. Besides, it would be best to have end-to-end semantics for scalability to keep off the efforts which should modify complicated network infrastructure. It is highly expected that a mobile device could have simultaneous Internet connectivity via multiple wireless network technologies, such as WLAN and 3G to increase resilience to path failure by distributing data across multiple end-to-end paths. SCTP carries crucial multihoming and multistreaming features and our scheme can measures the jitter and calculates jitter ratio independently per path. Consequently, the improved jitter-based congestion scheme of SCTP should adapt well in all kinds of wired-wireless networks such as the third generation, satellite, and sensor networks that there might exists asymmetric links since nodes have diverse radio transmission ranges.

The rest of this paper is organized as follows: Section 2 introduces several related work such as SCTP and TCP enhancements over wired-wireless networks. Section 3 describes the proposed scheme: a jitter-base congestion control scheme of SCTP. Beside, this paper strengthens the jitter-based loss differentiation scheme to avoid misjudging the loss events. Section 4 demonstrates the simulation results to evaluate our improvement of the proposed scheme. Section 5 concludes the proposed scheme and brings up future works.

2. Related Work

Standard SCTP scheme is effective for reliable data transfer in wired networks, but it suffers serious performance degradation in the heterogeneous networks due to misjudging the wireless and congestion losses. A good loss differentiation and congestion control scheme is required in the new generation IP networks. In this section, we briefly introduce recent solutions to address these problems in SCTP. Since TCP has the same problems as SCTP and end-to-end semantics of the proposed scheme is our main concern, wireless enhancement on TCP end-to-end approaches will also be introduced. Besides, we will specify why current SCTP modifications cannot work well in the hybrid wired-wireless topologies.

2.1. Wireless Enhancement on SCTP. There are several researches which aim at the issue of SCTP performance over heterogeneous networks. Current SCTP solutions over the wireless issue can be categorized into two categories: (1) intermediate node supported approach and (2) end-to-end approach. The main idea of intermediate node supported approach relies on the intermediate nodes, such as base station or router, using several mechanisms to help sender to differentiate loss events. Collaborative SCTP [15] and ECN-D SCTP [16] belong to this category. However, end-to-end approach only uses the information of sender and receiver transmission data to judge the loss event. WiSE [17] is a part of this category.

Collaborative SCTP, a new scheme with the collaboration of multiple entities and cross layer interactions, is designed to deal with variable bit error rate of wireless network, especially in high BER wireless channel. In contrast to other schemes, this approach exploits SCTP features, message-oriented and multistreaming, to cope with loss differentiation problem. Analysis indicates that the smaller the frame size, the greater the probability of successful transmission in the high BER wireless networks. Based on the characteristic of SCTP packet, SCTP sender can transmit chunks into small packets by disassembling large packets, to alleviate wireless losses in high BER wireless networks. The drawback of the disassembly function is to produce more overhead by IP and SCTP headers. In addition, other hosts, such as the base station and the receiver that receives fragmentation data should have the reassembly function to recover the original packets. The disassembly function also can help in differentiating wireless loss from congestion loss. Sender logs the bundle chunks and uses the records to distinguish loss events. If all chunks in an original packet have been lost, we would have viewed it as a congestion loss. Otherwise, the sender considers that wireless loss occurred and retransmits the lost packet without dropping the congestion window. After observing the simulation results, when BER is very low, the overhead of disassembly function may damage performance due to the transmission of more headers and control frames. SCTP hosts should try to estimate the current BER and decide when to activate these functions. Unfortunately, the implementation of this scheme is complicated because of cooperation of multiple layers and multiple entities. In addition, base stations may get burdened under the heavy traffic load since the execution of the disassembly function would exhaust the CPU resource and cause the poor performance.

ECN-D SCTP proposed a fine-tuned explicit congestion notification (ECN) mechanism [18] for SCTP in the wireless environment. Based on the ECN scheme over SCTP, it can differentiate loss events accurately to improve throughput performance. ECN is implemented in internal routers between sender and receiver. The router cooperates with active queue management (AQM) schemes, such as RED. If queue size in router exceeds the threshold, router is required to mark incoming packets to inform the congestion events instead of dropping the packets directly. When receiver gets the ECN signal, receiver will send ECN-echo marked acknowledgement to sender. After sender receives the acknowledgement which be marked with ECN-echo chunk, they can differentiate wireless losses from congestion losses by Congestion Coherence scheme. According to the scheme, there are two scenarios in which wireless losses occur: (1) only wireless losses occurred for current window, (2) wireless losses and congestion losses occurred simultaneously. In the former scenario, we can easily find out that wireless losses occurred because no ECN message is received. SCTP source should not reduce the congestion window size when no network congestion happens. In the latter, ECN-D SCTP proposed that it reacts to congestion only once for a window of data. In other words, SCTP sender does not identify the reason of lost packets; all packet losses will be viewed as noncongestion losses after reducing congestion window once.

Wireless SCTP Extension (WiSE) tries to exploit the multihoming feature of SCTP by selecting one of the available paths which has better condition for data transmission. WiSE uses available bandwidth estimation techniques to infer loss events which are due to congestion or radio channel errors. Furthermore, the proposed scheme continues to probe available bandwidth on the current transmission path and other alternative paths. If the primary path is under heavy load, such as severe congestion (Timeout), and alternative paths are slightly loaded, WiSE will switch the transmission to another better alternative path. Thus, the key point of this scheme is the accuracy of bandwidth estimation schemes. TCP Westwood [19, 20] is adopted for estimating the available bandwidth on primary path. WiSE has a good path selection scheme for SCTP multihoming, but the bandwidth estimation scheme of TCP Westwood still overestimates the available bandwidth. The incorrect estimation may result in poor performance in the wireless network. Besides, over asymmetric links, such as 3G wireless networks, cannot adapt this scheme because of its bandwidth estimation scheme based on monitoring ACK reception rate.

2.2. Wireless Enhancement on TCP End-to-End Approaches.

Congestion control of SCTP is a slight modification which is based on TCP congestion control. Therefore, we should refer to TCP enhancements over wireless networks. Intermediate node supported approaches violate the end-to-end semantics and need to modify intermediate nodes in support of detecting the network condition. In the hybrid wired-wireless networks, intermediate node supported approaches may lack of flexibility to extend network topology. Hence, we regard TCP end-to-end approaches as our main reference.

TCP Vegas [21, 22] aims to improve the end-to-end congestion avoidance mechanism of TCP. The main objective is to estimate the expected bandwidth for the connection in order to control the transmission rate that can avoid network congestion. This scheme defines *BaseRTT* value which represents the minimal round trip time during the transmission to calculate the expected transmission rate of this link. After receiving an acknowledgement, sender continues to update *ActualRTT* value which means the current round trip time to calculate the real transmission rate. The difference between *BaseRTT* and *ActualRTT* should be ranged between the thresholds which Vegas defined. If the difference is higher than upper bound threshold, congestion may occur since sending rate is too high. Thus, sender decreases one congestion window size. If the difference is smaller than lower bound, sender should increase one congestion window size so as to utilize the available bandwidth. Or else, sender should keep the sending rate. As we know, TCP Vegas suffers fairness problems when the connections start transmitting at different times. The *BaseRTT* is not the same in this circumstance. Besides, TCP Vegas is not suitable for wireless network since it cannot distinct loss events.

TCP Veno [23] is a loss differentiation scheme for the wireless environment and it is derived from TCP Vegas. This method provides another threshold to differentiate between wireless and congestion losses. Although the performance is improved in wireless environment, the loss differentiation scheme cannot work well when the random loss rate is high. And it still does not solve the problem of *BaseRTT*.

TCP Jersey [24] is another enhancement which improves network performance in wireless network. This scheme consists of two key components: one is congestion warning (CW) and the other is available bandwidth estimation (ABE). CW designs a simple packet marking scheme which modifies current ECN scheme to differentiate loss events. The main difference is that CW proposes that router should mark all packets while the average queue length exceeds the threshold. The nonprobabilistic scheme leaves sender to use proper congestion control strategy in different circumstance. Besides, it considers that original ECN information is not timely enough to react to variable network environment due to its parameter settings. The larger queue weight can be used to track the correct queue length and smooth the instantaneous queue length. ABE uses a rather simple estimator to estimate the available bandwidth by monitoring the returning ACK rate at the sender side. Sender calculates the optimum congestion window size for adjusting its rate when congestion occurred.

TCP Jersey makes good use of CW and ABE schemes to propose a rate-based congestion window control mechanism. With the help of these two schemes, if duplicated ACKs are received without CW mark, congestion window size should be kept the same size and sender retransmits the loss packets immediately. On the other hand, if duplicated ACKs are received with CW mark, sender will enter the rate control procedure to adjust its slow start threshold and congestion window size to the latest optimum congestion window size. This scheme sets its congestion window to a more sensitive value when different types of loss events occurred. But it still needs the router support, and this estimator cannot perform well when the traffic load gets heavy over reverse links or asymmetric links.

The main idea of JTCP [25] is to apply the jitter ratio to differentiate wireless losses from congestion losses and revise the Reno's congestion control scheme to adapt to wireless environments. Jitter ratio [26] is derived from the interarrival jitter, which is defined in Real-time Transport Protocol (RTP) [27]. Interarrival jitter is the variance of packet spacing at the receiver side and packet spacing at the sender side. In other words, it presents current path's status by the packet-by-packet delay. The interarrival jitter (D) is defined as follows:

$$D(i,j) = \left(R_j - R_i\right) - \left(S_j - S_i\right) = \left(R_j - S_j\right) - (R_i - S_i). \tag{1}$$

Note that i and j mean the index of continuous packets which sender sent. Rj represents the receiving time of packet j at receiver, and Sj represents the sending time of packet j at sender. When D is larger than zero, we can find that some cross-traffic is inserted into packet i and j. So it causes the packet j to be queued at the intermediate node for a while. The valuable information can be exploited to observe the congestion state of current transmission path approximately. Based on the above concept, jitter ratio (jr) is defined to estimate the ratio of queued packets. Relying on the interarrival jitter is sufficient to indicate the congestion event directly. JTCP provides an enhancement, jitter ratio, which can provide the estimation for the current status of bottleneck queue. The scheme tries to model the status of queue to prove jitter ratio is enough to provide effective information for detecting congestion events. Supposed that t_A (sec) is the packet-by-packet delay of the packets arrival at the router, and t_D (sec) is the delay of the packets departure from the router, and B is the service rate of router

$$B \approx \frac{1}{t_D}. \tag{2}$$

According to the equation, the ratio of queued packets can be defined as follows:

$$\begin{aligned}\frac{((1/t_A) - B)}{(1/t_A)} &\approx \frac{((1/t_A) - (1/t_D))}{(1/t_A)} \\ &= \frac{((t_D - t_A)/(t_A \times t_D))}{(1/t_A)} = \frac{t_D - t_A}{t_D} \\ &\approx \frac{(t_R(i) - t_R(i-1)) - (t_S(i) - t_S(i-1))}{(t_R(i) - t_R(i-1))} \\ &= \frac{D}{(t_R(i) - t_R(i-1))}.\end{aligned} \tag{3}$$

When traffic load becomes heavy, the queued packets are increasing in the bottleneck queue. If queued packets arrive at the maximum limit of the buffer, incoming packets at the router will be dropped right away. Thus, the ratio of dropped packets will approximated as the ratio of queued packets. Jitter ratio could be taken to predict the loss ratio. In the following, jitter ratio is formulated as follows:

$$\text{Jr} = \frac{D}{t_R(i) - t_R(i-1)}. \tag{4}$$

Due to the characteristic of predicting ratio of queued packets, jitter ratio plays an important role in JTCP for loss differentiation scheme. JTCP combines the jitter ratio with its congestion control and do proper actions for different loss events. Because of TCP, reaction time from sending a packet to receiving an ACK is about one RTT; average jitter ratio of JTCP will be sampled once per RTT. When sender's congestion window size is w, the average jitter can be defined as

$$\begin{aligned}\sum_{i=n-w}^{n-1} D_i &= \sum_{i=n-w}^{n-1} (R_{i-1} - R_i) - (S_{i-1} - S_i) \\ &= (R_{n-1} - R_{n-w}) - (S_{n-1} - S_{n-w}).\end{aligned} \tag{5}$$

Then the average jitter ratio is defined as follows:

$$\text{Jr} = \frac{D_{(n-m\times w, n-1)}}{R_{n-1} - R_{n-m\times w}}. \tag{6}$$

When TDACKs or timeout occurred, JTCP compares the average jitter ratio with the threshold, which is defined as the inverse of congestion window size. If average jitter ratio is greater than this threshold, JTCP regards the loss event as congestion loss and reduces its congestion window size to one half. Otherwise, the loss event will be viewed as wireless loss. Sender will not reduce the congestion window and will do fast retransmission immediately.

In [25], the jitter-based congestion approach has demonstrated performance advantage and interprotocol fairness over existing wireless TCP solutions, such as TCP Westwood and Newreno. Different from TCP Jersey, the jitter-based solution does not require routers to support ECN and there might be enough time to obtain ECN as the buffer at a congested router is overflowing. In this paper, we will apply the jitter-based congestion control mechanism to our wireless SCTP scheme. There are still some problems which need to be solved for jitter-based loss differentiation scheme in some circumstances. We will address this problem and increase the accuracy of loss differentiation scheme. Besides, we will adopt available bandwidth estimation scheme for more sensitive congestion control in order to achieve better throughput.

3. Proposed Scheme: JSCTP

Since SCTP becomes more attractive in the wired-wireless networks, this paper proposed a new congestion control scheme with end-to-end semantics which is based on jitter ratio over wired-wireless networks, called JSCTP. JSCTP adopts jitter-based loss differentiation scheme to improve the insufficiency of original congestion control scheme. Besides, we should do several modifications for SCTP due to the characteristic of multihoming. The original jitter-based loss differentiation scheme may make wrong decision for distinguishing loss events in some circumstances. We point out where the problem may occur and provide an enhancement to avoid this misjudgment. In order to minimize network congestion and improve performance, available bandwidth estimation scheme will be integrated into congestion control mechanism to make the bottleneck more stabilized. Later, we will describe our proposal specifically.

3.1. Jitter-Based Loss Differentiation Scheme over SCTP

3.1.1. Collect Samples of Interarrival Jitter. In order to distinguish loss event correctly, we adopt jitter ratio to observe the status of bottleneck queue. In the first step, we must record one-way interarrival jitter of packet which has been sent. SCTP specification does not mention how to record timestamp in the packet. Thus, we introduce a timestamp chunk to record the sending time and receiving time of packets. In our scheme, every JSCTP packet must bundle a timestamp chunk for recording the timestamp. JSCTP consumes extra network bandwidth due to using timestamp chunk which is 12 bytes long. But the one way time information can help us to eliminate noise which is caused by reverse channel. We will exploit this one way time information to calculate jitter ratio and estimate available bandwidth later.

As sender receives an ACK and intends to calculate jitter ratio, there would be some problems occurred due to TCP implicit acknowledge mechanism. If packets are lost during the transmission, TCP sender cannot get the correct receiving time of packets which are still outstanding before fast retransmission. There is a simple example shown in the Figure 1. Assume that sender sends the packets 111, 112, 113, 114, and 115 and packet 112 is lost. In a while, duplicated ACKs are received to trigger the sender to resend the lost packet 112 to receiver. After the retransmission is successful, receiver will return an ACK which informs sender that the sequence number of packet below 115 is received. The problem occurrs at the moment because we cannot know the real receive time of packets 113, 114, and 115. It may cause inaccurate jitter and influence the computation of jitter ratio.

The problem mentioned before would not occur in JSCTP. Gap ACK Blocks in the SACK of SCTP could solve this problem. Although data chunks that arrived at the receivers get lost or are out of order during the transmission, sender can figure out that the received SACK acknowledged which packet by Cumulative TSN ACK field and Gap ACK Blocks. Therefore, sender can record more stable receiving time to calculate jitter ratio.

3.1.2. Independent Jitter Ratio for Different Path. Because of SCTP multihoming feature, SCTP may use multiple network devices to establish an association to transmit data. When primary path is severely congested or failover, data will be transferred to alternative paths and transmitted continually. Different paths may have different propagation delays due to its routing paths or network devices. If jitter measurements of different paths are mixed up, sender may confuse to calculate accurate jitter ratio. For the reason, our scheme should be capable of this environment. JSCTP measures the jitter and calculates jitter ratio independently per path. We separate all the paths to use its own measurements. After loss events occurred, JSCTP executes congestion control mechanism according to jitter ratio of the transmission path. Thus, we can apply the jitter ratio mechanism for multihoming feature.

3.1.3. Decision Rule for Loss Differentiation. We introduce the average jitter ratio in JSCTP first. See the two subsections given above; average jitter ratio is redefined as follows:

$$\mathrm{Jr} = \frac{D_{(n-m\times(w/s),n)}}{R_n - R_{n-m\times(w/s)}}. \qquad (7)$$

Note that w means the congestion window size in the current transmission path and s is Maximum Segment Size (MSS). The quotient of these two parameters represents the sequence of segment which was sent in previous round trip time. The parameter m determines how much previous RTT should be taken into consideration. It is usually set to one to get the fast response. The main difference of average jitter ratio between original and redefinition is the parameter n which denotes the TSN of latest acked packet. Refering to Section 3.1.1, we can continue recording and updating the jitter ratio after the loss events occurred by using SACK. It means that we still can monitor the latest network condition. But original jitter ratio of TCP would not be updated until the retransmission is a success. Hence, we can get useful jitter ratio in time and make correct loss differentiation in JSCTP.

After introducing jitter ratio in JSCTP, we follow the original decision rule for loss differentiation. JSCTP distinguishes loss events by comparing average jitter ratio with specific threshold, k/w ratio. This ratio denotes that k packets may be queued at the bottleneck when SCTP sends w packets into network. Because of using AIMD scheme in SCTP, we infer that the value of k should not be greater than one MSS. A congestion loss event is presumed when jitter ratio is greater

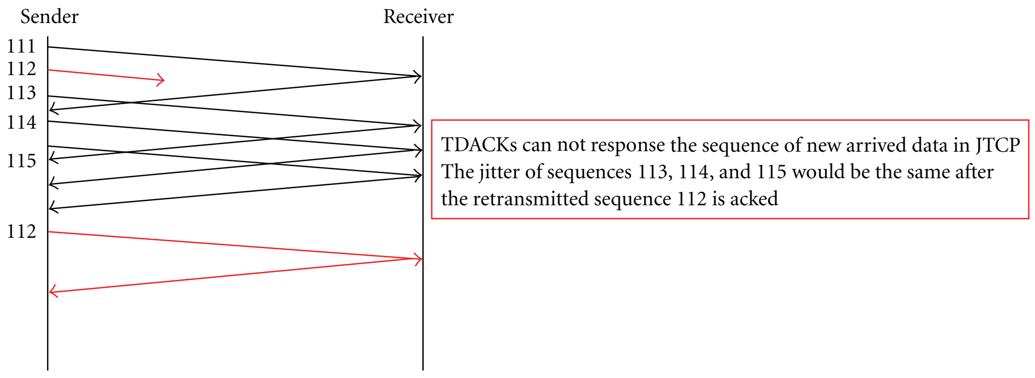
Sender
Receiver
111
112
113
114
115
112
TDACKs can not response the sequence of new arrived data in JTCP
The jitter of sequences 113, 114, and 115 would be the same after
the retransmitted sequence 112 is acked

Figure 1: Problem of the receiving time stamps.

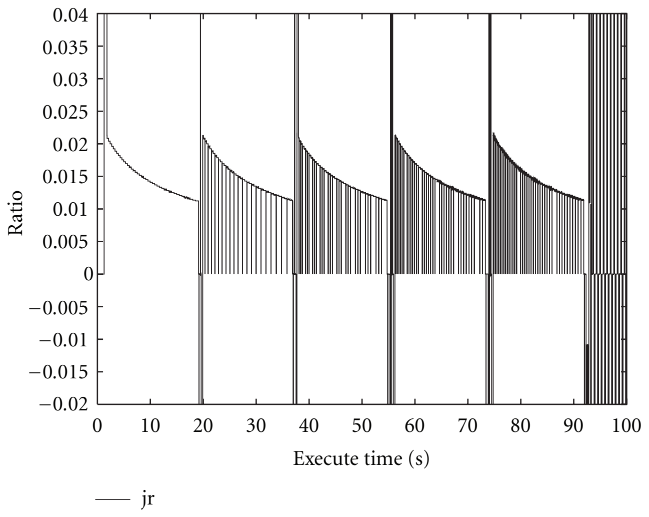

Figure 2: Average jitter ratio of JSCTP only congestion loss occurred.

than k/w ratio. Otherwise, we consider that the loss event is caused by wireless lossy links.

3.1.4. Ineffective Jitter Ratio Problem. In addition to the loss differentiation scheme, we found a problem called ineffective jitter ratio problem which makes JSCTP misjudge congestion loss as wireless loss in some situation. Misjudging congestion loss as wireless loss is more severe because the congestion window size would not be reduced to half. Sender may inject more packets that cause the bottleneck more congested and poor performance for connections over the bottleneck.

In order to address the problem properly, we try to simulate the scenario where only congestion losses occurred during the transmission. The jitter ratio is recorded in Figure 2. From the figure, we can find an irregular phenomenon that average jitter ratio is sometimes dropped to zero. After about 92 seconds, jitter ratio becomes instable. This is because congestion loss occurred and sender misjudges the loss event. The reason of misjudgment is that average jitter ratio is equal to zero at the moment. According to the decision rule, zero is always less than k/w ratio and sender misjudges loss event as wireless loss. So the sender still sends packets without reducing the congestion window size. Bottleneck queue is overflowed and burst packets may be dropped. So the variation of jitter ratio is large and performance is decreased.

By tracing the queue length of bottleneck carefully, we intend to identify the major reason why jitter ratio sometimes might be dropped to zero. Jitter ratio calculation is effective when bottleneck queue size rises and falls. As Figure 3, as long as the first packet and last packet experience the same propagation delay, it denotes that the bottleneck queue size is the same at the moment when transmitting the two packets. Therefore, the interarrival jitter $D(n - w/s, n - 1)$ is zero, which results in the average jitter ratio to be zero. At this moment, the calculation of average jitter ratio is ineffective to differentiate loss events.

In order to solve the problem, there are two solutions to be considered. The former, when calculating the jitter ratio, we use the jitter of packets which transmit in previous RTT ($m = 1$). If we set a larger value of m, we can consider more history to prevent the problem. But this method may not totally eliminate the problem. The latter, filter is used to smooth jitter ratio to lighten the influence of ineffective jitter ratio. The smooth jitter ratio is as follows:

$$\text{smooth}_{\text{jr}} = (1 - \alpha) \times \text{smooth}_{\text{jr}} + \alpha \times \text{jr}_{\text{sample}}. \quad (8)$$

If the sample of jitter ratio is zero, we ignore this ineffective jitter ratio to avoid influencing on the smooth jitter ratio. Experimental studies reveal that $\alpha = 0.05$ will achieve good performance. Besides, decision rule is revised that smooth jitter ratio is replaced with original jitter ratio to compare with k/w ratio.

3.2. Congestion Control Policy of JSCTP

3.2.1. Rate-Based Congestion Control. Original congestion control scheme of SCTP followed AIMD algorithm to probe available bandwidth roughly. But AIMD algorithm is not suitable for well utilizing available bandwidth due to blindly reducing congestion window. However, several available bandwidth estimation (ABE) schemes operate in terms of the returning rate of ACK packets over the reverse link.

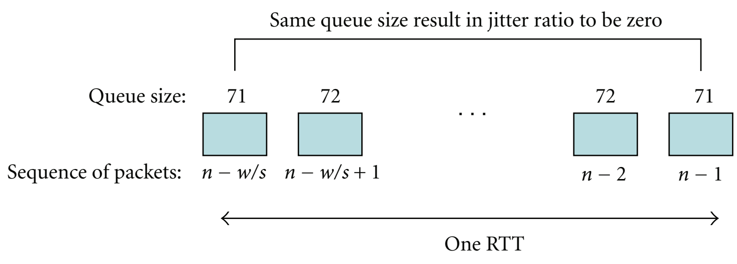

Figure 3: Ineffective jitter ratio calculation.

The assumption is based on good condition of reverse link and symmetric links. Since data is transmitted on forward link, the information for estimating available bandwidth should be provided by the forward rate of data packets. TCP New Jersey [28] proposed timestamp-based available bandwidth estimation (TABE) scheme which uses timestamp to record one way time information and return the information to sender for calculating available bandwidth. In our scheme, we integrate TABE scheme into congestion control of JSCTP. TABE calculation is according to the following equation:

$$R_n = \frac{\text{RTT} \times R_{n-1} + L_n}{(t_n - t_{n-1}) + \text{RTT}}. \quad (9)$$

R_n represents the estimate bandwidth when the nth SACK returns and L_n is the total packet size that nth SACK acknowledges. The value of t_n denotes the receiving time of nth packet at receiver side. RTT is the end-to-end round trip time delay.

Sender can adjust its sending rate to an optimal value by using the calculation of ABE. The optimal congestion window is calculated as follows:

$$\text{cwnd} = \frac{\text{RTT} \times R_n}{s}, \quad (10)$$

where s denotes the payload size of JSCTP packet.

3.2.2. Operation of JSCTP Congestion Control. Our proposed scheme, JSCTP integrates TABE and jitter-based loss differentiation scheme into congestion control of SCTP. After combining with these two schemes, JSCTP can differentiate loss event correctly and make proper congestion window adjustment to achieve better throughput in wired-wireless networks. There are two cases that JSCTP can detect loss events. The former is that sender receives four duplicated SACKs, the latter is that no SACK is returned to result in retransmission timer expired.

Figure 4 reveals the flowchart of JSCTP actions when receiving four duplicated SACKs. As NewReno, JSCTP only adjusts congestion window once during one RTT. So we should check duplicated SACKs by using Next RTT scheme at first. If duplicated SACKs occurred and sender just adjusted congestion window during the RTT, sender should enter immediate retransmit phase to retransmit packets. Else, JSCTP starts differentiating loss events in the next step. Follow the revised jitter-based loss differentiation scheme, if smooth_jr is greater than k/w ratio, we consider it as congestion loss and enter the rate adjustment state. Otherwise,

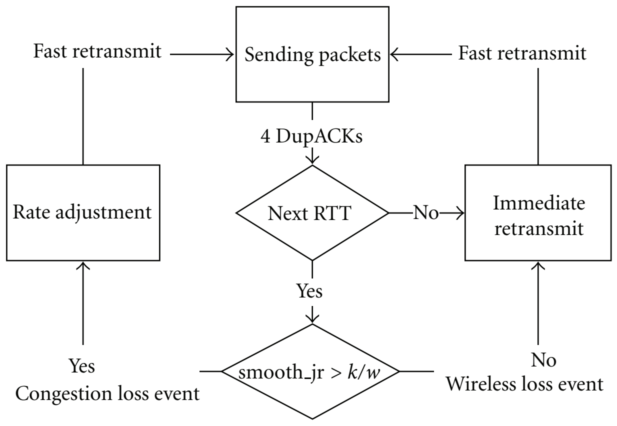

Figure 4: Flow chart of JSCTP when four DupSACKs occurred.

```
If (4-Dup SACKs are received) Then
    If (smooth_jr >= $k/w$) Then            // congestion event
        ssthresh = RTT* $R_n$ / s
        cwnd = ssthresh
        Retransmit lost packet
    Else                                     // wireless loss event
        Retransmit lost packet
    EndIf
EndIf
```

Algorithm 1: Pseudocode of JSCTP when four DupSACKs occurred.

the loss event is regarded as wireless loss and JSCTP enters immediate retransmit state. The procedure of two states can refer to Algorithm 1 which shows the pseudocode of duplicated SACKs that occurred on JSCTP. Rate adjustment state refers the optimal congestion window calculation of TABE. First, it set ssthresh to optimal congestion window. Subsequently, sender sets cwnd to ssthresh if the transmission is in congestion avoidance. In the immediate retransmit state, JSCTP retransmits lost packets without adjusting current congestion window size.

Another case is that retransmission timer expires when packets get lost. We still apply the jitter-based loss differentiation and rate adjustment in the algorithm. There is a little difference between duplicated SACKs that occurred.

Whether the loss event is considered as congestion or wireless loss, we set ssthresh to optimal congestion window which determined by TABE. The congestion window size is set to ssthresh when wireless loss event occurred. The main reason is that wireless loss may occurred by weak signal or mobility. We try to recover the transmission as soon as possible. Algorithm 2 shows the pseudocode of timeout occurred.

4. Simulation Results

After introducing our proposed scheme, we should validate JSCTP which is operative well over wired-wireless hybrid networks from simulation results. In our simulation, we use network simulator 2 (NS2; NS-2 network simulator. LBL (Online), Available: http://www.isi.edu/nsnam/ns/) for our experiment environment and have done some modifications of SCTP code to satisfy our demands. The reference simulation topology which describes wired-wireless networks is depicted in Figure 5. In this scenario, Node *S* denotes multihomed source node which establishes an association to node *D* through two wired network interfaces, and *D* is the multihomed destination node. R1 and R2 are routers which are connected to node *S* with wired links. Wireless channels placed on last hop and modeled with an error module between *D* and AP1, AP2. Bottleneck links are located on paths of router and access point. The upper path is selected for primary path, the other one is alternative. The propagation delay on the whole paths is set to about 45 ms. FTP traffic is applied to source node to generate long-live flow during the simulation. We assume that SCTP data chunk has the same size of 1456 Bytes which represents SCTP packets only bundle one data chunk. Besides, cross-traffic is considered for some experiments. If cross-traffic is UDP, the packet size is set to 100 Bytes. Otherwise, TCP is 1500 Bytes.

In the following, we will present several scenarios over wired-wireless networks. First, because of applying the jitter ratio in JSCTP, we should validate parameter settings of jitter ratio and k/w ratio to choose a suitable one for improving better performance. Bad value of parameters may raise the probability of misjudging loss events and degrade throughput. Second, we show that JSCTP can raise the throughput than original schemes outstandingly. JSCTP performance will be verified by different wireless loss rates, propagation delay, and network topology. Besides, JSCTP still need to keep the characteristics such as fairness and TCP friendliness. After these validations, we can demonstrate that JSCTP really performs well over wired-wireless networks.

4.1. Comparison of Different JSCTP Parameter Settings. Comparing with Figure 2, we use a filtered jitter ratio, smooth_jr to replace average jitter ratio against ineffective jitter ratio problem. We can observe that filtered jitter ratio is more approximate to the k/w ratio in the Figure 6. Moreover, jitter ratio which is equal to zero has been filtered out. Filtered jitter ratio follows the k/w ratio rise and fall regularly. In other words, the correct estimation of queued packets makes JSCTP perform well when differentiating congestion loss from wireless loss. We overcome the misjudgment which lets JSCTP slow down the sending rate in time.

Control parameter k is the significant factor which may influence the correctness of differentiating loss events. As we mentioned before, SCTP only add one MSS by AIMD scheme per RTT. Thus, the best setting of k should be the size of MSS. Because current simulation size of MSS is 1456 Byte. Figure 7 shows that we compared different values of k. k is equal to 1456 Bytes would gain better performance. If k is larger than MSS, congestion events may be misjudged for wireless losses which may cause more congestion. Thus, we define k as equal to MSS.

4.2. Wireless SCTP Throughput Comparison. In this subsection, we discuss wireless performance over JSCTP and original SCTP scheme. We also evaluate that embedded bandwidth estimation scheme in JSCTP and the performance is outstanding than without embedded TABE or not. The loss model is applied to generate loss events over wireless channel. There are several scenarios which we want to compare with. Bottleneck bandwidth is set to 2 Mb. In the following simulation, the error rate from 0% to 10% is put in one or both wireless links.

In Figure 8, error rate is applied only for primary path and there is no cross traffic through alternative link. In other words, there will be less timeouts during transmission because all retransmission through alternative path will be successful. In this scenario, it is suitable to evaluate the mentioned TCP congestion control schemes such as TCP Westwood and TCP New Jersey on the primary path. The main objective of this scenario is to observe that the difference of performance when only duplicated SACKs occurred. The simulation brings us a huge performance improvement by using JSCTP. Simulation result shows that JSCTP well-differentiates loss events to avoid unnecessary degrading window size and leads to higher throughput. Besides, JSCTP with available bandwidth estimation scheme can achieve better performance than without this scheme. TABE scheme makes JSCTP less congestion happen due to adjusting its congestion window to an optimum value. Thus, it can alleviate bottleneck loading to reduce amount of congestion.

Another scenario is shown in Figure 9; error rate is operated and the same as primary and alternative path. Currently, there are retransmission timeout expirations during transmission. Even if duplicated SACKs and timeout appeared simultaneously, throughput is still increased obviously over wireless lossy links, especially for lower BER rate. There is no obvious outstanding improvement for JSCTP with ABE. But it performs relative smooth during the variable loss rate.

We also compare JSCTP with original SCTP schemes in multihop scenario which is depicted in Figure 10. In this scenario, we set more hops and TCP cross-traffic in the primary and alternative paths. Here we study the scenario of congestion loss and wireless loss that coexisted on JSCTP and original SCTP scheme. As in Figure 11, we can find that JSCTP still outperforms original SCTP a lot in this scenario.

```
If (Timeout expired ) Then
    If (smooth_jr >= $k/w$) Then                //Occurred by congestion
        ssthresh = RTT * $R_n/s$
        cwnd = 1
        Retransmit lost packet
    Else                                        //Occurred by wireless loss
        ssthress = RTT * $R_n/s$
        cwnd = ssthresh
        Retransmit lost packet
    EndIf
EndIf
```

Algorithm 2: Pseudocode of JSCTP when timeout occurred.

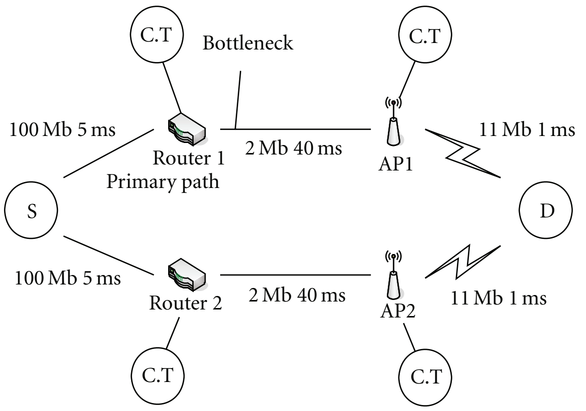

Figure 5: Single-hop topology.

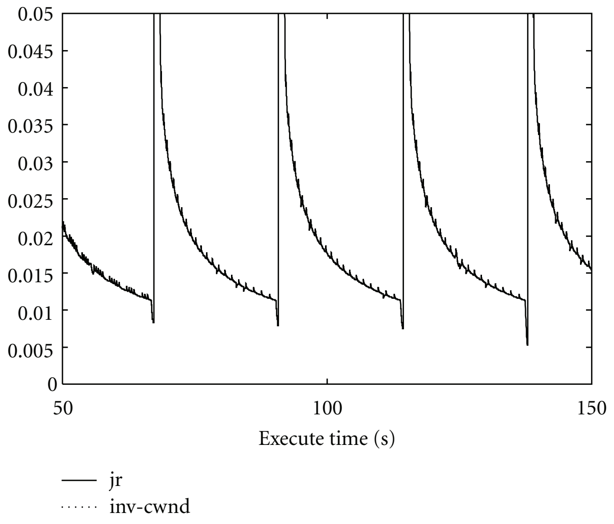

Figure 6: Comparison of filtered jitter ratio and k/w ratio.

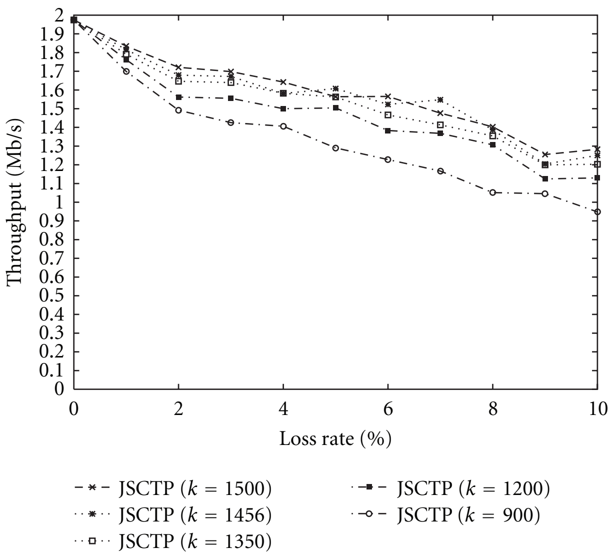

Figure 7: Different value of k.

If wireless loss rate is zero, JSCTP can utilize more bandwidth by available bandwidth scheme. When wireless loss rate is higher than zero, JSCTP can distinct loss events directly to achieve higher throughput than original SCTP.

In this case, propagation delay is the variance of simulation metric. Since SCTP sends packets about one RTT because of congestion control. Length of propagation Delay may affect average throughput. In our scenario, we apply 1% loss rate to the wireless channel. As Figure 12 shows, JSCTP has great accomplishment over this scenario. When the propagation delay is increased, the influence of loss misjudgment is more severe. JSCTP can maintain good throughput as the propagation delay is increased.

4.3. Interprotocol Fairness. One of the critical points which SCTP implementation should keep to maintain is fairness. No matter what environment is, every JSCTP connection should share fair bottleneck bandwidth during transmission. Thus, we should address this issue and check whether our scheme obeys this metrics or not. In this scenario, we change JSCTP and SCTP connections from 1 to 15. These connections transmit data at the same time. Hu and

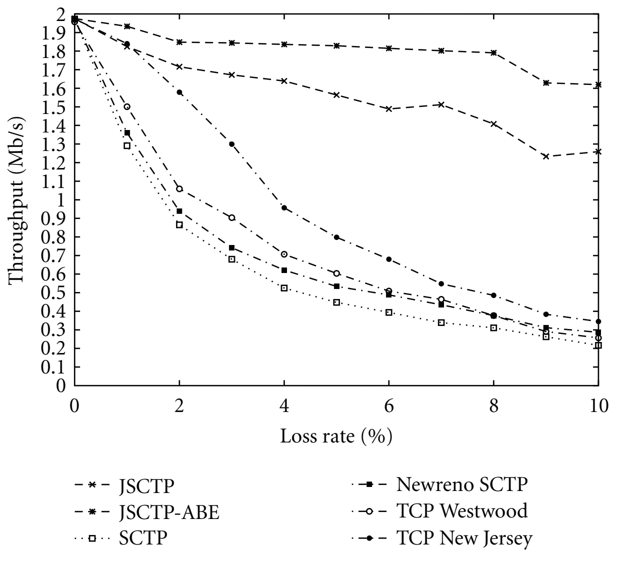

FIGURE 8: Throughput comparison with 1∼10% loss on primary path.

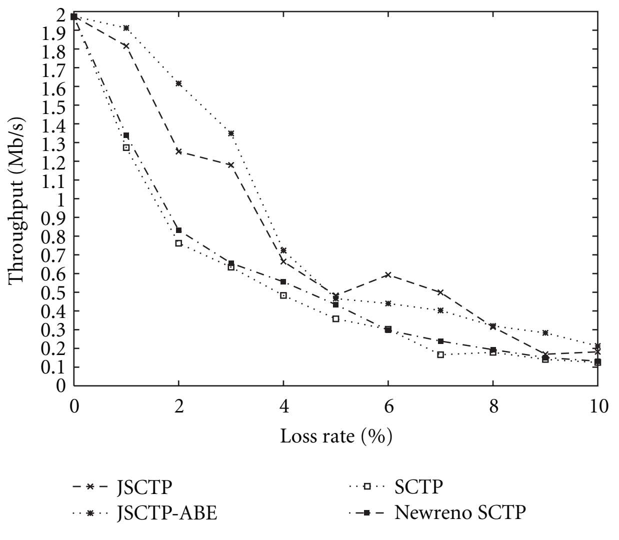

FIGURE 9: Throughput comparison with 1∼10% loss on both paths.

Yeung [29] proposed an evaluative fairness equation to qualify this metric

$$\text{Fairness} = \frac{\left(\sum_{i=1}^{n} b_i\right)^2}{n \times \left(\sum_{i=1}^{n} b_i^2\right)}. \tag{11}$$

Note that b_i represents how many parts of bandwidth which connection i used in this link, and n denotes the number of connections. If the fairness result is approached to 1, it means that all connections have been allocated fairer bandwidth.

We try to run this scenario over 1% and 5% wireless loss rate. Through Figures 13, 14, simulation results reveal that JSCTP has good ability for interprotocol fairness even if loss rate is variable. Our modification of congestion control scheme complies with fairness issue.

4.4. TCP Friendliness. Currently, TCP has been widely used in the Internet. Most of data traffic has been carried on TCP. Thus, our proposed protocol needs to follow TCP friendliness semantic to avoid stressing TCP traffic in wireless networks. To validate this characteristic, we design a scenario to address this issue. This scenario contains ten connections over 1% and 5% wireless loss rate environment, five for JSCTP and the others for TCP connections. When simulation starts, TCP connections begin data transmission. After 30 seconds, JSCTP connections start transmitting data.

Figure 15 shows that when 1% wireless loss rate is performed, TCP congestion control scheme cannot distinct loss events so that it drops the congestion window unnecessarily. When JSCTP connections start up, JSCTP connections perform better than TCP connections because JSCTP connections have ability to distinct loss in order to utilize bottleneck bandwidth which TCP connections released.

In Figure 16, we can easily observe that JSCTP can achieve higher throughput than TCP in 5% wireless loss rate environment. The main reason is that TCP cannot utilize bandwidth over high wireless error rate environment. Therefore, even if JSCTP connections start up at 30 seconds, TCP still remains the same throughput because they cannot utilize bandwidth anymore.

5. Conclusion

This paper presented a new congestion control scheme with end-to-end semantics to improve SCTP performance over wired-wireless networks. The new SCTP protocol, JSCTP adopts jitter ratio to differentiate wireless loss from congestion loss. In order to combine with jitter ratio, we add addition timestamp chunk in SCTP and use SACK to record correct one way time information for calculating jitter ratio. Because of multihoming, different paths should maintain its parameters of jitter ratio when transmitting through the path.

Besides, we correct the ineffective jitter ratio problem to avoid misjudging wireless loss to congestion loss. JSCTP could filter out ineffective jitter ratio and avoid sender overflowing the bottleneck queue. After adjusting decision rule of loss differentiation mechanism, we integrate timestamp-based available bandwidth estimation scheme into JSCTP congestion control mechanism. It can make up for the overhead of timestamp chunk and fully utilize the one way time information. TABE could eliminate the effect of reverse channel to calculate optimal congestion window correctly. Furthermore, the sensible value of dropping congestion window can stabilize bottleneck queue and cause less congestion.

Simulation results show that JSCTP is indeed a practical solution for wireless IP communications. We show that our propose scheme, JSCTP, guarantees better bandwidth utilization, fairness, and TCP friendliness over wireless lossy links.

Jitter-based loss differentiation scheme performs well for SCTP and TCP protocol over wired-wireless networks. But it still may misjudge loss events under high BER rate or complicated network topology. The control variable k of

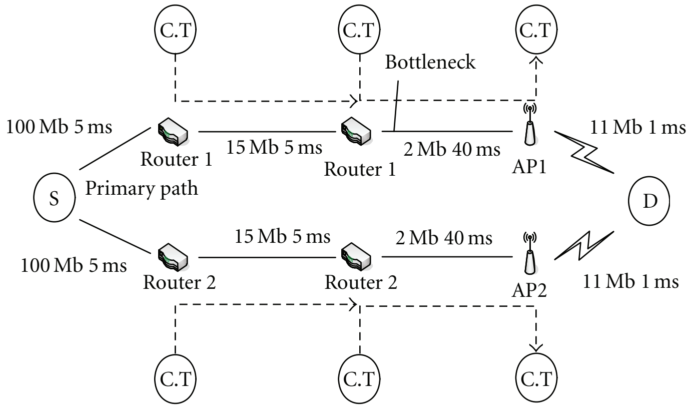

Figure 10: Multihop simulation topology.

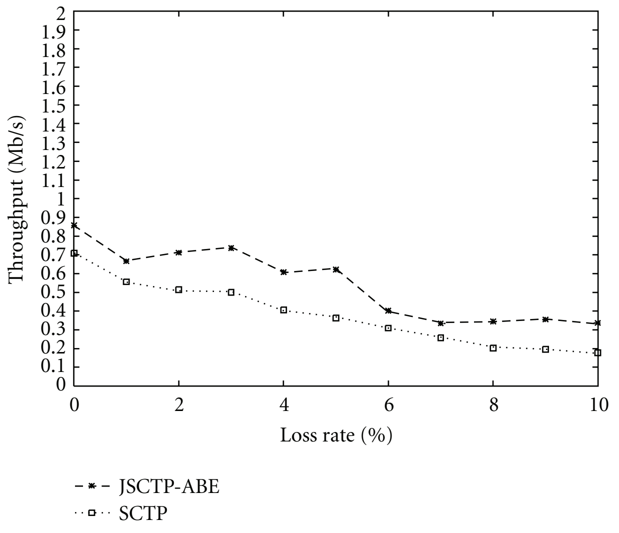

Figure 11: Throughput comparison in multihop scenario.

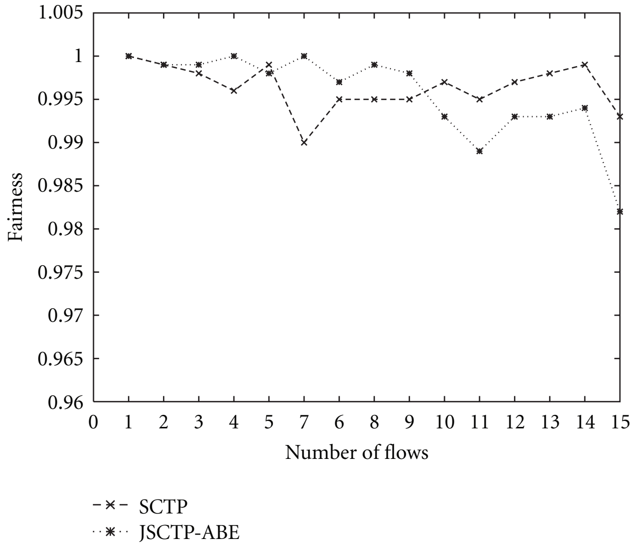

Figure 13: 1% loss rate of fairness.

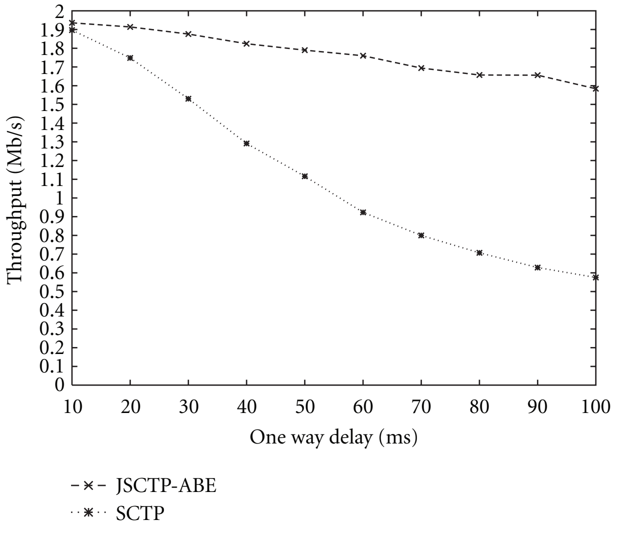

Figure 12: One way propagation delay from 10∼100 ms with 1% loss.

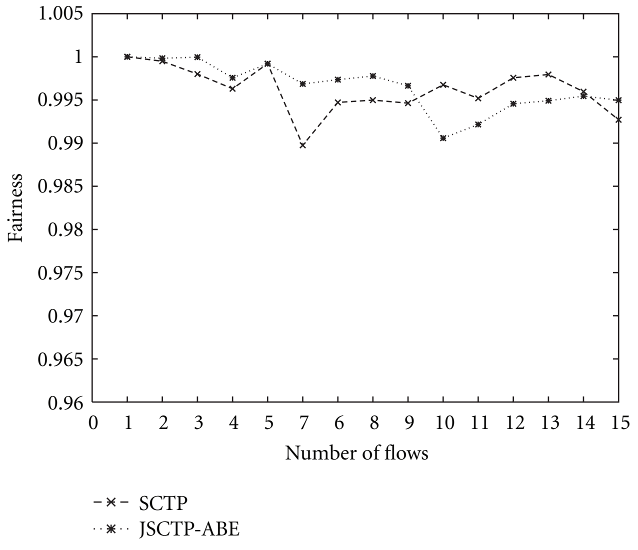

Figure 14: 5% loss rate of fairness.

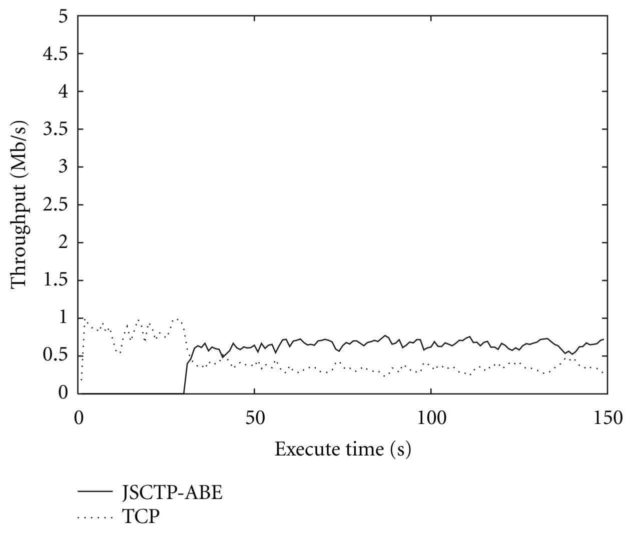

FIGURE 15: 1% loss rate of TCP friendliness with 5 Mb bottleneck.

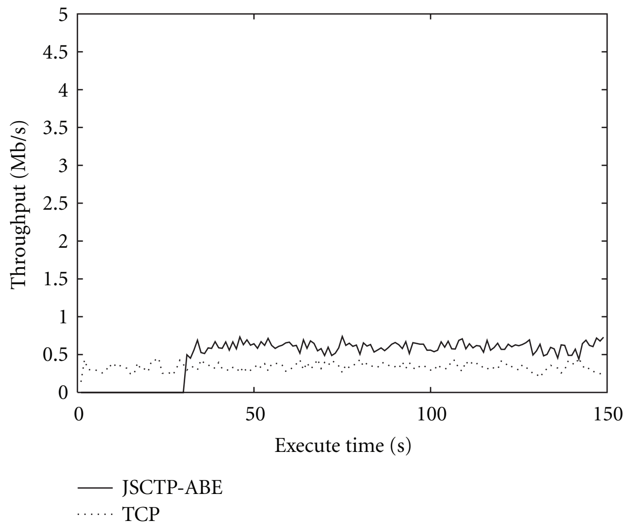

FIGURE 16: 5% loss rate of TCP friendliness with 5 Mb bottleneck.

our scheme is defined as a static value. Actually, static value could not react to the variation network. k should follow some factors to be tuned up dynamically. On the other hand, JSCTP needs the feedback information to calculate jitter ratio and available bandwidth. If links are congested or broken, feedback information may not return to sender. Thus, we cannot differentiate loss events. Fortunately, we can transfer data through alternative paths to keep off bad path by SCTP multihoming feature. Thus, a fine-tuned jitter-based congestion control mechanism should be improved continually over all wireless networks.

Acknowledgments

This work was sponsored in part by National Science Council Taiwan under project NSC 99-2219-E-002-021 and project NSC 99-2631-S-008-004, and sponsored in part by Taiwan Education Ministry under the project no. 995903 and sponsored in part by Institute for Information Industry under the vehicular parking system project.

References

[1] QI. Bi, G. I. Zysman, and H. Menkes, "Wireless mobile communications at the start of the 21st century," *IEEE Communications Magazine*, vol. 39, no. 1, pp. 110–116, 2001.

[2] A. Greenspan, M. Klerer, J. Tomcik, R. Canchi, and J. Wilson, "IEEE 802.20: Mobile broadband wireless access for the twenty-first century," *IEEE Communications Magazine*, vol. 46, no. 7, pp. 56–63, 2008.

[3] K.-C. Leung and V. O. K. Li, "Transmission Control Protocol (TCP) in wireless networks: issues, approaches and challenges," *IEEE Communications Surveys*, vol. 4, no. 4, pp. 64–79, 2006.

[4] R. Stewart and C. Metz, "SCTP: new transport protocol for TCP/IP," *IEEE Internet Computing*, vol. 5, no. 6, pp. 64–69, 2001.

[5] R. Stewart, "Stream Control Transmission Protocol.," Internet Engineering Task Force (IETF), RFC 2960, 2000.

[6] H.-S. Liu, C.-C. Hsieh, H.-C. Chen, C.-H. Hsieh, and W. J. Liao, "Exploiting mult-link SCTP for live broadcasting service," in *Proceedings of IEEE Vehicular Technology Conference*, Taipei, Taiwan, May 2010.

[7] M. Mathis, "TCP Selective Acknowledgments Options," Internet Engineering Task Force (IETF), RFC 2018, 1996.

[8] S. Fu and M. Atiquzzaman, "SCTP: state of the art in research products, and technical challenges," *IEEE Communications Magazine*, vol. 42, no. 4, pp. 64–76, 2004.

[9] J. Shi, Y. Jin, H. Huang, and D. Zhang, "Experimental Performance Studies of SCTP in Wireless Access Networks," in *Proceedings of the International Conference on Communication Technology (ICCT '03)*, pp. 392–395, April 2003.

[10] M. Allman, V. Paxson, and W. Stevens, "TCP Congestion Control," Internet Engineering Task Force (IETF), RFC 2581, 1999.

[11] G. Xylomenos, G. C. Polyzos, P. Mähönen, and M. Saaranen, "TCP performance issues over wireless links," *IEEE Communications Magazine*, vol. 39, no. 4, pp. 52–58, 2001.

[12] A. Bakre and B. R. Badrinath, "I-TCP: indirect TCP for mobile hosts," in *Proceedings of the 15th International Conference on Distributed Computing Systems*, pp. 136–146, June 1995.

[13] J.-H. Hu, K. L. Yeung, S. C. Kheong, and G. Feng, "Hierarchical cache design for enhancing TCP over heterogeneous networks with wired and wireless links," in *Proceedings of IEEE Global Telecommunications Conference*, vol. 1, pp. 338–343, 2000.

[14] YE. Tian, K. Xu, and N. Ansari, "TCP in wireless environments: problems and solutions," *IEEE Communications Magazine*, vol. 43, no. 3, pp. S27–S32, 2005.

[15] S. Liu, S. Yang, and W. Sun, "Collaborative SCTP: a collaborative approach to improve the performance of SCTP over wired-cum-wireless networks," in *Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks (LCN '04)*, pp. 276–283, November 2004.

[16] G. Ye, T. N. Saadawi, and M. J. Lee, "Improving stream control transmission protocol performance over lossy links," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 4, pp. 727–736, 2004.

[17] R. Fracchia, C. Casetti, C. F. Chiasserini, and M. Meo, "A WISE extension of SCTP for wireless networks," in *Proceedings of*

IEEE International Conference on Communications (ICC '05), pp. 1448–1453, May 2005.

[18] K. K. Ramakrishnan and S. Floyd, "A Proposal to add Explicit Congestion Notification (ECN) to IP," Internet Draft, Work in progress, January 1999.

[19] S. Mascolo, C. Casetti, M. Gerla, M. Y. Sanadidi, and R. Wang, "TCP Westwood: bandwidth estimation for enhanced transport over wireless links," in *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, pp. 287–297, July 2001.

[20] M. Gerla, M. Y. Sanadidi, R. Wang, A. Zanella, C. Casetti, and S. Mascolo, "TCP westwood: Congestion window control using bandwidth estimation," in *Proceedings of IEEE Global Telecommunicatins Conference (GLOBECOM '01)*, pp. 1698–1702, November 2001.

[21] L. S. Brakmo and L. L. Peterson, "TCP Vegas: end to end congestion avoidance on a global internet," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 8, pp. 1465–1480, 1995.

[22] K. Takagaki, H. Ohsaki, and M. Murata, "Analysis of a window-based flow control mechanism based on TCP Vegas in heterogeneous network environment," in *Proceedings of the International Conference on Communications (ICC '01)*, pp. 3224–3228, June 2000.

[23] C. P. Fu and S. C. Liew, "TCP Veno: TCP enhancement for transmission over wireless access networks," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 2, pp. 216–228, 2003.

[24] K. Xu, Y. Tian, and N. Ansari, "TCP-Jersey for wireless IP communications," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 4, pp. 747–756, 2004.

[25] E. H. K. Wu and M. Z. Chen, "JTCP: Jitter-based TCP for heterogeneous wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 4, pp. 757–766, 2004.

[26] S. Y. Chen, E. H. K. Wu, and M. Z. Chen, "A new approach using time-based model for TCP friendliness rate estimation," in *Proceedings of the International Conference on Communications (ICC '03)*, pp. 679–683, May 2003.

[27] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Application," Internet Engineering Task Force (IETF), RFC 1889, 1996.

[28] K. Xu, Y. Tian, and N. Ansari, "Improving TCP performance in integrated wireless communications networks," *Computer Networks*, vol. 47, no. 2, pp. 219–237, 2005.

[29] J. H. Hu and K. L. Yeung, "FDA: a novel base station flow control scheme for TCP over heterogeneous networks," in *Proceedings of the 20th Annual Joint Conference on the IEEE Computer and Communications Societies (IEEE INFOCOM '01)*, pp. 142–151, April 2001.