Throughput Modeling of TCP Parallelization

Yi-Cheng Chan, Fang-Chun Liu, and Yuan-Tzu Yen
Department of Computer Science and Information Engineering
National Changhua University of Education
No. 2, Shi-Da Road, Changhua 500, Taiwan, R.O.C.
Email: {ycchan@cc, m95612006@mail, s94610053@mail}.ncue.edu.tw

Abstract–*Modeling TCP performance is an important issue over the past decade. In this paper, we address how to predict parallel TCP throughput as a function of feasible network parameters without packet loss rate. The proposed model consider all the TCP's behavior in slow-start, congestion avoidance, and fast retransmit and recovery phase. The NS2 simulation results confirm the effectiveness of the proposed model.*

Keywords: parallel TCP, throughput modeling, Internet, transport layer protocol.

1. Introduction

Transmission Control Protocol (TCP), which has been described in IETF RFC793 [1] in 1981, is an underlying reliable data transfer protocol. It is widely used by many network applications, including WWW, FTP, e-mail, and the remote access. Recent measurement shows that from 60% to 90% of today's Internet traffic is carried by TCP [2]. The behavior of TCP is therefore tightly coupled with the overall Internet performance.

With the faster growth of the Internet traffic and the more varied types of networks than the past, traditional TCP protocol has been challenged gradually for efficient transit. Due to the widespread use of TCP, its performance modeling has received a lot of attention in the literature [3].

On today's Internet, the bandwidth delay product (BDP) continues to grow. Higher BDP of the Internet is problematic for TCP since its additive increase multiplicative decrease (AIMD) algorithm [4]. The AIMD algorithm becomes the performance bottleneck because of its slow response on adjusting window size. There are two problems described as follows. First, linearly increasing the congestion window makes it grow slowly. That leads the congestion window takes time to become large enough to fit the available bandwidth. Second, the congestion window would be decreased dramatically due to loss events, like timeout or received duplicated ACKs.

The AIMD algorithm has been shown to be a limiting factor of TCP performance. To deal with such throughput problem, the solutions can be divided into two categories. One is to modify TCP congestion control. The other is parallel TCP.

Some proposals modify the behavior of TCP to overcome the limitations of a single TCP connection, such as FAST TCP [5], HS-TCP [6], BIC-TCP [7], H-TCP [8], STCP [9], and LTCP [10]. By adjusting TCP control mechanism, they remove the restriction of the linearly increasing window size, and reduce the impact of the loss events which are not led by the congestion event. These TCP variants have successfully shown good performance in the utilization of the high bandwidth network. However, they have serious fairness problem with respect to the connections that use standard TCP [11].

Another approaches are described about parallel TCP, which creates multiple connections between a communication pair. Parallel TCP can achieve higher throughput than a single TCP connection does because of the following two reasons. First, the increase or recovery of parallel TCP's congestion window size can be fast to quickly fit the available bandwidth. Second, parallel TCP reduces the severity of a congestion event. When the network becomes congestion, it is probably that not all of the parallel TCP connections face the congestion lose.

Several applications use parallel TCP connections to increase TCP throughput, such as Netscape browser and SRB (Storage Resource Broker). Both GridFTP and XFTP are the examples that support parallel data transfer in the computational grids. Furthermore, some researches modify the congestion control of parallel TCP to provide better fairness with respect to single TCP connection. These modifications are TCP-P [12], TCP/DCA-C [13], A-TCP-P [14], etc.

With the importance of parallel TCP, we focus on its throughput prediction model. In order to derive the mod-

This work is supported by the National Science Council, Taiwan, R.O.C., under Grant NSC 97-2221-E-018-016.

eling expression, we consider completely about parallel TCP's behavior in slow-start, congestion avoidance, and the fast retransmit and recovery phase. Our aim is to find out the throughput equation of parallel TCP under certain real network parameters.

The rest of this paper is organized as follows. We begin by reviewing related work in Section 2. In this section, we present an overview of TCP throughput model. Section 3 presents the detailed development of the proposed model. Firstly, we describe the assumptions those are used to construct our model. Then build up the model. Section 4 we evaluate the performance of the proposed model. Finally, the conclusions and future work are given in Section 5.

2. Related Work

Modeling TCP performance is an important issue that attracts research attention over the past decade. Lots of researches used the network parameters to model the behavior of TCP transfer to predict throughput [15] [16] [17] [18]. The main focus on this section is to show the evolutions of two analytical TCP models described as follows.

Hacker, et al. present a model for the upper bound of the throughput of N parallel TCP connections [17]. Authors assume that MSS and RTT are stable across all TCP connections. All of the connections are between two hosts, and thus all connections have the equivalent RTT value when the network remains congested. The throughput model can be stated as:

$$Throughput \leq \frac{MSS}{RTT}[\frac{1}{\sqrt{p_1}} + \frac{1}{\sqrt{p_2}} + ... + \frac{1}{\sqrt{p_N}}], \quad (1)$$

where p_i is the packet loss rate for connection i. The upper bound is useful when the network is not congested.

The second parallel TCP model we address here is proposed by Wu, et al. [18]. This model shows the throughput of each parallel TCP connection for the congestion avoidance phase. Only congestive loss is considered. Authors express the throughput model by the buffer size, bandwidth, number of connections, and the delay time.

A simple dumbbell topology was used. Let B be the bottleneck bandwidth, Q be the buffer size, and T be the end-to-end round-trip delay. The maximum and minimum values of the parallel TCP congestion window size can be expressed as: $\frac{BT+Q}{2} \leq \Sigma_{i=1}^{N} W_i \leq BT + Q$. And the throughput formula for each connection can be stated as:

$$Throughput = \frac{3B}{4N} \times \frac{B^2T^2 + 2BTQ + Q^2}{B^2T^2 + BTQ + Q^2}. \quad (2)$$

The authors also make the following revisions to the expression of the duration time of transfer. Two round-trip times (T) are added for the delay of the congestion detection. Besides, one T is added for the fast retransmit and fast recovery. So we can get the revised expression of the throughput formula as follows.

$$Throughput = \frac{3B}{4N} \times \frac{B^2T^2+2BTQ+Q^2+8NS(BT+Q)}{B^2T^2+BTQ+Q^2+6NS(BT+Q)}. \quad (3)$$

This model can predict the throughput of parallel TCP without knowing the packet loss rate. However, we examine this model and find the predicted throughput is imprecise when B is large. For example, under the environment that B equals to 45Mbps, RTT is 0.1s, Q equals to BDP, and N is set to 1 to 50, the average prediction is 37Mbps. In this case, that of the simulated result used NS2 equals to 44Mbps. The second case, B equals to 100Mbps, RTT is 0.1s, Q equals to BDP, and N is 1 to 100, the average prediction is 83Mbps. In this case, that of the simulated result is 98Mbps. It shows the huge bias is within this throughput model when B is large.

The focus of this paper is to develop a parallel TCP throughput model under feasible network parameters. TCP's behavior in all three phases are considered. Unlike most analytic models, the proposed model does not need the packet loss rate of the network. It may be practical for TCP sources to predict the throughput. We discuss the proposed model in the following section.

3. Proposed Model

In this section, the assumptions of our model will be illustrated first. Then the developing of the model is described in detail.

3.1. Assumptions

Our model is based on the TCP Sack that is widely used on the current Internet. We assume a TCP source open all its connections at the same time and always has data to send. We also model TCP's behavior in terms of rounds. A round starts with the transmission of the current size of TCP congestion window. In our model, the duration of a round is equal to the round-trip time.

In our model, packets are lost in the cause of congestion. We do not consider the effect of random loss. Besides, to simplify the analysis, we assume all connections of a parallel TCP source perceive the loss event in the same round. The way which TCP source detects packet loss is base on receiving V duplicate ACKs. V usually be three. Based on the above assumptions, we derive the parallel TCP throughput expressions as follows.

3.2. Developing of the Model

We intend to derive parallel TCP throughput equations which are expressed by real network parameters. The four

variables or parameters used in our model are defined as follows. B is the bandwidth of the bottleneck link. T is the end-to-end round-trip delay (without queuing time). The drop-tail router with buffer size Q is in the bottleneck link. N is the number of parallel TCP connections

We consider the behavior of parallel TCP behavior completely in three phases: slow-start, congestion avoidance, and the fast retransmit and recovery phase. We define Y to be the number of packets sent during whole transfer, and A to be the duration of transfer. Then the throughput can be shown as

$$Throughput = \frac{Y}{A}. \quad (4)$$

3.2.1. Slow-Start Phase

The slow-start phase enables a TCP connection to discover the available bandwidth by gradually increasing the amount of data injected into the network. Upon receiving an ACK, the congestion window size is increased by one packet, like Fig. 1.

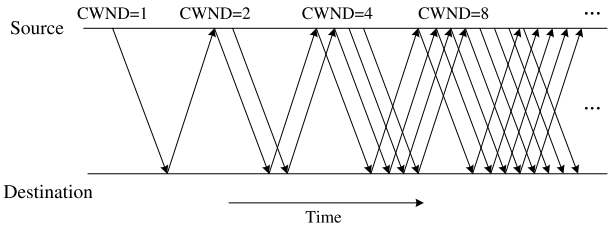


Figure 1. Packets in transit during slow-start for a TCP connection.

Let U_{SS} be the number of rounds in the slow-start phase (from 0^{th} to (U_{SS} -1)th round). We use Y_{SS} to indicate the total data sent in the slow-start phase. All the connections of parallel TCP exponentially increase their congestion window size, and we have

$$Y_{SS} = N(2^0 + 2^1 + ... + 2^{U_{SS}-1}) = N(2^{U_{SS}} - 1), \quad (5)$$

where N is the number of parallel TCP connections.

The congestion window of each TCP connection in the slow-start phase expands exponentially until one of the following two events occur. One is the value of congestion window size reaches the slow-start threshold $(SSTHRESH)$. At this time, it leaves slow-start phase and enters congestion avoidance phase. The other packet loss event happens. This moment the network becomes congestion and the total congestion window size of parallel TCP in slow-start will approximate the value of $B \times T + Q$. So we get:

$$N \times (2^{U_{SS}-1}) = min\{N \times SSTHRESH, B \times T + Q\}. \quad (6)$$

Then the number of rounds in slow-start phase can be expressed as follow:

$$U_{SS} = min\{1+log_2(\frac{B \times T + Q}{N}), 1+log_2(SSTHRESH)\}. \quad (7)$$

Let T_{SS} denote the period time during the slow-start phase. To simplify the analysis, we assume that the time length of each round in slow-start phase is equal to T. That is, $T_{SS} = U_{SS} \times T$. According to Eq. (7), we get that:

$$T_{SS} = T+min\{log_2(\frac{B \times T + Q}{N})^T, log_2(SSTHRESH)^T\}. \quad (8)$$

Finally, we have gotten Y_{SS} and T_{SS} to be part of our model. We next extend our model to include parallel TCP behaviors in the congestion avoidance phase.

3.2.2. Congestion Avoidance Phase

In congestion avoidance (CA) phase, the congestion window of each connection grows linearly. The increasing rate of parallel TCP's congestion window is N times faster than that of a single TCP connection. We specify the evolution of congestion window for parallel TCP in Fig. 2.

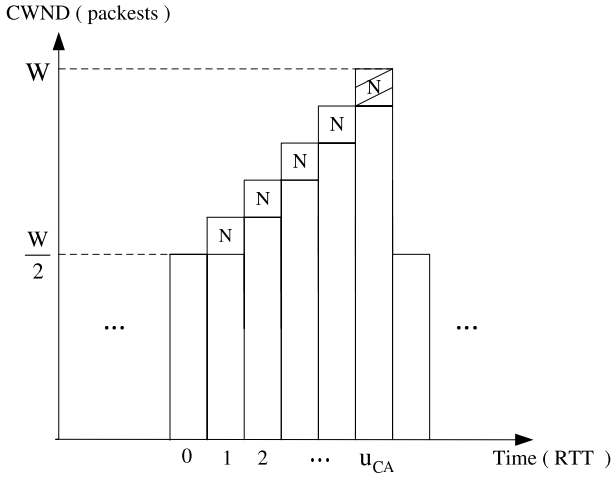


Figure 2. The evolution of the congestion window for parallel TCP in CA.

It is sure that all connections of parallel TCP may not perceive the loss event immediately. To simplify the analysis of the model, we assume that parallel TCP takes another round to perceive the loss. Let (U_{CA} + 1) be the number of rounds in CA (for 0^{th} to U_{CA}^{th} round). Consider the congestion window evolution of parallel TCP in each round. That is:

For the 0^{th} round , $CWND = \frac{B \times T + Q}{2}$.

For the 1^{th} round, $CWND = \frac{B\times T+Q}{2} + 1 \times N$.

...

For the $(U_{CA}-1)^{th}$ round,
$CWND = \frac{B\times T+Q}{2} + (U_{CA}-1) \times N = B \times T + Q$.

In our model, we assume the network pipe is filled with the packets in $(U_{CA}-1)^{th}$ round. So the congestion window in $(U_{CA}-1)^{th}$ round must be $B \times T + Q$. Thus, we can get that:

$$U_{CA} = \frac{B \times T + Q}{2N} + 1. \quad (9)$$

The total data sent in CA is the sum of congestion window in each round. It can be expressed as:

$$Y_{CA} = \frac{U_{CA}+1}{2} \times (B \times T + Q + N \times U_{CA}). \quad (10)$$

We consider the duration T_{CA} for parallel TCP in transit in CA. The duration T_{CA} consists of the following :

t_e : the time period that queue is empty.
t_q : the time period that queue is built up.
t_Q : the time period that queue is full.

During data transit in CA, when the congestion window size of parallel TCP grows from $\frac{B\times T+Q}{2}$ to $B \times T$, the bottleneck queue should be empty. So that the number of rounds during the duration that queue size is null is $\frac{B\times T-\frac{B\times T+Q}{2}}{N} + 1 = \frac{B\times T-Q}{2N} + 1$. And the length of each round is T. Considering that the queue may not be empty during the whole CA. In this case t_e may equal to zero. we get that:

$$t_e = max\{0, T \times (\frac{B \times T - Q}{2N} + 1)\}. \quad (11)$$

There are some packets accumulating in queue when the congestion window of parallel TCP is growing from $B \times T$ to $B \times T + Q$. Then the number of round during packets queuing in buffer is $\frac{B\times T+Q}{2N} - \frac{B\times T-Q}{2N} = \frac{Q}{N}$. The length of each round consists of T and the packet queuing time. That is:

$$t_q = \{(T + \frac{N}{B}) + (T + \frac{2N}{B}) + ... + (T + \frac{\frac{Q}{N} \times N}{B})\}.$$

By the equation given above, we have:

$$t_q = \frac{Q \times (2B \times T + N + Q)}{2NB}. \quad (12)$$

The buffer is full in the last round in CA, so that:

$$t_Q = \frac{B \times T + Q}{B}. \quad (13)$$

T_{CA} is gotten by add up t_e, t_q, and t_Q, from Eq. (11), Eq. (12), and Eq. (13), that is:

$$T_{CA} = max\{\frac{2BT(Q+N)+Q(3N+Q)}{2NB}, \frac{BT(BT+Q+4N)+Q(3N+Q)}{2NB}\}. \quad (14)$$

Now we have gotten Y_{SS} and T_{SS} above to be part of our model. We next extend our model to include parallel TCP behaviors in the fast retransmit and recovery phase.

3.2.3 Fast Retransmit and Recovery phase

We consider parallel TCP connections where all loss indications are due to V duplicate ACKs. When TCP perceives the loss event, it means that TCP source receive V duplicate ACKs. In this moment, the value of $SSTHRESH$ becomes half of the congestion window, and the value of congestion window is equal to $SSTHRESH$.

During the fast retransmit and recovery phase, the size of send window is increased by one packet every time an duplicate ACK is received. The process is continue until successfully retransmission and then go back to the congestion avoidance phase.

Our model is based on TCP Sack. Since TCP SACK option field in ACKs report noncontiguous blocks of data which have been received correctly. We assume that after retransmit, TCP costs just another single round waiting for successful retransmit. So the duration of the fast retransmit and recovery phase is that:

$$T_{FF} = \frac{B \times T + Q}{B}. \quad (15)$$

We also assume that each connection losses X number of packets in the end of CA. So the total packets sent in the fast retransmit and recovery phase is that:

$$Y_{FF} = (B \times T + Q + N) - VN - XN. \quad (16)$$

From Eq. (4), Eq. (5), Eq. (8), Eq. (10), Eq. (14), Eq. (15), and Eq. (16), we finally derive our model as following:

$$Throughput = \frac{Y_{SS} + m \times (Y_{CA} + Y_{FF})}{T_{SS} + m \times (T_{CA} + T_{FF})}. \quad (17)$$

For m is the number that the congestion avoidance phase and the fast retransmit and recovery phase during parallel TCP transit. It is related to the length of transfer time. If the total transmitting time is long enough, the influence of the slow-start phase can be negligible. So the model can be expressed as:

$$Throughput = \frac{Y_{CA} + Y_{FF}}{T_{CA} + T_{FF}}. \quad (18)$$

In next section, we prefer using Eq. (17) to do accurately performance analysis. The throughput predicted by our model will be compare with the one simulated by NS2.

4. Performance Evaluation

In this section, we present the performance evaluation in detail. We compare the results predicted by the proposed model against the simulated results used NS2 [19].

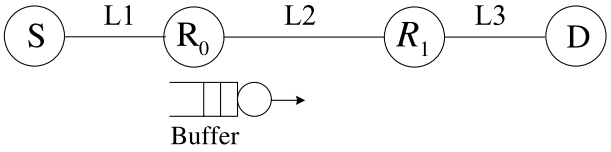


Figure 3. Topology for simulations.

4.1. Topology and Setting

We perform all the experiments using the dumbbell network topology shown in Fig. 3. We employ TCP SACK for all parallel TCP connections. Parallel TCP traffic goes from the source node S to the destination node D. R_0 and R_1 are drop-tail routers. Link L2 is the bottleneck link. Both L1 and L3 are access links between the end nodes and the routers. Each simulation time is 300 seconds. All parallel TCP connections start simultaneously in the beginning.

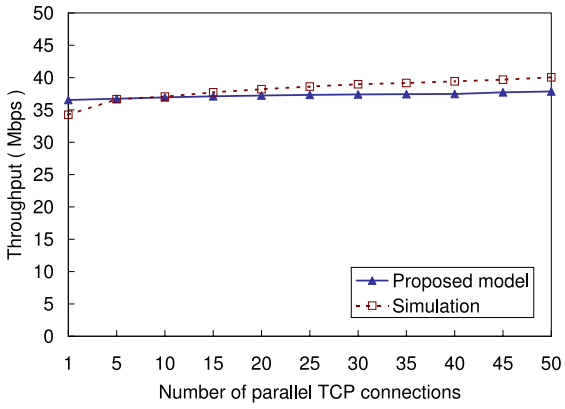
The setting of the parameters in our model is described as follows. Let the segment size equal to 1KB and $SSTHRESH$ equal to 32KB. We set X to be one, that is, there is one packet loss in the end of CA for each connection. We set V to be three, it means that the packet loss indication is based on the receipt of triple duplicate ACKs.

We perform the parallel TCP on these typical bottleneck links such as T1(B=1.54Mbps), T3(B=45Mbps), and the broadband network(B=100Mbps). Our model behaves similarly on each of them. Since the limitation of the space in the paper, we disclose just T3 case of the experimental results in detail as follows.

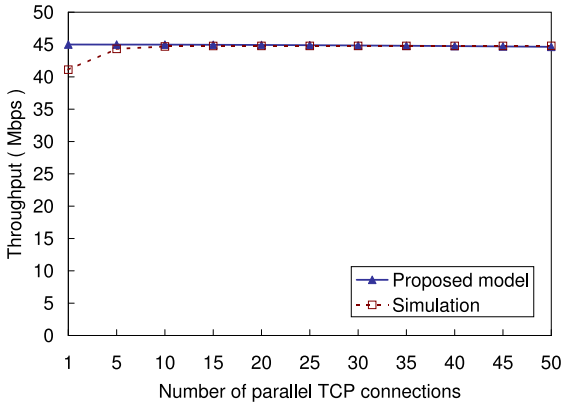
4.2. Simulations on different buffer sizes

This subsection presents the results of our model and that of NS2 simulation on different buffer sizes. In order to evaluate the impact on the buffer size, we measure the performance under two different circumstances. One is the buffer size is enough (equals to BDP). The other is the buffer is limited. We present the experimental results with a varying number of parallel TCP connections. Figure 4 plots the aggregated throughput of the model and that of NS2 simulation as a function of the number of parallel TCP connections. The details of them are described as follows.

Figure 4(a) is the results which is on T3 bottleneck link with limited buffer size 50KB. The bandwidth of L1 and L3 are both 100Mbps. RTT is set to 0.1s. The difference ratios in the throughput between the model and the simulation



(a) Q=50KB



(b) Q=BDP

Figure 4. Achieved throughput by the model and NS2 simulation on B=45Mbps and RTT=0.1s with different buffer sizes.

are in the range of 0% to 5.8%. From the graph, it is easy to see that the available bandwidth is not fully used. The bandwidth utilization in Fig. 4(a) is less than 88%. Especially with N equal to 1, the utilization of simulated results only equal to 76%. From the result we can find that parallel TCP can be more efficiently to utilize the bandwidth than the single TCP connection. The utilization of the bandwidth is raised while N is increased.

Figure 4(b) is the results which is on T3 bottleneck link with the enough buffer size that equals to BDP. The bandwidth of L1 and L3 are both 100Mbps and RTT is set to 0.1s. In comparison with that of Fig. 4(a), the results of the proposed model in Fig. 4(b) are much similar to the simulated ones. Except the result with N equal to 1, the difference ratio in the throughput between the model and the simulation is less than 1.5%. It is easy to see that our model performs well on the circumstance with enough buffer size.

Both in Fig. 4(a) and Fig. 4(b) with N equal to 1, it shows that the model predicts the throughput with a certain difference. We attribute the difference to the reason as

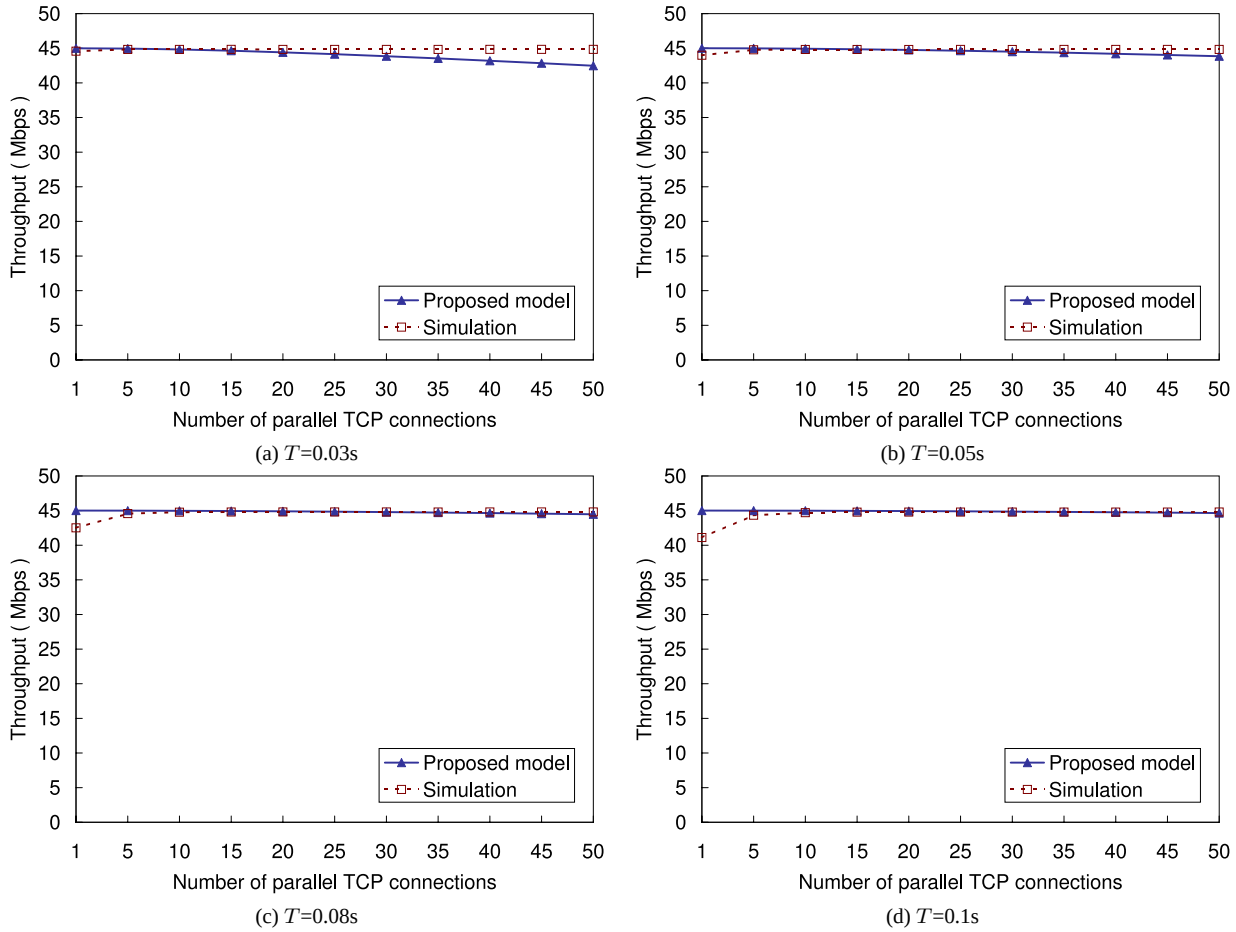


Figure 5. Achieved throughput by the model and NS2 simulation on B=45Mbps and Q=BDP with different RTTs.

follows. When the number of parallel TCP connection is small, the congestion window of each connection becomes large. Thus, the number of the lost packets within a lost event may be huge. It may be hard for a TCP source to retransmit them successfully in a single round. However, we assume in the model that all the lost packets retransmitted in a single round to simplify the analysis.

4.3. Simulations on different RTTs

This subsection depicts the results of our model and that of NS2 simulation in different end-to-end round-trip delay (RTT). In order to evaluate the influence on RTT, we measure the performance under four different length of RTTs: 0.03s, 0.05s, 0.08s, and 0.1s. We present these experimental results with a varying number of parallel TCP connections. Figure 5 plots the aggregated throughput of the model and that of NS2 simulation as a function of N.

Figure 5(a) is the results which is on T3 bottleneck link with RTT equal to 30ms. The bandwidth of L1 and L3 are both 100Mbps. The maximum percentage of the difference between the model and the simulation is 5%. From the graph, it is easy to see that with N growing, a small difference is performed between the results of the model and that of the simulation. It is due to the violation of the assumption in our model that all connections suffer the loss event in a round. Actually, not all connections of the parallel TCP perceive the loss in the same round. Some connections may not experience the loss event since other connections have reacted and cut the congestion window in half.

Figure 5(b) and Fig. 5(c) are the results which are both on T3 bottleneck link with the different RTT equal to 50ms and 80ms respectively. The bandwidth of L1 and L3 are both 100Mbps. In comparison with Fig. 5(a), the results of our model in Fig. 5(b) and Fig. 5(c) are much similar to that of the simulation. It represents the trend that our model predicts the behavior of parallel TCP accurately when RTT becomes large.

The results with RTT equal to 100ms is shown in Fig. 5(d). Except the prediction on N to be 1, the predicted throughput is almost equal to the result of the simulation. The maximum percentage of the difference between them is 1%. The appearance of the small difference with N equal to 1 has described above in subsection 4.2.

Overall, our model can be used to predict the parallel TCP throughput. As a result, this model is quiet suited in the circumstances with large buffer size and long RTT. It is significant since parallel TCP performs well in the higher BDP network. Besides, no need to the packet loss rate makes our model toward practical.

5. Conclusions

We have shown a throughput prediction model for parallel TCP in this paper. The model captures the essence of parallel TCP's behavior in slow-start, congestion avoidance, and the fast retransmit and recovery phase. It expresses throughput as a function of the feasible network parameters, such as the bandwidth, end-to-end round-trip delay, buffer size, and the number of connections that parallel TCP used. Packet loss rate is not needed in our model because of its hard obtain for a TCP source on the Internet. We have compared our model with the simulated results used NS2. Our model provides the throughput prediction and try to match the observed behavior in most case of the scenario. It is fairly suitable in the case of high BDP network which parallel TCP is much significant on it.

Our future work remain as follows. We plan to investigate the maximal parallelisation of parallel TCP. A few of researches start to focus the issue about parallelisation. We intend to study the maximum parallelisation for the parallel TCP to compute the proper amount of connections, to fully utilize the bandwidth of a transmission path. Furthermore, we intend to design a mechanism to retrieve the network parameters that are needed in our throughput model. By the aid from the routers along the transmission path, a parallel TCP sender may acquire the necessary parameters and then compute the proper amount of connections that can be set up between a traffic pair. The mechanism will realize the maximum parallelisation theory of the parallel TCP.

References

[1] J. Postel, "Transmission Control Protocol," *IETF RFC 793*, September 1981.

[2] M. Fomenkov, K. Keys, D. Moore, and K. Claffy, "Longitudinal study of internet traffic in 1998-2003," *WISICT*, pp. 1–6, January 2004.

[3] J. Olsen, "Stochastic modeling and simulation of the TCP protocol," Ph. D. Thesis, Department of Mathematics and Computer Science at Uppsala University, October 2003.

[4] Y. C. Chan, C. L. Lin, and C. Y. Ho, "Quick Vegas: Improving Performance of TCP Vegas for High Bandwidth-Delay Product Networks," *IEICE Transactions on Communications*, vol E91-B, no. 4,pp. 987-997, April 2008.

[5] C. Jin, D. X. Wei, and S. H. Low, "FAST TCP: Motivation, Architecture, Algorithms, Performance," *IEEE INFOCOM*, pp. 2490–2501, March 2004.

[6] S. Floyd, "HighSpeed TCP for Large Congestion Windows," *IETF RFC 3649*, December 2003.

[7] L. Xu, K. Harfoush, and I. Rhee, "Binary increase congestion control for fast long-distance network," *IEEE INFOCOM*, pp. 2514–2524, March 2004.

[8] D. J. Leith and R. Shorten, "H-TCP: TCP for high-speed long distance networks," *PFLDNet Workshop*, pp. 1–16, December 2004.

[9] T. Kell, "Scalable TCP: Improving Performance in Highspeed Wide Area Networks," *ACM CCR*, vol. 32, no. 2, pp. 83–91, April, 2003.

[10] S. Bhandarkar, S. Jain, and A. L. N. Reddy, "Improving TCP Performance in High Bandwidth High RTT Links Using Layered Congestion Control," *PFLDNet Workshop*, pp. 1–8, February 2005.

[11] S. Cho, "Congestion control schemes for single and parallel TCP flows in high bandwidth-delay product networks," Ph. D. Thesis, Department of Computer Science at Texas A&M University, May 2006.

[12] S. Cho and R. Bettati, "Aggregated Aggressiveness Control on Groups of TCP Flows," *IEEE/ACM Trans. NETWORKING 2005*, vol. 3462, pp. 586–597, May 2005.

[13] S. Cho and R. Bettati, "Collaborative Congestion Control in parallel TCP Flows," *IEEE ICC*, pp. 1026–1030, May 2005.

[14] S. Cho and R. Bettati, "Adaptive Aggregated Aggressiveness Control on Parallel TCP Flows Using Competition Detection," *IEEE ICCCN*, pp. 237–244, October 2006.

[15] E. Altman, K. Avrachenkov, and C. Barakat, "A stochastic model of TCP/IP with stationary random losses," *ACM Computer Communication Review*, vol. 30, no. 4, pp. 231-242, October 2000.

[16] M. Mellia, I. Stoica, and H. Zhang, "TCP model for short lived flows," *IEEE Communications Letters*, vol. 6, no. 2, pp. 85–87, February 2002.

[17] T. J. Hacker, B. D. Athey, and B. Noble, "The end-to-end performance effects of parallel TCP sockets on a lossy wide-area network," *16th IEEECS/ACM International Parallel and Distributed Processing Symposium*, pp. 1-16, April 2002.

[18] J. Wu and H. El-Ocla, "TCP congestion avoidance model with congestive loss," *IEEE ICON2004*, pp. 3–8, November 2004.

[19] UCB/LBNL/VINT, The Network Simulator (ns2), http://www.isi.edu/nsnam/ns/.