



This article was originally published in a journal published by Elsevier, and the attached copy is provided by Elsevier for the author's benefit and for the benefit of the author's institution, for non-commercial research and educational use including without limitation use in instruction at your institution, sending it to specific colleagues that you know, and providing a copy to your institution's administrator.

All other uses, reproduction and distribution, including without limitation commercial reprints, selling or licensing copies or access, or posting on open internet sites, your personal or institution's website or repository, are prohibited. For exceptions, permission may be sought for such use through Elsevier's permissions site at:

<http://www.elsevier.com/locate/permissionusematerial>



Prediction of short-lived TCP transfer latency on bandwidth asymmetric links[☆]

Guang Tan^{*}, Stephen A. Jarvis

Department of Computer Science, University of Warwick, Coventry, CV4 7AL, United Kingdom

Received 27 February 2005; received in revised form 31 July 2005

Available online 9 March 2006

Abstract

TCP over bandwidth asymmetric networks such as *Cable TV* and *Asymmetric Digital Subscriber Loop* (ADSL) exhibits different characteristics from TCP on symmetric links. A number of techniques have been proposed to address this problem. However, previous research has been largely focused on bulk transfers. This paper investigates the effects of bandwidth asymmetry on Web-like short-lived transfers. Two prediction models, one given in a closed form and the other in a recursive form, are presented for TCP transfers without and with *ACK Filtering* (AF), a representative optimizing technique for TCP transfers over bandwidth asymmetric links. *Ns-2* based experiments show that these models can predict TCP transfer latency with a high degree of accuracy. © 2006 Elsevier Inc. All rights reserved.

Keywords: Prediction; TCP transfer; Short-lived; Bandwidth asymmetry

1. Introduction

Most Internet applications are built on TCP (Transmission Control Protocol), an underlying transport-layer protocol providing reliable data transfers. To ensure reliability, TCP uses feedback in the form of acknowledgments from the receiver. In addition, TCP is *ACK-clocked*, relying on the timely arrival of acknowledgments to maintain progress and fully utilize the available bandwidth of the path. Usually the acknowledgments are small enough so that their transmission time is negligible compared to the transmission time of data. The implicit assumption is that the bandwidth over two directions are comparable. However, on today's Internet with a great diversity of network accessing technologies, this is not always the case. Emerging technologies such as Cable TV networks and *Asymmetric Digital Subscriber Loop* (ADSL) over dial-up telephone lines are able to provide network accessing services using different medias over a variety of links. In these networks, the bandwidth in the forward direction from the server to the user (also referred to as *downstream direction*) is often orders of magnitude larger than that in the reverse direction (*upstream direction*). This results in significant queuing delay at a transmission point for the ACKs from the receiver.

[☆] This research is sponsored in part by grants from the NASA AMES Research Center (administrated by USARDSG, contract No. N68171-01-C-9012), the EPSRC (contract No. GR/R47424/01) and the EPSRC e-Science Core Programme (contract No. GR/S03058/01).

^{*} Corresponding author.

E-mail addresses: gatan@dcs.warwick.ac.uk (G. Tan), saj@dcs.warwick.ac.uk (S.A. Jarvis).

Since the TCP sender expects ACKs from the receiver to advance its sliding window, the sending process will be slowed and the throughput is thus reduced.

To address this issue, a number of techniques have been proposed to alleviate ACK congestion. One notable technique is *ACK Filtering* (AF) [9]. For one connection, it reduces the ACK traffic by replacing earlier ACKs with the latest one using the cumulative property of ACK, that is, an ACK with a certain sequence number implicitly acknowledges those packets with smaller sequence numbers. This method extends the tolerable degree of network asymmetry¹ to a much higher value.

This paper provides an insight into the interactions between TCP data flow and ACK packets over the bottleneck links. Two models, the *ATCP Model* and the *ATCP-AF Model*, are presented for asymmetric TCP transfers without and with the AF technique, respectively. The former model gives a closed form expression for the predicted transfer time for a given amount of data, while the later gives this in a recursive form. Since we focus on the short-lived transfer, we do not consider the packet loss.

Ns-2 [3] based experiments have been conducted to validate the models. Both models prove to correctly capture the characteristics of asymmetric TCP and provide accurate predictions as compared with the simulation results. The next section gives an overview of related work; Sections 3 and 4 introduce the models of asymmetric TCP and asymmetric TCP with AF, respectively, and Section 5 concludes the paper.

2. Related work

Network performance modeling is a significant research topic. TCP models are usually classified based on the transfer length they focus on. In [5,7], models for long-lived transfers are discussed. They characterize the steady state throughput of bulk transfer as a function of several key parameters including maximum window size, round trip time, retransmission timeout and average loss rate. Cardwell et al. [1] extend the model in [7] to include connection establishment and slow start phases, thus giving a transfer latency prediction for an arbitrary data length. For short-lived transfers, Mellia et al. [6] compute average latency by exhaustively enumerating all loss cases. However, their model only handles transfers of a few segments, because the complexity grows exponentially. The data loss conditions and TCP congestion avoidance algorithms are central to these models, which share a common assumption that the network links are symmetric. Hasegawa et al. [2] discuss the appropriate combination of HTTP and TCP protocols on asymmetric networks. Their emphasis is on the interactions between existing protocols and their variants including HTTP 1.0/1.1, TCP Tahoe/Vegas, etc.

To examine the network conditions affected by different asymmetry factors, the research work presented in [4] suggests approaches to alleviate the upstream congestion. The technique proposed is to give the ACKs a fair chance of being transmitted without being delayed by many large data packets.

ACK Filtering [9] (also known as ACK suppression) is a technique performed at the entry point of the bottleneck link. It allows late coming ACKs to replace earlier ones of the same TCP flow using the cumulative property of ACK, that is, an ACK with a certain sequence number implicitly acknowledges those packets with smaller sequence numbers. This method can reduce the number of ACKs transmitted by choosing only the latest one. AF should only be applied to pure ACKs that contain no other flags such as SYN, RST, URG, and FIN. In addition, it should avoid deleting a series of 3 duplicate ACKs that indicate the need for Fast Retransmission [RFC2581] or ACKs with the Selective ACK option (SACK) [RFC2018]. AF is most effective for long-lived bulk transfers.

3. Modeling asymmetric TCP

3.1. Assumptions and notation

The model in this paper accounts only for the delays arising from TCP's performance and ignores factors like application processing latency or specific scheduling strategy. The receiver uses the delayed acknowledgment scheme specified in RFC2581.

¹ This is usually referred to as "normalized asymmetry," which is defined in [4] as the ratio of the raw bandwidth divided by the ratio of the packet sizes used in the two directions.

Table 1
Notation and variable definitions used in the models

Notation	Definition
W_r	Maximum size of the receiver's window
B_f	Bandwidth of forward link
B_r	Bandwidth of reverse bottleneck link
D_f	Delay of forward link
D_r	Delay of reverse link
D	$D_f + D_r$
MSS	Maximum segment size of data packet
S_d	Size of full data packet
S_a	Size of acknowledgement packet
T_d	Transmission time for a full-sized packet over the forward link
T_a	Transmission time for an ACK packet over the bottleneck link
D_{ack}	Delayed ACK interval

Our models are focused on short-term transfers and hence do not consider packet losses. It has been shown that packet losses are a very common phenomenon in practise and can have significant impact on TCP performance [4], especially for long-lived bulk transfers. However, TCP transfers such as static Web page retrieval spend a large fraction of their lifetime in the startup period, which is often short enough to see few, if any, losses and consequently their performance is dominated by startup effects.

Supporting notation and variable definitions used in the discussion are described in Table 1. Note that W_r will not change during the transfer since the application processing delay is ignored. Also, because the congestion avoidance mechanism of TCP is not considered here, W_r determines the maximum size of a sender's window (W_m).

3.2. The ATCP Model

The goal of this section is to establish a model to predict the transfer time T for a given amount of data (in terms of packets) observed at the receiver side. A typical TCP transfer usually experiences 3 phases: the Connection Establishment (CE) Phase, the Slow Start (SS) Phase and the Steady State Phase. In what follows, the model is divided into 3 parts and each part is discussed in turn.

3.2.1. The Connection Establishment (CE) Phase

Assuming the connection is initiated from the data sender, then from its viewpoint, the connection is established after it sends the SYN packet and receives the SYN + ACK packet. The CE latency is denoted by $T(0)$, and

$$T(0) = D + T_a. \quad (1)$$

Note that the transmission time of an ACK packet is accounted for in Eq. (1), which differentiates it from traditional models. In asymmetric networks with limited reverse channel bandwidth, the transmission time is no longer a trivial factor. For example, on a 9.6 Kbps dial-up line, the transmission of a 40 bytes ACK packet would take 33.3 ms, some 1000 times as long as that needed on a 10 Mbps link. This feature makes the RTT value largely dependent on the number of queuing ACKs (and how they are queuing) at the entry point of the slow channel, and thus complicates the transfer models as compared with those of symmetric TCP.

3.2.2. The Slow Start (SS) Phase

During the slow start phase, the sender increases its congestion window by one on every ACK it receives. Since the receiver sends ACKs every two (full) data packets (except for the first packet, for which the ACK would be sent after the delayed acknowledgement timer expires), the sender will send 3 packets back-to-back in response to any ACK that acknowledges two packets: 1 for the increased $cwnd$ and 2 for refilling the old window. Fig. 1 illustrates the time line of packets observed at the receiver side. A *block* is defined as a series of packets sent back-to-back from the sender. As shown in Fig. 1, the first block is the first packet; the second block consists of 2 packets: 2 and 3; and from the 3rd block until the end of the slow start phase, each block consists of 3 packets. Meanwhile the $cwnd$ continues to increase until it reaches W_r , the maximum receiver window. At this threshold, the receiver should have received

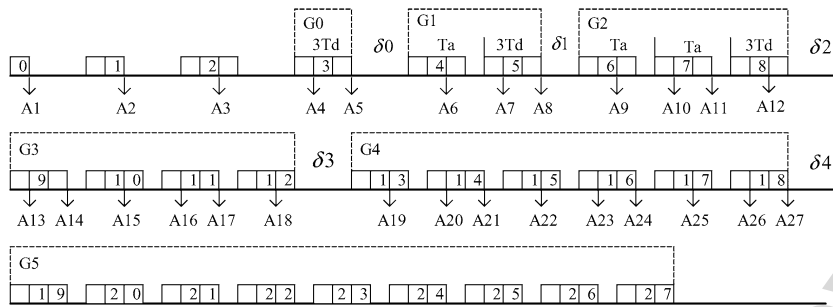


Fig. 1. Packet time line during the slow start phase at the receiver side.

$3(W_r - 1)$ packets. After the SS phase, each ACK received at the sender will cause 2 new data packets to be sent, therefore the block size will be constantly 2.

It can be observed that the ACKs of block 3, A_4 and A_5 , introduce blocks 4 and 5, whose ACKs, A_6 , A_7 and A_8 , introduce blocks 9, 10 and 11, ... and so on. A *group* of packets is therefore recursively defined as the set of data packets sent in response to all the ACKs caused by the previous group's data packets, with the initial group, G_0 , consisting of block 3. For the k th ($k \geq 1$) group, let B_k be the beginning of its first block, and E_k be the end of its last block. The *group length*, denoted by L_k , is thus equal to $E_k - B_k$. For example, in Fig. 1, L_3 covers the time period between the front edge of block 9 and the end of block 12. There is an interval between two consecutive groups, G_k and G_{k+1} , which is denoted by δ_k . Also let N_k^p , N_k^a and N_k^b be the number of data packets, number of ACKs and number of blocks of group k , respectively.

During the SS phase, the transfer over the reverse link proceeds with one ACK per T_a , and over the forward link, one block per $3T_d$ corresponding to each ACK packet. If $T_a \leq 3T_d$, the forward transfer will lag behind the ACK transfer and thus forms a bottleneck for the whole data stream. Since the focus is on the case where data transfer slowdown is originated from the reverse link, $T_a > 3T_d$ is assumed. This has two implications. First, the spacing of data blocks is at least T_a , because the data sending on the forward direction is "clocked" by the slower transferring of ACKs; second, if the interval between two blocks is T_a , the spacing of all the blocks generated by these two blocks' ACKs is also T_a . With the definition of group, this means that all blocks within one group are spaced by exactly T_a .

Here the aim is to obtain the transfer time, $T(M)$, for an arbitrary data packet P_M . Let $T'(M)$ be the arrival time of the first packet in the same block as P_M . Since $T(M)$ can be easily computed from $T'(M)$ with the offset inside the block, it is possible to use $T'(M)$ to approximate $T(M)$ (the difference is non-cumulative and bounded by $2T_d$). In the remainder of this paper, these two terms will be used interchangeably if the difference is not explicitly stated. After the CE phase, P_1 takes $D_f + T_d$ time to reach the receiver, therefore $T(1) = D + D_f + T_a$. Upon receipt of P_1 , the receiver should send back an ACK, which, however, maybe delayed until the delayed acknowledgement timer expires. Hence $T'(1) = \max\{T(0), D_{ack}\} + D + T_a$, and from Fig. 1, it follows that $T'(2) = \max\{T(0), D_{ack}\} + 2(D + T_a)$ and $T'(3) = \max\{T(0), D_{ack}\} + 3(D + T_a)$. Note that only the first delayed ACK is considered and its effect in later transfer is omitted. Now the task is to compute $T(M)$ for $M > 9$. This is done by first determining the group $G(M)$ that P_M ($M > 9$) belongs to, and then computing all the group intervals δ_k . The $T(M)$ is then given by adding its group offset to the beginning time of this group.

Theorem 1. Let $G(M)$ be the group that packet M ($M > 9$) belongs to, then

$$G(M) \geq \frac{\ln(M + 3) - 2 \ln 3}{\ln 3 - \ln 2}. \tag{2}$$

Proof. Since the receiver sends an ACK every 2 data packets, the number of ACKs issued in response to the packets of group $k - 1$ ($k \geq 1$), $N_{k-1}^a = \lfloor (3/2)N_{k-1}^b \rfloor$; furthermore, each of these ACKs corresponds with a block in G_k , so $N_k^b = N_{k-1}^a = \lfloor (3/2)N_{k-1}^b \rfloor$. It follows that $N_k^b \leq (3/2)N_{k-1}^b \leq (3/2)^2 N_{k-2}^b \leq \dots \leq (3/2)^{k-1} N_1^b = 2(3/2)^{k-1}$. Assume packet M belongs to group x ($x \geq 1$), then $M \leq 9 + 3 \sum_{k=1}^x N_k^b$, which can be rewritten as $M \leq 9 + 3 \sum_{k=1}^x 2(3/2)^{k-1}$, it can therefore be shown that $x \geq \lceil [\ln(M + 3) - 2 \ln 3] / (\ln 3 - \ln 2) \rceil$. \square

By this theorem, one packet's group can be approximated as

$$G(M) \approx \left\lceil \frac{\ln(M+3) - 2 \ln 3}{\ln 3 - \ln 2} \right\rceil. \quad (3)$$

When an ACK is sent out, it may be queued at the bottleneck link until previous ACKs are removed, so the actual transmitting time may be later than expected. To simplify the discussion, it is assumed that there is no queuing for ACKs, which can be done by counting the queuing time to the endpoints, and the actual *transmitting time*, $T'(A)$ and the *producing time* $T(A)$ are differentiated. Let A_k^f and A_k^l denote the first and last ACK of group k , respectively; a group G_k is said to be *Ack-aligned* if $T'(A_k^f) = T(A_k^f)$. This implies $T(A_k^f) - T'(A_{k-1}^l) \geq T_a$. Additionally, if group G_k is Ack-aligned, then $T'(A_k^f) = T(A_k^f) = B_k + \alpha_k T_d$, where α_k is 1 or 2. The A_k^f takes $D_r + T_a$ to reach the sender; the sender then sends out a block, which arrives at the receiver as the first block of G_{k+1} after D_f . Thus $T(A_k^f) + D + T_a = B_{k+1}$, and further

$$B_{k+1} - B_k = D + T_a + (\alpha_{k+1} - \alpha_k)T_d \approx D + T_a. \quad (4)$$

If $T(A_k^f) - T'(A_{k-1}^l) < T_a$, the A_k^f will be deferred by some time before A_{k-1}^l is cleared at the bottleneck link, which results in $T(A_k^f) < T'(A_k^f)$. In this case G_k is said to be *Ack-skewed*.

Theorem 2. *If there exists an Ack-skewed group and let the first such group be G_s ($s > 2$), then for any $k \geq s$, G_k is Ack-skewed.*

Proof. Since G_s is the first Ack-skewed group, G_{s-1} must be Ack-aligned, so $T'(A_s^f) - T'(A_{s-1}^l) = T_a$, which means $B_{s+1} - E_s = T_a - 3T_d$. Because $T'(A_s^f) = T(A_s^f) + \lfloor (3/2)N_s^b \rfloor T_a = B_s + \alpha_s T_d + N_s^b T_a + \lfloor (1/2)N_s^b \rfloor T_a > B_s + L_s = E_s$, $T(A_{s+1}^f) - T'(A_s^l) < B_{s+1} + 3T_d - T'(A_s^l) < T_a$, therefore G_{s+1} is Ack-skewed. Further, since $T'(A_{s+1}^f) - T'(A_s^l) = T_a$, by the same deduction, it can be proved that G_{s+2} is also Ack-skewed. It follows that for all $i > 0$, G_{s+i} is Ack-skewed. This establishes the theorem. \square

By Theorem 2, there are three cases for any consecutive G_k and G_{k+1} :

- (1) If both are Ack-skewed, then A_{k+1}^f will reach the receiver exactly T_a after A_k^l , which results in a time interval of T_a between the two blocks caused by them. Since these two blocks are the last block of G_{k+1} and the first block of G_{k+2} respectively, the interval between G_{k+1} and G_{k+2} , δ_{k+1} , is thus equal to $T_a - 3T_d$.
- (2) If G_k is Ack-aligned and G_{k+1} is Ack-skewed, then like the previous case, $\delta_{k+1} = T_a - 3T_d$.
- (3) If both are Ack-aligned, then by Eq. (4), $L_k + \delta_k = D + T_a$ and $L_{k+1} + \delta_{k+1} = D + T_a$, so $\delta_{k+1} = \delta_k - (L_{k+1} - L_k) \approx \delta_k - (L_{k+1} - L_k)$. Moreover, $\delta_{k+1} = \delta_k - (N_{k+1}^b - N_k^b)T_a = \delta_k - (\lfloor 3N_k^b/2 \rfloor - N_k^b)T_a$. Because $3N_k^b/2 - 1 < \lfloor 3N_k^b/2 \rfloor \leq 3N_k^b/2$, it is the case that $\delta_k - N_k^b T_a/2 < \delta_{k+1} \leq \delta_k - (N_k^b - 1)T_a/2$. δ_{k+1} can be approximated using the average of these two bounds, so $\delta_{k+1} \approx \delta_k - (N_k^b - 1)T_a/2$. Since the δ must be no less than $T_a - 3T_d$, $\delta_{k+1} \approx \max\{\delta_k - (N_k^b - 1)T_a/2, T_a - 3T_d\}$.

Therefore, $\delta_k \approx \max\{\delta_k - [(3/2)^{k-1} - 1/2]T_a, T_a - 3T_d\}$. As it is known that $\delta_1 \approx \max\{D, T_a - 3T_d\}$, resolving the recurrence gives

$$\delta_k \approx \max \left\{ D - \left[2 \left(\frac{3}{2} \right)^{k-1} - \frac{k}{2} - 1 \right] T_a, T_a - 3T_d \right\}, \quad k \geq 2. \quad (5)$$

By Theorem 1 and Eq. (3), an arbitrary packet P_M ($M > 9$) can be associated with a specific group; further, by Eq. (5) the beginning time of that group can be obtained. Adding the group offset of this packet to the group beginning time therefore gives the arrival time of this packet. Since Eq. (5) only holds for group $k \geq 2$, it is assumed that $M > 15$ (for $9 < M \leq 15$ the $T(M)$ can be computed from Fig. 1), and $x = G(M)$, then

$$T(M) = B_x + \left(\frac{M-9}{3} - \sum_{k=2}^{x-1} N_k^b \right) T_a = B_2 + \sum_{k=2}^{x-1} (L_k + \delta_k) + \left(\frac{M-9}{3} - \sum_{k=2}^{x-1} N_k^b \right) T_a$$

$$\begin{aligned}
&= B_2 + \sum_{k=2}^{x-1} \delta_k + \left[\frac{M-15}{3} - (x-2) \right] T_a + 3(x-2)T_d \\
&= T(15) + \delta_1 + \sum_{k=2}^{x-1} \delta_k + \left[\frac{M-15}{3} - (x-2) \right] T_a + 3(x-2)T_d.
\end{aligned} \tag{6}$$

From Fig. 1 it is straightforward to compute $T(15) = \max\{D + D_f + T_a, D_{\text{ack}}\} + 4D + 5T_a$, and substituting δ_k using Eq. (5) gives

$$\begin{aligned}
T(M) &= \max\{D + D_f + T_a, D_{\text{ack}}\} + \sum_{k=1}^{x-1} \max\left\{ D - \left[2\left(\frac{3}{2}\right)^{k-1} - \frac{k}{2} - 1 \right] T_a, T_a - 3T_d \right\} \\
&\quad + 4D + \left(\frac{M}{3} - x + 2 \right) T_a + 3(x-2)T_d.
\end{aligned} \tag{7}$$

3.2.3. The steady state

At a steady state, each ACK packet received by the sender will cause 2 packets to be sent, and the spacing of ACK is constantly T_a , so for $M > 3(W_r - 1)$,

$$T(M) = T(3W_r - 3) + \frac{M - 3W_r + 3}{2} T_a, \tag{8}$$

where $T(3W_r - 3)$ can be computed using Eq. (7).

Up to now, the arrival time of packet P_M ($M > 15$) during the transfer has been determined by Eqs. (7) and (8). Furthermore, the cases for $M \leq 15$ can also be computed by analyzing Fig. 1. Let $\varepsilon = \max\{D + D_f + T_a, D_{\text{ack}}\}$ and $x = \lceil \frac{\ln(M+3) - 2\ln 3}{\ln 3 - \ln 2} \rceil$, then the equations can be summarized and reorganized as follows:

$$T(M) = \begin{cases} D + D_f + T_a, & \text{if } M = 1; \\ \varepsilon + \lceil \frac{M}{3} \rceil (D + T_a), & \text{if } 1 < M \leq 15; \\ \varepsilon + \sum_{k=1}^{x-1} \max\{D - [2(\frac{3}{2})^{k-1} - \frac{k}{2} - 1]T_a, T_a\} \\ \quad + 4D + (\frac{M}{3} - x + 2)T_a + 3(x-2)T_d, & \text{if } 15 < M \leq 3W_r - 3; \\ T(3W_r - 3) + \frac{M - 3W_r + 3}{2} T_a, & \text{if } M \geq 3W_r - 3. \end{cases} \tag{9}$$

Note that $T(3W_r - 3)$ is used only for brevity, in fact, the result is a close-form expression.

3.3. Validating the ATCP Model

The *ns-2* simulator [3] has been used to validate this model. Fig. 2 shows the network topology used in the experiments. A data sender at node 0 connects to the receiver at node 3 via FullTCP [3], and then sends a certain amount of data. Node 1 and node 2 serve as intermediate routing devices, and the link between node 1 and 2 is asymmetric, with the forward bandwidth set to 10 Mbps and the reverse 56 K/28.8 K/9.6 Kbps. A selection of parameters associated with TCP and the network topology are listed in Table 2.

The transfer time vs. number of packet of the slow start phase for both simulation and model prediction is shown in Fig. 3(a), and Fig. 3(b) depicts the results of transfers in a longer time interval, including both the slow start and the steady state phases. Experiments have also been conducted by varying other parameters, and the comparative results are very similar. As can be seen, the ATCP Model tracks very accurately the simulation results, which indicates that the ATCP Model correctly captures the transfer characteristics of TCP over a lossless asymmetric network.

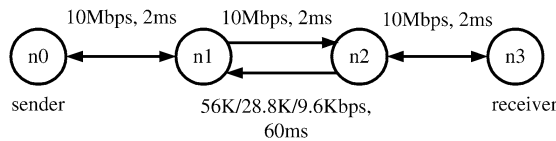


Fig. 2. Network topology for the evaluation of the ATCP Model.

Table 2
Parameters and values used in the simulation and model

Parameter	Value	Parameter	Value
W_r	40	B_f	10 Mbps
B_r	56/28.8/9.6 Kbps	D_f	6 ms
D_r	64 ms	S_d	1500 bytes
S_a	40 bytes	T_d	$(1500 * 8) / B_f$
T_a	$(40 * 8) / B_r$	D_{ack}	200 ms

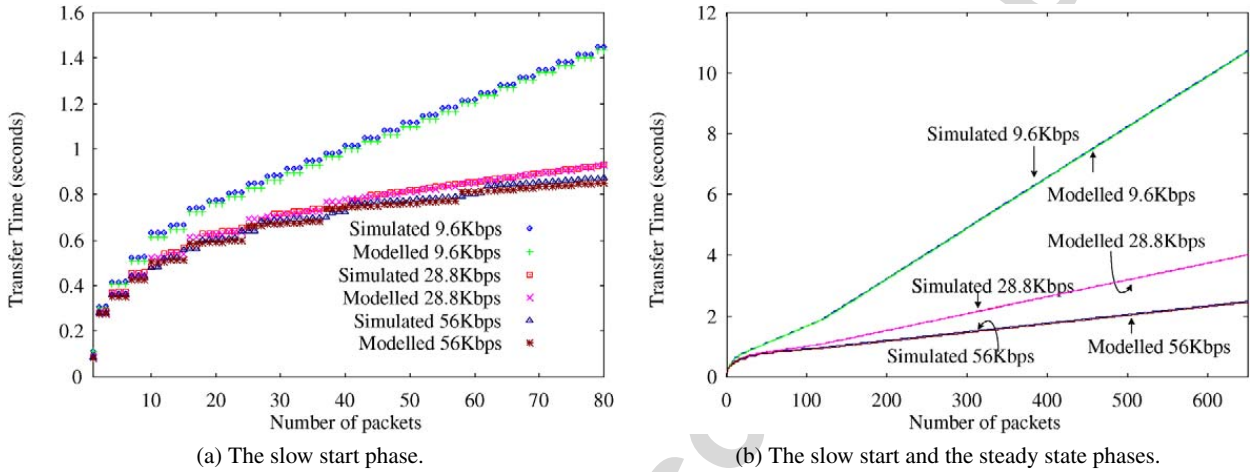


Fig. 3. Comparison between the simulated and the modeled results.

Data analysis shows that the relative error is less than 3% for more than 95% of all data points. Discrepancies between the simulated and modeled data stem from three factors. The approximations in Eqs. (3) and (5), which help achieve close-form expressions but at the expense of precision, cause the most notable deviation from the actual values; the omitted delayed acknowledgement effect also brings some small “disturbance” to the expected data interaction. Finally, averaging the transfer time over all packets within a block generates a minor difference. Despite these, the model proves to correctly reflect the asymmetric TCP behavior and can estimate the transfer latency at a high level of confidence.

4. Modeling asymmetric TCP with ACK Filtering

The ATCP Model is extended to the ATCP-AF Model by taking into account the effects of the AF technique. The ACK reconstruction [8] technique is assumed to be used in conjunction with AF to expedite the congestion window opening during the slow start phase. With this in use, the ACK stream through the bottleneck link will be changed only in its spacing of ACKs, and the concept of group and the interaction of one-ACK-three-packets is the same as in the ATCP Model. The same assumptions, notation and definitions as used in the ATCP Model are also used in this model unless otherwise stated. Fig. 4 gives an example of the packet time line for TCP transfer with AF. The connection setup and several initial packets are omitted in Fig. 4 since the initial process is also the same as that found in the in ATCP-AF Model. In the following the discussion is limited to the transfer during the SS phase and steady state phase.

On the basis of the ATCP Model, AF introduces a dynamic to the ACK behaviors in the ATCP-AF Model. The ACK entering the bottleneck link need not wait for previous ones to be transmitted, instead it removes all ACKs (of the same connection) queuing before it. When this ACK leaves the bottleneck link, the “filtered” ACKs are regenerated and forwarded to the sender back-to-back. These ACKs in turn trigger a series of back-to-back blocks, as shown in Fig. 4. A collection of consecutive blocks is defined here as a *chunk*, and the *chunk size* is defined as the number of blocks it contains. For a group k , let N_k^c be the number of chunks it contains, and its average chunk size $S_k^c = N_k^b / N_k^c$.

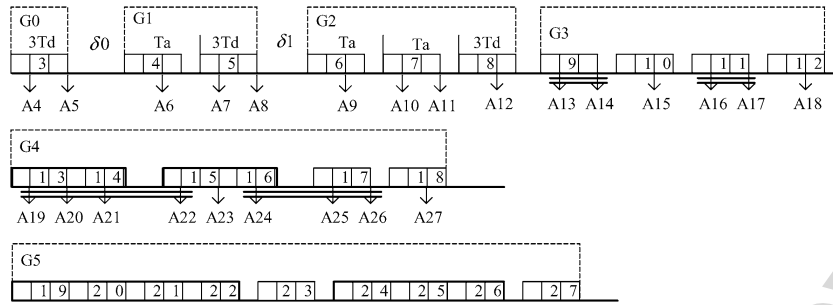


Fig. 4. Packet time line with AF during the SS phase at the receiver side (several initial packets are omitted.) The ACK packets marked by double bold lines are subject to filtering, which results in only the last one getting through the bottleneck link.

4.1. The slow start phase

In this model, each ACK still triggers 3 packets during the SS phase in the same way as in the ATCP Model. So Theorem 1 and Eq. (3) can still be used to determine the group of a specific packet, P_M . To obtain the arrival time of a packet, $T_{af}(M)$, it is necessary to know the beginning and the length of that group. By estimating the group offset of this packet and adding it to the group beginning time, it is possible to compute $T_{af}(M)$. Assuming δ_{k-1} , N_k^c and L_k are known, the derivation of δ_k and L_{k+1} are subsequently described.

Fig. 5 depicts the relationship between the block/chunk interval and the sending time of the ACKs. If G_k is Ack-aligned, and $T'(A_k^f) = B_k + \alpha_k T_d$ (α_k is 1 or 2), then it must transmit ACKs at the following time points: $T'(A_k^f)$, $T'(A_k^f) + T_a$, $T'(A_k^f) + 2T_a, \dots, T'(A_k^f) + (\lfloor L_k/T_a \rfloor - 1)T_a$, as shown in Fig. 5(a) and (b). At each time point, say $T'(A_k^f) + iT_a$ ($0 < i < \lfloor L_k/T_a \rfloor$), the ACK will carry the information of all ACKs produced between itself and previous time points, that is, $T'(A_k^f) + (i - 1)T_a \leq T(A) < T'(A_k^f) + iT_a$. Since the production of the last ACK A_k^l occurs either at $E_k - T_d$ or $E_k - 2T_d$, which are both beyond the scope of the ACK at $T'(A_k^f) + (\lfloor L_k/T_a \rfloor - 1)T_a$, there would be an extra ACK to be sent at $T'(A_k^f) + (\lfloor L_k/T_a \rfloor)T_a$. Hence for an Ack-aligned group, $N_k^a = \lfloor L_k/T_a \rfloor + 1$.

It can be observed that when $\delta_{k-1} > 2T_a$, G_k must be Ack-aligned. If, however, $\delta_k < 2T_a - 4T_d$ (see Fig. 5(c)), then the actual sending time for G_k 's first ACK, $T'(A_k^f)$, will be deferred to at least the end of G_k 's first chunk. This results in the ACK at $T'(A_k^f) + (\lfloor L_k/T_a \rfloor - 1)T_a$ covering the producing time of A_k^l , so $N_k^a = \lfloor L_k/T_a \rfloor$. If $2T_a - 4T_d \leq \delta_k \leq 2T_a$, there are several possible cases for N_k^a depending on the specific value of δ_{k-1} and the position of A_{k-1}^l . Here it is simply approximated as $\lfloor L_k/T_a \rfloor$, then

$$N_{k+1}^c = N_k^a = \begin{cases} \lfloor L_k/T_a \rfloor + 1, & \text{if } \delta_{k-1} \geq 2T_a, \\ \lfloor L_k/T_a \rfloor, & \text{if } \delta_{k-1} < 2T_a. \end{cases} \quad (10)$$

In ACK Filtering, the number of filtered ACKs should be limited by the maximum number of ACKs produced during a T_a . Since the ACK is produced at most every $2T_d$, the number of ACKs filtered is at most $T_a/2T_d - 1$. So

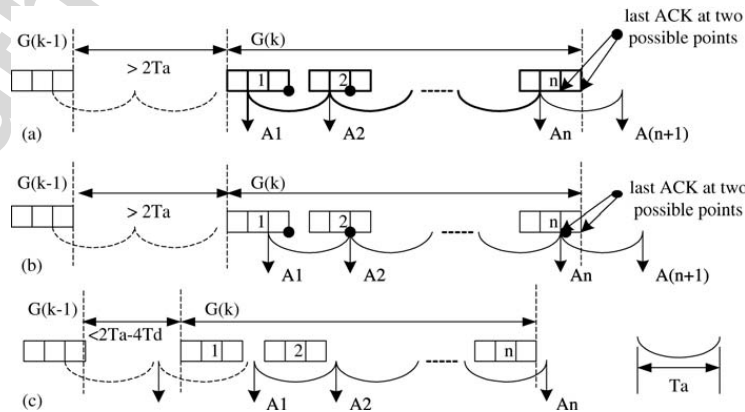


Fig. 5. Relationship between ACKs and chunks. In (a) and (b), the G_k is Ack-aligned, and in (c), the G_k is Ack-skewed.

the maximum possible chunk size is $(T_a/2T_d) \cdot 3T_d > T_a$. This means that the time interval among the chunks in fact shrinks to zero, although logically they are delivered independently from the sender. The chunk may grow so large that a series of blocks are left waiting for transmission on the forward link. This implies that the bottleneck effect has shifted from the reverse link to the forward link during a particular time period. This reflects on how AF improves bandwidth utilization of the forward link.

If the transfer over the forward link is absolutely smooth, that is, every chunk can be transmitted immediately after it is produced, the $L_k + \delta_k$ should be approximately $D + T_a$, as stated in Eq. (4). Therefore $\delta_k = D + T_a - L_k$. With the effect of AF, the chunk size grows and as a result queuing may occur on the forward link, and it is possible that $L_k + \delta_k > D + T_a$. In this case $\delta_k = \max\{T_a - 3N_k^c T_d, 0\}$. Therefore

$$\delta_k = \begin{cases} D + T_a - L_k, & \text{if } L_k \leq D + T_a, \\ \max\{T_a - 3N_k^b T_d/N_k^c, 0\}, & \text{if } L_k > D + T_a. \end{cases} \quad (11)$$

Knowing $L_{k+1} = (N_{k+1}^c - 1) \max\{3S_{k+1}^c T_d, T_a\} + 3S_{k+1}^c T_d$, then

$$L_{k+1} = (N_{k+1}^c - 1) \max\left\{\frac{3N_{k+1}^b}{N_{k+1}^c} T_d, T_a\right\} + \frac{3N_{k+1}^b}{N_{k+1}^c} T_d. \quad (12)$$

With $\delta_0 = D + T_a$ and $L_1 = T_a + 3T_d$, all δ_k and L_k with $k > 0$ are thus determined from the above recurrence. Now that all δ_k and L_k can be recursively obtained, the $T_{af}(M)$ can be computed as

$$T_{af}(M) = \sum_{i=1}^{x-1} (L_i + \delta_i) + \frac{M-9 - \sum_{i=1}^{x-1} N_i^b}{N_x^b} L_x = \sum_{i=1}^{x-1} (L_i + \delta_i) + \left[\frac{M+3}{2} \left(\frac{2}{3}\right)^{x-1} - 2 \right] L_x, \quad (13)$$

where $9 < M \leq 3(W_r - 1)$ and $x = \lceil \frac{\ln(M+3) - 2\ln 3}{\ln 3 - \ln 2} \rceil$.

4.2. The steady state phase

Let the index of the end group of the SS phase be S , then $S = \lceil (\ln W_r - \ln 3) / (\ln 3 - \ln 2) \rceil$. Since $cwnd$ stops increasing after group G_S , G_{S+1}^b can be estimated as G_S^b . Also G_S^c and δ_{S-1} are used to approximate G_{S+1}^c and δ_S , respectively. It is the case therefore that $L_{S+1} = L_S$. From G_{S+1} onwards, all following groups have the same chunk/block length and spacing as in the previous group. Therefore $G(M) = S + \lceil [(M/3) - W_r + 1] / N_S^b \rceil$ and,

$$T_{af}(M) = T_{af}(3W_r - 3) + \frac{\frac{M}{3} - (W_r - 1) - [G(M) - S]N_S^b}{N_S^b} L_S, \quad (14)$$

where $9 < M \leq 3(W_r - 1)$ and $N_S^b = 2(\frac{3}{2})^{S-1}$.

The transfer time during all phases has therefore been determined through Eq. (10) to Eq. (14).

4.3. Validating the ATCP-AF Model

The *ns-2* simulator is again used to validate this model. The same network topology and configurations as in the validation of ATCP Model are used in the experiment. Additionally the AF function is enabled in the link from node 2 to node 1, and ACK reconstruction in the link from node 1 to node 0. The comparisons between the simulated and modeled transfer time during the SS phase and in a longer time interval are plotted in Fig. 6(a) and 6(b), respectively. The error rate is less than 6% for more than 95% of predictions. It can be seen that the modeled results again provide a good match with the simulated results.

5. Conclusions

This paper studies the effects of bandwidth asymmetry on short-lived TCP transfer performance. In particular, it makes two contributions:

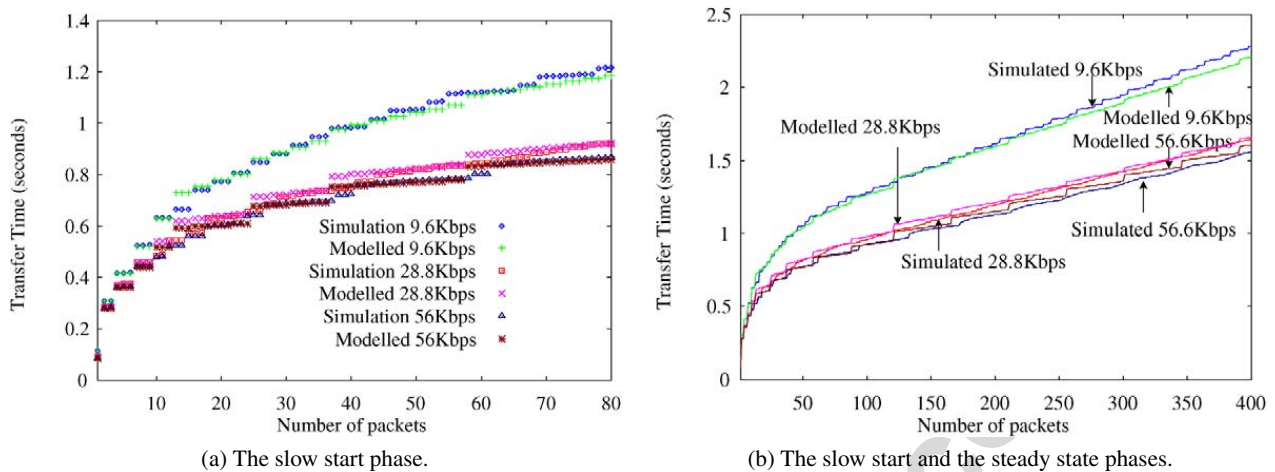


Fig. 6. Comparison between the simulated and the modeled results.

- An analytical model, the ATCP Model, is presented for short-lived TCP transfers over bandwidth asymmetric links. The model gives a close-form expression for transfer time prediction, which proves to accurately track the simulation results. To the best of our knowledge, this is the first effort in TCP modeling in the context of bandwidth asymmetric networks;
- An ATCP-AF Model for TCP transfer with ACK Filtering over asymmetric links is also presented. This model is given in a recursive form, and also provides very close predictions for the transfer time as compared to simulated results.

Simulations have been conducted to verify these models and the results show that they provide accurate predictions for short-lived TCP transfer latency.

References

- [1] N. Cardwell, S. Savage, T. Anderson, Modeling TCP latency, in: Proc. INFOCOM, Tel Aviv, Israel, March, 2000.
- [2] G. Hasegawa, M. Murata, H. Miyahara, Performance evaluation of HTTP/TCP on asymmetric networks, *Int. J. Comm. Systems* 12 (4) (July–August 1999) 281–296.
- [3] UCB/LBNL/VINT network simulator-ns, version 2, see <http://www.isi.edu/nsnam/ns/>.
- [4] T.V. Lakshman, U. Madhow, B. Suteret, Window-based error recovery and flow control with a slow acknowledgment channel: A study of TCP/IP performance, in: Proc. of INFOCOM '97, April, 1997.
- [5] M. Mathis, J. Semke, J. Mahdavi, T. Ott, The macroscopic behavior of the TCP congestion avoidance algorithm, *ACM Comput. Comm. Rev.* 27 (3) (July 1997) 67–82.
- [6] M. Mellia, I. Stoica, H. Zhang, TCP model for short lived flows, *IEEE Comm. Lett.* 6 (2) (February 2002) 85–87.
- [7] J. Padhye, V. Firoiu, D. Towsley, J. Kurose, Modeling TCP throughput: A simple model and its empirical validation, in: Proc. of ACM SIGCOMM '98, September, 1998, pp. 303–314.
- [8] N.K.G. Samaraweera, Return link optimization for Internet service provision using DVB-S networks, *Proc. ACM Comput. Comm. Rev.* 29 (3) (1999) 4–19.
- [9] K. Srinivan, Method and apparatus for collapsing TCP ACKs on asymmetric connections, US Patent 5,793,768, issued August 11, 1998, filed August 13, 1996.