Estimating TCP Flow Completion Time Distributions

Gan Luan

Abstract: **The flow completion time is an important performance metric for network users, and it is used in designing scheduling and congestion control algorithms. The existing TCP flow completion time distributions are calculated from complicated models, and they are only suitable for short TCP flows. This paper is motivated to build a simple model to derive TCP flow completion time distribution that is suitable for both short and long TCP flows. To build such a model, the distribution of the packet transmission latency under TCP congestion control is firstly calculated. The packet transmission latency is the time needed to successfully deliver a packet, including the time for retransmissions due to packet losses. Semi-Markov process is used to model the TCP congestion window evolution. Using the packet transmission latency distribution together with the TCP congestion window size information, a new parallel queueing model which describes how a file is transmitted by a TCP flow is formed. In the parallel queueing model, the flow completion time distribution is derived, and simulation results from NS2 demonstrate the accuracy of the proposed analysis.**

Index Terms: **Congestion control, flow completion time, packet loss, packet transmission latency, TCP.**

I. INTRODUCTION

FLOW completion time is the duration to finish the file transfer in a flow. It is an important performance metric for network users, in both wired and wireless networks [1]–[21]. Network users are sensitive to the time needed to finish the data transfer, rather than the route the flow takes or the packet loss probability it experiences. The flow completion time is also required by quality of service (QoS) requirements of a wide range of network services. In data center networks, the flow completion time is considered in designing both congestion control algorithms [2]–[6] and scheduling algorithms [6]–[11]. However, authors in [2]–[11] assume the flow is served at a constant rate without any retransmissions, which loses credibility. Whenever a packet loss happens, a retransmission is triggered to provide reliable service in the flow, and the retransmissions of that flow make the flow completion time random. According to [7], the flow completion time may rise to three times of its mean value. Therefore, to consider flow completion time distributions in designing congestion control and scheduling algorithms for deadline-aware network services is important.

Manuscript received May 28, 2018; approved for publication by SuKyoung Lee, Division III Editor, September 13, 2018.

This work was jointly supported by: (1) National Natural Science Foundation of China (No. 61771068, 61671079, 61471063, 61372120, 61421061); (2) Beijing Municipal Natural Science Foundation (No. 4182041, 4152039); (3) the National Basic Research Program of China (No. 2013CB329102).

G. Luan is with the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, China, email: luangan@bupt.edu.cn.

Digital Object Identifier: 10.1109/JCN.2019.000006

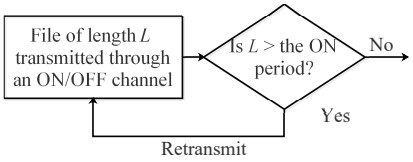

Fig. 1. File transmitted through ON/OFF wireless channel.

The flow completion time distribution is studied in both wired and wireless networks [12]–[20]. In wireless networks [12]–[14], the authors conclude, due to retransmissions, the file completion time is heavy-tailed even when the file size is light-tailed, provided that the file size distribution has infinite support; when the file size has finite support, the transmission duration distribution exhibits a transition from power-law body to exponential tail. In these papers, the channel dynamics are modeled as ON/OFF process, and the alternating on and off periods lead to retransmissions. As shown in Fig. 1, when an ON period is not long enough to transmit the whole file, the file is retransmitted in the next ON period. In wired networks, a file is segmented into multiple packets and transmitted in a flow, instead of being transmitted as a whole file as in wireless networks. The data links in wired networks do not have availability problems as in wireless channels. In wired networks, over 90% of the packet traffic is generated by applications using TCP. So the flow completion time distribution in wired networks is mostly analyzed under TCP. In [15] and [16], the authors propose methods to draw state machines according to the evolution of TCP congestion window to compute completion time distributions. The drawbacks of these methods are that a new diagram has to be generated to analyze the flow completion time distribution for a new set of network parameters and the models get complex when the maximum window size of the flow becomes large. In [17]–[19], the authors analyze different stages of the congestion window behavior, calculate the duration of each behavior stage, and derive a flow completion time as a weighted sum of these durations. The defect of these methods is that all possible packet loss situations are considered in the flow completion time equation. These techniques are suitable for finding the expectation value of the flow completion time, however, they require the enumeration of all packet loss scenarios in calculating the flow completion time distribution, which makes them not suitable for analyzing long TCP flows. In [20], the authors use a recursive model to derive the completion time of short-lived TCP flows, and the recursive equation takes all patterns of packet losses into consideration. The result is accurate for short flows up to about 10 segments. Similar to [17]–[19], the method quickly be-

come intractable for long TCP flows because of the exponential growth in the number of packet losses.

In this paper, we are motivated to build a new model to estimate the flow completion time distribution under TCP that is simple and accurate, suits for both short and long flows. To build the flow transmission model, distribution of transmission latency for each packet under TCP congestion control is calculated, i.e., the time to successfully deliver a packet is calculated, including the time needed for retransmissions due to packet losses. Then we model the TCP congestion window size development with a semi-Markov process. The congestion window size development process describes the steady state of congestion window behavior, and the packet transmission latency distribution reflects the time needed to deliver a packet in the flow. Employing the packet transmission latency distribution result and the TCP congestion window size steady state information, the flow transmission model is formed as a parallel queueing system. The flow completion time distribution is derived, by analyzing a single queue as a renewal process using the central limit theorem and combining the dependent parallel queueing processes according to the congestion control rules. We evaluate our model by using the NS2 simulator. The simulations show that the measured distribution of the packet transmission latency and the distribution of the flow completion time from the NS2 simulator match our theoretical results very well. The contributions of this paper are summarized as follows:

1. The closed-form solution to the distribution of packet transmission latency under TCP congestion control, considering the time for retransmissions due to packet losses, is derived for the first time.
2. The TCP flow transmission is modeled as a parallel queuing system for the first time.
3. Distribution of flow completion time is derived from the new flow transmission model, and it is suitable for analyzing both short and long TCP flows.

The remainder of this paper is organized as follows. Section II introduces the packet transmission model and the system assumptions for TCP flows. In Section III, the distribution of packet transmission latency is calculated. Section IV describes the congestion window semi-Markov model and the flow transmission parallel queuing model. The flow completion time distribution is derived in Section V. Simulation results are presented in Section VI. Section VII discusses about extensions and applications of our estimation model, and Section VIII concludes the paper.

II. PACKET TRANSMISSION MODEL UNDER TCP

Fig. 2 shows the TCP flows on a general network. TCP Reno is currently the most widely used version of TCP in different TCP implementations. Our analytical model is built based on the congestion control algorithm used in TCP Reno. The progression of the congestion window size can be recognized as a cycle of three phases, namely, slow start phase, congestion avoidance phase, and timeout phase, as shown in Fig. 3. We are going to describe the model for the packet transmission process, and in our model we assume the network has a constant packet loss probability p.

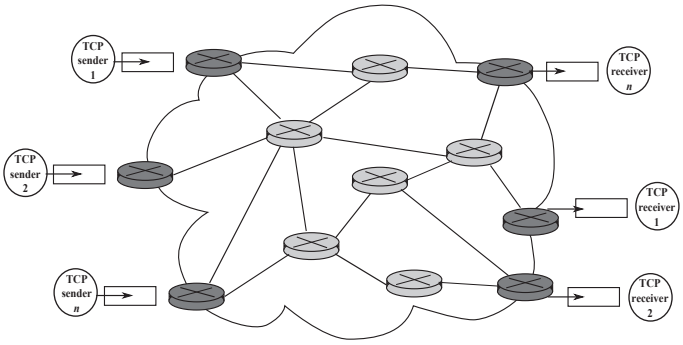

Fig. 2. TCP flows on a general network.

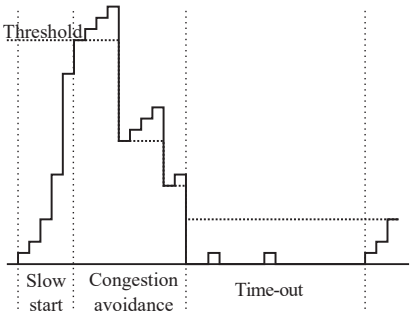

Fig. 3. Three phases of the congestion window process.

In both slow start phase and congestion avoidance phase, a packet will be successfully sent to the receiver with probability $1-p$ and get lost with probability p. A lost packet is identified by triple duplicated ACK's with probability $1-Q$, and identified by a timeout with probability Q [23]. When triple duplicated ACK's appear, the sender will operate the multiplicative-decrease procedure, resend the lost packet, and then continue its congestion avoidance phase. When timeout happens, the sender instantly changes its congestion window down to 1 and enters its timeout phase. In the timeout phase, the sender will wait for T_0, the preset timeout value, before retransmitting the lost packet. After that, with probability p the resent packet will get lost and trigger the next round of timeout. In the timeout phase, the timeout value doubles when each round of timeout happens until a value of $64T_0$ is reached. Beyond that, the timeout value remains $64T_0$, until the resent packet is successfully acknowledged by the receiver. Then the timeout phase ends and the slow start phase is entered.

We denote the single trip time (STT) as random variable T_{STT} and the round trip time (RTT) as T_{RTT}. Random variable T_{pkt} represents the time of successfully sending a packet under TCP, including the time needed for retransmissions due to packet losses. We use T_{TO} to denote the length of a timeout phase.

III. PACKET TRANSMISSION LATENCY DISTRIBUTION

Packet transmission latency distribution itself is also an important measure of congestion control performance, especially for short-lived TCP flows [21]. In [21], it is pointed out that the transmission latency of the last packet in a short flow has great influence on the total completion time of the flow. To the authors' knowledge, there has not been any closed-form solu-

tion to the packet transmission latency distribution under TCP, considering the time needed for retransmissions due to packet losses. We are going to derive a closed-form probability density function (PDF) for packet transmission latency under TCP Reno in this section.

Based on the TCP congestion control model described above, we concluded that the latency experienced by a TCP packet is

$$T_{\rm pkt} = \begin{cases} T_{\rm STT} & w.\ p.\ (1-p); \\ T_{\rm RTT} + T_{\rm pkt} & w.\ p.\ p(1-Q); \\ T_{\rm TO} + T_{\rm STT} & w.\ p.\ pQ, \end{cases} \tag{1}$$

where the abbreviation of w. p. represents "with probability," and based on (1) we have the pdf of $T_{\rm pkt}$:

$$\begin{aligned} f_{T_{\rm pkt}}(x) =& (1-p) f_{T_{\rm STT}}(x) + pQ f_{T_{\rm TO}} * f_{T_{\rm STT}}(x) \\ &+ p(1-Q) f_{T_{\rm RTT}} * f_{T_{\rm pkt}}(x). \end{aligned} \tag{2}$$

Notice this pdf is actually a defective improper renewal equation, hence, we have the following theorem,

Theorem 1: The bounded unique solution for the distribution of a packet transmission latency under TCP Reno, which fits (1), is

$$f_{T_{\rm pkt}}(x) = A(x) + A(x) * V(x), \tag{3}$$

where $A(x) = (1-p) f_{T_{\rm STT}}(x) + pQ f_{T_{\rm TO}} * f_{T_{\rm STT}}(x)$, $V(x) = \sum_{n=1}^{\infty} [p(1-Q)]^n f^{(n)}_{T_{\rm RTT}}(x)$, and $f^{(n)}_{T_{\rm RTT}}(x)$ denotes the n-fold convolution of $f_{T_{\rm RTT}}(x)$.

Proof: The proof has two parts.

First of all, (3) is a solution. Convolving both sides of (3) with $p(1-Q) f_{T_{\rm RTT}}(x)$, we have

$$\begin{aligned} & p(1-Q) f_{T_{\rm RTT}} * f_{T_{\rm pkt}}(x) \\ &= p(1-Q) f_{T_{\rm RTT}} * A(x) + p(1-Q) f_{T_{\rm RTT}} * A(x) * V(x) \\ &= A(x) * V(x). \end{aligned} \tag{4}$$

Since $A(x) * V(x) = f_{T_{\rm pkt}}(x) - A(x)$, (4) can be written as $p(1-Q) f_{T_{\rm RTT}} * f_{T_{\rm pkt}}(x) = f_{T_{\rm pkt}}(x) - A(x)$. Therefore, (3) is a solution that satisfies the original defective improper renewal equation.

Second, we are going to show solution (3) is unique. Assume there is another pdf satisfies (2), which is denoted as $f'_{T_{\rm pkt}}(x)$. The difference of the two distribution functions is represented as

$$W(x) = f_{T_{\rm pkt}}(x) - f'_{T_{\rm pkt}}(x).$$

Then

$$\begin{aligned} & p(1-Q) f_{T_{\rm RTT}} * W(x) \\ &= p(1-Q) f_{T_{\rm RTT}} * f_{T_{\rm pkt}}(x) - p(1-Q) f_{T_{\rm RTT}} * f'_{T_{\rm pkt}}(x) \\ &= [f_{T_{\rm pkt}}(x) - A(x)] - [f'_{T_{\rm pkt}}(x) - A(x)] \\ &= W(x), \end{aligned} \tag{5}$$

which means $W(x) = [p(1-Q)]^n f^{(n)}_{T_{\rm RTT}}(x) * W(x)$ must hold for any $n \in \mathbb{N}_+$. Therefore $W(x) = 0$, and the solution (3) is unique. □

When Laplace transforms of $f_{T_{\rm STT}}(x)$, $f_{T_{\rm TO}}(x)$, and $f_{T_{\rm RTT}}(x)$ exist, and we denote them as $\Phi_{T_{\rm STT}}(s)$, $\Phi_{T_{\rm TO}}(s)$, and $\Phi_{T_{\rm RTT}}(s)$, the closed-form solution to (2) is the inverse transform of

$$\Phi_{T_{\rm pkt}}(s) = \frac{(1-p)\Phi_{T_{\rm STT}}(s) + pQ\Phi_{T_{\rm TO}}(s)\Phi_{T_{\rm STT}}(s)}{1 - p(1-Q)\Phi_{T_{\rm RTT}}(s)}, \tag{6}$$

i.e.,

$$f_{T_{\rm pkt}}(x) = \mathcal{L}^{-1}\left[\Phi_{T_{\rm pkt}}(s)\right], \tag{7}$$

where $\mathcal{L}^{-1}[\cdot]$ is the inverse Laplace transform operation. When any of the Laplace transforms of $f_{T_{\rm STT}}(x)$, $f_{T_{\rm TO}}(x)$, and $f_{T_{\rm RTT}}(x)$ does not exist, there would still be a rather good approximation for the packet transmission latency distribution if one cuts off the tails of the summation $V(x)$ with large enough n, since the weights in the summation are geometrically decreasing.

The result in (3) is a new result for analyzing single packet's transmission delay considering the packet losses and retransmissions. It is different from the retransmission delay results in wireless networks in [12]–[14], since the ON/OFF channel availability is not the reason causing retransmissions and the retransmission rules are different. The packet transmission latency expression in (1) is concluded based on the TCP transmission rules. Laplace transform and method of solving renewal equations are used to achieve the closed-form solution for pdf of $T_{\rm pkt}$.

A. Example

ICMP experimental results show that RTT along a fixed path in the Internet follows a shifted gamma distribution [22], and the shifted gamma distribution can be denoted as

$$f_{T_{\rm RTT}}(x) = \begin{cases} \frac{(x-d)^{c-1} e^{-\frac{x-d}{b}}}{b^c \Gamma(c)},\ t \geq d & \text{if } c > 0; \\ \delta(x-d) & \text{if } c = 0, \end{cases} \tag{8}$$

where $b > 0$ is the scale parameter, $c > 0$ is the shape parameter, d is the shift from the origin, $\Gamma(\cdot)$ is the gamma function, $\Gamma(c) = \int_0^{+\infty} t^{c-1} e^{-t} {\rm d}t$, and $\delta(\cdot)$ is the unit impulse function. The Laplace transform for the shifted gamma distribution in (8) is,

$$\Phi_{T_{\rm RTT}}(s) = \begin{cases} e^{-sd}(1+bs)^{-c} & \text{if } c > 0; \\ e^{-sd} & \text{if } c = 0, \end{cases} \tag{9}$$

In this paper, we assume the one way delays between the two ends are symmetric, and $T_{\rm STT} = T_{\rm RTT}/2$. Hence the Laplace transform of the one way delay distribution function is

$$\Phi_{T_{\rm STT}}(s) = \frac{1}{2}\Phi_{T_{\rm RTT}}(\frac{s}{2}). \tag{10}$$

As pointed out in [23], when T_0 is preset by TCP algorithm, the timeout period $T_{\rm TO}$ takes the following values

$$T_{\rm TO} = \begin{cases} (2^k - 1)T_0 & \text{for } k \leq 6; \\ (63 + 64(k-6))T_0 & \text{for } k \geq 7, \end{cases} \tag{11}$$

where each value of k happens with probability $P(k) =$

$p^{k-1}(1-p)$. Therefore, the Laplace transform of T_{TO} is

$$\begin{aligned}\Phi_{T_{\mathrm{TO}}}(s) = {} & (1-p)e^{-sT_0} + p(1-p)e^{-s3T_0} \\ & + p^2(1-p)e^{-s7T_0} + p^3(1-p)e^{-s15T_0} \\ & + p^4(1-p)e^{-s31T_0} + p^5(1-p)e^{-s63T_0} \\ & + \frac{p^6(1-p)e^{-s127T_0}}{1-pe^{-s64T_0}}.\end{aligned} \tag{12}$$

So, taking (9), (10), and (12) into (7) we achieve closed-form pdf for the latency experience by a TCP packet. This example shows how our method can be used to find packet transmission latency distributions of the real network. The result of the single packet's transmission latency distribution is used in building models to analyze the flow completion time in the next section.

IV. FLOW TRANSMISSION MODEL

To find a simple and accurate estimation of the flow completion time distribution, which can analyze flows with any length and under various congestion control algorithms, we need to build a simple model describing the flow transmission in networks, other than using the combinations of time spent in different evolution stages of congestion window to explain the flow transmission latency.

The flow transmission is modeled as a parallel queueing system. In order to describe the parallel queueing system, we denote the time average congestion window size as $\mathrm{E}[W] = \lim_{N\to+\infty}\frac{1}{N}\sum_{n=1}^{N}W(n)$, where $W(n)$ represents the congestion window size during the nth RTT. Notice that value $\mathrm{E}[W]$ means, on average, there is always $\mathrm{E}[W]$ windows working to serve the packets at each moment, during the flow transmission. Therefore, we model the flow transmission as $\mathrm{E}[W]$ servers serve the same flow of length L_{flow} packets, and the service time for each packet is identically distributed as $T_{\mathrm{seg}} = T_{\mathrm{pkt}} + T_{\mathrm{ack}}$, where T_{ack} is the time it takes to ACK the sender that the packet has successfully received by the receiving end, which corresponds to T_{STT} in this paper.

The time average congestion window size can be found both analytically and empirically. Empirical method is to take samples of a congestion window size sample path, and then calculate the mean of the samples, whereas the analytical method is to analyze durations and behaviors of congestion window size in different stages of the evolution according to the congestion control algorithm, and then calculate $\mathrm{E}[W]$. In the following part, $\mathrm{E}[W]$ of TCP Reno is derived analytically, and how this model can be used to estimate the flow completion time distribution is discussed in the next section.

A. Time Average Congestion Window Size in TCP Reno

According to the description in [23] and taking the slow start phase into consideration, the behavior of the TCP congestion window size sample path can be described as a semi-Markov process, which is shown in Fig. 4. The evolution of congestion window size has three states, namely the additive-increase state, the timeout state, and the slow start state. When it is in the additive-increase state, indicated as the left circle in Fig. 4, any packet loss will trigger a state transition. The packet loss

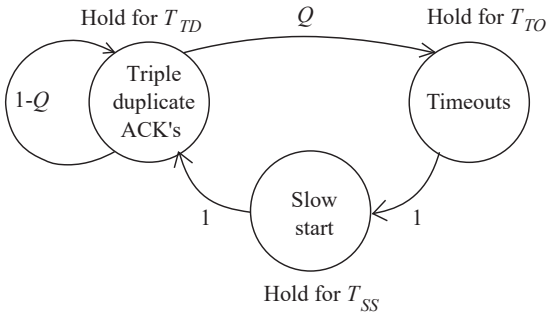

Fig. 4. The semi-Markov process for the TCP congestion window process.

is identified by a timeout with probability $Q = 3/W_{\mathrm{TD}}$, and is identified by three duplicate acknowledgements with probability $1-Q$ [23]. If it is identified by a timeout, the congestion window will start its timeout state, and if it is identified by three duplicate acknowledgements, the congestion window will halve its current congestion window size and window threshold, and then re-enter the additive-increase state. After each timeout state, there follows a slow start phase. The time spent in the additive-increase phase before a state transition happens for the semi-Markov process is denoted as T_{TD}, the time spent in the timeout phase before the next additive-increase phase starts is denoted as T_{TO}, and the time spent in the slow start phase is denoted as T_{SS}.

The embedded Markov chain is irreducible and positive recurrent [24]. The steady state probabilities of the embedded chain are

$$\{\pi_1 = \frac{1}{1+2Q}, \pi_2 = \frac{Q}{1+2Q}, \pi_3 = \frac{Q}{1+2Q}\}.$$

Thus, the time average probabilities of being in the three states are

$$\beta_1 = \frac{\pi_1 \times \mathrm{E}[T_{\mathrm{TD}}]}{\pi_1 \times \mathrm{E}[T_{\mathrm{TD}}] + \pi_2 \times \mathrm{E}[T_{\mathrm{TO}}] + \pi_3 \times \mathrm{E}[T_{\mathrm{SS}}]},$$

$$\beta_2 = \frac{\pi_2 \times \mathrm{E}[T_{\mathrm{TO}}]}{\pi_1 \times \mathrm{E}[T_{\mathrm{TD}}] + \pi_2 \times \mathrm{E}[T_{\mathrm{TO}}] + \pi_3 \times \mathrm{E}[T_{\mathrm{SS}}]},$$

and

$$\beta_3 = \frac{\pi_3 \times \mathrm{E}[T_{\mathrm{SS}}]}{\pi_1 \times \mathrm{E}[T_{\mathrm{TD}}] + \pi_2 \times \mathrm{E}[T_{\mathrm{TO}}] + \pi_3 \times \mathrm{E}[T_{\mathrm{SS}}]},$$

respectively. It is easy to see that $\mathrm{E}[T_{\mathrm{TD}}] = T_{\mathrm{RTT}}\mathrm{E}[W_{\mathrm{TD}}]/2$, $\mathrm{E}[T_{\mathrm{SS}}] = T_{\mathrm{RTT}}(\mathrm{E}[\lceil \log_2(W_{\mathrm{TD}}/2)\rceil] + 1)$, and it can be found in [23] that $\mathrm{E}[T_{\mathrm{TO}}] = T_0(1 + p + 2p^2 + 4p^3 + 8p^4 + 16p^5 + 32p^6)/(1-p)$, where T_0 is the initial timeout value set by the sender and p is the packet loss probability.

Since the timeouts and the slow start phases are now considered, the expectation of window size W during the whole time is now

$$\mathrm{E}[W] = \beta_1\frac{3}{4}\mathrm{E}[W_{\mathrm{TD}}] + \beta_2 \times 1 + \beta_3\mathrm{E}[W_{\mathrm{SS}}], w \geq 1. \tag{13}$$

As is pointed out in [23], the congestion window size at the end of a TDP

$$\mathrm{E}[W_{\mathrm{TD}}] = 1 + \sqrt{\frac{8(1-p)}{3p} + 1} \tag{14}$$

and $\mathrm{E}[W_{\mathrm{TD}}] \approx \sqrt{8/(3p)}$ for small values of p. Hence, the estimated value of Q is

$$\hat{Q} = \frac{3}{\mathrm{E}[W_{\mathrm{TD}}]} = \frac{3}{1+\sqrt{\frac{8(1-p)}{3p}+1}} \tag{15}$$

and $\hat{Q} \approx 3\sqrt{3p/8}$ for small values of p [23]. In slow start phase, the window size grows exponentially with base 2 and the index takes on natural numbers starts from 0, until it reaches the threshold value, which is half of the congestion window size at the end of the previous TDP. Therefore,

$$\begin{aligned}\mathrm{E}[W_{\mathrm{SS}}] &= \int_0^{+\infty} f_{W_{\mathrm{TD}}}(x)\frac{\sum_{i=0}^{\lceil \log_2(x/2)\rceil -1} 2^i + x/2}{\lceil \log_2(x/2)\rceil + 1}\mathrm{dx} \\ &= \int_0^{+\infty} f_{W_{\mathrm{TD}}}(x)\frac{2^{\lceil \log_2 x\rceil -1} - 1 + x/2}{\lceil \log_2(x/2)\rceil + 1}\mathrm{dx} \\ &\approx \frac{\int_0^{+\infty} f_{W_{\mathrm{TD}}}(x)\left(2^{\lceil \log_2 x\rceil -1} - 1 + x/2\right)\mathrm{dx}}{\int_0^{+\infty} f_{W_{\mathrm{TD}}}(x)\left(\lceil \log_2(x/2)\rceil + 1\right)\mathrm{dx}} \\ &\approx \frac{\mathrm{E}[W_{\mathrm{TD}}]-1}{\lceil \log_2 \mathrm{E}[W_{\mathrm{TD}}]\rceil}.\end{aligned} \tag{16}$$

Eq. (13) is the time average congestion window size that we derived.

This general way of modeling TCP flow transmission as a parallel queueing system has not been found anywhere else. Independencies are assumed between successive packets served by a same server and among packets served by different servers at the same time. The number of servers, the time average congestion window size $\mathrm{E}[W]$, can be understood as the throughput of the flow, however, the service time of each server follows the distribution of $T_{\mathrm{seg}} = T_{\mathrm{pkt}} + T_{\mathrm{ack}}$ we derived in Section III. This simplified multiple server single flow system model gives us an easy angle of viewing flow transmission, and it unveils a simple way to solve the flow completion time distribution, instead of using the complicated models and state machines used in [15]–[20].

V. DISTRIBUTION OF FLOW COMPLETION TIME

In our model, a flow of length L_{flow} will be served simultaneously by $\mathrm{E}[W]$ servers. In a multiple server single flow system, servers' service durations are identical, i.e., $T_{\mathrm{svr}_1} = \cdots = T_{\mathrm{svr}_{\mathrm{E}[W]}} = T_{\mathrm{svr}}$, where T_{svr} is used to represent the service time of any server. According to the law of large numbers, the mean number of segments served by each server is $\mathrm{E}[L_{\mathrm{svr}}] = L_{\mathrm{flow}}/\mathrm{E}[W]$. Notice that during TCP congestion window development, whenever a packet loss is detected, either by triple duplicated ACK's or by a timeout, all packets in the remaining flow are held up until the retransmission is finished. So the service of the parallel queues are not independent; they are influenced by each other. We need to further simplify the model, by supposing all the servers are independently serving $\mathrm{E}[L_{\mathrm{svr}}]$ packets with identical total service durations, but the service will be paused when any packet loss happens in the system until the retransmission is successfully ACKed.

On one hand, the service in a single server system serves $\mathrm{E}[L_{\mathrm{svr}}] = L_{\mathrm{flow}}/\mathrm{E}[W]$ packets with independent and identically distributed service time T_{seg} is a renewal process with $\mathrm{E}[L_{\mathrm{svr}}]$ arrivals. The total service time T'_{svr} of this simpler single server model is normally distributed with mean $\mu_{T'_{\mathrm{svr}}} = \mu_{T_{\mathrm{seg}}} \times \mathrm{E}[L_{\mathrm{svr}}]$ and standard deviation $\sigma_{T'_{\mathrm{svr}}} = \sqrt{\mathrm{E}[L_{\mathrm{svr}}] \times \sigma^2_{T_{\mathrm{seg}}}}$, based on the central limit theorem.

On the other hand, in the system with $\mathrm{E}[W]$ servers we assumed, the flow completion time is estimated as

$$T_{\mathrm{flow}} = \sum_{i=1}^{\mathrm{E}[W]} T_{\mathrm{svr_i}} - (\mathrm{E}[W]-1) \times \mu_{T_{\mathrm{svr}}}. \tag{17}$$

Because $T_{\mathrm{svr}_1} = \cdots = T_{\mathrm{svr}_{\mathrm{E}[W]}}$,

$$T_{\mathrm{flow}} = \mathrm{E}[W] \times T_{\mathrm{svr}} - (\mathrm{E}[W]-1) \times \mu_{T_{\mathrm{svr}}}. \tag{18}$$

To simplify this expression of T_{flow}, we substitute T_{svr} with T'_{svr}, hence, $T'_{\mathrm{flow}} = \mathrm{E}[W] \times T'_{\mathrm{svr}} - (\mathrm{E}[W]-1) \times \mu_{T'_{\mathrm{svr}}}$, and T'_{flow} is normally distributed with mean $\mu_{T'_{\mathrm{flow}}} = \mu_{T'_{\mathrm{svr}}} = \mu_{T_{\mathrm{seg}}} \times \mathrm{E}[L_{\mathrm{svr}}]$ and standard deviation $\sigma_{T'_{\mathrm{flow}}} = \mathrm{E}[W] \times \sigma_{T'_{\mathrm{svr}}}$. Therefore,

$$T'_{\mathrm{flow}} \sim \mathcal{N}(\mu_{T'_{\mathrm{flow}}}, \sigma^2_{T'_{\mathrm{flow}}}) \tag{19}$$

is our estimated distribution of the flow completion time.

The normal distribution in (19) is the key result of our paper. It is achieved by simplifying and analyzing the parallel queueing flow transmission model proposed in Section IV. The central limit theorem and the law of large numbers are used in the simplified renewal process of a single server, and the service time of different servers are combined together to reach the flow completion time, based on the congestion control retransmission rules. Hence the flow completion time distribution is achieved.

VI. SIMULATION RESULTS

We evaluated our results by using NS2 simulator. In order to show the accuracy of our distribution, we firstly simulated the single TCP flow which is transmitted on a 104 ms bidirectional link with capacity 10 Mbps that connects the sender and the receiver. Packets are randomly dropped with probability p. We simulated the packet transmission latency distribution and the flow completion time distribution for different packet loss probabilities and different flow lengths. Then we simulated multiple TCP flows transmitted by the same 104 ms, 10 Mbps link with $p = 0.01$. We observed the flow completion time distribution of one of the multiple flows and discussed the impact of the multiple flows on the flow completion time distribution. We also tested the packet transmission latency distribution and the flow completion time distribution of a single flow transmitted on a 22 ms, 1000 Mbps bidirectional link with $p = 0.01$ to test our theoretical results. The results are presented as follows.

A. *Single TCP Flow on a* 104 ms, 10 Mbps *Link*

The packet transmission latency CDF results are shown in Fig. 5. Since packet loss happens rarely for a single packet, we simulated the distributions with packet loss probabilities of $p = 0.1$ and $p = 0.01$, which are a little bit greater than the real network values, to show the accuracy of our theoretical results. The simulated CDFs of packet transmission latency under TCP Reno fits very well with the theoretical CDFs we derived using renewal equation and Laplace transform in (3).

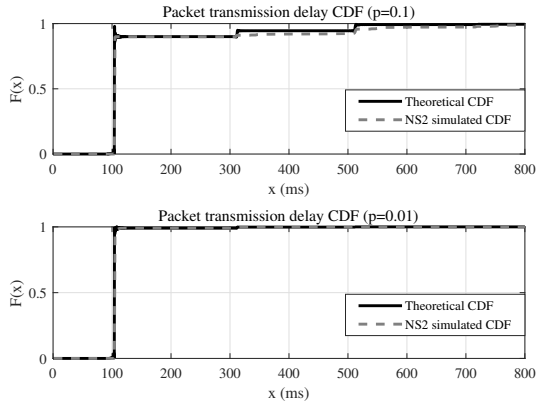

Fig. 5. Simulated and theoretical results for packet transmission latency CDFs of single TCP flow on a 104 ms, 10 Mbps link.

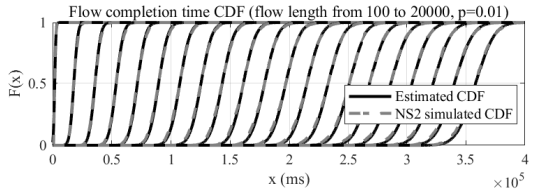

Fig. 6. Simulated and theoretical results for flow completion time CDFs of single TCP flow on a 104 ms, 10 Mbps link with different flow lengths and $p = 0.01$.

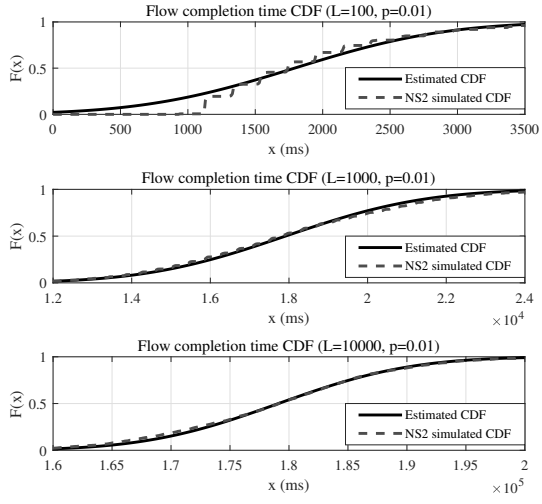

Fig. 7. Simulated and theoretical results for flow completion time CDFs of single TCP flow on a 104 ms, 10 Mbps link with $L_{\text{flow}} = 100$, 1000, and 10000 packets, and $p = 0.01$.

Figs. 6 and 8 show the flow completion time CDF simulation results and theoretical distribution results with different flow lengths under packet loss probabilities of $p = 0.01$ and $p = 0.006$, respectively. The flow lengths of the simulations were set to be $L_{\text{flow}} = 100$ packets and L_{flow} increases from 1000 to 20000 packets with step sizes of 1000 packets. It is shown in Figs. 6 and 8 that the NS2 simulation results fits our estimated CDFs very well. To zoom in and see details of the

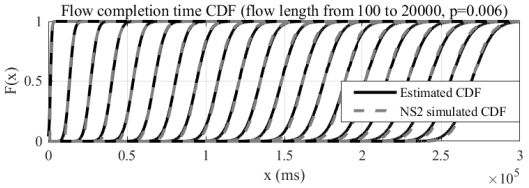

Fig. 8. Simulated and theoretical results for flow completion time CDFs of single TCP flow on a 104 ms, 10 Mbps link with different flow lengths and $p = 0.006$.

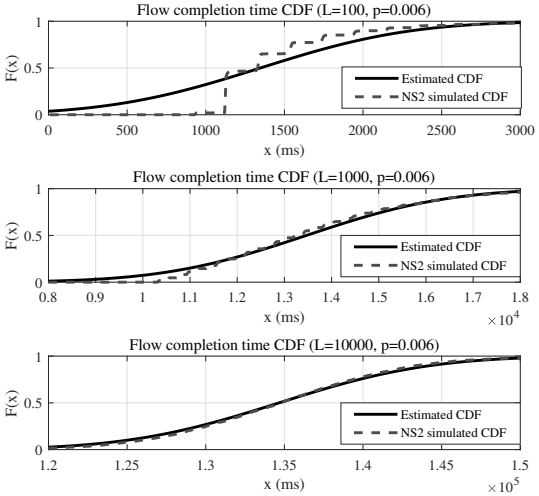

Fig. 9. Simulated and theoretical results for flow completion time CDFs of single TCP flow on a 104 ms, 10 Mbps link with $L_{\text{flow}} = 100$, 1000, and 10000 packets, and $p = 0.006$.

simulated results in Figs. 7 and 9 with flow lengths of 100, 1000, and 10000 packets, under packet loss probabilities of $p = 0.01$ and $p = 0.006$, respectively. The simulated distribution curves match the theoretical CDFs very well for both short and long flow lengths. The estimated means and the estimated standard deviations are accurate. The staircase phenomena appeared in Figs. 7 and 9 for $L_{\text{flow}} = 100$ packets are resulted from the fact that when flow lengths are short, packet losses are not likely to happen for a large number of packets, hence the simulated results are not smoothly normally distributed, and the steps in the simulated CDFs corresponds to the time delays brought in by packet losses in the flow. Even if the simulated CDFs of short flow completion time are not smoothly fitting estimated normal distributions, the estimated mean and spread (standard deviation) for short flows are still pretty accurate, as can be seen in Figs. 7 and 9.

B. Multiple TCP Flows on a 104 *ms,* 10 *Mbps Link*

We simulated 3 TCP flows transmitted on a same 104 ms, 10 Mbps link with the packet loss probability $p = 0.01$. Fig. 10 shows the flow completion time CDF results of one of the multiple TCP flows with different flow lengths. The flow completion time distribution in the chosen TCP flow is not infected by the other TCP flows sharing the same route. This is true because the throughput of a TCP flow is dynamically adjusted by the con-

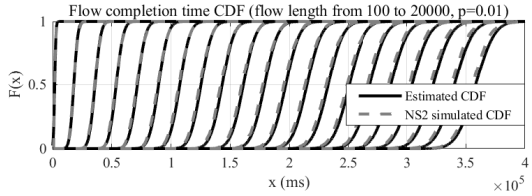

Fig. 10. Simulated and theoretical results for flow completion time CDFs of one of the multiple TCP flows on a 104 ms, 10 Mbps link with different flow lengths and $p = 0.01$.

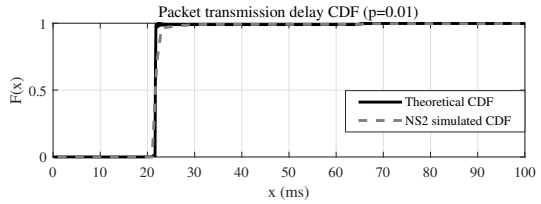

Fig. 11. Simulated and theoretical results for packet transmission latency CDFs of single TCP flow on a 22 ms, 1000 Mbps link.

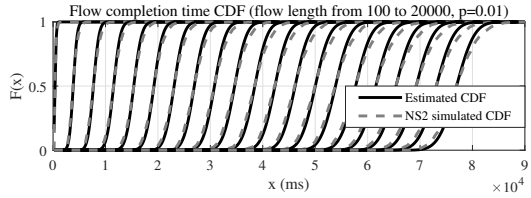

Fig. 12. Simulated and theoretical results for flow completion time CDFs of single TCP flow on a 22 ms, 1000 Mbps link with different flow lengths and $p = 0.01$.

gestion control algorithm, which adapts the flow data rate to the current network traffic load, reflected by packet loss probabilities. A network data link capacity is chosen with the ability to support multiple TCP flows. So when multiple flows share a same link, the completion time distribution of one of the flows is not influenced, given that the multiple flows did not cause more packet losses and the network packet loss probability remains stable. However, if there are too many TCP flows sharing a same link, the queueing delay will accumulate and the packet loss probability will increase markedly. In that situation the flow completion time distribution can be recalculated by taking the corresponding packet transmission latency distribution and packet loss probability into the parallel queueing flow transmission model.

C. Single TCP Flow on a 22 ms, 1000 Mbps Link

The packet transmission latency CDF results for a single TCP flow on a 22 ms, 1000 Mbps link with $p = 0.01$ is shown in Fig. 11. The packet transmission is much faster than in the previous simulation cases, when the link capacity is greater and the delay is lower. The flow completion time distribution is calculated and compared with the NS2 simulation results in Fig. 12. We can see that the simulated results fit the estimated flow completion time distributions very well, and the mean values of the flow completion time in Fig. 12 is much less than that in Fig. 6.

VII. DISCUSSION

In this section, discussions about how this estimation method can be extended to analyze other congestion control algorithms and how the flow completion time distributions can be used in designing scheduling and congestion control algorithms are made separately.

A. Extensions of the Estimation Method

For flows under different congestion control algorithms, distributions of the flow completion time can be reached by making two changes to the model we proposed. The first change is to rewrite (1) according to the new congestion control description about retransmission schemes for a single packet, and then, derive the packet transmission latency distribution. The second change is to reanalyze the congestion window size evolution or redo experiments to find the time average congestion window size in (13), and then build a parallel queueing flow transmission model as we proposed. Then the packet transmission latency distribution and the flow completion time distribution under any congestion control algorithm can be derived.

B. Applications in Designing Scheduling and Congestion Control Algorithms

In designing scheduling algorithms for deadline aware networks, since the flow completion time is normally distributed, and deriving the estimated distribution is simple and accurate, it is natural to consider spread information (standard deviations) of the distributions in updating bandwidth for different flows, which in return, will improve the number of flows complete in time. In designing congestion control algorithms, the flow completion time distribution can be used as a requirement metric measuring the efficiency of the algorithm.

VIII. CONCLUSION

In this paper, exact single packet transmission latency distribution was calculated. The packet transmission latency referred to the time to successfully deliver the packet, including eventual retransmissions due to packet losses. A semi-Markov process was used to model the TCP congestion window size evolution. Then we formed a new parallel queueing model, which described how a flow is transmitted, using the packet transmission distribution and the congestion window size result from the semi-Markov model. From the parallel queueing model we derived the flow completion time distribution for a flow with fixed flow length. NS2 simulations showed our estimated packet transmission latency distributions and flow completion time distributions are very accurate. Our estimation model can be easily extended to analyze other congestion control algorithms, and our results can be directly used in designing scheduling and congestion control algorithms.

REFERENCES

[1] N. Dukkipati and N. McKeown, "Why flow-completion time is the right metric for congestion control," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 1, pp. 59–62, Jan. 2006.

[2] M. Handley *et al.,* "Re-architecting datacenter networks and stacks for low latency and high performance," in *Proc. ACM SIGCOMM*, 2017, pp. 29–42.
[3] J. Zhang, F. Ren, X. Yue, R. Shu, and C. Lin, "Sharing bandwidth by allocating switch buffer in data center networks," *IEEE J. Sel. Area Comm.*, vol. 32, no. 1, Jan. 2014.
[4] C. Lee, C. Park, K. Jang, S. Moon, and D. Han, "DX: Latency-based congestion control for datacenters," *IEEE/ACM Trans. Netw.*, vol. 25, no. 1, pp. 335–348, Feb. 2017.
[5] M. Alizadeh *et al.,* "Data center TCP (DCTCP)," in *Proc. ACM SIGCOMM*, 2010, pp. 63–74.
[6] R. Rojas-Cessa, Y. Kaymak, and Z. Dong, "Schemes for fast transmission of flows in data center networks," *IEEE Commun. Surv. Tut.*, vol. 17, no. 3, pp. 1391–1422, 3rd Quart., 2015.
[7] S. Zhang, Z. Qian, H. Wu, and S. Lu, "Efficient data center flow scheduling without starvation using expansion ratio," *IEEE Trans. Parall. Distr.*, vol. 28, no. 11, pp. 3157–3170, Nov. 2017.
[8] W. Bai *et al.,* "PIAS: Practical information-agnostic flow scheduling for commodity data centers," *IEEE/ACM Trans. Netw.*, vol. 25, no. 4, pp. 1954–1967, Aug. 2017.
[9] L. Chen, K. Chen, and W. Bai, "Scheduling mix-flows in commodity datacenters with Karuna," in *Proc. ACM SIGCOMM*, 2016, pp. 174–187.
[10] C. Wilson, H. Ballani, T. Karagiannis, and A. Rowstron, "Better never than late: Meeting deadlines in datacenter networks," in *Proc. ACM SIGCOMM*, 2011, pp. 50–61.
[11] C. Hong, M. Caesar, and P. B. Godfrey, "Finishing flows quickly with preemptive scheduling," in *Proc. ACM SIGCOMM*, 2012, pp. 127–138.
[12] P. R. Jelenković and J. Tan, "Characterizing heavy-tailed distributions induced by retransmissions," Department of Electrical Engineering, Columbia University, EE2007-09-07, Tech. Rep., Sept. 2007, eprint arXiv: 0709.1138v2.
[13] J. Tan, T. T. Swapna, and N. B. Shroff, "Retransmission delays with bounded packets: Power-law body and exponential tail," *IEEE/ACM Trans. Netw.*, vol. 22, no. 1, pp. 27–38. Feb. 2014.
[14] J. Nair, M. Andreasson, L. L. H. Andrew, S. H. Low, and J. C. Doyle, "On channel failures, file fragmentation policies, and heavy-tailed completion times," *IEEE/ACM Trans. Netw.*, vol. 24, no. 1, pp. 529–541, Feb. 2016.
[15] Cs. Király, M. Garetto, M. Meo, M. A. Marsan, and R. L. Cigno, "Analytical computation of completion time distributions of short-lived TCP connections," *Perform. Eval.*, vol. 59, pp. 179–197, 2005.
[16] R. Gaeta, M. Gribaudo, D. Manini, and M. Sereno, "On the use of Petri nets for the computation of completion time distribution for short TCP transfers," in *Proc. Springer Petri Nets*, 2003, pp. 181–200.
[17] N. Cardwell, S. Savage, and T. Anderson, "Modeling TCP latency," in *Proc. IEEE INFOCOM*, 2000, pp. 1742–1751.
[18] B. Sikdar, S. Kalyanaraman, and K. S. Vastola, "An integrated model for the latency and steady-state throughput of TCP connections," *Perform. Eval.*, vol. 46, pp. 139–154, 2001.
[19] B. Sikdar, S. Kalyanaraman, and K. S. Vastola, "Analytic models and comparative study of the latency and steady-state throughput of TCP Tahoe, Reno and SACK," in *Proc. IEEE GLOBECOM*, Nov. 2001, pp. 1781–1787.
[20] M. Mellia, I. Stoica, and H. Zhang, "TCP model for short lived flows," *IEEE Commun. Lett.*, vol. 6, no. 2, pp. 85–87, Feb. 2002.
[21] D. Ciullo, M. Mellia, and M. Meo, "Two schemes to reduce latency in short lived TCP flows," *IEEE Commun. Lett.*, vol. 13, no. 10, pp. 806–808, Oct. 2009.
[22] L. Ma, K. E. Barner, and G. R. Arce, "Statistical analysis of TCP's retransmission timeout algorithm," *IEEE/ACM Trans. Netw.*, vol. 14, no. 2, pp. 383–396, Apr. 2006.
[23] J. Padhye, V. Firoiu, D. F. Towsley, and J. F. Lurose, "Modeling TCP reno performance: A simple model and its empirical validation," *IEEE/ACM Trans. Netw.*, vol. 8, no. 2, pp. 133–145, Apr. 2000.
[24] S. M. Ross, *Introduction to Probability Models*, 10th Edition, Elsevier, 2011.

Gan Luan received her B.Sc. degree in Telecommunications Engineering from Beijing University of Posts and Telecommunications, Beijing, China, in 2012. She is currently working towards the Ph.D. degree in Communication and Information System at the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, China. Her current research interests include network performance analysis and radio-frequency identification.