
© Institute of Electrical and Electronics Engineers, Inc. 2002
Reprinted with permission.
First-Come-First-Served Packet Dispersion and Implications for TCP

Bob Melander  Mats Björkman
(Bob.Melander@docs.uu.se) (Mats.Bjorkman@docs.uu.se)

Per Gunningberg
(Per.Gunningberg@docs.uu.se)

Dept. of Computer Systems, Uppsala University
Box 325, SE-751 05 Uppsala, Sweden.

Abstract

We study the packet dispersion phenomenon that a traffic flow experiences when it passes through a router. We show that when there are competing flows and the router schedules packets first-come-first-served (FCFS), the dispersion is not described well by the bottleneck spacing effect. We therefore introduce the term FCFS-spacing effect. We also show that for a router implementing weighted fair queuing, dispersion is due either to the bottleneck spacing effect or the FCFS spacing effect. Finally, we discuss the implications of FCFS packet dispersion on TCP’s self-clocking mechanism.

1 Introduction

The question addressed in this paper is whether the scenarios depicted in Figures 1a) and b) are equivalent or not with respect to packet dispersion (i.e. the separation of packets in time)? In other words, is the dispersion of packets having traversed an empty bottleneck link with link bandwidth $l$ the same as the (average) dispersion of packets having traversed a link with link bandwidth $l' > l$ and available bandwidth $l$? In scenario b), competing traffic is consequently using $x = l' - l$ bps of the link capacity. Providing an answer to this question is important because packet dispersion plays a central role in many bandwidth measurement methods [3, 4, 6, 10, 11, 15] and in TCP’s self-clocking mechanism [8].

The dispersion phenomenon in Figure 1a) is usually referred to as the bottleneck spacing effect. One of the first to describe it was Jacobson [8]. In short, the bottleneck spacing effect dictates the following: Suppose that two packets

⋄ This work is supported in part by the SITI/Ericsson CONNECTED project.
of size $s_p$ and separated by $\Delta t_0$ are about to be sent across a link with link bandwidth $l$. Then, if $\Delta t_0 < \frac{s_p}{l}$, the packets will be separated by $\Delta t = \frac{s_p}{l}$ when they have traversed the link.

We will show that the two scenarios in Figure 1 are not equivalent when the router schedules packets according to a first-come-first-served (FCFS) discipline. Therefore, we will refer to the dispersion phenomenon in such a setting as the FCFS spacing effect. We also consider the weighted fair queueing scheduling discipline. In such a setting, we show that the packet dispersion is due either to the bottleneck spacing effect or the FCFS spacing effect.

Finally, we discuss the implications of the FCFS spacing effect for TCP’s self-clocking mechanism. This complements a study by Jain and Dovrolis [9], where the relation between TCP throughput and available bandwidth has been studied. For slow start we show that the dispersion of acknowledgements does not correspond to the available bandwidth. This is of importance for TCP variants [1, 7, 12] that attempt to estimate the available bandwidth based on those dispersion values, and use that estimate to set TCP’s $ssthresh$ appropriately. We also show that in congestion avoidance, the convergence of TCP’s send rate to available bandwidth is only asymptotic and requires several round-trips. This is in contrast to cases where the bottleneck spacing effect operates, where convergence is achieved in one round-trip.
2 FCFS Spacing Effect

When a first-come-first-served packet scheduling regime is used, incoming packets from different flows are (logically) queued together. The packets are then serviced in the order of their arrival. This means that flows are not separated from each other and that the behavior of one flow can affect all other flows. In particular, it implies that the more aggressive a flow is, the larger its share of the link capacity will be. If the flows are approximated as fluids, the share that a certain flow gets can therefore be expressed mathematically as

\[ p = \begin{cases} 
  o & \text{if } o < l - x \\
  \frac{o}{x+o} & \text{if } o \geq l - x
\end{cases} \] (1)

where \( o \) is the incoming (or offered) rate of the flow, \( p \) is the outgoing rate of the flow, \( x \) is the aggregate rate of the other flows and \( l \) is the link capacity.

Hence, in an underload situation (i.e. when the combined rates of the incoming flows are less than the link bandwidth), the share that a certain flow gets is equal to that flow’s offered rate. In an overload situation, a flow will instead get a share of the link capacity that is proportional to that flow’s fraction of the total offered rate.

To verify Equation 1 we have performed real measurements on a Linux router and simulations using the network simulator ns [13]. The results from the real measurements and simulations corroborate each other. Due to the limited space the graphs presented here are taken from the simulations. The topology used in the measurements is shown in Figure 2. Two routers, \( R \) and \( R' \) are interconnected by a link with capacity \( l \). Four sets of hosts are connected to each of the routers, with \( n_0 \) hosts in set 0, \( n_1 \) hosts in set 1, etc. Each host set belongs

![Figure 2](image-url)
to a separate flow class, although this division into classes is irrelevant for FCFS scheduling. In all measurements, \( n_0 = n_1 = n_2 = n_3 = 4 \). The competing (or cross) traffic is generated by on-off Pareto traffic generators (with the shape parameter set to the default value in ns, 1.5).

Figure 3 shows the measured bandwidths\(^1\), \( p \), from a set of measurements where the offered bandwidth, \( o \), for a UDP probe flow is increased linearly from 1 Mbps to 10 Mbps in steps of 0.5 Mbps. The size of the probe packets is \( s_p = 1400 \) bytes and the average size of the cross traffic packets is 500 bytes. For each rate level, \( o_i \), 100 packets are sent separated by \( \Delta t_0 = \frac{8 \cdot s_p}{o_i} \) seconds. The measured bandwidth is calculated as \( p = \frac{s_p}{\overline{\Delta t}} \), where \( \overline{\Delta t} \) is the mean value of the inter-packet times, \( \Delta t_i \), observed by the probe destination.\(^2\)

\[\text{Figure 3: The measured bandwidth as a function of offered bandwidth. The router schedules packets first-come-first-served.}\]

All dashed lines in the graph are the theoretical curves obtained using Equation 1. The two curves for the cases with no cross traffic (i.e. when \( x = 0 \) Mbps) illustrate the bottleneck spacing effect. The remaining five curves are for cases when there is cross traffic, i.e. \( x > 0 \) Mbps. These curves illustrate the FCFS spacing effect. The measured bandwidth equals the offered bandwidth up to the breakpoint where \( o = l - x \) as dictated by the first part of Equation 1. For the curves with no cross traffic that happens when \( o = 3 \) Mbps and \( o = 5 \) Mbps, respectively. When \( o \) is increased beyond the breakpoint value the second part of Equation 1 comes into effect. As can be seen in the graph, the curves from the measurements are in good agreement with the theoretic curves.

\(^1\)We use bandwidth and rate interchangeably.
\(^2\)\(i = 1, \ldots, 99\).
As pointed out earlier, Equation 1 assumes that flows are continuous. This is obviously not the case for network traffic flows, which are discrete (information is sent in packets). Hence, real flows will only approximate continuous flows. To study how packet size affects the validity of Equation 1 we have performed measurements where the packet sizes of the cross traffic and the probe traffic have been varied in a systematic way. Figure 4 shows the mean of $\Delta t_{\text{measured}}^i - \Delta t_{\text{theory}}$ for different offered probe rates when the probe packet size is 1400 bytes. The first term are the inter-packet times observed by the probe destination and the second term is the theoretic inter-packet time dictated by Equation 1, $\Delta t_{\text{theoretic}} = sp/p$. The confidence intervals at the 95% level are also drawn. Figures 8 and 9 in the appendix show similar graphs for probe packet sizes 700 bytes and 300 bytes, respectively.

These graphs show that the variance decreases as the packet size of the cross traffic gets smaller. This is not surprising, since smaller packets means that the cross traffic will more closely approximate continuous flows. That translates into improved agreement with Equation 1. Another observation is that, at the breakpoint probe rate ($l - x = 10 - 5 = 5$ Mbps), $\Delta t_{\text{measured}}^i$ is consistently somewhat larger than $\Delta t_{\text{theory}}$. We have not been able to explain this. A third observation is that the variance decreases when the probe packets become smaller. This would seemingly speak in favor of using small probe packet when probing for available bandwidth. That would contradict the results of previous studies where large packet were found to give the best accuracy [2,
However, as the probe packets get smaller, $\Delta t$ will also become smaller in order to achieve a certain rate (since, if $p = \frac{s_p}{w} = \frac{s'_p}{w'}$ and $s'_p < s_p$, then $\Delta t' < \Delta t$). Although not immediately apparent from the graphs, it turns out that the relative variance ($\frac{\text{var}(\Delta t)}{\Delta t}$) is less for large packets. From that perspective, our study also indicates that large probe packets are preferable to smaller ones.

If the mean $\Delta t_{\text{measured}} - \Delta t_{\text{theory}}$ differences are instead expressed in bandwidth, i.e. as the mean of $p_{i_{\text{measured}}} - p_{i_{\text{theory}}}$, we find that they are less than 0.5 Mbps in most cases. The exception is when $s_p = 300$ bytes where the differences are between 0.5 Mbps and 1.5 Mbps in a few cases. The overall conclusion in this section is that the theoretical predictions using Equation 1 and the measurement results agree well, especially when the cross traffic is composed of small packets.

## 3 Weighted Fair Queueing

In weighted fair queueing [5], flows are divided into a set of classes $\{C_i\}$ and each class is assigned a weight $\omega_i$. During any interval of time, class $C_i$ is guaranteed to receive a fraction of service (i.e. link capacity) at least equal to $\frac{\omega_i}{\sum_j \omega_j}$ given that there are class $C_i$ packets to send. Fair queueing is a special case of weighted fair queueing where all weights $\omega_i$ are equal.

As opposed to first-come-first-served scheduling, flows in different classes are separated from each other. They are (logically) put in different queues. This means that a flow cannot grab an arbitrary portion of the link capacity at the expense of other flows by being aggressive enough. This has consequences for packet dispersion; as illustrated in Figure 5. Here the measurements are similar to those in Figure 3 except that the scheduling regime in router $R$ is now weighted fair queuing. The guaranteed share for the class that the UDP probe flow belongs to ($C_0$) is $\frac{\omega_0}{\sum_j \omega_j} \cdot 1 = 0.4 \cdot 10 = 4$ Mbps. The two curves for which the actual share of class $C_0$ is 4 Mbps correspond to the case where all other classes fully use their guaranteed shares. The remaining two curves for which the actual share is 6.3 Mbps correspond to the case where classes $C_2$ and $C_3$ use less than their guaranteed share.

Consider first the case when the UDP probe flow is alone in its class. This case is represented by the two curves where $x = 0$ Mbps. As can be seen, the measured bandwidth equals the offered bandwidth as long as the offered bandwidth is less than class $C_0$’s share (4 Mbps and 6.3 Mbps, respectively). After that, the measured bandwidth cannot increase more and the curves thus flatten out. Note that this is exactly what the bottleneck spacing effect for links with link capacities 4 Mbps and 6.3 Mbps yields (although without the small variations that is due to the burstiness of the flows in the other classes) as shown by the two dashed curves.

If several flows belong to the same class $C$, the packets from these flows have
4 Implications for TCP

In this section we discuss what the implications of the FCFS spacing effect are for TCP. To date, numerous TCP variants have been proposed. We consider the one that has gained the most wide-spread use, TCP Reno. There are essentially two adaptation mechanisms in TCP\(^1\), *self-clocking* and *congestion window ad-

---

\(^1\)Henceforth, TCP Reno will be assumed when writing just “TCP”.

---

Figure 5: The measured bandwidth as a function of offered bandwidth. The router uses weighted fair queueing to schedule packets. Probe packet size, \(s_p = 1400\) bytes. The average cross traffic packet size is 500 bytes.
justment in response to packet losses. Packet dispersion is only of relevance for the self-clocking mechanism so that is what the rest of this presentation focuses on.

Self-clocking is a term introduced in Jacobson’s classic paper [8]. It basically refers to the fact that a TCP sender’s strobing of new segments into the network is triggered by the reception of acknowledgements. When an acknowledgement is received it also causes the congestion window (cwnd) to increase, in slow start by 1, in congestion avoidance by 1/cwnd. If the dispersion of the acknowledgements reflects the capacity of the network, then at steady-state (i.e. in TCP’s congestion avoidance phase), self-clocking will effectively cause a TCP to use the appropriate send rate (i.e. the rate that does not overload the network path traversed). Jacobson showed that the bottleneck spacing effect will separate the acknowledgements so that their dispersion correspond to the capacity of the bottleneck link. However, his discussion of self-clocking does not take into account the effect that competing flows have on packet dispersion. As showed in the previous sections, if the routers schedule packets FCFS, the dispersion phenomenon may then be different from the one dictated by the bottleneck spacing effect.

To continue the discussion we need the following lemma, which follows directly from Equation 1.

**Lemma 1** If Equation 1 holds for p and o then \( p \leq o \). \( \blacksquare \)

That is, the outgoing rate of a flow passing through a FCFS-router is (on average) never higher than the incoming rate of that flow. By using Lemma 1, it is not difficult to prove the following:

**Proposition 2** Consider a TCP Reno style feedback system where the input rate \( p_n \) at time step \( n \) equals the output rate from the previous time step, \( p'_{n-1} \), scaled by \( k \). Suppose that \( p'_n \) is related to \( p_n \) as shown below.

\[
\begin{cases} 
  p_n & \text{if } p_n \leq l - x \\
  \frac{p_n}{x + p_n} & \text{if } p_n \geq l - x 
\end{cases}
\]

Then \( \lim_{n \to \infty} |p_n - \min(p_0, k \cdot l - x)| = 0 \) where \( k = 2 \) in slow start and \( k = 1 + 1/cwnd \) in congestion avoidance. \( \blacksquare \)

Proposition 2 encapsulates the impact that both the bottleneck spacing effect and the FCFS spacing effect have on TCP’s self-clocking mechanism. Should there be no cross traffic (i.e. \( x = 0 \text{ Mbps} \)), the convergence is actually instantaneous due to the bottleneck spacing effect. Hence, already after the first

\[\text{Due to the limited space here we refer to [14] for all proofs.}\]
feedback round, the sending rate of TCP will be \( \min(p_0, k \cdot l) \). If \( x > 0 \) Mbps, the FCFS spacing effect makes the sending rate approach the fixed-point rate \( p_\infty = \min(p_0, k \cdot l - x) \) only asymptotically.

Using Lemma 1, the following proposition concerning TCP’s send rate during slow start can also be shown:

**Proposition 3** In a TCP Reno style feedback system, if \( l \leq p_0 \leq 2l - x \) then slow-start batch \( n = 1, 2, \ldots \) will be sent with a rate greater than or equal to the link bandwidth of the bottleneck link (i.e. back-to-back with respect to that link’s capacity).

To summarize Propositions 2 and 3 we have for slow start: If \( p_0 \leq 2l - x \), then TCP will keep sending packet batches that will arrive back-to-back at the bottleneck link. If \( p_0 > 2l - x \), \( p_i \) tends to \( 2l - x \), directly if \( x = 0 \), asymptotically if \( x > 0 \). These three cases are illustrated in Figure 6, where the sending rate of a TCP Reno (simulated in ns) is compared to the theoretical figures as given by the above equations. The simulation topology is the same as in previous sections. The end-to-end delay is 40 ms. The size of TCP segments is 1400 Bytes and the average size of the cross traffic packets is 500 Bytes.

![Figure 6: TCP Reno during slow-start with scenarios corresponding to the three cases described in the text.](image-url)

For the first scenario, \( l = 10 \) Mbps and \( x = 7 \) Mbps. Hence, \( p_0 < 2l - x \), and thus we expect \( p_\infty = p_0 = 10 \) Mbps. As can be seen from the figure, TCP behaves as expected. For the second scenario, \( l = 3 \) and \( x = 0 \). Here we expect the TCP sending rate to go directly to \( 2l = 6 \) Mbps, and in the figure this is what happens. For the last scenario, \( l = 3 \) Mbps and \( x = 2 \) Mbps. Here, we
expect the sending rate to approach $2l - x = 4$ Mbps, but only asymptotically. As can be seen, this also happens.

![Graph](image)

**Figure 7:** TCP congestion avoidance examples. The same three scenarios as in the previous figure.

As TCP enters the congestion avoidance phase, the value of $k$ will change, and thus also the fixed-point $p_\infty$. Hence, a new phase of adjustment will occur, where TCP finds a new sending rate corresponding to the new fixed-point. This is illustrated in Figure 7, using the same example bottlenecks as in the slow-start figure. For simplicity, we have in these measurements used an approximation of congestion avoidance where $k = 1$. Hence, the expected fixed-point rates would be 3 Mbps, 3 Mbps and 1 Mbps, respectively. As can be seen from Figure 7, the TCP sending rate goes only asymptotically towards the expected fixed-point rates when there is cross traffic, whereas for the example without cross traffic, the sending rate goes directly to the fixed-point rate.

## 5 Conclusions

We have shown that the packet dispersion experienced by a flow traversing a path with available bandwidth $l$ due to competing traffic is different from the packet dispersion the flow would experience traversing a path with bottleneck link bandwidth $l$ and no competing traffic. We call the spacing phenomenon in the former case, the first-come-first-served (FCFS) spacing effect to differentiate it from the bottleneck spacing effect in the latter case. We have showed that for weighted fair queuing-routers, the packet dispersion is either due to the FCFS spacing effect or the bottleneck spacing effect.
5 Conclusions

The implications of the FCFS spacing effect for TCP’s self-clocking are two-fold: 1) The dispersion of the acknowledgements causes TCP to over-estimate the available bandwidth. 2) The send rate of TCP only approaches the correct bandwidth during congestion avoidance, and then only asymptotically. This is in contrast to the cases where no cross traffic is present and where TCP finds the available bandwidth directly after the first round trip in congestion avoidance.

Appendix

The two figures discussed in section 2. The probe packet size is $s_p = 700$ bytes and $s_p = 300$ bytes, respectively.

Figure 8: $\Delta t_{\text{measured}} - \Delta t_{\text{theory}}$ as a function of cross traffic packet size with 95% confidence intervals. $l = 10$ Mbps, $x = 5$ Mbps. Probe packet size $s_p = 700$ bytes. a) Probe rate (i.e. offered rate) 10 Mbps. b) Probe rate 7 Mbps. c) Probe rate 5 Mbps. d) Probe rate 3 Mbps.
Figure 9: $\Delta t_{\text{measured}} - \Delta t_{\text{theory}}$ as a function of cross traffic packet size with 95% confidence intervals. $l = 10$ Mbps, $x = 5$ Mbps. Probe packet size $s_p = 300$ bytes. a) Probe rate (i.e. offered rate) 10 Mbps. b) Probe rate 7 Mbps. c) Probe rate 5 Mbps. d) Probe rate 3 Mbps.
References


