On the Characteristics and Origins of Internet Flow Rates

Yin Zhang Lee Breslau
AT&T Labs–Research
{yzhang,breslau}@research.att.com

Vern Paxson Scott Shenker
International Computer Science Institute
{vern,shenker}@icsi.berkeley.edu

ABSTRACT

This paper considers the distribution of the rates at which flows transmit data, and the causes of these rates. First, using packet level traces from several Internet links, and summary flow statistics from an ISP backbone, we examine Internet flow rates and the relationship between the rate and other flow characteristics such as size and duration. We find, as have others, that while the distribution of flow rates is skewed, it is not as highly skewed as the distribution of flow sizes. We also find that for large flows the size and rate are highly correlated. Second, we attempt to determine the cause of the rates at which flows transmit data by developing a tool, T-RAT, to analyze packet-level TCP dynamics. In our traces, the most frequent causes appear to be network congestion and receiver window limits.

Categories and Subject Descriptors

C.2.2 [**Computer Communication Networks**]: Network Protocols

General Terms

Measurement

Keywords

Network measurement, TCP, flow rates

1. INTRODUCTION

Researchers have investigated many aspects of Internet traffic, including characteristics of aggregate traffic [8, 16], the sizes of files transferred, traffic of particular applications [4] and routing stability [7, 17], to name a few. One area that has received comparatively little attention is the rate at which applications or flows transmit data in the Internet. This rate can be affected by any of a number of factors, including, for example, application limits on the rate at which data is generated, bottleneck link bandwidth, network congestion, the total amount of data the application has to transmit, whether or not the application uses congestion control, and host buffer limitations. An Internet link may well contain traffic aggregated from many flows limited by different factors elsewhere in the network. While each of these factors is well understood in isolation, we have very little knowledge about their prevalence and effect in the current Internet. In particular, we don't have a good understanding of the rates typically achieved by flows, nor are we aware of the dominant limiting factors.

A better understanding of the nature and origin of flow rates in the Internet is important for several reasons. First, to understand the extent to which application performance would be improved by increased transmission rates, we must first know what is limiting their transmission rate. Flows limited by network congestion are in need of drastically different attention than flows limited by host buffer sizes. Further, many router algorithms to control per-flow bandwidth algorithms have been proposed, and the performance and scalability of some of these algorithm depends on the nature of the flow rates seen at routers [9, 10, 14]. Thus, knowing more about these rates may inform the design of such algorithms. Finally, knowledge about the rates and their causes may lead to better models of Internet traffic. Such models could be useful in generating simulation workloads and studying a variety of network problems.

In this paper we use data from packet traces and summary flow level statistics collected on backbone routers and access links to study the characteristics and origins of flow rates in the Internet. Specifically, we examine the distribution of flow rates seen on Internet links, and investigate the relationship between flow rates and other characteristics of flows such as their size and duration. Given these macroscopic statistics, we then attempt to understand the causes behind these flow rates. We have developed a tool, called T-RAT, which analyzes traces of TCP connections and infers which causes among several possibilities limited the transmission rates of the flows.

Among our significant findings are the following. First, confirming what has been observed previously, the distribution of flow rates is skewed, but not as highly skewed as flow sizes. Second, we find, somewhat surprisingly, that flow rates strongly correlated with flow sizes. This is strong evidence that user behavior, as evidenced by the amount of data they transfer, is not intrinsically determined, but rather, is a function of the speed at which files can be downloaded. Finally, using our analysis tool on several packet traces, we find that the dominant rate limiting factors appear to be congestion and receiver window limits. We then

SIGCOMM'02, August 19-23, 2002, Pittsburgh, Pennsylvania, USA.

Trace	Date	Length	# Packets	Sampled	Bidirectional
Access1a	Jan. 16, 2001	2 hours	22 million	—	Yes
Access1b	Dec 13, 2001	30 minutes	5.5 million	—	Yes
Access1c	Jan. 3, 2002	1 hour	32 million	—	Yes
Access2	Jan. 2, 2002	1 day	10 million	1 in 256	Yes
Peering1	Jan. 24, 2001	45 minutes	34 million	—	No
Regional1a	Jan. 2, 2002	1 hour	1.2 million	1 in 256	No
Regional1b	Jan. 3, 2002	2 hours	2.3 million	1 in 256	No
Regional2	Jan. 3, 2002	2 hour	5 million	1 in 256	No

Table 1: Characteristics of 8 packet traces

examine the distribution of flow rates among flows in the same causal class (*i.e.,* flows whose rate is limited by the same factor).

While we believe our study is the first of its kind to examine the causes of Internet flow rates and relate these causes to other flow characteristics, it is by no means the last word in this area. This paper raises the question, but it leaves many issues unaddressed. However, the value in our work is a new tool that allows for further investigation of this problem, and an initial look at the answers it can provide.

Also, while we address flow rates from a somewhat different angle, our paper is not the first to study Internet flow rates. A preliminary look at Internet flow rates in a small number of packet traces found the distribution of rates to be skewed, but not as highly skewed as the flow size distribution [14]. This result was consistent with observation in [10] that a small number of flows accounted for a significant number of the total bytes. In recent work, Sarvotham et al [20] found that a single high rate flow usually accounts for the burstiness in aggregate traffic. In [2], the authors look at the distribution of throughput across connections between hosts and a web server and find that the rates are often consistent with a log-normal distribution. These papers have all made important observations. In this paper, we aim to go beyond this previous work, looking at flow rates making up aggregate traffic and attempting to understand their causes.

The rest of this paper is organized as follows. In the next section we describe the data sets and methodology used in this study. In Section 3 we present various statistics concerning flow rates and related measures. We then describe our rate analyzing tool in Section 4, describe our efforts to validate its performance in Section 5, and present results of applying it to packet traces in Section 6. We present some conclusions in Section 7.

2. DATASETS AND METHODOLOGY

We used data from two sources in our study. The first set of data consisted of 8 packet traces collected over a 14 month period. The traces were collected at high speed access links connecting two sites to the Internet; a peering link between two Tier 1 providers; and two sites on a backbone network. The latter 3 traces were sampled pseudo-randomly (using a hash on the packet header fields) at a rate of 1/256. Sampling was on a per-flow basis, so that all packets from a sampled flow were captured. The packet monitors at the access links saw all traffic going between the monitored sites and the Internet, so both directions of connections were included in the traces. For the other traces, because of asymmetric routing often only one direction of a connection is visible. The finite duration of the traces (30 minutes to 2 hours) introduces a bias against the largest and most long-lived flows. However, the effect of truncation on flow rates, the statistic in which we are most interested, should not be significant. The characteristics of the traces are summarized in Table 1.

We supplemented the packet level traces with summary flow level statistics from 19 backbone routers in a Tier 1 provider. Data was collected for 24 hours from the 19 routers on each of 4 days between July, 2000 and November, 2001, yielding 76 sets of data. Because this data was collected concurrently from routers in the same backbone provider, a single flow can be present in more than one of the datasets. We do not know how often this occurred. However, the 19 routers represent a relatively small fraction of the provider's routers, so we expect that each dataset contains a relatively unique set of flows.

Records in these datasets contain the IP addresses of the endpoints, port numbers, higher layer protocol, the start time and end time for the flow, the total number of packets and the total number of bytes. Since these datasets lack packet level details, we cannot use them for the trace analysis in Section 4. However, they provide a useful supplement to our results in Section 3, greatly broadening the scope of the data beyond the limited number of packet traces. Each of the 4 days of summary statistics represents between 4 and 6 billion packets and between 1.5 and 2.5 terabytes of data.

Flows can be defined by either their source and destination addresses, or by addresses, port numbers and protocol. The appropriateness of a definition depends in part on what one is studying. For instance, when studying router definitions that do per-flow processing, the former definition may be appropriate. When examining the characteristics of individual transport layer connections the latter is preferred. For the results reported in this paper, we used the 5-tuple of IP addresses, port numbers, and protocol number. We also generated results defining flows by source and destination IP addresses only. Those results are not qualitatively different. Also, for the results presented here, we used a 60 second timeout to decide that an idle flow has terminated. Repeating the tests with a 15 second timeout again did not qualitatively affect the results.

In the analysis that follows we report on some basic per-flow statistics, including flow size, duration and rate. Size is merely the aggregate number of bytes transferred in the flow (including headers), and duration is the time elapsed between the first and last packets of a flow. Flow rate is also straightforward (size divided by duration) with the exception that determining a flow rate for very short flows is

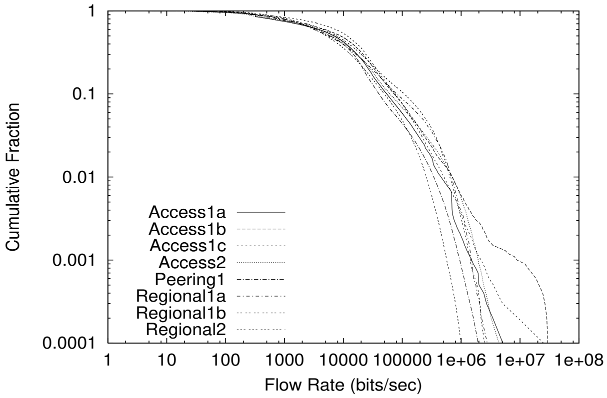

Figure 1: Complementary distribution of flow rates

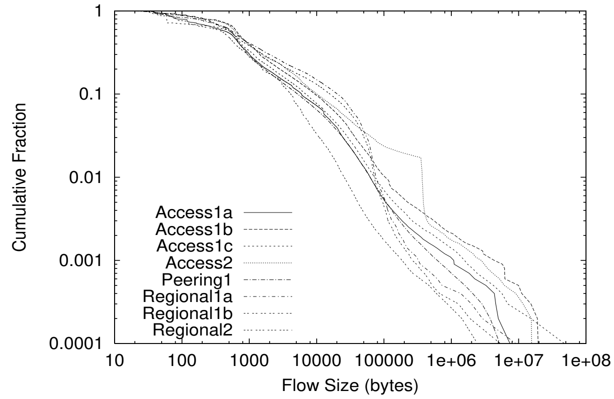

Figure 2: Complementary distribution of flow sizes

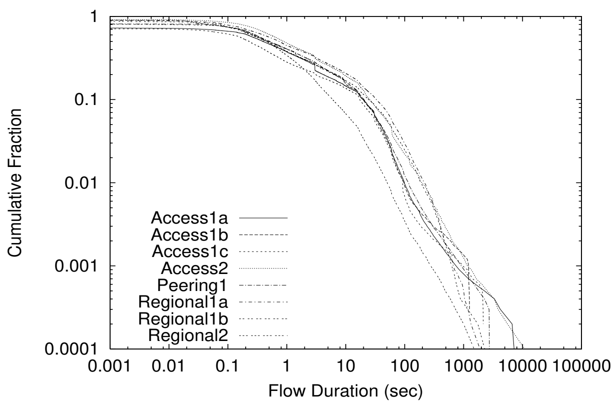

Figure 3: Complementary distribution of flow duration

problematic. In particular, rate is not well-defined for single packet flows whose duration by definition is zero. Similarly, flows of very short (but non-zero) duration also present a problem. It does not seem reasonable to say that a 2-packet flow that sends these packets back-to-back has an average rate equal to the line rate. In general, since we are most interested in the rate at which applications transmit data, when calculating rates we ignore flows of duration less than 100 msec, since the timing of these flows' packets may be determined as much by queueing delays inside the network as by actual transmission times at the source.

3. CHARACTERISTICS

In this section we examine the characteristics of Internet flows. We begin by looking at the distributions of rate, size and duration, before turning to the question of relationships among them. Throughout, we start with data from the packet traces, and then supplement this with the summary flow data.

3.1 Rate Distribution

Figure 1 plots the complementary distribution of flow rates, for flows lasting longer than 100 msec, in the 8 packet traces. The distributions show that average rates vary over several orders of magnitude. Most flows are relatively slow, with average rates less than 10kbps. However, the fastest flows in each trace transmit at rates above 1Mbps; in some traces the top speed is over 10Mbps. For comparison, we also show the complementary distributions of flow size and duration in Figures 2 and 3, respectively. The striking difference here is the longer tail evident in the distributions of flow sizes for the packet traces. One possible explanation of this difference is that file sizes are potentially unbounded while flow rates are constrained by link bandwidths.

A previous study of rate distributions at a web server suggested that the rate distributions were well described by a log-normal distribution [2]. To test that hypothesis, we use the quantile-quantile plot (Q-Q plot) [3] to compare the flow rate distribution with analytical models. The Q-Q plot determines whether a data set has a particular theoretical distribution by plotting the quantiles of the data set against the quantiles of the theoretical distribution. If the data comes from a population with the given theoretical distribution, then the resulting scatter plot will be approximately a straight line. The Q-Q plots in Figures 4 and 5 compare the log of the rate distribution to the normal distribution for two of the traces (Access1c and Regional2). The fit between the two is visually good. As in Reference [2], we further assess the goodness-of-fit using the Shapiro-Wilk normality test [5]. For Access1c (Figure 4), we can not reject the null hypothesis that the log of rate comes from normal distribution at 25% significance level; for Regional2 (Figure 5), we can not reject normality at any level of significance. This suggests the fit for a normal distribution is indeed very good. Applying the Shapiro-Wilk test on all the packet traces and flow summary data, we find that for 60% of the data sets we can not reject normality at 5% significance level. These results give evidence that the flow rates can often be described with a log-normal distribution.

The next question we address is how important the fast flows are. In particular, how much of the total bytes transferred are accounted for by the fastest flows? Note that a skewed rate distribution need not imply that fast flows account for a large fraction of the bytes. This will depend on the size of fast flows. Figure 6 plots the fraction of bytes accounted for in a given percentage of the fastest flows for the 8 packet traces. We see that in general, the 10% fastest flows account for between 30% and 90% of all bytes transferred, and the 20% fast flows account for between 55% and 95%. This indicates that while most flows are not fast, these fast flows do account for a significant fraction of all traffic. Figure 7 shows results for the summary flow data. This

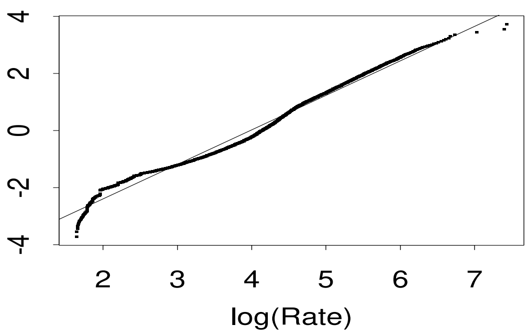

Figure 4: Q-Q plot for Access1c trace

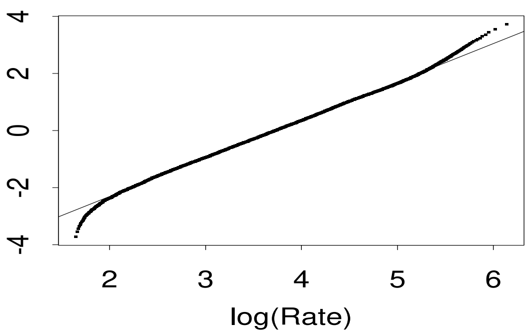

Figure 5: Q-Q plot for Regional2 trace

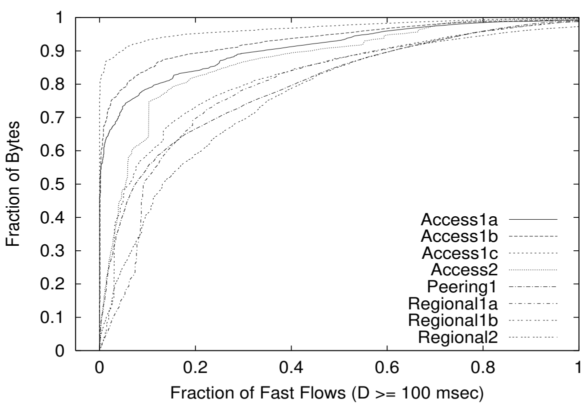

Figure 6: Fraction of bytes in fastest flows

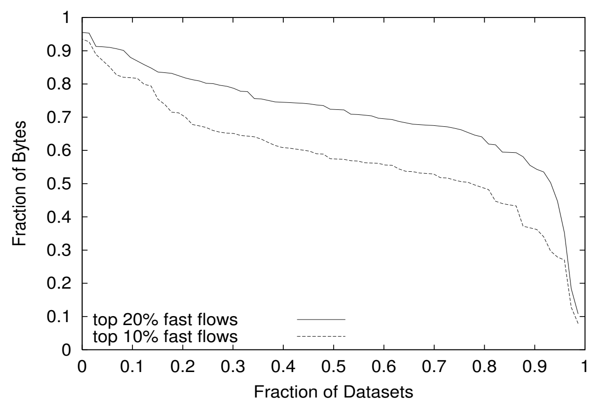

Figure 7: Distribution of the fraction of bytes in the 10% and 20% fastest flows for summary flow data.

figure plots the distribution of the percentage of bytes accounted for by the 10% and 20% fastest flows across the 76 sets of data. The skewed distributions exhibited in the traces are evident here as well. For example, in over 80% of the datasets, the fastest 10% of the flows account for at least 50% of the bytes transferred. Similarly, the fastest 20% of the flows account for over 65% of the bytes in 80% of the datasets. For comparison, the fraction of bytes in the *largest* flows (not shown) is even greater.

We now characterize flows along two dimensions: big or small, and fast or slow. We chose 100 KByte as a cutoff on the size dimension and 10 KByte/sec on the rate dimension. These thresholds are arbitrary, but they provide a way to characterize flows in a two-by-two taxonomy.

Table 2 shows the fraction of flows and bytes in each of the 4 categories for the packet traces. Flows that are small and slow, the largest group in each trace, account for between 44% and 63% of flows. However, they account for a relatively small fraction of the total bytes (10% or less.) There are also a significant number of flows in the small-fast category (between 30% and 40%) but these too represent a modest fraction of the total bytes (less than 10% in all but one trace.) On the other hand, there are a small number of flows that are both big and fast (generally less than 10%). These flows account for the bulk of the bytes transferred—at least 60% in all of the traces, and over 80% in many of them. The big-slow category is sparsely populated and these flows account for less than 10% of the bytes. Data for the 76 sets of summary flow statistics (not shown here) are generally consistent with the packet trace results.

One question about Internet dynamics is the degree to which traffic is dominated (in different ways) by small flows. In terms of the number of flows, there is little doubt that the vast majority are indeed small. More than 84% of the flows in all of our traces (and over 90% in some of them) meet our (arbitrary) definition of small. However, before we conclude that the Internet is dominated by these small flows and that future designs should be geared towards dealing with them, we should remember that a very large share of the bytes are in big and fast flows. In 6 of the 8 traces we examined, these flows comprised over 80% of the bytes. Thus, when designing mechanism to control congestion or otherwise deal with traffic arriving at a router, these big and fast flows are an important (and sometimes dominant) factor.

3.2 Correlations

We next examine the relationship between the flow characteristics of interest. Table 3 shows 3 pairs of correlations—duration and rate, size and rate, and duration and size—for the 8 packet traces. We computed correlations of the log of these data because of the large range and uneven distribution. We restricted the correlations to flows with durations longer than 5 seconds. Results for the other flow definitions are similar.

The correlations are fairly consistent across traces, and show a negative correlation between duration and rate, a

	Small-Slow		Small-Fast		Big-Slow		Big-Fast	
Trace	flows	bytes	flows	bytes	flows	bytes	flows	bytes
Access1a	0.602	0.025	0.326	0.031	0.016	0.043	0.057	0.901
Access1b	0.436	0.016	0.468	0.024	0.02	0.022	0.076	0.938
Access1c	0.528	0.006	0.395	0.01	0.018	0.01	0.059	0.974
Access2	0.518	0.018	0.381	0.024	0.021	0.073	0.08	0.885
Peering1	0.581	0.07	0.354	0.066	0.017	0.1	0.048	0.764
Regional1a	0.506	0.056	0.345	0.05	0.027	0.078	0.122	0.816
Regional1b	0.463	0.044	0.406	0.05	0.023	0.068	0.108	0.837
Regional2	0.626	0.103	0.341	0.193	0.007	0.092	0.026	0.612

Table 2: Fraction of flows and bytes in Small/Slow, Small/Fast, Big/Slow and Big/Fast flows.

Trace	logD,logR	logS,logR	logD,logS
Access1a	-0.366	0.870	0.140
Access1b	-0.187	0.883	0.296
Access1c	-0.319	0.877	0.175
Access2	-0.319	0.885	0.159
Peering1	-0.319	0.847	0.235
Regional1a	-0.453	0.842	0.100
Regional1b	-0.432	0.835	0.136
Regional2	-0.209	0.877	0.287

Table 3: Correlations of size, rate and duration in 8 packet traces

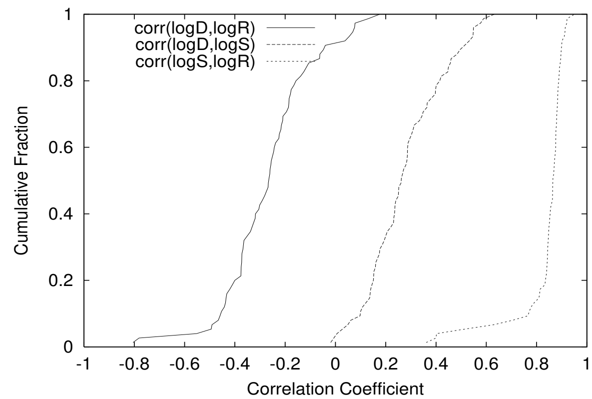

Figure 8: CDF of correlations of size, rate and duration across all datasets

slight positive correlation between size and duration and a strong correlation between the size and rate. The correlation between rate and size is also evident in other subsets of flows. For flows longer than 1 second, the correlations range from .65 to .77. For flows lasting longer than 30 seconds, the correlations range from .90 to .95.

Figure 8 shows CDFs of the 3 correlations taken across each of our datasets (packet traces and summary flow level statistics). This figure shows that the general trend exhibited in the packet traces was also evident in the summary flow data we examined.

The most striking result here is the correlation between size and rate. If users first decided how much data they wanted to transfer (*e.g.,* the size of a file) independent of the network conditions, and then sent it over the network, there would be little correlation between size and rate,[1] a strong correlation between size and duration, and a strongly negative correlation between rate and duration. This is not what we see; the negative correlation between rate and duration is fairly weak, the correlation between size and duration is very weak, and the correlation between size and rate is very strong. Thus, users appear to choose the size of their transfer based, strongly, on the available bandwidth. While some adjustment of user behavior was to be expected, we were surprised at the extent of the correlation between size and rate.

[1]TCP slow-start could cause some correlation between rate and size. In order to assess the impact of slow-start on the correlations we observed, we eliminated the first 1 second of all flows and recomputed the correlations. For flows lasting longer than 5 seconds, the resulting correlations between size and rate in the 8 traces ranged from .87 to .92, eliminating slow-start as a significant cause of the correlation.

4. TCP RATE ANALYSIS TOOL

In the previous section we looked at flow rates and their relationship to other flow characteristics. We now turn our attention to understanding the origins of the rates of flows we observed. We restrict our attention to TCP flows for two reasons. First, TCP is used by most traffic in the Internet [21]. Second, the congestion and flow control mechanisms in TCP give us the opportunity to understand and explain the reasons behind the resulting transmission rates. In this section we describe a tool we built, called T-RAT (for TCP Rate Analysis Tool) that examines TCP-level dynamics in a packet trace and attempts to determine the factor that limits each flow's transmission rate.

T-RAT leverages the principles underlying TCP. In particular, it uses knowledge about TCP to determine the number of packets in each flight and to make a rate limit determination based on the dynamics of successive flights. However, as will become evident from the discussion below, principles alone are not sufficient to accomplish this goal. By necessity T-RAT makes use of many heuristics which through experience have been found to be useful.

Before describing how T-RAT works, we first review the requirements that motivate its design. These include the range of behavior it needs to identify as well as the environment in which we want to use it.

The rate at which a TCP connection transmits data can be determined by any of several factors. We characterize the possible rate limiting factors as follows:

- Opportunity limited: the application has a limited amount of data to send and never leaves slow-start.

This places an upper bound on how fast it can transmit data.

- Congestion limited: the sender's congestion window is adjusted according to TCP's congestion control algorithm in response to detecting packet loss.
- Transport limited: the sender is doing congestion avoidance, but doesn't experience any loss.
- Receiver window limited: the sending rate is limited by the receiver's maximum advertised window.
- Sender window limited: the sending rate is constrained by buffer space at the sender, which limits the amount of unacknowledged data that can be outstanding at any time.
- Bandwidth limited: the sender fully utilizes, and is limited by, the bandwidth on the bottleneck link. The sender may experience loss in this case. However, it is different from congestion limited in that the sender is not competing with any other flows on the bottleneck link. An example would be a connection constrained by an access modem.
- Application limited: the application does not produce data fast enough to be limited by either the transport layer or by network bandwidth.

We had the following requirements in designing T-RAT. First, we do not require that an entire TCP connection, or even its beginning, be observed. This prevents any bias against long-lived flows in a trace of limited duration. Second, we would like the tool to work on traces recorded at arbitrary places in the network. Thus, the analyzer may only see one side of a connection, and it needs to work even if it was not captured near either the sender or receiver. Finally, to work with large traces, our tool must work in a streaming fashion to avoid having to read the entire trace into memory.

T-RAT works by grouping packets into flights and then determining a rate limiting factor based on the behavior of groups of adjacent flights. This entails three main components: *(i)* estimating the Maximum Segment Size (MSS) for the connection, *(ii)* estimating the round trip time, and *(iii)* analyzing the limit on the rate achieved by the connection. We now describe these components in more detail. As mentioned above, T-RAT works with either the data stream, acknowledgment stream, or both. In what follows, we identify those cases when the algorithm is by necessity different for the data and the acknowledgment streams.

4.1 MSS Estimator

The analysis requires that we have an estimate of the MSS for a connection. When the trace contains data packets, we set the MSS to the largest packet size observed. When the trace contains only acknowledgments, estimating the MSS is more subtle, since there need not be a 1-to-1 correspondence between data and acknowledgment packets. In this case, we estimate the MSS by looking for the *most frequent common divisor*. This is similar to the greatest common divisor, however, we apply heuristics to avoid looking for divisors of numbers of bytes acknowledged that are not multiples of the MSS.

4.2 Round Trip Time Estimator

In this section, we present a general algorithm for estimating RTT based on packet-level TCP traces. RTT estimation is not our primary goal but, rather, a necessary component of our rate analyzer. As such, we ultimately judge the algorithm not by how accurately it estimates RTT (though we do care about that) but by whether it is good enough to allow the rate analyzer to make correct decisions.

There are three basic steps to the RTT estimation algorithm. First, we generate a set of candidate RTTs. Then for each candidate RTT we assess how good an estimate of the actual RTT it is. We do this by grouping packets into flights based on the candidate RTT and then determining how consistent the behavior of groups of consecutive flights is with identifiable TCP behavior. Then we choose the candidate RTT that is most consistent with TCP. We expand on each of these steps below.

We generate 27 candidates, T_k, between 0.003 sec and 3 sec, where $T_k = 0.003 * 1.3^k$ sec. This covers the range of round trip times we would normally expect anywhere beyond the local network.

Assume we have a stream of packets, P_i, each with arrival time T_i and an inter-arrival interval $\Delta P_i = T_i - T_{i-1}$. For a candidate RTT, we group packets into flights as follows. Given the first packet, P_0, in a flight, we determine the first packet in the next flight by examining ΔP_i for all packets with arrival times between $T_0 + RTT$ and $T_0 + fac \cdot RTT$, where *fac* is a factor to accommodate variation in the round trip time. We identify the packet P_1 with the largest inter-arrival time in this interval. We also examine P_2, the first packet that arrives after $T_0 + fac \cdot RTT$. If $\Delta P_2 \geq 2 \cdot \Delta P_1$, we choose P_2 as the first packet of the next flight. Otherwise, we choose P_1.

There is an obvious tradeoff in the choice of *fac*. We need *fac* to be large enough to cover the variation of RTT. However, setting *fac* too large will introduce too much noise, thereby reducing the accuracy of the algorithm. Currently, we set *fac* to 1.7, which is empirically optimal among 1.1, 1.2, ..., 2.0.

Once a set of flights, F_i ($i \geq 0$) has been identified for a candidate RTT, we evaluate it by attempting to match its behavior to that of TCP. Specifically, we see whether the behavior of successive flights is consistent with slow-start, congestion avoidance, or response to loss. We elaborate on how we identify each of these three behaviors.

Testing for Packet Loss: When the trace contains data packets, we infer packet loss by looking for retransmissions. Let seq_B be the largest sequence number seen before flight F. We can conclude that F has packet loss recovery (and a prior flight experienced loss) if and only if we see at least one data packet in F with upper sequence number less than or equal to seq_B. For the acknowledgment stream, we infer packet loss by looking for duplicate acknowledgments. Like TCP, we report a packet loss whenever we see three duplicate acknowledgments. In addition, if a flight has no more than 4 acknowledgment packets, we report a packet loss whenever we see a single duplicate. The latter helps to detect loss when the congestion window is small, which often leads to timeouts and significantly alters the timing characteristics. These tests are robust to packet reordering as long as it does not span flight boundaries or cause 3 duplicate acknowledgments.

Testing for Congestion Avoidance: Given a flight F, define its *flight size*, S_F, in terms of the number of MSS packets it contains:

$$S_F = \left\lceil \frac{seq_F^+ - seq_F^-}{MSS} \right\rceil$$

where seq_F^- is the largest sequence number seen before F, and seq_F^+ is the largest sequence number seen before the next flight. We define a flight's *duration* D_F as the lag between the arrival of the first packet of F and the first packet in the subsequent flight.

Testing whether four consecutive flights[2] F_i $(i = 0, 1, 2, 3)$ are consistent with congestion avoidance requires determining whether the flight sizes, S_{F_i}, exhibit an additive increase pattern. The test is trivial when the receiver acknowledges every packet. In this case, we only need to test whether $S_{F_{i+1}} - S_{F_i} = 1$ holds for $i = 0, 1, 2$.

The test is considerably more complex with delayed acknowledgments. In this case, the sizes of successive flights need not increase by 1. Because only every other packet is acknowledged, the sender's congestion window increases by 1 on average every second flight. Further, because the flight size is equal to the sender's window minus unacknowledged packets, the size of successive flights may decrease when the acknowledgment for last packet in the prior flight is delayed. Hence, sequences of flight sizes like the following are common:

$$n, \underbrace{n+1, n+1}_{\Delta S=0}, \underbrace{n+2, n+1}_{\Delta S=-1}, n+2, n+3, n+4, n+3, ...$$

In our algorithm, we consider flights $F_i (i = 0, 1, 2, 3)$ to be consistent with congestion avoidance if and only if the following three conditions are met:

1. $-2 \leq S_{F_i} - predicted_i \leq 2$ for $i = 1, 2, 3$, where $predicted_i = \max_{0 \leq k < i}(S_{F_k})$ is the predicted number of segments in flight F_i.
2. The flight sizes are not too small and have an overall non-decreasing pattern. Specifically, we apply the following three tests. *(i)* $S_{F_i} \geq 2$ for $i = 0, 1, 2, 3$; *(ii)* $S_{F_3} \geq \max(3,\ S_{F_0})$; *(iii)* $\max F_i \geq 4$.
3. The flight durations are not too different. More specifically, $\max D_{F_i} \leq 4 \cdot \min D_{F_i}$.

The first condition above captures additive increase patterns with and without delayed acknowledgments. The second and third conditions are sanity checks.

Testing for Slow-Start: As was the case with congestion avoidance, TCP dynamics differ substantially during slow-start with and without delayed acknowledgments. We apply different tests for each of the two cases and classify the behavior as consistent with slow-start if either test is passed.

To capture slow-start behavior without delayed acknowledgments, we only need to test whether $S_{F_{i+1}} = 2 \cdot S_{F_i}$ holds for $i = 0, 1, 2$.

The following test captures slow-start dynamics when delayed acknowledgments are used. We consider flights $F_i (i = 0, 1, 2, 3)$ to be consistent with slow-start behavior if and only if the following two conditions are met:

1. $-3 \leq S_{F_i} - predicted_i \leq 2$ for $i = 1, 2, 3$, where $predicted_i = S_{F_{i-1}} + ACK_{F_{i-1}}$ is the predicted number of segments in flight F_i, $ACK_{F_{i-1}}$ is the estimated number of non-duplicate acknowledgment packets in flight F_{i-1}. (For an acknowledgment stream, $ACK_{F_{i-1}}$ can be counted directly. For a data stream, we estimate $ACK_{F_{i-1}}$ as $\lfloor S_{F_{i-1}}/2 \rfloor$.)
2. The flight sizes are not too small and have an overall non-decreasing pattern. Specifically, we apply the following tests: *(i)* $S_{F_i} \geq S_{F_{i-1}}$ for $i = 1, 2, 3$; *(ii)* $S_{F_3} \geq \max(3,\ 2 \cdot S_{F_0})$.

The first condition captures the behavior of slow-start with and without delayed acknowledgments. The second and third are sanity checks.

Analyzing TCP Dynamics: Having described how we identify slow-start and congestion avoidance, and how we detect loss, we now present our algorithm for assessing how good a set of flights, F_i, generated for a candidate RTT is. Let c be the index of the current flight to be examined. Let s be the state of the current flight: one of *CA*, *SS* or *UN*, for congestion avoidance, slow-start and unknown, respectively. Initially, $s = UN$. For a given flight, F_c, we determine the state by examining F_c, F_{c+1}, F_{c+2} and F_{c+3} and applying the following state transitions.

- $s = CA$:
  - If there is loss in at least one of the 4 flights, then s transitions to *UN*.
  - If the 4 flights show additive increase behavior as described above then we remain in state *CA*.
  - Similarly, we also remain in state *CA* even if we don't recognize the behavior. As with TCP, we only leave *CA* if there is packet loss.
- $s = SS$:
  - If there is loss in at least one of the 4 flights, then s transitions to *UN*.
  - If the 4 flights are consistent with multiplicative increase, then s remains *SS*.
  - Otherwise, s transitions to *UN*.

  Note that we can leave state *SS* when there is packet loss or there is some flight we do not understand.
- $s = UN$:
  - If there is loss in at least one flight, s remains *UN*.
  - If the four flights are consistent with the multiplicative increase behavior then s transitions to *SS*.
  - If the four flights are consistent with additive increase, s transitions to *CA*.
  - Otherwise, s remains *UN*.

[2]As the discussion below on delayed acknowledgments indicates, and as confirmed by experience, four consecutive flights is the smallest number that in most cases allows us to identify the behavior correctly.

As we analyze the set of flights, we sum up the number of packets in flights that are either in *CA* or *SS*. We assign this number as the score for a candidate *RTT* and select the candidate with the highest score.

We have made several refinements to the algorithms described above, which we briefly mention here but do not describe further. First, when testing for slow-start or congestion avoidance behavior, if an initial test of a set of flights fails, we see whether splitting a flight into two flights or coalescing two adjacent flights yields a set of flights that matches the behavior in question. If so, we adjust the flight boundaries. Second, to accommodate variations in RTT, we continually update the RTT estimate using an exponentially weighted moving average of the durations of successive flights. Third, in cases where several candidate RTTs yield similar scores, we enhance the algorithm to disambiguate these candidates and eliminate those with very large or very small RTTs. This also allows us to reduce the number of candidate RTTs we examine.

4.3 Rate Limit Analysis

Using the chosen RTT, we apply our rate limit analysis to determine the factor limiting a flow's transmission rate. Since conditions can change over the lifetime of a flow, we continually monitor the behavior of successive flights. We periodically check the number of packets seen for a flow. Every time we see 256 packets, or when no packets are seen for a 15 second interval, we make a rate limit determination. We now describe the specific tests we apply to determine the rate limiting factor.

Bandwidth Limited: A flow is considered bandwidth limited if it satisfies either of the following two tests. The first is that it repeatedly achieves the same amount of data in flight prior to loss. Specifically, this is the case if: *(i)* there were at least 3 flights with retransmissions; and *(ii)* the maximum and minimum flight sizes before the loss occurs differ by no more than the MSS.

The second test classifies a flow as bandwidth limited if it sustains the link bandwidth on its bottleneck link. Rather than attempting to estimate the link bandwidth, we look for flows in which packets are nearly equally-spaced. Specifically, a flow is considered bandwidth limited if $T_{hi} < 2*T_{lo}$, where T_{lo} is the 5^{th} percentile of the inter-packet times[3] and T_{hi} is the P^{th} percentile. We set $P = max(95, 100*(1-.75\cdot\frac{\#flights}{\#packets}))$. P must be a function of the flight size. Otherwise, we risk classifying sender and receiver window limited flows that have large flight sizes as bandwidth limited.

Congestion Limited: A flow is considered congestion limited if it experienced loss and it does not satisfy the first test for bandwidth limited.

Receiver Window Limited: We can only determine that a flow is receiver window limited when the trace contains acknowledgments since they indicate the receiver's advertised window. We determine a flow to be receiver window limited if we find 3 consecutive flights $F_i (i = 1, 2, 3)$, with flight sizes $S_i \cdot MSS > \text{awnd}_{\max} - 3 \cdot MSS$, where $\text{awnd}_{\max}$ is the largest receiver advertised window size. The difference of $3 * \text{MSS}$ is a heuristic that accommodates variations due to delayed acknowledgments and assumes that the MSS need not divide the advertised window evenly.

Sender Window Limited: Let $S_{F_{med}}$ and $S_{F_{80}}$ be the median and the 80^{th} percentile of the flight sizes. A flow is considered sender window limited if the following three conditions are met. First, the flow is not receiver window limited, congestion limited, or bandwidth limited. Second, $S_{F_{80}} < S_{F_{med}} + 3$. Finally, there are four consecutive flights with flight sizes between $S_{F_{80}} - 2$ and $S_{F_{80}} + 1$.

Opportunity Limited: A flow is deemed opportunity limited if the total number of bytes transferred is less than $13 * MSS$ or if it never exits slow-start. The limit of 13 is needed because it is difficult to recognize slow-start behavior with fewer than 13 packets.

Application Limited: A flow is application limited if a packet smaller than the MSS was transmitted followed by a lull greater than the RTT, followed by additional data.

Transport Limited: A flow is transport limited if the sender has entered congestion avoidance, does not experience any loss, and the flight size continues to grow.

T-RAT is not able to identify unambiguously the rate limiting behaviors in all cases. Therefore, the tool reports two additional conditions.

Host Window Limited: The connection is determined to be limited by either the sender window or the receiver window, but the tool cannot determine which. When acknowledgments are not present and the flow passes the sender window limited test above, it is classified as host window limited.

Unknown Limited: The tool is unable to match the connection to any of the specified behaviors.

5. VALIDATION

Before using T-RAT to analyze the rate limiting factors for TCP flows in our packet traces, we first validated it against measurement data as well as simulations. Specifically, we compared T-RAT's round trip time estimation to estimates provided by `tcpanaly` [15] over the NPD $\mathcal{N}_2$ [18] dataset.[4] Accurate RTT estimation is a fundamental component of the tool since making a rate-limit determination is in most cases not possible without being able to group packets into flights. Once we validated the RTT estimation, we then needed to determine whether the rate analyzer returned the right answer. Validating the results against actual network traffic is problematic. After all, that is the problem we intend to solve with this tool. Thus, we validated T-RAT against packet traces produced by network simulations and by controlled network experiments in which we could determine the specific factors that limited each flow's transmission rate.

5.1 RTT validation

The NPD $\mathcal{N}_2$ dataset contains packet traces for over $18,000$ TCP connections. We used $17,248$ of these in which packets were captured at both ends of the connections, so the dataset contains data and acknowledgment packets recorded at both the sender and receiver. We ran `tcpanaly` over this data and

[3]In fact, because delayed acknowledgments can cause what would otherwise be evenly spaced packets to be transmitted in bursts of 2, we cannot use the inter-packet times directly in this calculation. For data packets, instead of using the inter-arrival distribution, ΔP_i, directly, we use $\Delta P_i' = max(\Delta P_i, \Delta P_{i+1})$.

[4]`tcpanaly` requires traces of both directions of a connection. Therefore, we can use it to validate our tool using 2-way traces, but it cannot address the RTT estimation problem when only a single direction of the connection is available.

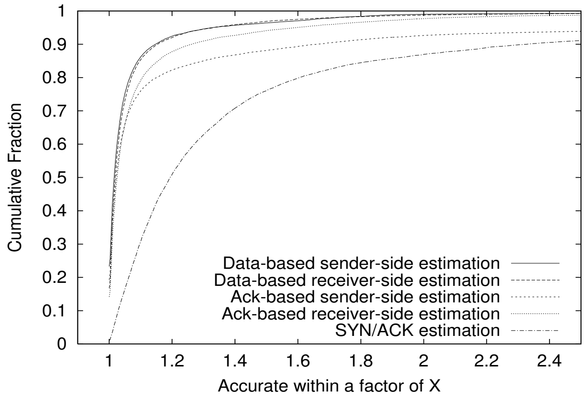

Figure 9: RTT validation against NPD $\mathcal{N}_2$ data

recorded for each connection the median of the RTT estimates it produced. We used these medians to compare to the performance of the RTT estimation of T-RAT.

Even though the NPD data includes both directions of connections, we tested our RTT estimation using only a single direction at a time (since the algorithm is designed to work in such cases.) Hence, we consider separately the cases in which the tool sees the data packets at the sender, acknowledgment packets at the sender, data packets at the receiver and acknowledgment packets at the receiver. For each RTT estimate computed by T-RAT we measure its accuracy by comparing it to the value produced by `tcpanaly`.

The results of the RTT validation are shown in Figure 9, which plots the CDF of the ratio between the two values for each of the 4 cases. The figure shows that with access to the data packets at either the sender or the receiver, for over 90% of the traces the estimated RTT is accurate within a factor of 1.15, and for over 95% of the traces the estimated RTT is accurate within a factor of 1.3. Accuracy of RTT estimates based on the acknowledgment stream, while still encouraging, is not as good as data stream analysis. In particular, with ack-based analysis at the receiver, 90% of estimates are accurate within a factor of 1.3 and 95% of traces are accurate within a factor of 1.6. Using the sender-side acknowledgment stream, estimates are accurate within a factor of 1.6 about 90% of the time. We suspect that delayed acknowledgments may be in part responsible for the inferior RTT estimation using the acknowledgment stream. By reducing the number of packets observable per RTT and perturbing the timing of some packets, they may make the job of RTT estimation more difficult. Further, we speculate that the sender side performance with acknowledgments also suffers because the acknowledgments at the sender have traversed an extra queue and are therefore subject to additional variation in the network delays they experience.

Previous studies have used the round trip time for the initial TCP SYN-ACK handshake as an estimate of per-connection round trip time [6, 11]. We also compared this value to the median value produce by `tcpanaly`. As shown in Figure 9, this estimate is significantly worse than the others. In general, the SYN-ACK handshake tends to underestimate the actual round trip time.

The overall results produced by our tool are encouraging. They show that RTT estimation works reasonably well in most cases. The real question, however, is how the rate analyzer works. Are the errors in RTT estimation small enough to allow the tool to properly determine a rate limiting factor, or do the errors prevent accurate analysis? We now turn to the question of the validity of the rate limiting factors.

5.2 Rate Limit Validation

We validated the rate limit results of T-RAT using both simulations and experiments in a controlled testbed. In our simulations, we used the *ns* simulator [13]. By controlling the simulated network and endpoint parameters we created TCP connections that exhibited various rate limiting behaviors. For example, congested limited behavior was simulated using several infinite source FTP connections traversing a shared bottleneck link, and application limited behavior was simulated using long-lived Telnet sessions. Our simulations included approximately 400 connections and 340,000 packets. T-RAT correctly identified the proper rate limiting behavior for over 99% of the connections.

While these simulations provided positive results about the performance of T-RAT, they suffered from several weaknesses. First, we were not able to validate all of the rate limiting behaviors that T-RAT was designed to identify. In particular, the TCP implementation in *ns* does not include the advertised window in TCP packets, preventing experiments that exhibited receiver window limited behavior. Second, the simulations varied some relevant parameters, but they did not explore the parameter space in a systematic way. This left us with little knowledge about the limits of the tool. Finally, simulations abstract away many details of actual operating system and protocol performance, leaving questions about how the tool would perform on real systems.

To further validate the performance of T-RAT we conducted experiments in a testbed consisting of PCs running the FreeBSD 4.3 operating system. In these experiments, two PCs acting as routers were connected by a bottleneck link. Each of these routers was also connected to a high speed LAN. Hosts on these LANs sent and received traffic across the bottleneck link. We used the `dummynet` [19] facility in the FreeBSD kernel to emulate different bandwidths, propagation delays and buffer sizes on the bottleneck link. We devised a series of experiments intended to elicit various rate limiting behaviors, captured packet traces from the TCP connections in these experiments using `tcpdump`, analyzed these traces using T-RAT, and validated the results reported by T-RAT against the expected behavior. Unless otherwise noted, the bandwidth, propagation delay, and buffer size on the emulated link were 1.5 Mbps, 25 msec, and 20 KBytes, respectively. We used an MTU of 540 bytes on all interfaces, allowing us to explore a wider range of window sizes (in terms of packets) than would be afforded with a larger MTU.

For some of the rate limiting behaviors, we captured TCP connections on both unloaded and loaded links. In order to produce background load, we generated bursts of UDP traffic at exponentially distributed intervals. The burst size was varied from 1 to 4 packets across experiments, and the average inter-burst interval was 30 msec, generating 10%, 20%, 30% and 40% load on the link. This was not intended to model realistic traffic. Rather the intention was to perturb the timing of the TCP packets and assess the effect of this perturbation on the ability of T-RAT to identify correctly the rate limiting behavior in question.

Experiments were repeated with and without delayed acknowledgments. All TCP packets were captured at both endpoints of the connection. We tested T-RAT using only a single direction of a connection at a time (either data or acknowledgment) to emulate the more challenging scenario of only observing one direction of a connection. Thus, for each connection we made four independent assessments using data packets at the source, data packets at the destination, acknowledgment packets at the source, and acknowledgment packets at the destination.

For each behavior we varied parameters in order to assess how well T-RAT works under a range of conditions. Our exploration of the relevant parameter space is by no means exhaustive, but the extensive experiments we conducted give us confidence about the operation of the tool.

In the vast majority of cases T-RAT correctly identified the dominant rate limiting factor. That is, for a given connection, the majority of periodic determinations made by T-RAT were correct. Further, for many connections, all of the periodic determinations were correct. In what follows, we summarize the experiments and their results, focusing on those cases that were most interesting or problematic.[5]

Receiver Window Limited: In these experiments, the maximum advertised receiver window was varied (by adjusting the receiver's socket buffer) for each connection, while the sender's window was larger than the bandwidth delay product of the link (and hence did not impact the sender's window.) The parameters of the bottleneck link were such that a window size of 18 packets saturated the link. We tested window sizes between 2 and 20 packets with no background load. Even when the link was saturated, there was sufficient buffering to prevent packet loss. With background load, we only tested window sizes up to 10 packets to avoid loss due to congestion. A 5 MByte file was transferred for each connection.

T-RAT successfully identified these connections as receiver window limited (using the acknowledgement stream) and host window limited (using the data stream) in most cases. Using the data stream, it did not correctly identify window sizes of 2 packets as receiver window limited. It is not possible to disambiguate this case from a bandwidth limited connection captured upstream of the bottleneck link when delayed acknowledgments are present. In both cases, the trace shows periodic transmission of a burst of 2 packets followed by an idle period. We would not expect receiver window limits to result in flight sizes of 2 packets, so we are not concerned about this failure mode.

T-RAT was able to identify a wide range of window sizes as receiver window limited (or host window limited using data packets.) As the number of packets in flight approaches the saturation point of the link, and as a consequence the time between successive flights approaches the inter-packet time, identifying flight boundaries becomes more difficult. When the tool had access to the data stream, it correctly identified the window limit until the link utilization approached 80%-90% of the link bandwidth. Beyond that it identified the connection as bandwidth limited. With access to the acknowledgment stream, the tool correctly identified the behavior as receiver window limited until the link was fully saturated.

As we applied background traffic to the link, the dominant cause identified for each connection was still receiver window limited for acknowledgement and host window limited for data packets. However, for each connection T-RAT sometimes identified a minority of the periodic determinations as transport limited when it had access to the data packets. With access to the acknowledgment packets, virtually all of the periodic determinations were receiver window limited. Thus, the advertised window information available in acknowledgments made T-RAT's job easier.

Sender Window Limited: These experiments were identical to the previous ones with the exception that in this case it was the sender's maximum window that was adjusted while the receiver window was larger than the bandwidth delay product of the bottleneck link.

The results were very similar to the those in the receiver window limited experiments. The tool was again unable to identify flight sizes of 2 packets as sender window limited (which in practice should not be a common occurrence.) T-RAT was able to identify window sizes as large as 80-90% of the link bandwidth as sender window limited. Beyond that it had trouble differentiating the behavior from bandwidth limited. Finally, as background load was applied to the link, the tool still correctly identified the most common rate limiting factor for each connection, though it sometimes confused the behavior with transport limited.

Transport Limited: To test transport limited behavior, in which the connection does congestion avoidance while not experiencing loss, we set the bottleneck link bandwidth to 10 Mbps and the one-way propagation delay to 40 msec, allowing a window size of more than 180 packets (recall we used a 540 byte MTU). In addition, we set the initial value of *ssthresh* to 2000 bytes, so that connections transitioned from slow-start to congestion avoidance very quickly. With no background traffic, each connection transferred a 4 MByte file. Without delayed acknowledgments, the window size reached about 140 packets (utilizing 75% of the link) before the connection terminated. When we tested this behavior in the presence of background load, each connection transferred a 2.5 MByte file and achieved a maximum window size of approximately 100 packets (without delayed acknowledgments). The smaller file size was chosen to prevent packet loss during the experiments. The experiments were repeated 10 times for each set of parameters.

T-RAT successfully identified transport limited as the dominant rate limiting cause for each connection. It made errors in some of the periodic determinations, with the errors becoming more prevalent as the burst size of the background traffic increased. Whenever T-RAT was unable to determine the correct rate limiting behavior, its estimate of the RTT was incorrect. However, correct RTT estimation is not always necessary. In some cases, the tool was robust enough to overcome errors in the RTT estimation and still determine the proper rate limiting behavior. In assessing transport limited behavior, T-RAT was more successful using data packets than acknowledgment packets, particularly when delayed acknowledgments were used. In contrast to the receiver window limited case above, the acknowledgment packets provide no additional information, and by acknowledging only half of the packets, T-RAT has less information with which to work.

Bandwidth Limited: In these experiments, a 10 MByte file was transferred across the bottleneck link with no competing traffic. The router buffer was large enough to avoid packet loss, and the sender and receiver windows were large

[5]More detailed information about the results is available at http://www.research.att.com/projects/T-RAT/.

enough to allow connections to saturate the link. We tested bottleneck link bandwidths of 500 Kbps, 1.5 Mbps, and 10 Mbps, with and without delayed acknowledgments. Each experiment was repeated 10 times.

In the vast majority of cases, T-RAT properly identified the rate limiting behavior. There are two points to make about these results. First, the RTT estimation produced by the tool was often incorrect. For a connection that fully saturates a bottleneck link, and is competing with no other traffic on that link, the resulting packet trace consists of stream of evenly spaced packets. There is, therefore, little or no timing information with which to accurately estimate RTT. Nonetheless, the test for bandwidth limiting behavior depends primarily on the distribution of inter-packet times and not on proper estimation of the flight size, so the tool still functions properly. The second observation about these experiments is that the connections were not exclusively bandwidth limited. Rather, they started in congestion avoidance (*ssthresh* was again set to 2000 bytes) and opened the congestion window, eventually saturating the link. The tool identified the connections as initially transport limited, and then as bandwidth limited once the bottleneck link was saturated. Visual inspection of the traces revealed that the tool made the transition at the appropriate time. In a few instances, the tool was unable to make a rate limiting determination during the single interval in which the connection transitioned states, and deemed the rate limiting behavior to be unknown.

Congestion Limited: Congestion limited behavior was tested by transferring 5 MByte files across the bottleneck link with random packet loss induced by `dummynet`. Tests were repeated with both 2% and 5% loss on the link in a single direction and in both directions. As with our other experiments, we repeated tests with and without delayed acknowledgments, and we repeated 5 transfers in each configuration.[6]

In nearly all cases, T-RAT identified these connections as congestion limited across all loss rates, acknowledgment strategies, and directionality of loss. For a very small number of the periodic assessments, connections were deemed transport limited. However, a connection that does not experience any loss over some interval will be in congestion avoidance mode and will be appropriately deemed transport limited. Visual inspection of a sample of these instances showed that this was indeed the case.

Opportunity Limited: In these experiments, we varied the amount of data transferred by each connection from 1 to 100 packets. The connection sizes and link parameters were such that the sources never left slow-start. However, at the larger connection sizes, the congestion window was large enough to saturate the link. Hence, while the source remained in slow-start, this was not always obvious when examining packet traces.

We first review the results without delayed acknowledgments. Using the trace of data packets at the source, T-RAT correctly identified all of the connections as opportunity limited. In the other 3 traces, T-RAT identified between 83 and 88 of the connections as opportunity limited. Most of the failures occurred at connection sizes greater than 80 packets, with a few occurring between 40 and 60 packets. None occurred for connection sizes less than 40 packets. When it failed, T-RAT deemed the connections either transport or bandwidth limited. These cases are not particularly troubling, as the window sizes are larger than we would expect to see with regularity in actual traces. With delayed acknowledgments, T-RAT reached the right conclusion in 399 out of 400 cases, failing only for a single connection size with acknowledgments at the receiver.

Application Limited: Characterizing and identifying application limited traffic is perhaps more challenging than the other behaviors we study. The test T-RAT uses for application limited traffic is based on heuristics about packet sizes and inter-packet gaps. However, there are certainly scenarios that will cause the tool to fail. For example, an application that sends constant bit rate traffic in MSS-sized packets will likely be identified as bandwidth limited. Further, since this traffic is by definition limited by the application our tool needs to recognize a potentially wider range of behaviors than with the other limiting factors. Understanding the range of application limited traffic in the Internet remains a subject for future study.

In our effort to validate the current tests for application limited traffic in T-RAT we had the application generate application data units (ADUs) at intervals separated by a random idle times chosen from an exponential distribution. We tested connections with average idle times of 1, 2, 3, 10, 20, 30, 50 and 100 msec. Furthermore, rather than generating MSS-sized ADUs as in our other experiments, we chose the size of the ADUs from a uniform distribution between 333 and 500 bytes, the latter being the MSS in our experiments. The resulting application layer data generation rates would have been between 3.3 Mbps (1 msec average idle time) and 33 kbps (100 msec idle time) without any network limits. In our case (1.5 Mbps bottleneck bandwidth) the highest rates would certainly run into network limits. Since we did not use MSS-sized packets, the resulting network layer traffic depended on whether or not the TCP Nagle algorithm [12], which coalesces smaller ADUs into MSS-sized packets, is employed. Hence, in addition to repeating experiments with and without delayed acknowledgments, we also repeated the experiments with and without the Nagle algorithm turned on.

Assessing the results of these experiments was difficult. Given that we used a stochastic data generation process, and that one cannot know *a priori* how this random process will interact with the transport layer, we could not know what the resulting network traffic would look like. Without a detailed packet-by-packet examination, the best we can do is to make qualitative characterizations about the results.

With the Nagle algorithm turned on, T-RAT characterized the two fastest data generation rates (3.3 Mbps and 1.65 Mbps) as a combination of congestion and bandwidth limited. This is what one would expect given the bottleneck link bandwidth. At the lowest data rates (33 Kbps and 66 Kbps) T-RAT deemed the traffic to be application limited. This is again consistent with intuition. In between, (from 110 Kbps to 1.1 Mbps) the traffic was characterized vari-

[6] We also performed the more obvious experiment in which multiple TCP connections were started simultaneously with loss induced by the competing TCPs. However, an apparent bug in the version of TCP we used sometimes prevented a connection from ever opening its congestion window after experiencing packet loss. Validating these results was more difficult since the TCP connections experienced a range of rate limiting factors (congestion, host window, transport.) Nonetheless, visual inspection of those results also indicated that the tool was properly identifying cases of congestion.

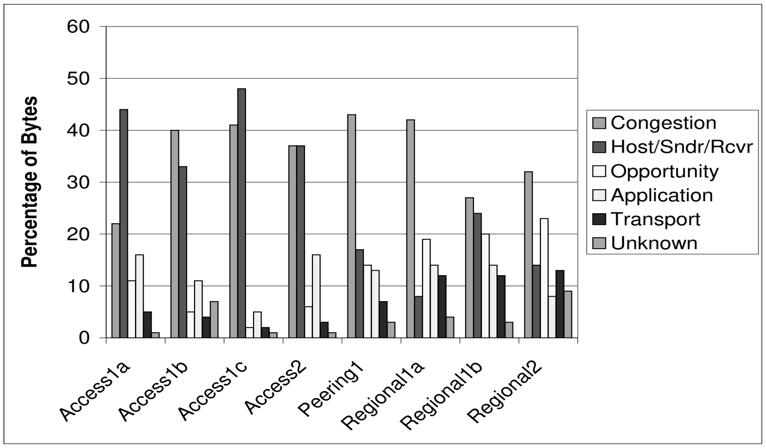

Figure 10: Fraction of bytes for each rate limiting factor

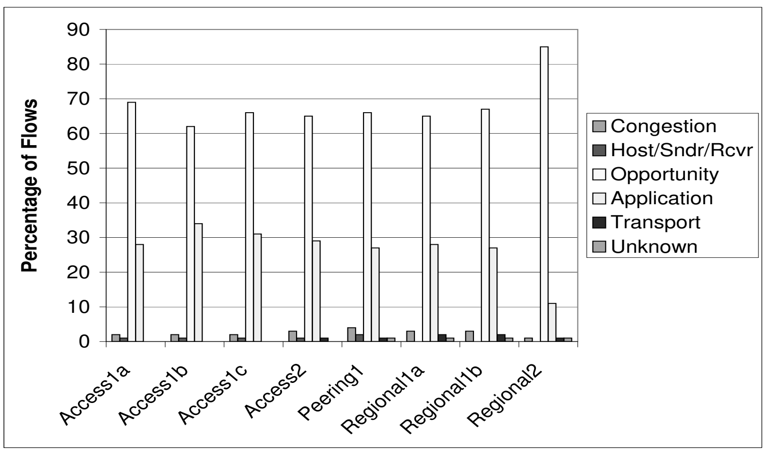

Figure 11: Fraction of flows for each rate limiting factor

ously as transport, host window, application, or unknown limited.

With the Nagle algorithm disabled, the fastest generation rates were again characterized as congestion and bandwidth limited. At all the lower rates (1.1 Mbps down to 33 Kbps), T-RAT deemed the connections as exclusively application limited when using the data stream, and a combination of application and transport limited when using the acknowledgment stream. Thus, application limited behavior is easier to discern when the Nagle algorithm is turned off.

6. RESULTS

The results of applying T-RAT to the 8 packet traces are shown in Figure 10. For each trace, the plot shows the percentage of bytes limited by each factor. The 4 traces taken from access links are able to differentiate between sender and receiver limited flows since they see data and acknowledgment packets for all connections. The peering and regional traces, on the other hand, often only see one direction of a connection and are therefore not always able to differentiate between these two causes. We have aggregated the 3 categories identified by T-RAT—sender, receiver and host window limited—into a single category labeled "Host/Sndr/Rcvr" limited in the graph.

As shown in Figure 10, the most common rate limiting factor is congestion. It accounts for between 22% and 43% of the bytes in each trace, and is either the first or second most frequent cause in each trace. The aggregate category that includes sender, receiver and host window limited is the second most common cause of rate limits accounting for between 8% and 48% of the bytes across traces. When we were able to make a distinction between sender and receiver window limited flow (*i.e.,* when the trace captured the acknowledgment stream), receiver window limited was a much more prevalent cause than sender window limited, by ratios between 2:1 and 10:1. Other causes—opportunity

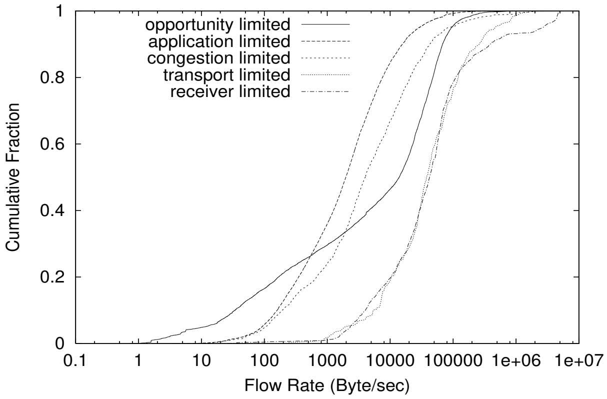

Figure 12: Rate distribution by rate limiting factor, Access1b trace

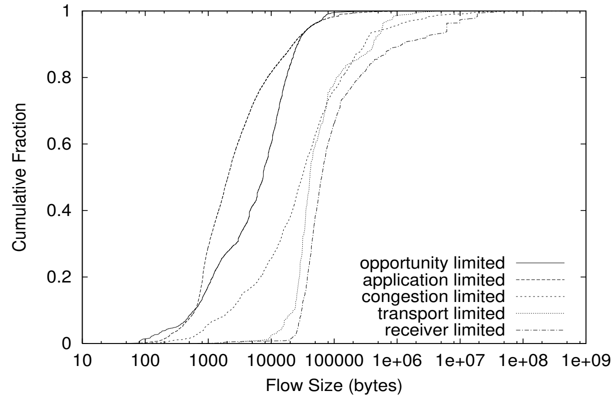

Figure 13: Size distribution by rate limiting factor, Access1b trace

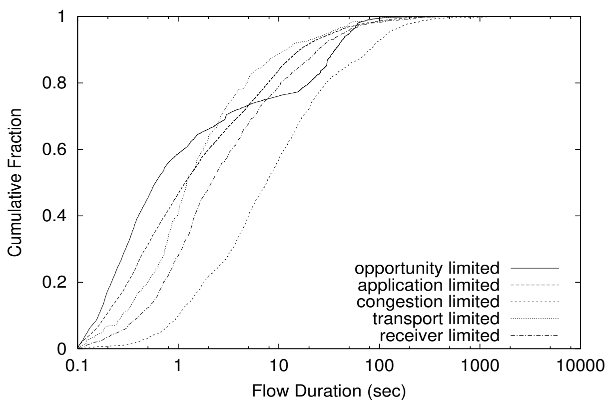

Figure 14: Duration distribution by rate limiting factor, Access1b trace

limited, application limited and transport limited—usually accounted for less than 20% of the bytes. Bandwidth limited flows accounted for less than 2% of the bytes in all traces (and are not shown in the plot). For most traces, the unknown category accounted for a very small percentage of the bytes. We examined the 2 traces in which the rate limiting cause for more than 5% of the bytes was unknown and identified 3 factors that prevented the tool from making a rate limiting determination. First, T-RAT cannot accurately estimate round trips on the order of 3 msec or less, and therefore, cannot determine a rate limiting factor for these connections. Second, when the traces were missing packets in the middle of a connection (which may have resulted either from loss at the packet filter or from multi-path routing) estimating the round trip time and the rate limiting cause becomes difficult. Finally, multiple web transfers across a persistent TCP connection also presented problems. When one HTTP transfer uses the congestion window remaining from the end of the previous transfer a moderate size file may not be opportunity limited (because it is larger than 13 packets and it never enters slow-start) and it may not have enough flights (because the initial flight size is large) for T-RAT to make a rate limit determination.

Not surprisingly, when we look at rate limiting factors by flows rather than bytes, the results are very different. Recall that we continuously update the reason a flow is limited, and a single flow may have multiple limiting factors throughout its lifetime. For example, it may be congestion limited for one interval, and after congestion dissipates, become window limited. In those cases when a flow experienced multiple causes, we classified it by the factor that most often limited its transmission rate. Figure 11 shows the percentage of flows constrained by each rate limiting factor for the 8 traces. The most common *per-flow* factors are opportunity and application limited. Collectively, they account for over 90% of the flows in each of the 8 traces, with opportunity limited accounting for more than 60% and application limited accounting for between 11% and 34%. No other cause accounted for more than 4% of the flows in any trace. These results are consistent with the results reported in Section 3. Namely, most flows are small and slow. Small flows are likely opportunity limited (they don't have enough packets to test buffer or network limits), and slow flows are likely application limited (not sending fast enough to test buffer or network limits.)

A general trend is evident when comparing the traces taken at access links to those taken at the peering and regional links. The former tend to have a higher percentage of bytes that are window limited. The access links are high speed links connecting a site to the Internet. As such, they support a population with good connectivity to the Internet. The other links are likely seeing a more general cross section of Internet users, some of whom are well-connected and others of whom are not. Since window limits are reached when the bandwidth delay product exceeds buffer resources, the well-connected users are more likely to reach these limits. This difference between the two kinds of traces was evident in Figure 1. That graph shows that the distribution of rates has a longer tail for the access links than for the regional and peering links.

We next ask whether these different rate limiting factors can be associated with different performance for users. Figure 12 plots the CDF of the rates for each of the rate limiting factors for the Access1b trace. The graph shows very distinct differences between subgroups. Overall, receiver limited and transport limited flows have the largest average rates, followed by congestion limited, application limited and opportunity limited. This same trend was exhibited across the other 7 traces. Figures 13 and 14 plot the distributions of size and duration for each rate limiting factor in the Access1b trace. Receiver limited flows have the largest size distribution, followed by transport and congestion limited. In the duration distribution, congestion limited flows have the longest duration, which is consistent with the observation that flows experiencing congestion will take longer to transmit their data than flows not experiencing congestion.

7. CONCLUSION

The rates at which flows transmit data is an important and not well-understood phenomenon. The rate of a flow can have a major impact on user experience, and the rates of flows traversing the Internet can have a significant effect on network control algorithms. We had two goals in this paper. First, we wanted to better understand the characteristics of flow rates in the Internet. Using packet traces and summary flow statistics we examined the rates of flows and the relationship between flow rates and other flow characteristics. We found that fast flows are responsible for most of the bytes transmitted in the Internet, so understanding their behavior is important. We also found a strong correlation between flow rate and size, suggesting an interaction between bandwidth available to a user and what the user does with that bandwidth.

Our second goal was to provide an explanation of the reasons why flows transmit at the rates they do. This was an ambitious goal, and we have by no means completely answered this question. We have seen, for a set of Internet packet traces, the reasons that flows are limited in their transmission rates and have looked at differences among different categories of flows. We believe our main contribution, however, is to open up an area of investigation that can lead to valuable future research. The tool we developed to study rate limiting behavior provides a level of analysis of TCP connections that can answer previously unanswerable questions. Thus, our tool has applicability beyond the set of results we have obtained with it thus far.

Acknowledgments

We would like to thank Rui Zhang for her work with us on flow characterization that helped launch this project. We would also like to express our thanks to the anonymous reviewers for many helpful comments on this paper.

8. REFERENCES

[1] M. Allman, "A Web Server's View of the Transport Layer," *Computer Communication Review*, 30(5), Oct. 2000.

[2] H. Balakrishnan, S. Seshan, M. Stemm, and R. Katz, "Analyzing Stability in Wide-Area Network Performance," In *Proc. ACM SIGMETRICS' 97*, June 1997.

[3] J. Chambers, W. Cleveland, B. Kleiner, and P. Tukey, "Graphical Methods for Data Analysis," Wadsworth Int'l. Group, Belmont, CA, 1983.

[4] M. Crovella and A. Bestavros, "Self-similarity in World Wide Web Traffic: Evidence and Possible Causes," *IEEE/ACM Transactions on Networking*, 5(6):835-846, December, 1997.

[5] R. D'Agostino and M. Stephens, Eds., "Goodness-of-Fit Techniques," Marcel Dekker, New York, 1986.

[6] H. Jiang and C. Dovrolis, "Passive Estimation of TCP Round-Trip Times," To appear in *Computer Communications Review*, July, 2002.

[7] C. Labovitz, G. Malan, and F. Jahanian, "Internet Routing Instability," *IEEE/ACM Transactions on Networking*, 6(5), pp. 515–528, 1998.

[8] W. Leland, M. Taqqu, W. Willinger, and D. Wilson, "On the Self-Similar Nature of Ethernet Traffic (Extended Version)," *IEEE/ACM Transaction on Networking*, 2(1), pp. 1-15, Feb. 1994.

[9] D. Lin and R. Morris, "Dynamics of Random Early Detection," in *ACM Sigcomm 97*, September, 1997.

[10] R. Mahajan, S. Floyd, and D. Wetherall, "Controlling High-Bandwidth Flows at the Congested Router," In *Proc. 9th International Conference on Network Protocols (ICNP)*, Nov. 2001.

[11] H. Martin, A. McGregor, and J. Cleary, "Analysis of Internet Delay Times," In *Proc. Passive and Active Measurements (PAM) workshop*, 2000.

[12] J. Nagle, "Congestion Control in IP/TCP Internetworks", RFC 896, January, 1984.

[13] ns—Network Simulator (Version 2.1b8). `http://www.isi.edu/nsnam/ns/`

[14] R. Pan, L. Breslau, B. Prabhakar, and S. Shenker, "Approximate Fairness through Differential Dropping," ACIRI Technical Report, 2001. `http://www.icir.org/shenker/afd-techreport.ps`

[15] V. Paxson, "Automated Packet Trace Analysis of TCP Implementations," In *Proc. ACM SIGCOMM '97*, Sep. 1997.

[16] V. Paxson, and S. Floyd, "Wide-Area Traffic: The Failure of Poisson Modeling," *IEEE/ACM Transactions on Networking*, 3(3), pp. 226-244, June 1995.

[17] V. Paxson, "End-to-End Routing Behavior in the Internet," *IEEE/ACM Transactions on Networking*, 5(5), pp. 601–615, Oct. 1997.

[18] V. Paxson, "End-to-End Internet Packet Dynamics," *IEEE/ACM Transactions on Networking*, 7(3), pp. 277-292, Jun. 1999.

[19] L. Rizzo, "Dummynet: A Simple Approach to the Evaluation of Network Protocols", *Computer Communications Review*, 27(1), Jan. 1997.

[20] S. Sarvotham, R. Riedi, and R. Baraniuk, "Connection-level Analysis and Modeling of Network Traffic," In *Proc. ACM Internet Measurement Workshop (IMW' 2001)*, Nov. 2001.

[21] K. Thompson, G. Miller, and R. Wilder, "Wide Area Internet Traffic Patterns and Characteristics," *IEEE Network*, vol. 11, no. 6, pp. 10-23, Nov. 1997.