On the Sensitivity of Network Simulation to Topology

Kostas G. Anagnostakis†, Raphael S. Ryger⋆, Michael B. Greenwald†

⋆ CS Department, Yale University
P.O. Box 208285, New Haven CT, USA
ryger@cs.yale.edu

†CIS Department, University of Pennsylvania
200 S. 33rd Street, Philadelphia PA 19104, USA
{*anagnost,mbgreen*}*@dsl.cis.upenn.edu*

Abstract

While network simulations for congestion control studies have often varied traffic loads and protocol parameters, they have typically investigated only a few topologies. The most common is by far the so-called "barbell" topology. In this paper we argue, first, that the barbell topology is not representative of the Internet. In particular, we report that a measurable fraction of packets pass through multiple congestion points. Second, we argue that the distinction between the "barbell" topology and more complex topologies is relevant by presenting a scenario with multiple congestion points that exhibits behavior that seems unexpected based on intuition derived from the barbell topology (in particular, a TCP-only system that exhibits behavior technically considered "congestion collapse"). We make the larger argument that the typical methodology currently accepted for evaluating network protocols is flawed. Finally, we briefly comment on some issues that arise in designing a simulation methodology that will be better suited to comparison of network protocol performance.

Keywords: network simulation, topology, congestion control, benchmarking, measurement

1 Introduction

No prudent engineer will deploy a new network protocol on a large scale until *after* the protocol has been validated in some way. The ultimate validation is performance measurement of the protocol deployed in a running system of reasonable scale and over a long duration. But because this deployment cannot occur without some previous validation, most analysis and validation of proposed network protocols or queue management schemes must depend, at least initially, upon simulation.

Simulation models for congestion control have typically investigated a small number of topologies. The most common, by far, is the so-called barbell (or "dumb-bell") topology: sources fanning-in to a single congested link, followed by fan-out to receivers ([14, 11, 3, 9, 6] are a small sample of such papers). Still common, but less frequent, are models that have included cross-traffic across the congested link or links (e.g. [6, 17]). Some work has even investigated multiple congested links, but usually with only a single path through the congested links (c.f. [7]). A totally distinct simulation methodology eschews the use of fixed or hand-crafted small topologies and instead relies on the generation of random network topologies based on realistic topology generators such as [5, 26, 4, 20].

There are (at least) two problems with both approaches: the lack of comprehensive realism and difficulty of useful reproducibility. First, scenarios (network topology, traffic load, etc.) must duplicate realistic and representative network behaviors. By "representative" we mean that the set of simulation scenarios should cover the broad range of behaviors that arise in the real networks. Second, scenarios must be useful in comparing the relative behavior of two (or more) different network protocols or algorithms.

This short paper presents a simple argument. First, that the barbell topology is not *representative* of the Internet. We mean this both in the sense that other topologies exist and that common protocols (such as TCP) exhibit behavior on those other topologies not captured in the barbell topology. In support of this claim, we report that a measurable fraction of packets in the Internet pass through multiple congestion points. Second, we argue that this distinction between topologies is *relevant* by presenting a scenario with multiple congestion points that exhibits behavior that seems unexpected based on intuition derived from the barbell topology. We do not claim to have "a better topology". Rather, we claim that no single topology

or scenario is sufficiently representative. More generally, we make the larger argument that the typical methodology currently accepted for evaluating network protocols is flawed.

We focus on barbell topologies not because they are particularly evil or flawed, but because they are particularly common. For the purpose of this paper we need to analyse a specific example, and (i) barbell topologies are the predominant model used today, and, despite this, (ii) in a general sense, the contemporary Internet actually looks quite different than the barbell topology: the backbone is typically overprovisioned, while areas of potential congestion exist near every sender *and* receiver (e.g. peering points, slow tail circuits, the uplink between small ISPs and *their* ISP ...). In contrast, the barbell topology generally assumes that a single point of congestion exists somewhere in the middle of the network; our informal description of the Internet makes it seem likely that some number of paths will encounter two or more congestion points with more than adequate capacity in between them.

The paper is organized as follows. We first present measurements that show non-barbell like behavior on real paths. Section 2 describes our measurement techniques, with our measurement results presented in Section 2.4. In Section 3 we show how such topologies can produce results that seem surprising in light of received wisdom derived from the barbell topology. Finally, based on these results, we argue that current methods of using network simulation to validate network protocols are inadequate, and we sketch the general outlines that might characterize a better approach.

2 The existence of connections with multiple congestion points

We start with the hypothesis that a non-negligible fraction of connections in the Internet encounter more than one congested link. To prove this, we examine the paths formed between a small number of sources (our measurement sites) and a large number of target hosts. For detecting congestion points within a path, we use the *network delay tomography* technique described in [2, 1]. Given a path and a set of nodes partitioning the path into segments, the technique uses packet pairs to estimate one-way delay distributions *per-segment*, which can then be used to characterize segments as containing a congestion point.

In the following paragraphs, we briefly describe the measurement technique and present its use for determining the fraction of connections interacting with multiple congestion points.

2.1 Measuring network-internal delays

Let s denote our measurement source and d denote a destination of interest. Suppose we wish to measure the delay on individual segments of the path $s \rightarrow d$. We can directly estimate the delay on the link (or segment) $x \rightarrow y$ between nodes x and y as follows. Let t_x and t_y denote the timestamps returned from ICMP Timestamp Request packets[1] sent back-to-back from s to nodes x and y respectively. If the path $s \rightarrow x$ is a prefix of $s \rightarrow y$, then we can also assume that the two packets will experience approximately the same delay on $s \rightarrow x$. In such a case $\delta = t_y - t_x$ will yield an initial estimate of the total delay on the measured segment $x \rightarrow y$. The total delay includes both the fixed link delay (transmission time and propagation delay over the measured segment) *and* the queuing delay on $x \rightarrow y$.

As clocks on x and y are not necessarily synchronized, we must assume that they differ by an offset $O_{x,y}$. (For now, let us assume that the skew in the rate that the clocks advance is small enough so that the change in $O_{x,y}$ over the period of measurement is within error tolerance.) $\delta = t_y - t_x = t_{\text{queueing}} + t_{\text{link}} + O_{x,y}$. Given δ_i and δ_j, two observations of δ, then $\delta_i - \delta_j = \Delta t_{\text{queueing}}$ (because t_{link} and $O_{x,y}$ cancel out).

Let $\delta_{\text{min}} = \min(\delta_i), i \in [1, m]$ after m observations. Assuming constant clock-offset, for sufficiently large m, δ_{min} denotes with "high" probability the event of zero queueing delay on $x \rightarrow y$. Then $\delta'_j = \delta_j - \delta_{\text{min}}$ gives the desired network internal queueing delay on path $x \rightarrow y$.

The computation of δ'_j depends upon the path to x being a prefix of the path to y. Measurements suggest that irregular routing in the Internet often violates this assumption [2] thus demanding extreme care. We say that nodes x, y are members of the same *tomography group*, if (i) $s \rightarrow x$ is a prefix of $s \rightarrow y$, and (ii) if the path from x to y in $s \rightarrow y$ lies on $s \rightarrow d$. ($s \rightarrow x$ need not overlap with $s \rightarrow d$ at all, as long as condition (i) is satisfied). Tomography groups partition the nodes of the path into equivalence classes. A node is called an *orphan* if it is the only member of its tomography group. We can compute δ'_j using direct measurements for any pair of nodes, x, y, in the same tomography group.

[1] In the past, some concern has been raised about techniques that rely on router-based ICMP measurements [15, 18], that routers may sometimes delay processing of ICMP messages, a practice that may introduce errors. Such concerns have recently been examined by Govindan and Paxson [12] (and confirmed in our own measurements [2]). These measurements indicate that the delay distributions of ICMP packets rarely differ noticeably from the delay distributions of regular traffic.

Tomography groups are determined using TTL-constrained probes similar to the `traceroute` tool [13]: we first determine the end-to-end-path from s to the destination d and then the path from s to each hop in the end-to-end path. This information is used to determine whether a pair of nodes satisfies conditions (i) and (ii).

The computation of δ'_j also depends upon the clock-offset, $O_{x,y}$, being constant. As this is rarely the case in practice, it is necessary to compensate for such errors for our measurements to be useful. We discuss our approach to clock calibration in Section 2.3.1.

2.2 Measurement setup

The experiment involves determining which paths to measure, determining how to partition paths for network tomography, performing the measurements and analyzing the results.

The paths considered in this study are based on IP addresses provided by the University of Pennsylvania SEAS Web server. We choose random targets from this list, determining hosts that reply to ICMP Timestamp requests. We first "screen" each path to determine if there is congestion end-to-end. For this, the tomography method is used in a null configuration (e.g. measuring end-to-end one-way queuing delays to the target). We send 300 probes in 200 ms intervals.

When an end-to-end path is determined to be congested, we need to partition the path into measurable segments. Given the path structure (e.g. tomography groups and their members) and a target number of segments, n_t, we consider possible partitions as follows. First, we consider every possible choice of $n_t - 1$ nodes from a single tomography group, for each tomography group. Then, we choose the set of nodes that minimizes the sum of squared distances (in terms of hop count) between adjacent nodes. This favors well balanced paths. For simplicity, we do not consider the case of overlapping pairs, although this could greatly improve our ability to achieve insightful partitions.

Once the path is appropriately segmented, we send 3000 probes to the head and tail of each segment, probing each segment in 500 ms intervals. To avoid introducing congestion due to probe traffic, we limit the number of concurrent experiments to 6, and set the number of segments measured per path to 4.

2.3 Analysis method

We discuss how clock calibration is implemented and how measured segments and paths are characterized as congested.

2.3.1 Clock calibration

Clock calibration issues must be addressed practically in the context of our particular experiments to retain as much of the data we collect as possible with a high degree of confidence in the calibration corrections we apply. We have simplified the general problem somewhat by using source machines whose clocks we know to be adjustment-free – no jumps, no skew changes. The length of the probing runs, generally around 30 minutes, affords approaches that may not be adaptable to much shorter probe durations.

We divide the probing run into intervals, and examine the minimum one-way transit times (OTTs) in each interval in each direction. A 12-second interval size generally gives a good distribution of sequences of at least four consecutive minimal OTTs for the same direction that are colinear, probably corresponding to congestion-free transits – even for target machines with clocks regularly jump-adjusted as frequently as once every 60 seconds. After the first pass that identifies the colinear-OTT stretches in each direction, a further pass, involving cross-stretch collinearity checks, reveals the target clock's jump adjustments and skew changes, and allows the coalescing of stretches, for the same or opposite directions, even across gaps occasioned by congestion-ridden intervals not usable in the first pass.

The result of these passes is a map of the probing run into regions of homogeneous target clock linear correction. Readings outside the mapped correctable regions are discarded. The details of the algorithms are beyond the scope of the present paper. Other strategies for skew and adjustment detection are discussed in [27, 23, 21].

2.3.2 Characterizing a segment as congested

We need an appropriate measure of congestion, based on the estimates provided by the tomography technique, to determine whether a path segment is congested. A simple approach is to use a *quantile-threshold* test on the measured queueing delay estimates within a certain measurement window, w. A path segment is characterized as congested in the interval $[t, t + w]$ if the x-quantile of queueing delay estimates is larger than a threshold th e.g. at least $(1 - x) * w$ samples within a window of w estimates are larger than the threshold th. It becomes crucial to select appropriate values for x, th and w. Higher values of x (and lower values of th) make the test more sensitive to transient queueing. Selecting an arbitrary (x, th) pair

seems inappropriate: we do not have strong evidence suggesting specific values, and the technique, as used, does not offer any insights on the correlation between queueing delay and packet loss (that would affect TCP flows). We thus have to rely on a (seemingly) reasonable *range* for those parameters, focusing on values of x between 0.7 and 0.9, and th between 30 ms an 80 ms (although we also report on higher values for illustration, to aid in examining the consistency of the results).

Use of a measurement window is needed because of the relatively long duration of our measurements on each path (approximately 30 minutes). We need to capture congestion phenomena at time-scales that match the lifetime of flows that are subject to congestion on the path, and using the quantile-threshold test on the full 30 minutes will hide shorter periods of congestion that would indeed interact with flows on the path. Besides that, it is also necessary to distinguish between simultaneously congested path segments and possibly migrating congestion points. For the purposes of this paper, we thus consider (somehow arbitrarily) a window size of 100 samples, which covers approximately 50 seconds of observations.

2.4 Results

We present results using measurements from 1869 paths[2] examined in April-May 2002.

In Figure 1 we show the number of paths that have multiple congestion points (simultaneously) in *at least* one measurement window, as a fraction of congested paths. We present figures for different values of the x-threshold. There are three observations to make. First, there is a striking difference in the figures between for higher values of x between $x = 0.88$ and $x = 0.92$, due to both transient queuing as well as possible estimation error (due to the packet-pair assumption).

Second, for the lower values of x and as th increases, the fraction of paths with multiple congestion points appears to be reasonably consistent.

Third, regardless of x and th parameters, the fraction of paths with multiple congestion points appears significant but may be misleading, as the long duration of our measurements increases the chances that congestion will be observed in at least one measurement interval. It becomes necessary to examine in more detail if multiple congestion points appear occasionally for every path (for reasons including measurement error), or if this behavior is persistent.

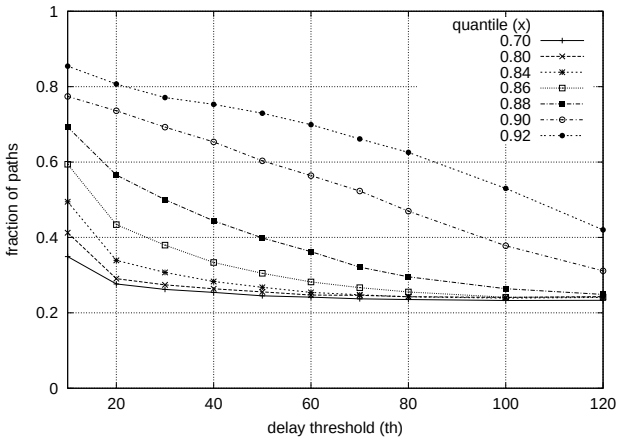

Figure 1: **Fraction of congested paths that have multiple congestion points in at least one measurement interval**

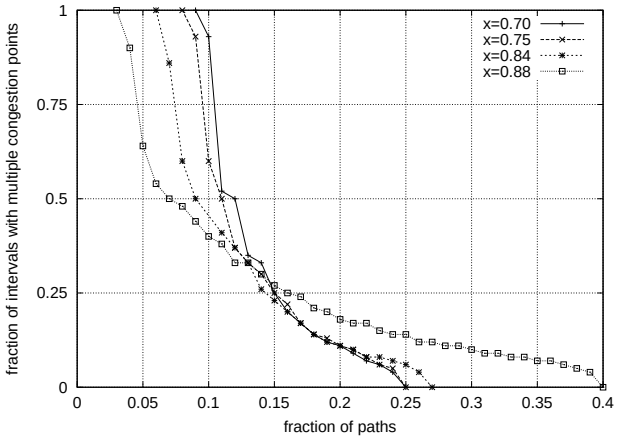

Figure 2: **Fraction of congested paths vs. fraction of multiple congestion point intervals (variable x, fixed $th = 50ms$)**

For each path that was ever found to be congested, we count the number of measurement intervals in which the path had multiple congestion points, as a fraction of the total number of intervals in which the path was found to be congested. In Figure 2 we display the resulting distribution. We observe that between 3 and 9% of congested paths that ever experienced congestion for even one interval, in fact went through multiple congestion points in every interval, 6 to 12% in roughly half the intervals and 12 to 16% in a quarter of all intervals.

It seems safe to say that roughly a bit under 1 out of 10 congested paths pass through multiple congestion points.

[2]Our first pass initially probed 38,275 paths. Of those, 2,777 were selected as appropriate candidates to investigate in more detail, breaking them down into segments. 908 of those were ultimately rejected for a variety of reasons such as target host disconnecting, clocks that could not be accurately calibrated, etc.

2.5 Limitations

There are a number of limitations of the measurement study which need to be considered in interpreting the results. Our design aims to err on the side of conservative estimates.

- As measurements are taken from a small set of sites, the paths we examine always have one endpoint at one of our measurement source. This could be a problem if measurements were taken behind a congested link, as this would introduce at least one congestion point per path. Fortunately, our measurements are taken from relatively well-connected networks, with segments close to the measurement source rarely characterized as congested. This makes our estimate safe but also rather conservative (data taken from a home via a cable-modem may look significantly worse).

- Irregular routing and the need to keep probe traffic load low restricts our selection of nodes that partition a path into measurable segments. Further, our desire to sample as many paths as possible limits the time we can spend finding optimal segment partitions for any one path, even when a better partition exists. Consequently, we may sometimes choose partitions that include multiple congestion points within a single segment; such a path will be incorrectly labeled as having a single point of congestion.

- Our traffic measurements may not be representative. The distribution of packets is (obviously) denser during times of heavier loads. However, our probes are sent at uniformly distributed times during the day. Therefore, proportionally, a higher percentage of our packets will experience light load, thereby biasing our results towards measuring less congestion. This bias may also be reinforced because our probes do not coincide with actual data transfers to the target address. (Our set of potential target addresses is collected in a separate pass before we begin any measurements.)

- We depend on delay statistics as an indicator of congestion, that may affect connections on several congestion points along a path. However, the interaction between the identified congestion points and the connections is not examined.

- It may seem unreasonable that we ignore packet losses in our determination of congestion. However, as discussed in Section 4, the congestion effects we see in Section 3 are not the results of either experiencing or ignoring packet loss.

- Results may not reflect the long-term behavior of Internet paths. This is an inherent limitation of many measurement studies. We note that the results presented in this paper appear to agree with our preliminary experiment [2] taken 6 months earlier. Establishing the long-term persistence of this phenomenon in the Internet is, however, beyond the scope of this work.

Our results should thus be interpreted as indicative rather than quantitative.

3 Performance implications of multiple congestion points

Is there any significance to the fact that a measurable fraction of congested flows actually experience congestion at multiple points, and that backbone links tend to be over-provisioned? Or, is this just an irrelevant distinction from the common barbell topology?

We argue that the distinction is relevant by presenting a scenario engineered to display two counter-intuitive behaviors. First, a system consisting only of long-lived TCP flows is undergoing (in the technical sense) congestion collapse; this is somewhat unexpected after the congestion avoidance algorithms in [14] and the arguments in [8]. Second, the culprit that is being too aggressive are the flows going through multiple congested links, with (relatively) long RTT. It is well known that the network is biased against TCP flows going through multiple congestion points: they get "beaten down" and receive less than equal share of the net[7, 19] by roughly a factor of the square root of the number of congestion points. It is also well known[10] that flows with long RTTs increase their window size at a slower rate than flows with short RTTs, and therefore converge to smaller cwnd sizes, on the average, with flows getting bandwidth roughly in proportion to the inverse of their RTT.

3.1 Simulation Scenario

We consider the 12-node topology of Figure 3, and three sets of TCP flows labeled A, X and Y. Flows in X and Y create background traffic, competing with flows in A on links $\langle 0, 2\rangle$ and $\langle 3, 4\rangle$. Flows in A thereby pass through two congested links. To avoid synchronization, all flows are started at random times in $t = [1s, 10s]$ and

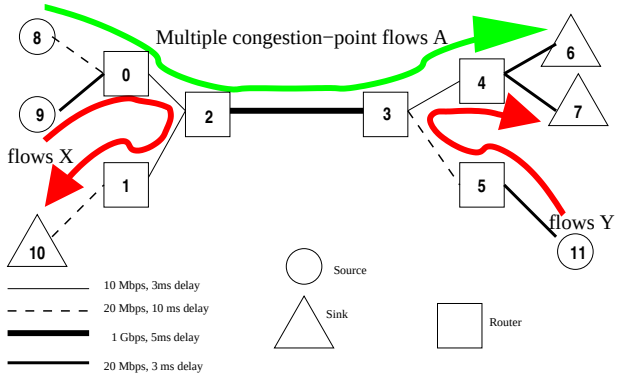

Figure 3: **Topology for simulation experiments**

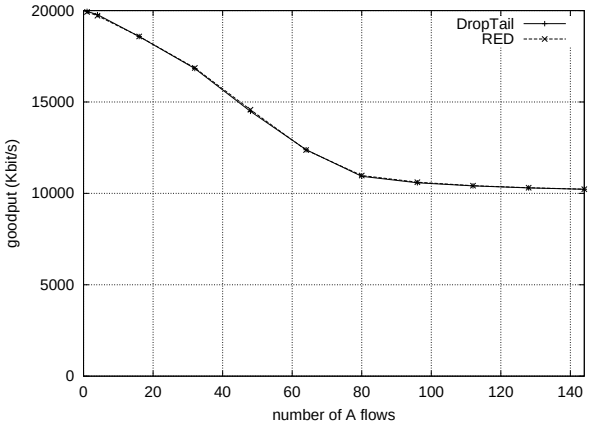

Figure 4: **Goodput vs. number of A flows**

are active throughout the experiment. All experiments last for 300 seconds of simulation time and the results reported are medians over 100 experiments.

The performance metric of interest for this study is aggregate *goodput* rate, defined as the average cumulative rate of *useful* data received by applications. The effect of multiple congestion points on other aspects of TCP behavior, such as fairness, are explored in other studies (c.f. [7]). We start measuring goodput only at the time *all* flows have successfully sent at least one segment to the application layer at the receiver[3].

We initially consider 5 flows in X and 5 flows in Y and vary the number of flows in A. The minimum round-trip time (RTT) is 48 ms for flows in A, and 38 ms for X and Y. For all flows, we use the Reno variant of TCP. All queues have a size of 40 packets; we use both Drop-Tail and RED scheduling. The results of this experiment are shown in Figure 4. The important feature to note is that as the number of flows over path A increases, the aggregate goodput drops. This shows that the system undergoes congestion collapse [22, 8], because increasing the offered network load (the number of flows A) results in a decrease in aggregate goodput (the sum of the packets delivered to all 3 data sinks).

It is easy to see why this occurs. The aggregation of flows traveling over path A simultaneously takes link capacity away from both flows X and Y. If flow A were to settle on a lower average `cwnd` that effectively decreased its rate by R, then X and Y would *each*, similarly, increase their effective rate by R, for a net increase in total goodput of R. On the other hand, if flow A were to settle on a higher average `cwnd` that effectively increased its rate by R, then X and Y would *each* decrease their effective rate by R, for a net decrease in total goodput rate of R. As more individual flows travel over path A, the behavior of the aggregate flow gets more aggressive and settles on a higher cumulative `cwnd`, thereby decreasing total goodput.

Note that the congestion collapse would not occur with only a single point of congestion (although the allocation of bandwidth might be unfair), nor would it occur if $\langle 2, 3 \rangle$ were a low capacity link (because that would limit A's aggregate throughput, and A would not be able to steal bandwidth from both X and Y).

3.2 Results are pervasive

Figure 4 shows that congestion collapse occurs under both RED[11] and Tail-drop. Further experiments confirm that, given the topology of Figure 3, this phenomenon persists across a wide range of parameters, with different queue sizes (Figure 5), different RTT ratios between multiple congestion point flows A and "background" flows X (Figure 6), and different TCP algorithms (Figure 7). Larger queue sizes exacerbate the initial collapse, but approach the limit of aggregate goodput more slowly. Increasing RTT of X, or decreasing RTT for A, both[4] result in making the aggregate flows over A more aggressive. Consequently, congestion collapse occurs more rapidly. Congestion collapse seems to occur regardless of the version of TCP; we experimented with Reno, Sack and Vegas (we did not simulate mixed implementations).

[3]Retransmission intervals for SYN packets are fixed, so lost SYN packets may result in arbitrary delays for connection startup time. Measurements taken before all flows have passed the SYN/ACK stage will not reflect the reported "number of flows".

[4]We only show the results of changing the RTT for A; changing the RTT for X displays similar results.

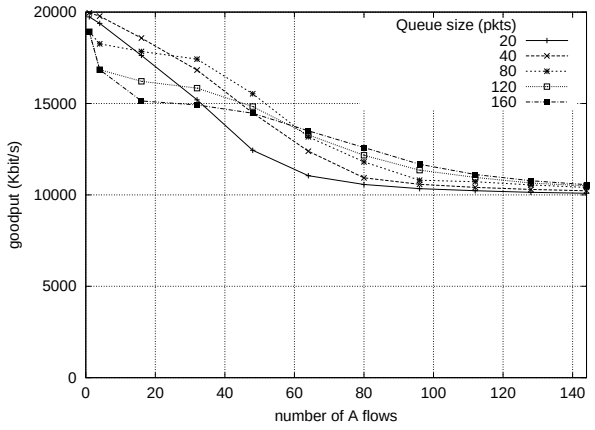

Figure 5: **Performance degradation with multiple congestion points, with different queue sizes**

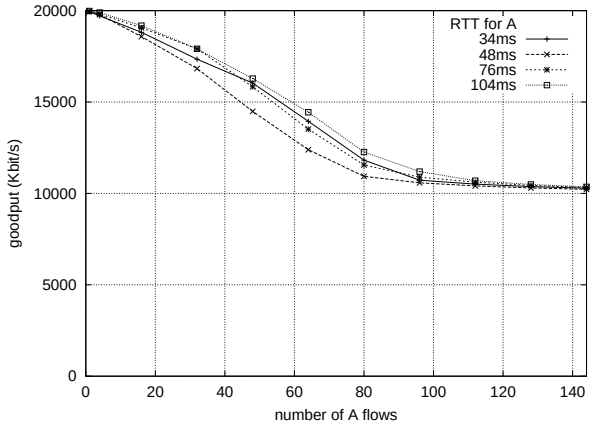

Figure 6: **Performance degradation with multiple congestion points, with different RTTs for A flows**

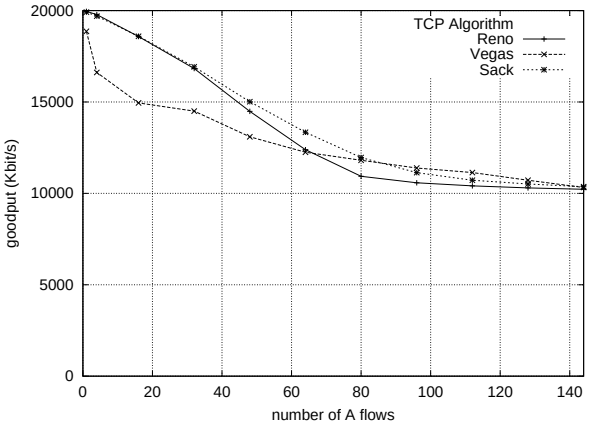

Figure 7: **Performance degradation with multiple congestion points, with different TCP algorithms**

4 Discussion

The non-standard topology used in this paper induces unfamiliar behavior in familiar protocols. The congestion collapse displayed in Figures 4 through 7 seems unusual in three respects. First, unlike the cases of congestion collapse described in [8], the type of collapse here is neither unnecessary retransmissions nor undelivered packets, and the primary factor has nothing to do with unresponsive flows. Second, although *technically* demonstrating "congestion collapse", the term seems somewhat misapplied here because it is unclear what the correct behavior should be. Finally, the culprit in this case seems to be the least likely party; the flows with both the longer RTT and the most points of congestion. We will discuss each of these three points, briefly, in turn.

- We consider the problem of classifying the type and cause of the congestion collapse described in Section 3. In [8], Floyd et. al. categorize five forms of congestion collapse: (1) *classical congestion collapse*, (2) *collapse from undelivered packets*, (3) *fragmentation-based collapse*, (4) *collapse from increased control traffic*, and (5) *collapse from stale packets*. It is easy to see that none of these categories apply here. The form of congestion collapse here can best be described as *collapse from underutilized links*.

  The essential factor behind this form of congestion collapse is the *lack of global cooperation between switches*. This particular form of congestion collapse is totally independent of the end-to-end congestion avoidance algorithms. Flow A cannot avoid collapse by unilaterally being less aggressive — unless it possessed some global knowledge: knowledge about two links it does not use. To see this, consider the case where flows X and Y each share the last link in their paths with flows X' and Y' respectively. If X and Y get squeezed more by the aggregate flow over A, then X' and Y' take up the slack — X and X' (or Y and Y') together, will utilize 100% of the capacity to the sink. However, if the aggregate flow over A backs off, then X and Y get more capacity at the expense of X' and Y', but nobody uses the excess capacity on the final link to A's sink. This contrasts with the example in Figure 3 in that the most efficient bandwidth allocation is achieved here by the aggregate flow, A, being aggressive and greedy while in Figure 3 the most efficient allocation occured when A backed off — yet, in *both* cases A encounters the same traffic on all of

the links it traverses — it cannot distinguish between the cases. There is no way for the flows over A to determine ideal behavior without knowing about X' and Y'.

- Although *technically* the behavior described here is in fact congestion collapse (because goodput decreases with an increase in offered load), nevertheless the term seems somewhat inappropriate.

  The issue is not that the loss in goodput approaches a limit other than 0, and each increase in load causes less and less goodput degradation. Upon reflection, the existence of a non-zero limit is not an issue — although different than classical congestion collapse [22], this is actually consistent with the use of the term in [8]. The value of the limit is not an issue, either — we can construct scenarios such as this that can have *arbitrarily* bad global utilization, by simply adding more independent paths that partially overlap with A. The example in Figure 3 has only two congestion points, corresponding to the partial overlap of A with independent flows X and Y. The goodput converges to 1/2 of the combined capacity of $\langle 1, 2\rangle$ and $\langle 3, 4\rangle$ (the min-cut that provides the max-flow between the sources and destinations). As a coarse approximation, if N distinct paths overlap with A, then global utilization (achieved aggregate goodput divided by optimal aggregate goodput) approaches $1/N$ as we increase the number of flows over A. Thus, by increasing N, we can achieve arbitrarily bad utilization (as close to 0% as we want).

  Rather, our discomfort with the term collapse stems from the fundamental tension, in this case, between local (and global) fairness and global goodput. Collapse is a pejorative term; it seems to imply that the system should have achieved a better result, somehow. In fact, absent a policy specification of the relative importance of fairness to flows and global goodput, it is unclear that *any* allocation of bandwidth is satisfactory. Optimal goodput can only be achieved by starving the A flows. Fairness to the flows over A can only be achieved by degrading global utilization.

  If such a policy were specified, then an omniscient router could achieve a good allocation of bandwidth, but in practice the lack of global cooperation between switches would render that unachievable in our case (and would justify the term "congestion collapse").

- It may superficially seem as if the behavior of long-RTT and multiple-congestion-point flows contradict existing analytic models of TCP behavior. A distinction must be drawn between our (flawed) intuition derived from such models, and the models themselves. A is an aggregation of flows, and not a single flow – as such it behaves far more aggressively than a single flow. We have not verified that individual component flows of A behave according to standard TCP models (we have every reason to suppose they do!), but note that a sufficiently large number of flows should overcome the penalty of long RTT as well as multiple congestion-points.

  However, informally we have observed that even researchers knowledgeable about TCP dynamics often inadvertently expect aggregations of flows to behave similarly to the individual component flows, unless the issue of the aggressiveness of multiple flows is explicitly brought to their attention.

It would be a mistake to conclude based on our measurements and example that the barbell topology itself was somehow more fundamentally flawed for evaluation of congestion control than any other choice of network topology. It would be similarly silly to suppose that the topology of Figure 3 was especially useful in highlighting the most common cases of congestion. On the contrary, the barbell topology seems to be a good candidate to understand and evaluate the behavior of protocols under the simplest and most basic conditions of congestion. Our topology seems useful in highlighting a different set of behaviors.

The reasonable conclusion is that no single topology is adequate to cover all interesting cases.

5 Conclusion

5.1 Towards a better evaluation methodology

We have argued that the use of a single topology, or even a single class of topologies, is inadequate for the evaluation of network protocols because it is possible that topology-sensitive properties may be overlooked or misunderstood. We have also argued that simulation is necessary to validate protocols before real-world deployment. If simulation is necessary, but a single scenario is insufficient, then clearly a suite of simulation scenarios is needed.

The specific composition of such a suite is not yet clear, but a preliminary sketch of some pitfalls to avoid and some guidelines to follow in constructing such a suite is possible.

It is desirable that any practical suite should be usable as a matter of course to evaluate *every* network proposal. One of the advantages that have attracted so many researchers to use the barbell topology is that so many others have used it before, and therefore simulation results are comparable. Introducing complexity in the model makes the results hard to interpret; care must be used in constructing such models (resemblance to the real world, realistic behavior that is *likely* to highlight problems in protocols under consideration, and careful explanation and interpretation of possible results). Standardizing such models as reference testbeds amortizes the cost of model construction over many more experiments and provides a basis of comparison.

If a single standardized suite is to be used for all protocol evaluations, and it is intended to be used as a tool for comparing protocol evaluations, then this argues for such a suite to be as small as practicable, else the burden on researchers will be too high, and the suite will not be used, and it will therefore not be available as a tool of comparison.

Diverse topologies and scenarios are required to highlight behaviors. Even admirably comprehensive attempts to include a broad range of network loads and topologies (c.f. Keshav [17]) have still been biased toward variants on the barbell topology, and would have missed, for example, the behavior described in this paper.

Simply adding new scenarios to increase diversity conflicts with the goal of small size. Therefore only carefully crafted scenarios will be reasonable — scenarios chosen to capture behaviors that are known to be interesting and relevant, and with evidence that such behavior may exist in real networks.

Randomly generated topologies are not likely to capture specific behaviors without careful intervention and design in the topology generator. Such generators will be useful, however, in testing problems of scale.

It is tempting to look to existing architectural benchmarks such as SPEC [25] or SPLASH [24] as a model. However, although such benchmarks have been proposed in the past (c.f. [16]), there are several compelling reasons why a network simulation suite should not be a benchmark intended to yield a single number. First, unlike processor design where the end result must be a single instruction set architecture and a small number of compatible processors, multiple protocols can exist and co-exist. Therefore the appropriate question is not "what is the best protocol?", but rather "what is the most appropriate protocol in such a situation?" Answers may differ depending on the situation. Second, validation through simulation is not performed simply to evaluate performance; rather it is a form of regression testing, to make sure that small modifications to programs in one area do not break behavior in other areas. In this vein, we often are willing to trade off properties that may not simply be measurable. The qualitative danger of, say, a distributed denial of service attack may not be commensurable with expected completion time of file transfers. Third, given that simulation is meant to provide confidence that large-scale deployment is reasonable, one function of such a suite would be to stress situations which had provoked race conditions and anomalies in earlier protocols.

Given these constraints, it seems inevitable that such a suite will need be a communal effort in design and maintenance. Fortunately, the Web has made efforts easier. It is easy to imagine a web site for the collection, development, and publication of annotated standard topologies with careful explanation and interpretation of possible results.

5.2 Concluding Remarks

In this paper we have shown that by suitably choosing a network topology for simulation, we can exhibit behavior that runs counter to received wisdom. In particular, we have exhibited a case where a set of congestion-aware TCP flows undergo congestion collapse, where flows going through more congestion points and with longer round trip times get bandwidth at the expense of flows with single bottlenecks and short round trips. Further, we have evidence that the necessary ingredients for such behavior exist in the Internet.

It is unfair to conclude from this that TCP is flawed, or that flows through multiple congestion points deserve special investigation. We conclude only that we should not become complacent about our understanding of network protocols, that protocols (including TCP) exhibit more complex behaviors than we might expect, and that what may seem like fundamental properties of protocols (for example, that a congestion-aware TCP can avoid congestion collapse) can depend upon the simulation topology. Therefore, reliance upon a small set of simulation topologies is inadvisable, and may miss important protocol interaction. Active work is needed to carefully collect a suite of simulation scenarios that represent both common configurations and anomalous behavior.

Acknowledgments

This work was supported by the DoD University Research Initiative (URI) program administered by the Of-

fice of Naval Research under Grant N00014-01-1-0795. We thank John C. Parker, Dave Millar and Charles Buchholtz at UPenn, C. Powell and J. Feigenbaum at Yale, and M. Hogsett and P. Lincoln at SRI for making it possible to obtain measurements from different sites. We also thank S. Ioannidis, B. Knutsson, J. M. Smith and the members of the Distributed Systems Laboratory for valuable comments and suggestions throughout the course of this work, and Vern Paxson for providing us with reference code for his clock calibration algorithms.

References

[1] K. G. Anagnostakis and M. B. Greenwald. Direct measurement versus indirect inference for determining network-internal delays, March 2002. Submitted to *Performance 2002*.

[2] K. G. Anagnostakis and M. B. Greenwald. On the feasibility of network delay tomography using only existing infrastructure, January 2002. Submitted to *Computer Communications Journal, Special Issue on Performance Evaluation of IP Networks and Services*.

[3] L. S. Brakmo, S. W. O'Malley, and L. L. Peterson. TCP Vegas: New techniques for congestion detection and avoidance. In *Proceedings of ACM SIGCOMM*, pages 24–35, August 1994.

[4] M. Doar. A better model for generating test networks. In *Proceedings of Globecom '96*, November 1996.

[5] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. In *Proceedings of ACM SIGCOMM*, pages 251–262, August 1999.

[6] A. Feldmann, A. C. Gilbert, P. Huang, and W. Willinger. Dynamics of IP traffic: a study of the role of variability and the impact of control. In *Proceedings of ACM SIGCOMM*, pages 301–313, August 1999.

[7] S. Floyd. Connections with multiple congested gateways in packet-switched networks part 1: One-way traffic. *ACM Computer Communications Review*, 21(5):30–47, October 1991.

[8] S. Floyd and K. Fall. Promoting the use of end-to-end congestion control in the internet. *IEEE/ACM Transactions on Networking*, 7(4):458–472, August 1999.

[9] S. Floyd, M. Handley, J. Padhye, and J. Widmer. Equation-based congestion control for unicast applications. In *Proceedings of ACM SIGCOMM*, pages 43–56. August 2000.

[10] S. Floyd and V. Jacobson. On traffic phase effects in packet-switched gateways. *Internetworking: Research and Experience*, 3(3):115–156, September 1992.

[11] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, 1(4):397–413, August 1993.

[12] R. Govindan and V. Paxson. Estimating router icmp generation delays. In *Proceedings of the Passive and Active Measurements Workshop (PAM) 2002*, March 2002.

[13] V. Jacobson. traceroute software. Original release at *ftp://ftp.ee.lbl.gov/traceroute.tar.gz*.

[14] V. Jacobson. Congestion avoidance and control. In *Proceedings of Symposium on Communications Architectures and Protocols (SIGCOMM '88)*, pages 314–329, 1988.

[15] V. Jacobson. pathchar - a tool to infer characteristics of internet paths. available from `ftp://ftp.ee.lbl.gov/pathchar`, April 1997.

[16] H. Kanakia, S. Keshav, and P. P. Mishra. A comparison of reactive host-based flow control schemes in reservationless networks. Technical Memorandum http://simon.cs.cornell.edu/Info/People/skeshav/92/3-12.ps, AT&T Bell Laboratories, July 1992.

[17] S. Keshav. Packet-pair flow control. Available at *http://www.cs.cornell.edu/skeshav/doc/94/2-17.ps*, 1994.

[18] M. Mathis and J. Mahdavi. Diagnosing internet congestion with a transport-layer performance tool. In *Proceedings of INET*, June 1996.

[19] M. Mathis, J. Semke, J. Mahdavi, and T. Ott. The macroscopic behavior of the TCP congestion avoidance algorithm. *ACM Computer Communication Review*, 27(3), July 1997.

[20] A. Medina, A. Lakhina, I. Matta, and J. Byers. BRITE: An approach to universal topology generation. In *Proceedings of IEEE MASCOTS 2001*, August 2001.

[21] S. B. Moon, P. Skelly, and D. Towsley. Estimation and removal of clock skew from network delay measurements. In *Proceedings of IEEE INFOCOM 1999*, March 1999.

[22] J. Nagle. Congestion control in IP/TCP internetworks. RFC 896, http://www.rfc-editor.org/, January 1984.

[23] V. Paxson. On calibrating measurements of packet transit times. In *Proceedings of ACM SIGMETRICS*, pages 11–21, 1998.

[24] J. P. Singh, W.-D. Weber, and A. Gupta. SPLASH: Stanford parallel applications for shared-memory. Technical Report CSL-TR-91-469, Stanford University, April 1991.

[25] Standard Performance Council. The SPEC95 CPU Benchmark Suite. *http://www.specbench.org*, 1995.

[26] E. W. Zegura, K. L. Calvert, and M. J. Donahoo. A quantitative comparison of graph-based models for internet topology. *IEEE/ACM Transactions on Networking*, 5(6):770–783, December 1997.

[27] L. Zhang, Z. Liu, and C. H. Xia. Clock synchronization algorithms for network measurements. In *Proceedings of IEEE INFOCOM 2002*, June 2002. To appear.