Performance Preserving Network Downscaling

Fragkiskos Papadopoulos, Konstantinos Psounis, Ramesh Govindan
University of Southern California
Los Angeles, CA 90089
E-mail: fpapadop, kpsounis, ramesh@usc.edu.

Abstract

The Internet is a large, complex, heterogeneous system operating at very high speeds and consisting of a large number of users. Researchers use a suite of tools and techniques in order to understand the performance of networks: measurements, simulations, and deployments on small to medium-scale testbeds. This work considers a novel addition to this suite: a class of methods to scale down *the* topology *of the Internet that enables researchers to create and observe a smaller replica, and extrapolate its performance to the expected performance of the larger Internet.*

The key insight that we leverage in this work is that only the congested links along the path of each flow introduce sizable queueing delays and dependencies among flows. Hence, one might hope that the network properties can be captured by a topology that consists of the congested links only. We show that for a network that is shared by TCP flows it is possible to achieve this kind of performance scaling. We also show that simulating a scaled topology can be up to two orders of magnitude faster than simulating the original topology.

1. Introduction

Networking research has taken a multi-pronged approach to understanding the performance of the Internet, and to predicting its behavior under new algorithms, protocols, architectures and load conditions. The community focuses on techniques ranging from analytic modeling, to measurement-based performance characterizations to simulation studies [3, 4, 5]. This is appropriate given the overwhelming size, complexity, heterogeneity, and the speed of operation of the Internet.

This multi-pronged approach has its limitations. First, the heterogeneity and complexity of the Internet makes it very difficult and time consuming to devise realistic traffic models, and models for the network. Second, for some of the same reasons, as well as the increasingly large bandwidths in the Internet core, it is very hard to obtain accurate and representative measurements. Even when such data are available, it is very expensive to run realistic simulations at meaningful scales since the memory and CPU requirements of such simulations seem to be well beyond the reach of available hardware. Of course, there are several approaches to alleviate some of these problems. The volume of measurements can be reduced by traffic sampling, and researchers attempt to use "realistic" topology models. However, topology modeling is still in its infancy—realistic models that include notions of capacity and latency are some years away. Further, from sampled traffic, it is hard to infer the performance of the original system.

The focus of this work and of its predecessor [1, 2] is the addition of a new prong—a class of *performance-preserving network downscaling* techniques that lets a designer study the behavior of new services or mechanisms in a large network using a scaled-down version of the network, where the downscaling is *designed to preserve one or more aspects of the original network*.

But what does it mean to design a performance-preserving network downscaling technique? Consider a large network (such as that of a backbone ISP), consisting of several hundred nodes and links, and traversed by hundreds of thousands of flows. Psounis et al. [1, 2] have introduced a method called SHRiNK [1] that preserves some network properties by creating a *slower* downscaled version of the original network. Specifically, SHRiNK downscales link capacities (but not network size) such that, when a sample of the original set of flows is run on the downscaled network, a variety of performance metrics, *e.g.* the packet delay distributions, are preserved.

But is there more than one instance of a performance-preserving downscaling technique? Yes. In this work, we propose two methods to perform *topological* downscaling. In particular, starting from a complex network, we create a smaller version of it with fewer links and nodes, observe the behavior of a sample of the original traffic on this network, and extrapolate the observed performance back to the origi-

[1] SHRiNK: Small-scale Hi-fidelity Reproduction of Network Kinetics.

Proceedings of the 38th Annual Simulation Symposium (ANSS'05)
1080-241X/05 $20.00 © 2005 **IEEE**

nal network.

Topological downscaling has three benefits. First, by relying only on a sample of the traffic, it reduces the amount of data we need to work with. Second, by using samples of actual traffic, it short-cuts the traffic characterization and model-building process. Finally, by reducing the number of links and nodes, it reduces the complexity of the network that we work with. This, in turn, allows operators to manage existing networks more efficiently, and researchers to test new architectures and algorithms in small experimental testbeds while ensuring the relevance of the results.

This approach also presents challenges. At first sight, it appears optimistic. For example, it is possible that a group of flows share some links in the original network but not in the replica. Hence, the replica may not capture some of the correlations between these flows. However, a more careful look at the network reveals that it is only the congested links (i.e. the ones where drops occur) along the path of each flow that introduce dependencies among flows and sizeable queueing delays [6, 7, 8, 9, 10, 11]. Hence, one might hope that the network properties can be captured by a topology that consists of the congested links only. The very small number of congested links along the path of a flow makes this approach quite effective in reducing the size of the network that one works with.

We have applied our approach in the topology of the CENIC backbone [12] and have successfully predicted queue and flow statistics of the original network from the small replica with very high accuracy. The accuracy of the approach does not depend on the load conditions, the active queue management schemes used, etc. The approach can be automated, that is, given the original topology and traffic, a tool can easily create the scaled network, run simulations, and extrapolate the performance of the original network from the simulation results. And simulating the scaled topology can be up to two orders of magnitude faster than simulating the original topology.

The rest of this paper is organized as follows: Section 2 briefly discusses prior work on scaling down networks. Section 3 introduces the proposed methods and applies them to simple topologies and to realistic topologies where multiple congested links exist (CENIC backbone [12]). Extensive simulation results are presented in Section 4. It is shown that a variety of metrics, including packet queueing delays, end-to-end flow delays, number of active flows, etc. can be predicted from the replica. Finally, Section 5 presents the savings - in terms of the time needed to run an experiment - from the usage of these methods, and Section 6 concludes the paper.

2. Prior work

Psounis et al [1, 2] introduced a method called SHRiNK that creates a *slower* version of the original network. The main steps in the creation of the replica are to reduce link capacities, increase propagation delays, and reduce the traffic arrival rate by sampling incoming flows.

The main results of this work are the following [1, 2, 13]: (i) For networks in which flows arrive at random times and whose sizes are heavy-tailed, performance measures such as the distribution of the number of active flows and of their normalized transfer times are left virtually unchanged. (ii) For networks which carry long-lived TCP-like flows arriving in clusters, and which are controlled by a variety of active queue management schemes, a slightly different scaling to the previous one leaves the queueing delay and drop probability unchanged as a function of time. However, as mentioned before, this class of work has not considered downscaling the size of the topology.

There have been very few studies of the question of whether one can reduce the topology of a network without losing important network properties, e.g. [14]. These studies have used graph properties as metrics to judge the quality of the scaling. For example, they have used metrics like the average degree of a graph, the clustering coefficient [15], or the degree exponent [16]. Our approach towards scaling down the topology of a network is very different. The goal is to be able to predict the performance of the original network from the scaled network. Like Psounis et al. [1], we want to be able to predict flow transfer times, packet delays, queue sizes, and so on.

Another line of research that is relevant to our work attempts to replace time consuming packet-level simulations by modelling. Bohacek et al. [17] propose a hybrid modelling framework that uses averaging of discrete variables over short time intervals, and can accurately predict network behavior faster than packet-level simulations. Liu et al. [18] extend the fluid models introduced in [19] to be topology-aware, take into account congested links only, and accurately predict network dynamics by solving the fluid models. They show that this takes less time than packet-level simulations, especially when workloads and bandwidths are high. Our work does not attempt to replace packet-level simulations with faster methods, but rather to run them in smaller networks with fewer traffic, which results in faster execution times. Further, the question of whether network topology can be scaled while preserving performance is interesting in its own right.

3. Scaling down network topology

In this section we present two methods for scaling down the topology of a network while preserving its performance. The methods operate on a given topology and need to know the traffic matrix (i.e. source/destination pairs and corresponding rates), the paths followed by network flows, and the links that are congested. Under both methods, the downscaled versions consist only of congested links from the orig-

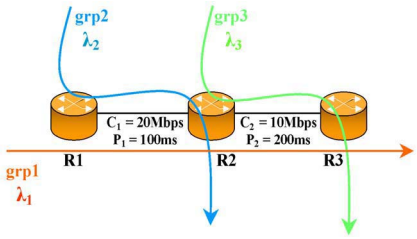

Figure 1. Original network topology.

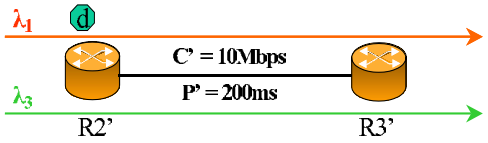

Figure 2. Scaled system when using DSCALEd.

inal topology. The first method, called DSCALEd (downscale using delays), accounts for the missing uncongested links by adding appropriate fixed delays to all packets. The second method, called DSCALEs (downscale using sampling), accounts for the missing links by sampling flows and properly adjusting the capacities and propagation delays of the replica network.

We argue that ignoring uncongested links does not result in a loss of information about the original network. This is based on the following two observations: (i) the packet arrival process is almost the same before and after an uncongested link, and (ii) queueing delays due to uncongested links do not have a significant contribution to end-to-end packet delays. These observations are used in [11], where a scheme was proposed for computing the end-to-end queueing delay through a backbone network, and are validated in detail in [9]. These are also the observations made in [6], where a method for modelling the backbone traffic at the flow level was introduced. Similar results have been derived in [20] for traffic which observes the Large Deviations Principle, and were used in [21] to justify that the effective bandwidth of a flow remains unchanged throughout a network.

We now describe a simple topology used to introduce the methods. Consider two links in tandem, as shown in Figure 1. There are three routers *R1, R2* and *R3*, and three groups of flows, *grp1, grp2,* and *grp3*. A group of flows comprises all the flows that follow the same path from the source to the destination. [2] The link *R1-R2* has a speed of $C_1 = 20$Mbps and a propagation delay of $P_1 = 100$ms. The link *R2-R3* has a speed of $C_2 = 10$Mbps and a propagation delay of $P_2 = 200$ms. The buffers on the routers can hold 300 packets and the AQM scheme is RED with parameters $min_{th} = 100$, $max_{th} = 250$ and averaging parameter $w = 0.00005$. (Note that there is nothing special about choosing RED as the AQM scheme or about choosing this particular set of parameters. Simulations show that similar results hold if DropTail is used instead.) Within each group, flows arrive with some rate λ_i, $i = 1, 2, 3$. We vary λ_i's so that the link *R2-R3* becomes congested whereas the link *R1-R2* remains uncongested.

Given the network setup shown in Figure 1, we are interested in creating a replica and predicting various performance measures on link *R2-R3*, which is the bottleneck link. This can be accomplished by using either of the following two methods.

3.1. DSCALEd: Downscaling using delays

The first method is quite straightforward and can be summarized as follows:

1. Ignore uncongested links and retain all congested links.
2. Groups of flows that traverse congested links in the original network, will traverse the corresponding bottleneck links in the scaled system.
3. Groups of flows that do not traverse bottleneck links are ignored.
4. Assign to the links of the scaled system, speed and propagation delay values equal to those they have in the original network.
5. For every group of flows that had in their original paths at least one link that is not included in the scaled system, add a constant delay factor such that the end-to-end propagation and transmission delays for each group of flows is equal to that in the original system.

Let's apply the method to the network in Figure 1. Since link *R1-R2* is highly uncongested and no drops occur, there isn't any interdependence between *grp1* and *grp2* flows. Moreover, the rate by which *grp1* packets enter the link *R2-R3* is not affected by the queueing on link *R1-R2*. Finally, the only link that has an impact on the total queueing delay of *grp1* flows is the congested link *R2-R3*. Having the above comments in mind, we remove link *R1-R2* from the original topology, and consequently, ignore *grp2* flows. Hence, the scaled system will consist of two groups of flows (*grp1*, *grp3*) and one link, denoted by *R2'-R3'*, as shown in Figure 2.

To compensate for the absence of link *R1-R2* from the scaled system, a fixed delay d is introduced to each *grp1*

[2] Notice that in accordance with the usual practice [22, 23, 10], packets are said to belong to the same flow if they have the same source and destination IP address, and source and destination port number. A flow is "on" if its packets arrive more frequently than a timeout of some seconds. This timeout is usually set to something less than 60 seconds.

packet. In this example, the value of d equals the sum of the propagation and transmission delay of link *R1-R2*. That is, $d = P_1 + L/C_1 = 100.416$ms, where $L = 1040$bytes is the packet size. Notice that a fixed delay like this can be very easily added to the round trip time of a packet by a simulator.
Remarks: It is straightforward to study uncongested links using this method. To do so, one only needs to add to the scaled system the uncongested links of interest, together with the groups of flows that traverse them.

It is easy to use this method in conjunction with SHRiNK [1, 2] to produce a smaller and slower replica. After the smaller replica is produced, one can employ SHRiNK independently.

3.2. DSCALEs: Downscaling using sampling

The previous method adds a fixed delay to each packet. While in Section 5 we argue that this can be done easily, it is still interesting to investigate if it is possible to avoid it. With this in mind, we propose a method that preserves performance by sampling flows and carefully choosing the capacities and propagation delays of the links of the replica network. The method can be summarized as follows:

1. Ignore uncongested links and retain all congested links.
2. Groups of flows that traverse congested links in the original network, will traverse the corresponding bottleneck links in the scaled system.
3. Groups of flows that do not traverse bottleneck links are ignored.
4. Sample each group of flows with different probabilities. (details described below)
5. Compute the capacities and propagation delays of the links of the scaled network such that (i) the round trip times of each group of flows remain unchanged (except by a constant multiplicative factor), and (ii) the traffic intensities of the links in the original and scaled network are equal.

Before describing the method in detail, we make the assumption that the queueing (and tranmission) delay is negligible in comparison to the *total* end-to-end delay. This assumption is not unrealistic for IP backbone networks [7]. For example, in [8] it is reported that the average measured queueing delay on an operational OC-3 link for two data sets was as low as 70μs and 33μs respectively. In [9, 10, 11], it was observed that packets not only experience insignificant queueing inside a backbone, but also that the jitter is insignificant and that the link utilizations can reach $80\% - 90\%$ before queueing delays begin to exceed several ms. In particular, in [11], it was shown that a minor excess bandwidth is needed to support end-to-end queueing delay requirements as low as 4ms.

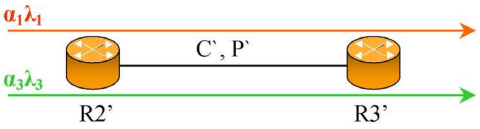

Figure 3. Scaled system when using DSCALEs.

In Section 4 we present simulation results for DSCALEs under the above assumption. However despite this assumption, DSCALEs can accurately predict metrics averaged over all groups of flows even when queueing delays are large. This is explained and demonstrated in [30].

Now, let's describe the method in detail. In the original topology (Figure 1), the link *R2-R3* is traversed by two group of flows: *grp1* and *grp3*. A packet of a flow that belongs to *grp1* will experience a (one-way) average end-to-end latency, equal to the sum of its queueing, transmission, and propagation delays. Thus, the total average latency from the source to the destination of a *grp1* packet equals

$$\frac{q_1 L}{C_1} + \frac{L}{C_1} + P_1 + \frac{q_2 L}{C_2} + \frac{L}{C_2} + P_2, \qquad (1)$$

where q_1 and q_2 are the average queue sizes (measured in number of packets, not counting the packet being transmitted) for the queues *R1-R2* and *R2-R3* respectively, and L is the packet size. Similarly, the end-to-end average latency of a *grp3* packet equals $\frac{q_2 L}{C_2} + \frac{L}{C_2} + P_2$. Assuming that the queueing and transmission delays are negligible in comparison to the propagation delays, the round-trip time for *grp1* flows is approximately equal to:

$$RTT_1 = 2\left(P_1 + P_2\right). \qquad (2)$$

Similarly, for *grp3* flows we have:

$$RTT_3 = 2P_2. \qquad (3)$$

For the same reasons as before, we remove link *R1-R2* from the original topology, and consequently, ignore *grp2* flows. The scaled system will consist of two groups of flows (*grp1*, *grp3*) and one link, denoted by *R2'-R3'*. Let C' be the capacity and P' be the propagation delay of this link.

On the scaled system (shown in Figure 3), the average one-way end-to-end latency of a packet belonging to either *grp1* or *grp3* equals $\frac{q' L}{C'} + \frac{L}{C'} + P'$, where q' is the average queue size on the scaled system. Assuming that the queueing and transmission delays are negligible (comparing to the end-to-end delays) in the scaled system as well, the round-trip time on this system is $RTT' = 2P'$.

Notice that the round-trip time on the scaled system is the same for both groups of flows. To capture the delay dynamics of both groups, we introduce two constants, a_1 and a_3,

and require that the following two equations hold simultaneously:

$$(P_1 + P_2) = a_1 P', \tag{4}$$

$$P_2 = a_3 P'. \tag{5}$$

In other words, the round-trip time of *grp1* flows is scaled by a factor of $\frac{1}{a_1}$ and the round-trip time of *grp3* of flows is scaled by a factor of $\frac{1}{a_3}$.

This is reminiscent of the situation in [1, 2], where the authors prove the following downscaling law: If network flows are sampled with some factor α and fed into a network replica whose link speeds are multiplied by α and propagation delays by $1/\alpha$, then performance extrapolation is possible. The main point of the law is to reduce the arrival rate and hence increase interarrival times via sampling by some amount, and slow down the network such that the round-trip time for every flow is increased by the same amount.

The difference here is that the round-trip times are scaled with different factors depending on the group of flows. Hence, we sample flows that belong to *grp1* with a factor α_1 and flows that belong to *grp3* with a factor α_3, as shown in Figure 3. Note that sampling only dictates whether a particular flow is present at the replica network or not. Once a flow is sampled, its packets traverse the network according to the TCP and network dynamics of the original or scaled system.

Let $a_1 + a_3 = \delta$ for some constant δ that we choose such that $a_1 \leq 1$ and $a_3 \leq 1$. (The specific value of this constant is not important.) Then, this equation together with Equations (4) and (5) can be solved to find a_1, a_3 and P'.

To find C' we require the traffic intensity in the congested link under study to be the same in the original and the scaled network. Traffic intensity is proportional to the ratio of the flow arrival rate over the link capacity. The total arrival rate of flows on link *R2-R3* is $\lambda_1 + \lambda_3$. Due to sampling, the total arrival rate of flows on the scaled system is $\alpha_1\lambda_1 + \alpha_3\lambda_3$. Hence, for traffic intensities to match,

$$C' = s \cdot C_2, \text{ where } s = \frac{\alpha_1\lambda_1 + \alpha_3\lambda_3}{\lambda_1 + \lambda_3}. \tag{6}$$

Applying the method with $\delta = 1$ we get $\alpha_1 = 0.6$, $\alpha_3 = 0.4$, $C' = 5$Mbps ($s = 0.5$) and $P' = 500$ms.

Remark: It is possible that the location of the congested links changes over time, e.g. due to changes in load conditions. It is easy to see that this will only result in changes to the scaled topology and does not affect the applicability of the methods.

3.3. Multiple bottlenecks

To demonstrate how our methods can be applied in cases where multiple bottlenecks exist, we consider the topology of the CENIC backbone [12]. The part of the CENIC backbone that we are interested in, along with link and flow information, is shown in Figure 4. The traffic matrix is given in Table 1. Note that for the ease of simulations we divided all the actual capacities by a factor of 100 [3]. This does not interfere with the study of topological scaling. We also multiplied all the real propagation delays by a same factor in order to maintain a same relation between transmission and propagation delays as in the actual topology.

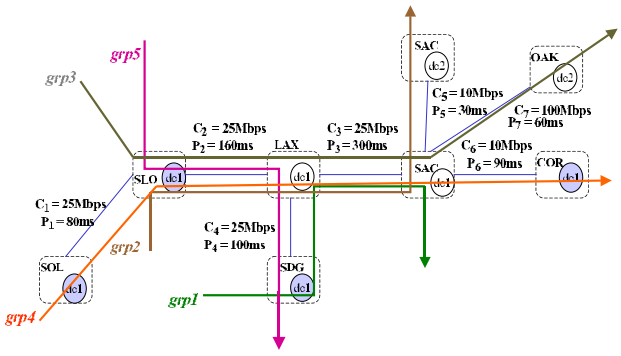

Figure 4. Part of the CENIC Backbone.

Group	Network Path
grp1	*SDG(dc1)-LAX(dc1)-SAC(dc1)*
grp2	*SLO(dc1)-LAX(dc1)-SAC(dc1)-SAC(dc2)*
grp3	*SLO(dc1)-LAX(dc1)-SAC(dc1)-OAK(dc2)*
grp4	*SOL(dc1)-SLO(dc1)-LAX(dc1)-SAC(dc1)-COR(dc1)*
grp5	*SLO(dc1)-LAX(dc1)-SDG(dc1)*

Table 1. Traffic matrix for Figure 4.

The input traffic that we impose forces links *SLO-LAX* and *LAX-SAC* to be congested. We are interested in studying link *LAX-SAC* which is traversed by four groups of flows: *grp1, grp2, grp3*, and *grp4*. Since some of the flows that traverse the congested link of interest, link *LAX-SAC*, also traverse the other congested link, link *SLO-LAX*, the scaled replica should consist of both links. Otherwise, the scaled topology will not capture the effect that the congested link *SLO-LAX* has on the flows that go through it, and performance prediction will be inaccurate. Flows within each group have exactly the same characteristics as in the simple topology. The routers use RED, with buffers sizes and parameters same as before.

Using DSCALEd, we can easily construct the scaled system shown in Figure 5. In this Figure, d_1, d_2, d_3, d_4 and d_5,

[3] By doing so, the total number of flows required to keep some links congested throughout the experiment are around half a million. Due to the static memory allocation that ns uses, the required RAM to run simulations with more flows is prohibitively large.

represent the fixed delay values that are added by the simulator to the round-trip time of groups *grp1*, *grp2*, *grp3*, *grp4* and *grp5* respectively. According to the method, d_1 must be equal to the sum of the propagation and transmission delays of link *SDG-LAX*, d_2 must be equal to the sum of these delays on link *SAC(dc1)-SAC(dc2)*, and so on. Their values are $d_1 = 100.3328$ms, $d_2 = 30.832$ms, $d_3 = 60.0832$ms, $d_4 = 171.1648$ms and $d_5 = 100.3328$ms.

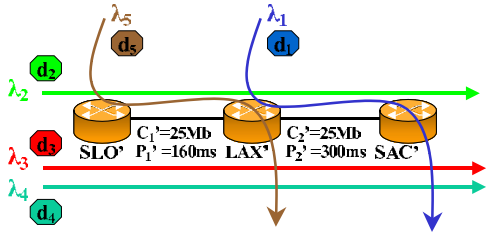

Figure 5. Scaled system with link and flow information (DSCALEd).

Using DSCALEs, we can construct the scaled system shown in Figure 6. Let $C1'$ be the capacity of link *SLO'-LAX'*, $C2'$ be the capacity of link *LAX'-SAC'*, and P' be the propagation delay of both links.

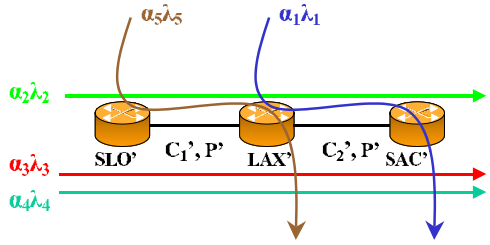

Figure 6. Scaled system with link and flow information (DSCALEs).

Following the same methodology as before, we can write the following set of equations:

$$(P3 + P4) = a_1 P', \quad (7)$$

$$(P2 + P3 + P5) = a_2 (P' + P'), \quad (8)$$

$$(P2 + P3 + P7) = a_3 (P' + P'), \quad (9)$$

$$(P1 + P2 + P3 + P6) = a_4 (P' + P'), \quad (10)$$

$$(P2 + P4) = a_5 P', \quad (11)$$

$$\sum_{i=1}^{5} a_i = \delta, \quad (12)$$

$$s_1 = \frac{\alpha_2\lambda_2 + \alpha_3\lambda_3 + \alpha_4\lambda_4 + \alpha_5\lambda_5}{\lambda_2 + \lambda_3 + \lambda_4 + \lambda_5}, \quad (13)$$

$$s_2 = \frac{\alpha_1\lambda_1 + \alpha_2\lambda_2 + \alpha_3\lambda_3 + \alpha_4\lambda_4}{\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4}, \quad (14)$$

$$C_1' = s_1 \cdot C2, \quad (15)$$

$$C_2' = s_2 \cdot C3. \quad (16)$$

Equations (7)...(11) correspond to groups of flows *grp1...grp5*. Equations (13)...(16) ensure that the traffic intensity on the two congested links is the same in the original and scaled network. All the equations together comprise a system of ten equations and ten unknowns ($\alpha_1 ... \alpha_5$, $C1'$, $C2'$, s_1, s_2, and P'). We solve the system using $\delta = 3$ and get: $\alpha_1 = 0.81$, $\alpha_2 = 0.49$, $\alpha_3 = 0.53$, $\alpha_4 = 0.64$, $\alpha_5 = 0.53$, $C_1' = 13.6875$Mbps, $C2' = 15.4375$Mbps and $P' = 493.34$ms.

In general, if we have N groups of flows that traverse K bottleneck links in total, we can always write a set of $N + 2K + 1$ equations with $N + 2K + 1$ unknowns.

4. Simulation results

In this section we use ns-2 [24] to investigate whether our methods can accurately predict the performance of IP networks from scaled-down replicas. We work with the simulation setups shown in Figures 1 and 4.

A word on network traffic properties is in order. It has been shown that the size distribution of flows on the Internet is heavy-tailed [25]. Hence, Internet traffic consists of a large fraction of short flows, and a small fraction of long flows that carry most of the traffic. Also, it has been recently argued that since network sessions arrive as a Poisson process [26, 27, 28] [4], network flows are *as if* they were Poisson [29]. (In particular, the equilibrium distribution of the number of flows in progress is *as if* flows arrive as a Poisson process.) We have taken these observations into account and considered heavy-tail distributed, Poisson flows.

We use the ns-2 built-in routines to generate sessions consisting of a single object each. This is what we call a flow

[4] That network sessions are Poisson is not surprising since a Poisson process is known to result from the superposition of a large number of independent user processes.

in the simulations. Each flow consists of a Pareto-distributed number of packets, with an average size of 12 packets and a shape parameter equal to 1.2.

4.1 Simple topology experiments

We begin by using DSCALEd to show that a number of performance measures of the original network depicted in Figure 1 can be predicted by the scaled replica depicted in Figure 2. (Recall that $d = 100.416ms$.) In particular, we compare the distribution of the number of active flows, the histogram of the flow transfer times (delays) and the distribution of the queue length in front of link *R2-R3* of the original topology and link *R2'-R3'* of the scaled topology. The flow arrival rates are the same within each group and equal 50 flows/sec. The results do not depend on whether the rates are equal or not, as we have verified by simulations.

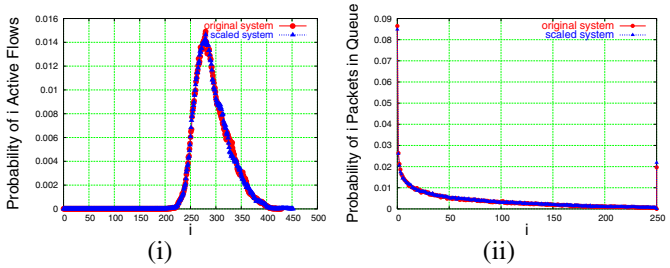

Figure 7. Distribution of (i) the number of active flows, and (ii) the queue length on the bottleneck link. (DSCALEd)

Figure 7(i) plots the distribution of active flows on links *R2-R3* and *R2'-R3'*. It is evident from the plot that the two distributions match. Figure 7(ii) plots the distribution of the queue lengths for the two links. Again, the two distributions match. Note that the right-most point on the plot gives the probability that there are more than 250 packets in the queue.

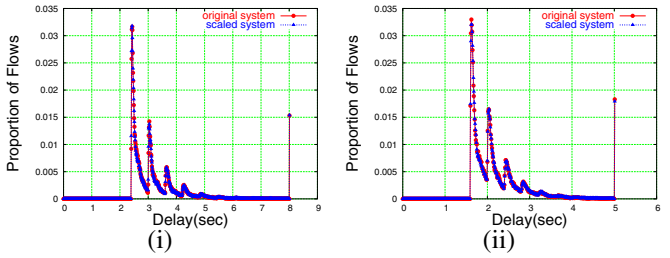

Figure 8. Histogram of delays of (i) *grp1*, and (ii) *grp3* flows. (DSCALEd)

Figure 8(i) plots the histogram of the flow transfer times (delays) for the flows of *grp1*, and Figure 8(ii) does the same for *grp3*. In both cases, we use delay chunks of 10ms each. It is evident from the plots that the distribution of the delays match. The peaks in the delay plot are due to the TCP slow-start mechanism. The left-most peak corresponds to flows which send only one packet and face no congestion, the portion of the curve between the first and the second peaks corresponds to flows which send only one packet and face congestion (but no drops), the next peak corresponds to flows which send two packets and face no congestion, and so on. The right-most point of Figure 8(i) represents the proportion of flows that belong to *grp1* and have a delay of more than 8sec. The right-most point of Figure 8(ii) represents the proportion of flows that belong to *grp3* and have a delay of more than 5sec.

We now proceed to use DSCALEs. Recall that the scaled replica is shown in Figure 3, where $\alpha_1 = 0.6$, $\alpha_3 = 0.4$, $C' = 5$Mbps and $P' = 500$ms.

A comment on how we sample flows is in order. Because of the heavy-tailed nature of the traffic, there is a small number of very large flows that has a large impact on congestion. To guarantee that we sample the correct number of these flows within each group, we separate flows into large (elephants) and small (mice) and sample exactly a proportion α_i, $i = 1, 3$, of them.

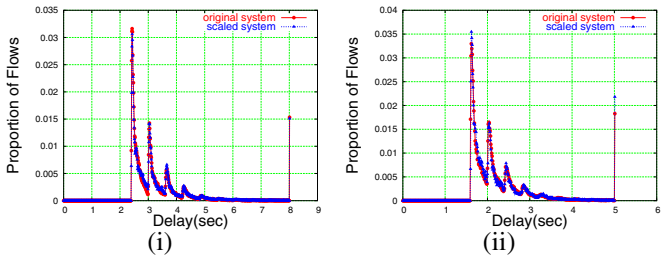

Figure 9. Histogram of normalized delays of (i) *grp1*, and (ii) *grp3* flows. (DSCALEs)

The plots of the distribution of active flows and the distribution of the queue lengths are identical between the two methods so we don't show them again. However, under the second method, the histogram of flow transfer times requires some normalization. Figure 9(i) plots the histogram of the normalized flow transfer times (delays) for the flows of *grp1*, and Figure 9(ii) does the same for *grp3*. In both cases, we use delay chunks of 10ms each, and the normalization is done as follows: (Normalized Flow Delay) $= a_i$ (Delay due to Propagation Time) $+ s$ (Total Queueing Delay), where a_i, $i = 1, 3$, are the sampling ratios and s equals $\frac{C'}{C_2}$, the ratio of the capacities of the congested link in the original and scaled network. While Equations (4) and (5) provide a justification for the multiplication with the a_i's, there is no rigorous justification for the use of s. The intuition behind this decision is

that if average queue sizes are the same in the two systems, queueing delay is inversely proportional to the link capacity. Simulations show that this decision yields accurate results. (Note that from a practical point of view, it is quite easy to separate in simulations the queueing delay from the rest of the delay. Hence, it is easy to perform the required normalization.) Returning to the figures, it is evident from the plots that the distribution of the normalized delays match.

4.2 CENIC backbone experiments

We now present simulation results for the topology shown in Figure 4, when using either of the proposed methods (the corresponding scaled systems are shown in Figures 5 and 6). The flow arrival rates are the same within each group and equal 70 flows/sec. Because of limitations in space we do not present results for all the groups of flows. We only present the flow delay histogram and the distribution of active flows for *grp5*. We also present the packet delay histograms for both links. Similar results hold for the rest of the groups of flows and for the queue length distributions.

We begin by using DSCALEd. The scaled system is shown in Figure 5. The fixed delay factors $d1...d5$ were computed in Section 3.

Figures (10) and (11) represent the simulation results when using DSCALEd. From the plots we can conclude that the scaled system produced by DSCALEd can predict the various performance measures of interest with a high accuracy.

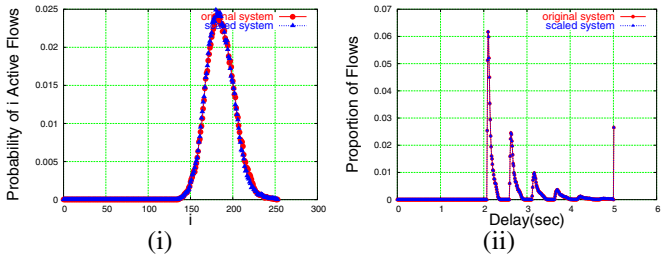

Figure 10. (i) Distribution of number of active *grp5* flows, and (ii) Histogram of delays of *grp5* flows. (DSCALEd)

We now move to the second method for which the scaled system is shown in Figure 6. The values of $\alpha_1...\alpha_5, C_1', C_2'$ and P' were computed in Section 3. The results are shown in Figures (12) and (13). The normalization of the packet queueing delays is done as follows: (Normalized Queueing Delay) $= s_i$ (Queueing Delay), where s_i, $i = 1, 2$, the ratio of the capacities of the congested links in the original and scaled network. For the packet delay histograms we use delay chunks of 0.3328ms each (equal to the transmission time

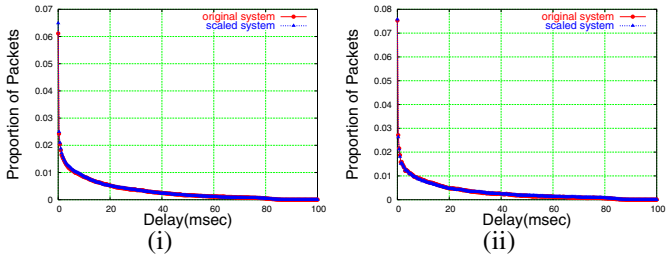

Figure 11. (i) histogram of packet delays on links *SLO-LAX* and link *SLO'-LAX'*, and (ii) histogram of packet delays on links *LAX-SAC* and link *LAX'-SAC'*. (DSCALEd)

of a packet over a 25Mbps link). The plots are similar as before and their explanation is the same. Also, the normalization of the flow delays is done in the same manner as before.

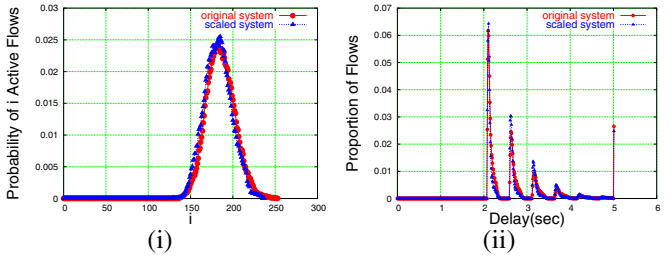

Figure 12. (i) Distribution of number of active *grp5* flows, and (ii) Histogram of normalized delays of *grp5* flows. (DSCALEs)

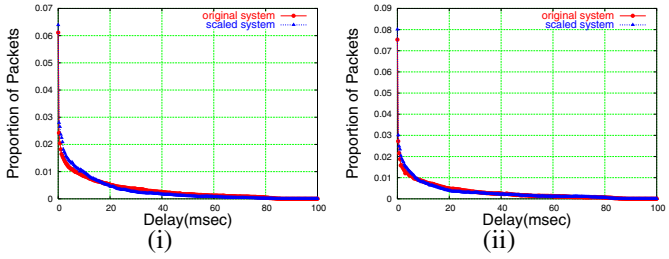

Figure 13. (i) histogram of normalized packet delays on links *SLO-LAX* and link *SLO'-LAX'*, and (ii) histogram of normalized packet delays on links *LAX-SAC* and link *LAX'-SAC'*. (DSCALEs)

It is obvious from the plots that the scaled system produced by DSCALEs can accurately predict the various per-

formance measures of interest.

Due to limitations of space, we do not provide here theoretical arguments explaining why our methods work so accurately. For such a discussion, the interested reader is referred to [30].

5. Performance savings

In this section, we present results which illustrate the effectiveness of the proposed methods in reducing the computational requirements of simulations. The metric of interest is the time needed to run an experiment. (This time should not be confused with the simulator's "virtual" time.) For our experiments we used a 2GHz processor with 3GB memory (RAM).

Method	Time(min)
Original	47
DSCALEs ($\delta = 1$)	15
DSCALEd ($\alpha = 1$)	42
DSCALEd ($\alpha = 0.5$)	14
DSCALEd ($\alpha = 0.1$)	4

Table 2. Simulation time for the simple topology.

Table 2 shows how effective our methods are when applied to the simple tandem topology shown in Figure 1. Simulating the simple tandem topology with a total of 240000 flows takes 47min, whereas using DSCALEs with $\delta = 1$, [5] takes only 15min. For this simple topology, the gain we get by using DSCALEd is not terribly exciting. To obtain more impressive savings, we can, of course, scale the system in time by a factor of $\alpha < 1$ [1, 2]. When we use an α of 0.1, the simulation time is only 4 min.

Table 3 shows the effectiveness of our methods when applied in a somewhat more complex topology like the one shown in Figure 4. The total number of flows used in this case was 450000. The corresponding scaled systems are shown in Figures 5 and 6.

Once again, the gain we get by using either of the methods is notable. In addition, it is worth noting the computational savings from DSCALEd even when sampling does not occur ($a = 1$).

Remarks: It is important to clarify the following about the nature of the performance savings due to sampling. The metrics of interest are distributions. For the observed histograms to converge to the corresponding distributions, enough events

[5] According to the method, different values of δ will result in different values of the sampling factors (α_i) and hence in a different total number of flows that are required to simulate the scaled system. This, in turn, will affect the time needed for the experiment.

Method	Time(min)
Original	815
DSCALEs ($\delta = 3$)	87
DSCALEd ($\alpha = 1$)	192
DSCALEd ($\alpha = 0.5$)	81
DSCALEd ($\alpha = 0.1$)	12

Table 3. Simulation time for the CENIC topology.

(e.g. packet arrivals) should occur. Hence, for example, instead of using DSCALEd with a total of N flows and sampling probability α, one might use a total of αN flows and no sampling. Similarly, choosing a smaller value for δ under DSCALEs might require a larger initial number of flows (before sampling) for convergence to occur. The true computational gains of the two methods come from the reduction of the number of links and nodes. These gains are expected to increase dramatically as the size of the network increases.

One question about DSCALEd remains unanswered: What is the complexity of adding a fixed delay to each packet? The reported simulation times imply that it is insignificant, but one might argue that as the size of the network increases this might be an issue. In particular, while it is very easy for a simulator to add a fixed delay to the round trip time of each packet, classifying each packet in order to add the proper delay might be computationally expensive if a very large number of source/destination pairs exist. However, hashing techniques can be used very efficiently for this purpose. In a network of n hosts there are at most $O(n^2)$ pairs, and for any meaningful values of n the associated complexity is trivial. For example, from an IP backbone point of view, the hosts mentioned above correspond to Points of Presence (POPs), and n is quite small. Indeed, according to the geographic maps found in [31] n is never more than 70.

6. Conclusion and Future Work

We propose two methods, DSCALEd and DSCALEs, to scale down an arbitrary network topology that is shared by TCP-like flows and controlled by various AQM schemes.

Both methods preserve the performance of the original system. Hence, they can be used to study large networks via small replicas. We show this via extensive simulations. For simple theoretical arguments the reader is referred to [30]. We also show that simulating a replica can be up to two orders of magnitude faster than simulating the original network. The computational gains might be even more pronounced for larger topologies than the ones that we study.

For future research we plan to perform experiments with real network topologies. In addition, we aim to investigate

optimal ways to achieve topological downscaling, and to study the applicability of these methods in wireless/ad-hoc networks. Finally, we plan to create a software tool that takes as input the original topology and traffic and produces a user-configurable downscaled network configuration.

References

[1] K. Psounis, R. Pan, B. Prabhakar, and D. Wischik, "The scaling hypothesis: Simplifying the prediction of network performance using scaled-down simulations," in *Proceedings of ACM HOTNETS*, 2002.

[2] R. Pan, B. Prabhakar, K. Psounis, and D. Wischik, "Shrink: A method for scalable performance prediction and efficient network simulation," in *Proceedings of IEEE INFOCOM*, 2003.

[3] C. Courcoubetis, G. Kesidis, A. Ridder, J. Walrand, and R. Weber, "Admission control and routing in ATM networks using inferences from measured buffer occupancy," *IEEE Transactions on Communications*, pp. 1778–1784, 1995.

[4] C. Estan and G. Varghese, "New directions in traffic measurement and accounting," in *Proceedings of ACM SIGCOMM*, 2002.

[5] D. Rubenstein, J. Kurose, and D. Towsley, "Detecting shared congestion of flows via end-to-end measurement," in *Proceedings of ACM SIGMETRICS*, 2000.

[6] C. Barakat, P. Thiran, G. Iannaccone, C. Diot, and P. Owezarski, "A flow-based model for internet backbone traffic," in *Internet Measurement Workshop*, 2002.

[7] K. Papagiannaki, *Provisioning IP Backbone Networks Based on Measurements*, Ph.D. thesis, University College London, March 2003.

[8] K. Papagianaki, S. Moon, C. Fraleigh, P. Thiran, F. Tobagi, and C.Diot, "Analysis of measured single-hop delay from an operational backbone network," in *Proceedings of IEEE INFOCOM*, 2002.

[9] C. Fraleigh, *Provisioning Internet Backbone Networks to Support Latency Sensitive Applications*, Ph.D. thesis, Stanford University, June 2002.

[10] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, and C. Diot, "Packet-level traffic measurements from the sprint IP backbone," in *IEEE Network*, 2003.

[11] C. Fraleigh, F. Tobagi, and C. Diot, "Provisioning IP backbone networks to support latency sensitive traffic," in *Proceedings of IEEE INFOCOM*, 2003.

[12] "Intermapper web server," `https://intermapper.engineering.cenic.org`.

[13] R. Pan, B. Prabhakar, K. Psounis, and M. Sharma, "A study of the applicability of a scaling hypothesis," in *Proceedings of ASCC*, 2002.

[14] V. Krishnamurthy, J. Sun, M. Fabloutsos, and S. Tauro, "Sampling internet topologies: How small can we go?," in *Proceedings of International Conference on Internet Computing*, 2003.

[15] T. Bu and D. Towsley, "On distinguishing between internet power law topology generators," in *Proceedings of IEEE INFOCOM*, 2002.

[16] M. Faloutsos, P. Faloutsos, and C. Faloutsos, "On power-law relationships of the internet topology," in *Proceedings of ACM SIGCOMM*, 1999.

[17] S. Bohacek, J. Hespanha, J. Lee, and K. Obraczka, "A hybrid systems modeling framework for fast and accurate simulation of data communication networks," in *Proceedings of SIGMETRICS*, June 2003.

[18] Y. Liu, F. L. Presti, V. Misra, and D. Towsley Y. Gu, "Fluid models and solutions for large-scale IP networks," in *Proceedings of ACM/SIGMETRICS*, June 2003.

[19] V. Misra, W. Gong, and D. Towsley, "A fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED," in *Proceedings of SIGCOMM*, 2000.

[20] D. Wischik, "The output of a switch, or, effective bandwidths for networks," *Queueing Systems*, vol. 32, 1999.

[21] G. de Veciana, G. Kesidis, and J. Walrand, "Resource management in wide-area ATM networks using effective bandwidths," in *IEE J. on Selected Areas in Comm., vol. 13, no. 6, pp. 1081–1090*, Aug. 1995.

[22] W. Fang and L. Peterson, "Inter-as traffic patterns and their implications," in *Proceedings of the 4th Global Internet Symposium*, December 1999.

[23] C. Fraleigh, C. Diot, B. Lyles, S. Moon, P. Owezarski, D. Papagiannaki, and F. Tobagi, "Design and deployment of a passive monitoring infrastructure," in *Proceedings of the Workshop on Passive and Active Measurements, PAM*, April 2001.

[24] "Network simulator," `http://www.isi.edu/nsnam/ns`.

[25] W. Willinger, M. S. Taqqu, R. Sherman, and D. V. Wilson, "Self-similarity through high-variability: Statistical analysis of ethernet lan traffic at the source level," *IEEE/ACM Transactions on Networking*, vol. 5, no. 1, pp. 71–86, 1997.

[26] A. Feldmann, A. C. Gilbert, and W. Willinger, "Data networks as cascades: Investigating the multifractal nature of internet wan traffic," in *Proceedings of ACM SIGCOMM*, 1998.

[27] C. J. Nuzman, I. Saniee, W. Sweldens, and A. Weiss, "A compound model for tcp connection arrivals," in *Proceedings of ITC Seminar on IP Traffic Modeling*, Monterey, 2000.

[28] V. Paxson and S. Floyd, "Wide area traffic: the failure of Poisson modeling," *IEEE/ACM Transactions on Networking*, vol. 3, no. 3, pp. 226–244, 1995.

[29] S. Ben Fredj, T. Bonalds, A. Prutiere, G. Gegnie, and J. Roberts, "Statistical bandwidth sharing: a study of congestion at flow level," in *Proceedings of SIGCOMM*, 2001.

[30] F. Papadopoulos, K. Psounis, and R. Govindan, "Performance preserving network downscaling," Tech. Rep. CENG-2004-13, University of Southern California, 2004.

[31] "Rocketfuel: An ISP topology mapping engine," `http://www.cs.washington.edu/research/networking/rocketfuel/interactive`.