Performance Preserving Topological Downscaling of Internet-Like Networks

Fragkiskos Papadopoulos, Konstantinos Psounis, *Member, IEEE*, and Ramesh Govindan

***Abstract*—The Internet is a large, 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 complex networks like the Internet: 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. This is complementary to the work of Psounis *et al.*, 2003, where the authors presented a way to scale down the Internet in *time*, by creating a slower replica of the original system.**

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. Using extensive simulations with transmission control protocol (TCP) traffic and theoretical analysis, we show that it is possible to achieve this kind of performance scaling even on topologies the size of the CENIC backbone (that provides Internet access to higher education institutions in California). We also show that simulating a scaled topology can be up to two orders of magnitude faster than simulating the original topology.

***Index Terms*—Efficient network simulation, performance prediction, topology downscaling, transmission control protocol (TCP)/closed-loop networks.**

I. Introduction

NETWORKING research has taken a multipronged 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 [2]–[7], to measurement-based performance characterizations [8]–[12] to simulation studies [13]–[16]. This is appropriate given the overwhelming size, complexity, heterogeneity, and the speed of operation of the Internet.

This multipronged 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, but it can be hard to work backwards—to infer the performance of the original system from sampled traffic, and, researchers attempt to use "realistic" topology models in their simulations, but topology modeling is still in its infancy—realistic models that include notions of capacity and latency are some years away.

The focus of this work and of its predecessor [1], [17] 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*. Studying engineering artifacts by scaling them down (in addition to simulating them) has long been an established tradition in other disciplines. For example, civil engineers not only study finite-element models of large structures, they also build scale models that attempt to faithfully replicate the proportional loading on various structural elements. We attempt to bring this methodology to computer networking.

But what does it mean to design a performance-preserving network downscaling technique? Consider a large network [such as that of a backbone Internet service provider (ISP)], consisting of several hundred nodes and links, and traversed by hundreds of thousands of flows. Psounis *et al.* [1] 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 original network.

Topological downscaling has three benefits. First, by relying only on a sample of the traffic, it reduces the amount of data we

Manuscript received October 1, 2005; revised April 29, 2006.

This paper was presented in part at the 38th Annual Simulation Symposium, San Diego, CA, April 2005.

F. Papadopoulos is with the Department of Electrical Engineering, University of Southern California, Los Angeles, CA 90089 USA (e-mail: fpapadop@usc.edu).

K. Psounis is with the Department of Electrical Engineering and the Department of Computer Science, University of Southern California, Los Angeles, CA 90089 USA (e-mail: kpsounis@usc.edu).

R. Govindan is with the Department of Computer Science, University of Southern California, Los Angeles, CA 90089-0781 USA (e-mail: ramesh@usc.edu).

Digital Object Identifier 10.1109/JSAC.2006.884029

[1]SHRiNK: Small-scale hi-fidelity reproduction of network kinetics.

need to work with. Second, by using actual traffic, it shortcuts 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, expedites simulations dramatically, and allows 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 along the path of each flow that introduce dependencies among flows and sizeable queueing delays [3], [8], [9], [18]–[20]. Further, it has been recently shown that links with capacities large enough to carry many flows without getting congested, are in a sense, transparent to the flows that pass through them [21]–[24]. 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 [25] 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, the existence of two-way traffic, 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 II briefly discusses prior work on scaling down networks. Section III gives a formal definition for congested links and discusses the impact of congested and uncongested parts of a network to performance. Section IV introduces the proposed methods and applies them to simple topologies, and to realistic topologies, where multiple congested links exist and two-way traffic is present (CENIC backbone [25]). Extensive simulation results are presented in Section V. It is shown that a variety of metrics, including packet queueing delays, end-to-end flow delays, number of active flows, acknowledgment (ACK) delays, etc., can be predicted from the replica. Formal proofs establishing the validity of the methods are presented in Section VI. Finally, Section VII examines how the amount of downscaling used by the methods is related to computational savings—in terms of the time needed to run an experiment—and accuracy, and Section VIII concludes this paper.

II. Related Work

Psounis *et al.* [1], [17], [26], [27] 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. An important result of this work is that for networks in which TCP flows arrive at random times and whose sizes are heavy-tailed, i.e., for networks representative of the Internet, performance measures such as the distribution of the number of active flows and of their transfer times are left virtually unchanged. This is verified using extensive simulations and theoretical arguments. 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. The most relevant to our work is the one by Eun *et al.* [21]–[24]. In this line of work, the authors show analytically that in a tandem of queues where arrivals are "fluid-like" or are dictated by Poisson-point-processes, as the capacity of upstream queues and the number of flows through them increases, the performance of a downstream queue is about the same, as if all the upstream queues did not exist. This work considers the network as being an "open-loop" system. In our work, we want to investigate if network downscaling is possible in the Internet, where traffic is dictated by TCP-like mechanisms, that is, we are studying "closed-loop" systems.

Other studies, e.g., by Krishnamurthy *et al.* [28], have compared small and large network graphs with respect to their graph properties. In particular, they have used metrics like the average degree of a graph, the clustering coefficient [29], or the degree exponent [30], in an effort to find connections between small and large networks. Our approach is very different. We want to create network miniatures and predict the performance of the original networks from the miniatures, and we are primarily interested in network-centric metrics, like flow transfer times, packet delays, queue sizes, and so on.

One of the practical benefits of working with network miniatures is significantly faster simulations. Relevant to this are some works that attempt to replace time consuming packet-level simulations by modeling only some parts/aspects of the original system. Liu *et al.* [2] extend the fluid models introduced in [31] to be topology-aware, and predict network dynamics by solving the fluid models over a subset of the original links. They show that this takes less time than packet-level simulations, especially when workloads and bandwidths are high. Bohacek *et al.* [32] propose a hybrid modeling/simulation framework that uses averaging of discrete variables over short time intervals, and can predict network behavior faster than packet-level simulations. Of course, packet level simulations are more realistic than these approaches, and even more so are network experiments in testbeds. Our methods create small-scale networks that can be directly used in simulations or in testbeds, ensuring the relevance of the results.

III. Congested Links: Definition and Impact

The widely used term "congested link" has various meanings in the literature today, see, for example, [10] and [33]–[35]. Typically, it is defined to be a link with either high utilization, low available bandwidth, high loss rate, long queueing

delay, or a combination thereof.[2] While these definitions are not necessarily equivalent, a common property of most types of "congested" links is that they alter the packet arrival process as packets go through the link. For example, this is clearly the case when drops occur, or when queueing delays are long and change over time. Taking into consideration our goal to create small, performance-preserving network replicas by ignoring "uncongested" links, it is natural to define the latter as precisely those links that do not alter the packet arrival process. For simulation purposes, we consider a link to be "uncongested" if no drops occur and the average queueing delay incurred to a packet of any flow that traverses this link is one order of magnitude smaller than the total end-to-end queueing delay of the packet.

Topological downscaling leverages the observation that ignoring uncongested links may not result in a loss of performance-related information about the original network, since the packet arrival process is almost the same before and after an uncongested link, no dependencies among flows are introduced by such links, and queueing delays due to uncongested links do not have a significant contribution to end-to-end packet delays.[3] Similar arguments have been used to compute end-to-end queueing delays through a backbone network [18]–[20], and to model the backbone traffic at the flow level [3]. Real traces have verified the insignificance of queueing delays due to uncongested links [8], [9]. Similar results have been derived in [5] and [38] for traffic which observes the large deviations principle, and in [21]–[24] for "fluid-like" traffic and traffic that is dictated by Poisson point-processes in the context of open-loop networks.

IV. Scaling Down Network Topology

In this section, we present two methods for scaling down the topology of a network, while preserving its performance. The proposed methods operate on a given topology with known traffic. (There is a large number of efficient tools that can estimate source/destination pairs, the corresponding rates, the paths followed by network flows, and the links that are congested, see, for example, [39]–[41].) Under both methods, the downscaled network consists of congested links from the original topology. The first method, called downscale using delays (DSCALEd), accounts for the missing uncongested links by adding appropriate fixed delays to all packets. The second method, called downscale using sampling (DSCALEs), accounts for the missing links by sampling flows and properly adjusting the capacities and propagation delays of the scaled network.

A simple topology used to introduce the methods is shown in Fig. 1(i), consisting of two links in tandem and three routers $R1$, $R2$, and $R3$. Packets are considered to belong to flows in accordance with the usual practice [8], [42], [43], and flows that follow the same path are grouped into groups. There are three groups of flows, $grp1$, $grp2$, and $grp3$. We vary the flow arrival rates within each group, λ_i, $i = 1, 2, 3$, so that the link $R2-R3$ becomes congested, whereas the link $R1 - R2$ remains uncongested. We are interested in creating a replica and predicting various performance measures on link $R2 - R3$, which is the congested link. A detailed description of the proposed methods that accomplish this task follows.

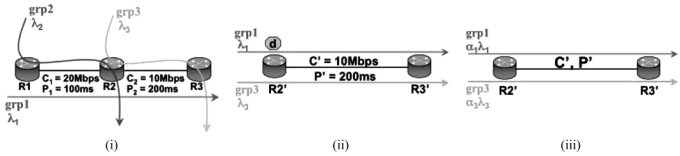


Fig. 1. (i) Original network topology used to introduce the downscaling techniques. (ii) Scaled system when using DSCALEd. (iii) Scaled system when using DSCALEs.

A. DSCALEd: Downscaling Using Delays

The first method is quite intuitive 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 congested links in the scaled system.
3) Groups of flows that do not traverse congested 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 flow is equal to that in the original system.

Let us illustrate the method by applying it to the network in Fig. 1(i). Since link $R1 - R2$ is uncongested, we remove it 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 Fig. 1(ii). To compensate for the absence of link $R1 - R2$ from the scaled system, a fixed delay d is introduced to each $grp1$ 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.

Note that when the original topology and the number of source/destination pairs are large, adding the fixed delays at exactly the point where uncongested network segments existed in the original network may require significant bookkeeping. For this reason, we choose to add the fixed delays at the source of each packet, irrespectively of where along its original path these delays occur. (This approach introduces different time shifts to packet arrival times, depending on the path of each packet. In Section VI, we establish theoretically that the packet arrival process is not altered by these different time shifts as long as flow arrivals are independent and stationary.)

[2]Every link is fed with packets from a buffer, e.g., the buffer of a router line card. The properties of this buffer, e.g., queueing delay, are also considered properties of the corresponding link.

[3]In a network where paths are very long, a large number of uncongested links might have collectively a significant impact on the total end-to-end queueing delay of a packet. We ignore this situation based on the well-known fact that Internet paths are quite short; they are typically less than 14 hops long, and rarely more than 18 hops long [36], [37].

Remark: While it is very easy for a simulator to add a fixed delay to the round-trip time (RTT) 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 (Internet protocol) backbone point of view, the hosts correspond to points of presence (POPs), and n is less than 100 [44].

B. DSCALEs: Downscaling Using Sampling

In Section II, we referred to a different type of network downscaling that creates a slower replica of the original network [1], [17], [26]. This time downscaling can be applied independently to a small network replica created using the previous method, in order to further reduce the amount of traffic one works with. Yet, it is interesting to investigate if one can seamlessly combine size and time downscaling in a single method. Another reason to look for more methods to downscale a network is to investigate if it is possible to avoid adding a fixed delay to each packet, which is a requirement of DSCALEd.

With the above in mind, we present a method that preserves performance by sampling flows and carefully choosing the capacities and propagation delays of the links of the replica network. (The reason why this method combines size and time downscaling will become apparent shortly.) The first three steps of the method are identical to those of DSCALEd. The last two steps can be summarized as follows.

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 the round trip times of each group of flows remain unchanged (except by a constant multiplicative factor), and 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 *total end-to-end* queueing (and transmission) delay is negligible in comparison to the *total* end-to-end delay. This assumption is primarily made for ease of exposition. DSCALEs can accurately predict a large number of metrics, but not all, even when queueing delays are large, as we show via theory and simulations in Sections VI and V, respectively. Nevertheless, it is interesting to point out that the negligible queueing delay assumption is not unrealistic for IP backbone networks, where although a packet may experience significant queueing in some links compared with others (e.g., at network access points versus links at the core), its total end-to-end delay is mainly dominated by the propagation delay [8], [18]–[20].

Now, let us describe the method in detail. Given our assumption, in the original topology [Fig. 1(i)], the RTT for $grp1$ flows is approximately equal to $RTT_1 = 2(P_1 + P_2)$. Similarly, for $grp3$ flows, we have $RTT_3 = 2P_2$. 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. The RTT on the scaled system [shown in Fig. 1(iii)] is approximately equal to $RTT' = 2P'$ 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:

$$(P_1 + P_2) = a_1 P', \tag{1}$$

$$P_2 = a_3 P'. \tag{2}$$

In other words, the RTT of $grp1$ flows is scaled by a factor of $1/a_1$ and the RTT of $grp3$ of flows is scaled by a factor of $1/a_3$.

This is reminiscent of the situation in [1] and [17], where the following time downscaling law is proved: *If network flows are sampled with some factor a and fed into a network replica whose link speeds are multiplied by a and propagation delays by $1/a$, then performance extrapolation is possible*. Intuitively, this is possible because the flow interarrival times and the RTTs of all packets are increased by the same factor. The difference here is that RTTs are scaled with different factors depending on the group of flows. With this in mind, we sample flows that belong to $grp1$ with a factor a_1 and flows that belong to $grp3$ with a factor a_3,[4] as shown in Fig. 1(iii). In Section VI, we provide a rigorous justification for these sampling factors.

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 smaller the value of δ is, the more intense traffic sampling is.) Then, this equation together with (1) and (2) can be solved to find a_1, a_3, and P'. Finally, 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. Since the total arrival rate of flows on link $R2 - R3$ is $\lambda_1 + \lambda_3$, and on the link of the scaled system is $a_1\lambda_1 + a_3\lambda_3$, for the traffic intensities to match, we require

$$C' = s \cdot C_2, \quad \text{where } s = \frac{a_1\lambda_1 + a_3\lambda_3}{\lambda_1 + \lambda_3}. \tag{3}$$

Applying the method with $\delta = 1$, we get $a_1 = 0.6$, $a_3 = 0.4$, $C' = 5$ Mb/s, and $P' = 500$ ms. Notice that the scaled system runs slower than the original, since the link speeds are smaller $(C' < C_2)$, and the flow interarrival times are larger due to sampling.

Remarks: The methods do not depend on the potential location of the congested links. These can be anywhere, i.e., inside the core, at public exchange points, at the edges of the network, etc. It is also possible that the location of the congested links changes over time, e.g., due to changes in load conditions. In such situations our methods would merely need to be reapplied to the altered topology and traffic matrix.

Further, it is straightforward to study uncongested links using either of the methods. 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.

C. Multiple Bottlenecks

To demonstrate the applicability of our methods to larger topologies with multiple bottlenecks, we consider the topology of the CENIC backbone [25], which consists of 25 nodes and

[4]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.

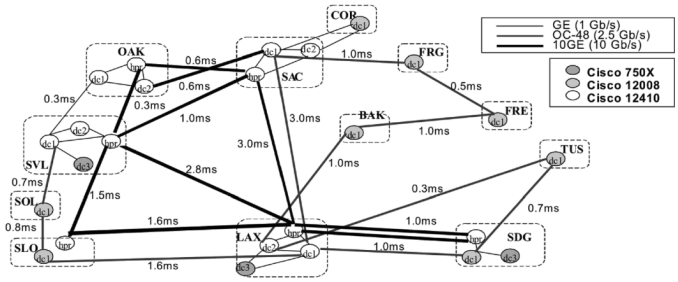


Fig. 2. The CENIC backbone.

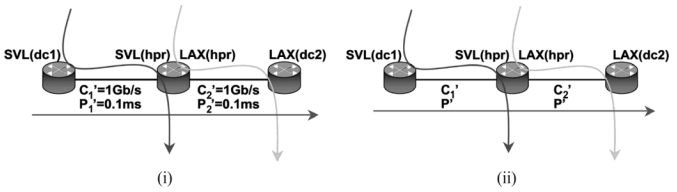


Fig. 3. (i) Scaled system with link and flow information when using DSCALEd. (ii) Scaled system with link and flow information when using DSCALEs.

41 links and is shown along with link information in Fig. 2. Note that the CENIC maps do not include information about the propagation delays of the links and the paths of the packets that traverse them. We estimate the propagation delay of a link by dividing the length of the link over the propagation velocity of the signal (taken as 133 000 miles/s). The propagation delay for all the links that belong to the same geographic area is taken as 0.1 ms and for the rest of the links is shown in Fig. 2 (appended next to each link).

We let each possible source-destination pair in the topology to correspond to a group of flows. (Notice that links are bidirectional.) Hence, in total, there are 600 groups of flows. The input traffic that we impose forces links $SVL(hpr) - SVL(dc1)$ and $LAX(dc2) - LAX(hpr)$ to be congested.[5] We are interested in studying the link $SVL(hpr) - SVL(dc1)$, which is traversed in both directions by a total of 70 groups of flows (35 per direction of the link). Since some of the flows that traverse the congested link of interest, link $SVL(hpr) - SVL(dc1)$, also traverse the other congested link, link $LAX(dc2) - LAX(hpr)$, the scaled replica should consist of both links. Otherwise, the scaled topology would not capture the effect that the congested link $LAX(dc2) - LAX(hpr)$ has on the flows that go through it, and performance prediction would be inaccurate. (For results where we consider only the one congested link, see [45].)

Using DSCALEd, we can easily construct the scaled system shown in Fig. 3(i).[6] Each arrow represents all groups of flows that traverse the corresponding link in the original topology in the direction of the arrow. Note that the figure shows only the left-to-right direction of the traffic. The two links are being traversed in the opposite direction in exactly the same way. Also, the figure shows the capacity and propagation delay of the links in the left-to-right direction only. The other direction has the same values. In the scaled topology there are a total of 120 groups of flows. An appropriate fixed delay value is added to the RTT of all the packets of each group, as explained before. For example, let $grp1$ be the group of flows which in the original topology follows the path: $SDG(dc3) - SDG(dc1) - SDG(hpr) - LAX(hpr) - SVL(hpr) - SVL(dc1) - SVL(dc3)$. In the scaled topology this group traverses link $SVL(hpr) - SVL(dc1)$ only, and the fixed delay added to all $grp1$ packets equals $\sum_i (P_i + L/C_i) = 4.127$ ms, where P_i is the propagation delay of link i, C_i is the capacity of link i, and the sum is over all links along the path of the group in the original system that are not present in the scaled system. For another example, let $grp2$ be another group of flows which in the original topology follows the path: $SVL(dc3) - SVL(dc1) - SVL(hpr) - LAX(hpr) - LAX(dc2) - TUS(dc1)$. In the scaled topology, $grp2$ traverses links $SVL(dc1) - SVL(hpr)$ and $LAX(hpr) - LAX(dc2)$, and the fixed delay added to all $grp2$ packets is 3.213 ms. Continuing this way we can compute the fixed delay for all the groups of flows.

Using DSCALEs, we can construct the scaled system shown in Fig. 3(ii). The arrows show the direction of the traffic as before. We focus again on the left-to-right direction. Let $C_1 = 1$ Gb/s be the capacity of link $SVL(dc1) - SVL(hpr)$ in the original topology, and C_1' in the scaled topology. Similarly, let $C_2 = 1$ Gb/s be the capacity of link $LAX(hpr) - LAX(dc2)$ in the original topology, and C_2' in the scaled topology. Finally, let P' be the propagation delay of both links in the scaled topology, and let I_i^j equal 1 if grp_i traverses link j and 0 otherwise. In the rest of this section, we refer to link $SVL(hpr) - SVL(dc1)$ as link 1, and to link $LAX(dc2) - LAX(hpr)$ as link 2.

Following the same methodology as in the simple topology scenario, we can write the following equations for $grp1$ and $grp2$, respectively

$$\sum_{i=1}^{41} I_1^i P_i = 4.2 = a_1 P' \tag{4}$$

$$\sum_{i=1}^{41} I_2^i P_i = 3.4 = a_2 (P' + P') \tag{5}$$

where P_i is the propagation delay of link i in the original topology. Notice that the first sum has nonzero terms for all links along the path of $grp1$, and the second sum for all links along the path of $grp2$. Continuing this way we can write down similar equations for the rest of the 118 groups of flows.

Following the methodology, we also write the following set of equations that ensure that the traffic intensity on the two congested links (from left-to-right) is the same in the original and scaled network:

$$C_1' = s_1 \cdot C_1, \quad \text{where } s_1 = \frac{\sum_{i=1}^{120} a_i \lambda_i I_i^1}{\sum_{i=1}^{120} \lambda_i I_i^1} \tag{6}$$

$$C_2' = s_2 \cdot C_2, \quad \text{where } s_2 = \frac{\sum_{i=1}^{120} a_i \lambda_i I_i^2}{\sum_{i=1}^{120} \lambda_i I_i^2}. \tag{7}$$

[5]In this scenario, the congested links of the original network were identified through the simulator. However, in practice a simple tool, e.g., *pathchar*, can easily identify the congested links of the actual original network. For a taxonomy of such performance measurement tools, see [41].

[6]This process can be automated. We have tools, available upon request, that automatically perform most of the required tasks, and we plan to create a software tool that puts everything together.

Notice that because of traffic symmetry, the factors s_1 and s_2 are the same on both directions of the links, and hence it suffices to compute them on one direction only. (In the next section, we discuss two-way traffic scenarios in more depth.)

Finally, as in the simple topology scenario, we have

$$\sum_{i=1}^{120} a_i = \delta \tag{8}$$

where δ is a constant design parameter that dictates the overall sampling intensity, chosen such that the sampling factors $a_i \leq 1$ for all i.

All the equations together comprise a system of 125 equations (one from each of the 120 groups, two from each of the two congested links, and one from the constant δ) and 125 unknowns ($a_1, \ldots, a_{120}$, C_1', C_2', s_1, s_2, and P'). We solve the system using $\delta = 60$ and get: $C_1' = 450$ Mb/s, $C2' = 550$ Mb/s, $P' = 4.78$ ms, and the values of $a_1 \ldots a_{120}$ ($a_1 = 0.88$, $a_2 = 0.36$, etc.). Note that 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.

D. A Closer Look at Two-Way Traffic Scenarios

Traffic on bidirectional network links is often asymmetric [46]. This traffic asymmetry may result from the nature of the applications. For example, web and ftp applications are inherently asymmetric; one direction carries small request messages and the other direction carries the actual data. In this case, one can assume that the TCP ACKs for the data will experience negligible queueing delays. This is the case in the simple topology used to introduce the two methods.

However, there are links in which data transfer occurs in both directions (two-way traffic). In such cases, ACK packets interact with data packets and may experience significant queueing delays. They can also become compressed together, and hence loose their important clocking properties. In the literature, this phenomenon is called *ACK compression*, see, for example, [12] and [47].

DSCALEd can be applied in the case of two-way traffic without any modification. Applying DSCALEs is more involved. In general, the corresponding set of equations of the DSCALEs method yield two different scaling factors, one for each direction. Hence, one would scale the capacity along one direction with a different factor than that along the other direction. This would scale the dynamics of data packets and acknowledgments of the same flow differently, and, as a result, DSCALEs may suffer from inaccuracies. Note that this is not an issue when the two-way traffic is "symmetric," that is, when flows on both directions of a link arrive with the same rates and follow the same paths inside the network. In this case, the capacities of both directions are scaled by the same amount. This is the case in the scenario with the CENIC backbone topology. In the next section, we provide simulation results for both one-way, and two-way traffic scenarios. Due to limitations of space, we do not show results for scenarios with two-way "asymmetric" data traffic. The interested reader is referred to [45] for results from such scenarios.

Remark: While tools for available bandwidth and capacity measurement exist, e.g., see [41], these are insufficient for our purposes, for several reasons. First, to offer comparable functionality to downscaling, a measurement infrastructure would need to be deployed that allows network operators to use these tools between arbitrary pairs of nodes in an ISP network. By contrast, downscaling uses information already known (via simple network management protocol (SNMP) or other standard techniques) to ISP operators: congested routers, the traffic matrix, and topology. Second, since these tools only test traffic along a single path, it would be difficult (and perhaps nonscalable) to use them to infer delay and active flow distributions across all flows traversing a congested router, quantities that downscaling can estimate.

V. Simulation Results

In this section, we use ns-2 [48] 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 Figs. 1(i) and 2.

Each source in the network generates flows that arrive at random times and have heavy-tailed sizes. This ensures the relevance of the input traffic since: 1) it is a well-known fact that the size distribution of flows on the Internet is heavy-tailed, see, for example, [7], and 2) flow interarrival times on the Internet are random. In most of our experiments, we have used exponential flow interarrival times. (Of course, while this implies that flow arrival times are Poisson, packet arrivals are dictated by the TCP dynamics.) We did this, following recent trends in traffic modeling according to which since network sessions arrive as a Poisson process [11], [49], [50],[7] network flows are *as if* they were Poisson [4]. (In particular, the equilibrium distribution of the number of flows in progress is *as if* flows arrive as a Poisson process.) We have also run simulations where flow arrival times are dictated from real traces and found very similar results. Note that DSCALEd works for any arrival process, whereas DSCALEs, in theory, requires arrivals to be Poisson (see Section VI for formal proofs.) Interestingly, the simulations with real traces show that, in practice, DSCALEs is relatively accurate even when arrivals are arbitrary.

We use the ns-2 built-in routines to generate sessions consisting of a single object each. This is what we call a flow 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. Finally, the buffers on the routers use either DropTail or random early detection (RED). In the simple topology, they can hold 300 packets and the RED's parameters are $\min_{\text{th}} = 100$, $\max_{\text{th}} = 250$, and averaging parameter $w = 0.00005$. In the CENIC topology they can hold 400 packets and the RED's parameters are $\min_{\text{th}} = 150$, $\max_{\text{th}} = 350$, and $w = 0.00005$.

In the rest of the section, we compare the distribution of the number of active flows on the original and the scaled network, the end-to-end delay histograms of the groups of flows in the two networks, and the distribution of the packet queueing delays

[7]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.

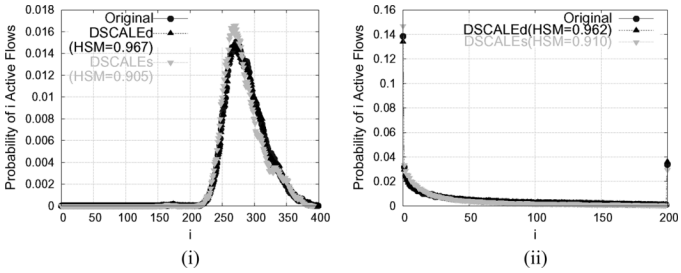


Fig. 4. (i) Distribution of the number of active flows on the bottleneck link. (ii) Distribution of the queue length on the bottleneck link (simple tandem topology).

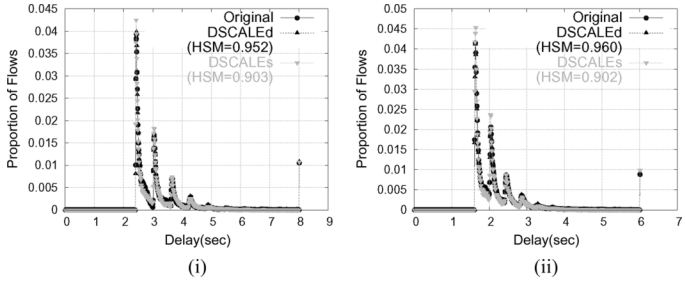


Fig. 5. Histogram of end-to-end flow delays of (i) $grp1$, and (ii) $grp3$ flows. (Simple tandem topology.)

and the ACK queueing delays on the congested links of the two networks. As usual, a flow in the network is considered active if the session between the source and the destination has not been terminated yet. And, the end-to-end delay of a flow is the time interval between the arrival of its first packet to the network and the departure of its last packet from the network.

In addition to visually comparing distributions and histograms to evaluate the accuracy of performance prediction, it is also important to quantify differences using statistical measures. To this end, to compare two histograms, say H_1 and H_2, we use the following histogram similarity measure: $HSM = 1 - C_v$, where $C_v = \sqrt{\chi^2/2}$ is the Cramer's V coefficient, and $\chi^2 = \sum_i ((H_1(i) - H_2(i))^2/(H_1(i) + H_2(i)))$ is the well-known chi-square statistic [51]. The sum is over all histogram chunks and $H_j(i)$ (j = 1,2) is the frequency of the events observed in the i^{th} chunk. Notice that HSM takes values in [0,1], with 1 indicating that the two histograms are identical (in this case $\chi^2 = 0$). The HSM values for each plot and method are shown in the captions of the plots.

A. Simple Topology Experiments

We show that a number of performance measures of the original network depicted in Fig. 1(i) can be predicted by the scaled replica depicted in Fig. 1(ii) (DSCALEd) and the scaled replica depicted in Fig. 1(iii) (DSCALEs). The flow arrival rates are set to 51 flows/s for all groups of flows. (The results do not depend on whether the rates are equal or not.) The resulting average packet queueing delay on link $R2 - R3$ is 41 ms and on link $R1 - R2$ is 2 ms. The drop probability on link $R2 - R3$ is approximately 4% and no drops occur on link $R1 - R2$. The link of interest is the congested link $R2 - R3$ of the original topology.

We begin by using DSCALEd. Recall that the fixed delay factor d that is added to all $grp1$ packets equals 100.416 ms. Fig. 4(i) plots the overall (over all groups of flows) distribution of active flows on links $R2 - R3$ and $R2' - R3'$. It is visually evident from the plot that the two distributions match. [The distributions of the number of active $grp1$ and $grp2$ flows also match between the two systems. We do not present the plots since they are similar to Fig. 4(i).] Fig. 4(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 200 or more packets in the queue.) Finally, Fig. 5(i) and (ii) plot the histogram of the end-to-end flow delays for the flows of $grp1$ and $grp3$, respectively. It is evident from the plots that the distribution of the delays match. [In the flow delay histograms, we use delay chunks of 10 ms. The peaks in the 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 Fig. 5(i) [Fig. 5(ii)] represents the proportion of flows that belong to $grp1$ ($grp3$) and have a delay of more than 8 s (6 s).]

We now proceed to DSCALEs. Recall that the sampling factors equal $a_1 = 0.6$ and $a_3 = 0.4$, and that the link capacity and propagation delay equal $C' = 5$ Mbps and $P' = 500$ ms, respectively. The results for DSCALEs are also shown in Figs. 4 and 5, and are very similar to those for DSCALEd. Note that under DSCALEs, the histogram of flow transfer times requires some normalization. Fig. 5(i) and (ii) plot the histogram of the *normalized* end-to-end flow delays for the flows of $grp1$ and $grp3$, respectively. The normalization is done as follows: $(\text{normalized flow delay}) = a_i$ $(\text{delay due to propagation time}) + s$ (queueing delay), where a_i, i = 1,3, are the sampling ratios and s equals C'/C_2, the ratio of the capacities of the congested link in the original and scaled network. Equations (1) and (2) provide the justification for the multiplication with the a_i's. A rigorous justification for the use of s is given in Section VI. Intuitively, if queue sizes are the same in the two systems, the queueing delay is inversely proportional to the link capacity.

In summary, both methods are quite accurate: they achieve HSM values that are at least 0.90. Clearly, it is not possible to achieve HSM values equal to one, since both scaled systems ignore the small, yet nonzero, queueing delay of link $R1 - R2$. DSCALEd is a bit more accurate than DSCALEs. The former yields HSM values larger than 0.95, while the later yields values that are at least 0.90. This is because DSCALEs employs traffic sampling. As a result, it uses a smaller fraction of the original traffic than DSCALEd (see Table I for the values of these fractions), and, the smaller the fraction of the original traffic that is present in the scaled system, the less the accuracy. This issue is discussed in detail in Section VII.

B. CENIC Backbone Experiments

We now present simulation results for the topology shown in Fig. 2, when using either of the proposed methods. The flow arrival rates are the same within each

TABLE I
AMOUNT OF DOWNSCALING, SIMULATION TIME, AND ACCURACY (SIMPLE TOPOLOGY)

Method	Size	Traffic	Time(min)	Avg. HSM (95% Confidence Interval)
Tandem Original	1	1	61	1
DSCALEd ($a = 1$)	1/2	2/3	31	0.96
DSCALEd ($a = 0.5$)	1/2	1/3	15	[0.91, 0.93]
DSCALEd ($a = 0.1$)	1/2	1/15	3	[0.79, 0.83]
DSCALEs ($\delta = 1$)	1/2	1/3	15	[0.88, 0.90]
DSCALEs ($\delta = 0.6$)	1/2	1/5	8	[0.84, 0.88]
DSCALEs ($\delta = 0.3$)	1/2	1/10	4	[0.77, 0.81]

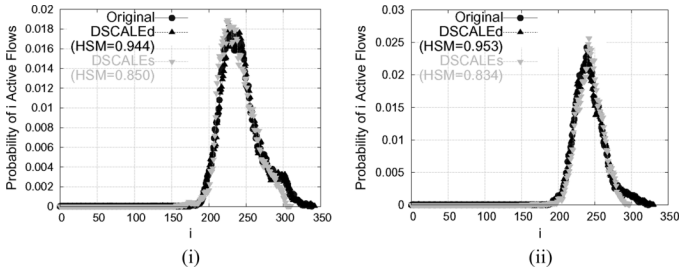


Fig. 6. Distribution of number of active flows (i) on link $SVL(hpr) - SVL(dc1)$, and (ii) on link $LAX(dc2) - LAX(hpr)$. (CENIC topology.)

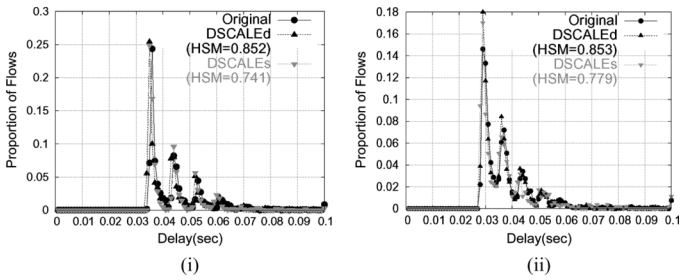


Fig. 7. Histogram of end-to-end flow delays of (i) $grp1$, and (ii) $grp2$ flows. (CENIC topology.)

group and equal 250 flows/s. The average queueing delays on the congested links $SVL(dc1) - SVL(hpr)$ and $LAX(dc2) - LAX(hpr)$ are, respectively, 0.15 and 0.18 ms. In the other (backward) direction, the average queueing delays equal 0.31 ms (link $SVL(hpr) - SVL(dc1)$) and 0.13 ms (link $LAX(hpr) - LAX(dc2)$). Drops occur only on $SVL(hpr) - SVL(dc1)$, with a probability of 2%.

For DSCALEd, the scaled system is shown in Fig. 3(i). The fixed delay factor for each of the 120 groups present in the scaled system is computed, as described in Section IV. For DSCALEs, the scaled system is shown in Fig. 3(ii). Recall from Section IV that for $\delta = 60$ we get $C_1' = 450$ Mb/s, $C_2' = 550$ Mb/s, $P' = 4.78$ ms, $a_1 = 0.88, a_2 = 0.36$, etc., for all a_i's, $(i = 1 \ldots 120)$.

Fig. 6 plots the distribution of the number of active flows on links $SVL(dc1) - SVL(hpr)$ and $LAX(dc2) - LAX(hpr)$. Fig. 7 plots the end-to-end flow delay histograms for $grp1$ and $grp2$ flows. (These groups were defined in Section IV.) For DSCALEs, the normalization of the flow delays of group i is done as follows: $(\text{normalized flow delay}) = a_i$ $(\text{delay due to propagation time}) + \sum_j I_i^j s_j$ (queueing delay in front of link j), where a_i, $i = 1, \ldots, 120$, are the sampling factors, s_j, $j = 1,2$ are the ratio of the capacities of the

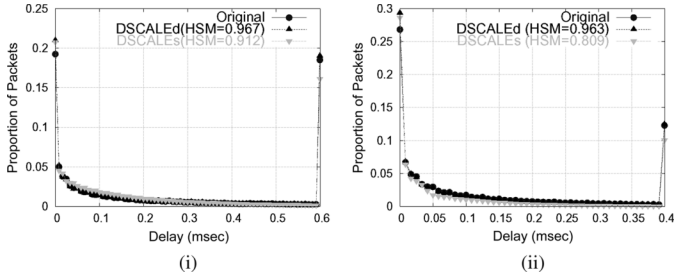


Fig. 8. Histogram of packet delays (i) on link $SVL(hpr) - SVL(dc1)$, and (ii) on link $LAX(dc2) - LAX(hpr)$. (CENIC topology.)

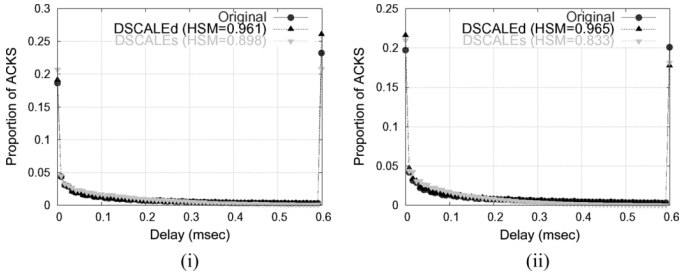


Fig. 9. Histogram of ACK delays (i) on forward direction, and (ii) on backward direction of link $SVL(hpr) - SVL(dc1)$. (CENIC topology.)

congested links in the original and scaled network, and I_i^j equals 1 if group i traverses link j and 0, otherwise. Due to limitations of space, we do not present results for the rest of the groups of flows. Fig. 8 plots the packet delay histograms on links $SVL(hpr) - SVL(dc1)$ and $LAX(dc2) - LAX(hpr)$. Finally, Fig. 9 plots the the ACK delay histograms in both directions of link $SVL(hpr) - SVL(dc1)$, to test whether the methods can predict ACK compression.

It is evident from the figures that both methods can predict performance. Based on these figures, we make a couple of interesting observations. First, the methods are slightly less accurate in the CENIC topology scenario than in the simple topology scenario. This is expected: CENIC is a very large network, and downscaling ignores a large number of uncongested links. Second, notice that both methods do a bit worse when predicting end-to-end delays than when predicting other per-link measures. This is because end-to-end delays depend on all the links along the path, and the small queueing delays due to the uncongested links are not captured in the scaled systems. Finally, notice that DSCALEd is again more accurate than DSCALEs. As in the simple topology scenario, this is due to the fact that DSCALEs employs traffic sampling, and as a result, it uses a smaller fraction of the original traffic than DSCALEd (see Table II for the values of these fractions).

C. A Closer Look at Large Queueing Delays

Recall that during the derivation of the DSCALEs equations in Section IV, the assumption was that end-to-end queueing delays are small in comparison to total end-to-end delays. To illustrate how DSCALEs works when queueing delays are relatively large compared with the end-to-end delays, we consider the original topology shown in Fig. 1(i) with the only difference that $P_2 = 25$ ms and the flow arrival rate is 47 flows/s (same for all the groups). In this case, the average queueing delay on link

TABLE II
AMOUNT OF DOWNSCALING, SIMULATION TIME, AND ACCURACY (CENIC TOPOLOGY)

Method	Size	Traffic	Time(min)	Avg. HSM (95% Confidence Interval)
CENIC Original	1	1	1725	1
DSCALEd ($a = 1$)	2/41	1/5	40	0.92
DSCALEd ($a = 0.5$)	2/41	1/10	16	[0.81, 0.83]
DSCALEd ($a = 0.1$)	2/41	1/50	5	[0.74, 0.78]
DSCALEs ($\delta = 70$)	2/41	7/60	17	[0.80, 0.84]
DSCALEs ($\delta = 60$)	2/41	1/10	15	[0.79, 0.83]
DSCALEs ($\delta = 30$)	2/41	1/20	11	[0.76, 0.80]

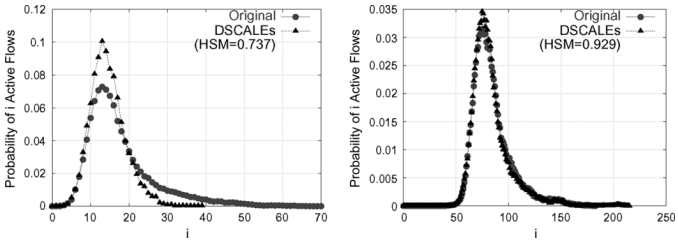


Fig. 10. (i) $grp3$ distribution of flows. (ii) Overall distribution of flows. (DSCALEs, simple topology, large queueing delays.)

$R2-R3$ (as calculated by the simulator) is 26 ms, which is comparable to the end-to-end latency of a $grp3$ packet. The scaled system is shown in Fig. 1(iii), where $a_1 = 0.83$, $a_3 = 0.17$, $P' = 150.4$ ms, and $C' = 5$ Mb/s.

Fig. 10(i) plots the $grp3$ distribution of active flows for both the original and the scaled system. The method is not very accurate. This is evident visually and from the relatively low, for such a simple topology scenario, HSM value. Yet, the method can accurately predict "aggregate-based" performance metrics, like the overall distribution of active flows and the queue length distributions. Section VI clearly defines which metrics are "aggregate-based," and proves the above statements. Fig. 10(ii) shows the overall distribution of active flows on the two systems, and it is evident that the method is quite accurate.

D. Real Internet Traces

We now illustrate how our methods work with real Internet traffic. We use the setup shown in Fig. 1(i) with $C_1 = 1$ Gb/s, $C_2 = 400$ Mb/s, $P_1 = P_2 = 1$ ms, and with routers having the same characteristics as before. In this experiment, the flow arrival times are no longer dictated by an exact Poisson process generated from the simulator, but instead are extracted from the Abilene-I data set [52]. The flow size is Pareto-distributed with the same parameters as before.

Fig. 11 plots the end-to-end $grp1$ and $grp3$ flow delay histograms, as obtained from the original and the scaled systems. Looking at the plots, we observe that DSCALEd is quite accurate, whereas DSCALEs is a bit less accurate. The same conclusion is also reached by examining the HSM values of the two methods. The small inaccuracy of DSCALEs is partially due to the non-Poisson flow arrivals. Indeed, in Section VI, we show theoretically that DSCALEs requires flow arrivals to be Poisson. (In the same section, we also show that this is not a requirement for DSCALEd.) Nevertheless, note that the inaccuracy in Fig. 11 is small. This implies that DSCALEs may be resilient to deviations from the Poisson requirement.

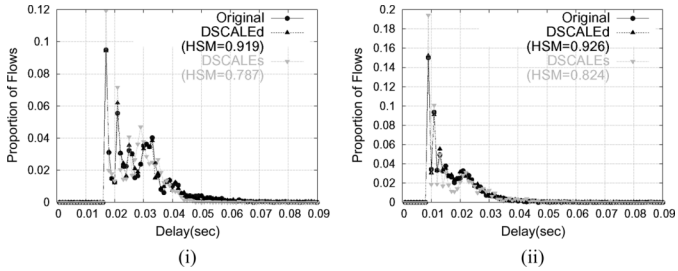


Fig. 11. Histogram of end-to-end flow delays of (i) $grp1$, and (ii) $grp3$ flows. (Simple tandem topology, real trace.)

VI. THEORETICAL SUPPORT

In this section, we formalize the important results of this paper. First, we show that it is always possible to construct a scaled network from any original network by removing uncongested links, without violating the following natural constraint: the links of the scaled system must be traversed by the same groups of flows and in the same direction as in the original system. Second, we prove that DSCALEd preserves network performance under the assumption that uncongested links do not contribute to the end-to-end queueing delays of packets. This result holds for any stationary arrival process. Third, we prove that DSCALEs preserves aggregate performance metrics under the additional assumption that flow arrivals are Poisson, and per-group performance metrics under the additional assumption that end-to-end delays are mainly dictated by propagation delays.

A. Downscaling Topology

Recall that the first three steps in the method summaries presented in Section IV are identical. In other words, both methods perform the same tasks when removing uncongested links and reconnecting together the remaining links to form the downscaled topology. The way by which the methods choose to attach one link to another and build the scaled topology depends on the direction that groups of flows are traversing the links in the original system. In order to preserve performance, one has to make sure that the links of the scaled system are being traversed by the same groups of flows and in the same direction as in the original system. But is this always possible? The following lemma shows that it is.

Lemma 1: The construction of the scaled topology is always possible, as long as minimum cost routing is used.

Proof: Without loss of generality, consider two links in the original system, say links (e_1, e_2) and (e_3, e_4), and two groups of flows, where the first group traverses these links in the direction (e_1, e_2), (e_3, e_4) and the second group in the direction (e_1, e_2), (e_4, e_3). It is easy to see that building the scaled system by attaching these two links together is problematic, because we must connect e_2 to both e_3 and e_4. However, the above situation is not possible: Since both groups follow the path with the minimum cost to their destination, they follow the minimum subpath from e_2 to link (e_3, e_4), and they must enter the link from

the same point. (In case there are equal minimum cost paths and routers use load-balancing, the above situation is possible, yet quite unlikely. In this case, adding a zero-delay link to the scaled topology allows one to connect e_2 to both e_3 and e_4.) ■

B. Preserving Performance With DSCALEd

Under the assumption that uncongested links do not contribute to the end-to-end queueing delays, it is easy to see why the proposed method would preserve performance if one would add the right fixed delays at exactly the point where uncongested network segments existed in the original network. By adding the right fixed delays to packets, we make sure that the transmission and propagation delays remain unaltered between the original and the scaled system, and, by retaining all congested links, we make sure that end-to-end queueing delays are unaltered.

However, DSCALEd adds fixed delays at the source of each packet. This minimizes bookkeeping since one does not have to retain information about where each uncongested link used to be in the original network. The following lemma establishes that adding the fixed delays at the sources does not alter the packet arrival processes at the links.

Lemma 2: If flow arrivals are independent among different groups and stationary, adding the fixed delays at the sources does not alter the packet arrival processes at the links.

Proof: Consider two scaled systems. In the first, fixed delays are added at exactly the points where uncongested network segments existed in the original network. In the second, fixed delays are added at the sources of the packets. We prove that the packet arrival processes at the links of these systems are the same.

Without loss of generality, let $I_1(t)$ and $I_2(t)$ denote the stationary flow arrival processes of two groups of flows, $grp1$ and $grp2$, respectively, at a congested link. Now, let d_1 denote the delay factor added to all the packets of $grp1$ flows and d_2 the delay factor added to all the packets of $grp2$ flows, due to downstream uncongested segments that were removed from the network. In the second system, this results in a time shift of the $grp1$ arrival process at the congested link by d_1, and of the $grp2$ arrival process by d_2. Since flows between different groups arrive independently and the arrival processes are stationary, $(I_1(t), I_2(t)) \stackrel{d}{=} (I_1(t+d_1), I_2(t+d_2))$, that is, the overall flow arrival process to the link remains the same if the fixed delays are added prior to the link. This takes care of the first packet of each flow.

What about subsequent packets of a flow? First, note that networks are discrete-event systems, clocked by transmissions and ACKs of packets. Now, consider the first packet of a flow that arrives at the congested link in the first scaled network (where fixed delays are added at exactly the point where uncongested network segments existed in the original network). The packet will experience an amount of queueing, and then, an amount of transmission and propagation delay, before it reaches its destination. Now, consider the lifetime of the same packet in the second network (where fixed delays are added at the sources). This packet experiences the same amount of queueing delay, but then, is delivered directly to its destination. This means that the source of the packet will receive an ACK earlier, compared with the first system. But the amount of propagation and transmission delays that the packet did not experience after leaving the congested link in the second system, is accounted in the fixed delay imposed at the source of the packet. This means that while the ACK was received earlier, the next packet arrival (triggered by the ACK) is delayed until the ACK is received in the first system as well. As a result, the second packet arrives at the congested link of the second system at the same time as the corresponding packet arrives at the congested link of the first system. Continuing inductively, we can see that the packet arrival process at the links of the two systems has not been altered. As a final note, it is easy to account for multiple congested links along the paths of the packets using the same arguments as above. ■

We can now state the following theorem whose proof follows immediately from the discussion above.

Theorem 1: If flow arrivals are independent among different groups, and the arrival process is stationary, DSCALEd preserves the network performance in distribution. For example, the distribution of metrics like the number of active flows, the flow delays, the queue lengths, etc., remain unchanged.

It is interesting to point out that if the fixed delays are added at exactly the point where uncongested network segments exist in the original network, DSCALEd preserves the network performance as a function of time. In contrast, the previous theorem states that performance prediction occurs for distributions.

C. Preserving Performance With DSCALEs

First, we briefly review one important result from [17] that we use. In a network where flows arrive as a Poisson process, do the following operations to construct a slower replica: 1) sample each incoming flow independently with probability a; 2) reduce link capacities by the same factor a; and 3) increase propagation delays (and protocol timeouts) by a factor $1/a$. In summary, arrival rates λ, capacities C, and propagation delays P change to $a\lambda$, aC, and P/a, respectively. Then, the state of the original network at time t, is the same in distribution with the state of the slower replica at time t/a. (Note that the state of the network at time t includes all the information needed to resume the evolution of the network from time t onwards.) In simple words, the only difference between the original and scaled system is that the latter runs slower by a factor a. We call this result the *time-downscaling law*.

We now state two results for DSCALEs. We distinguish between performance metrics that depend on the properties of the "supergroup" consisting of all groups of flows going through a link, called aggregate-based metrics, and metrics that also depend on the properties of a specific group of flows, called group-based metrics. To make the distinction clear, supposed we have a single link that is traversed by two groups of flows. If we vary the value of the arrival rates of the two groups such that their sum remains the same, aggregate-based metrics remain unchanged, whereas group-based metrics change. The distribution of the number of active flows (overall distribution), and the queue length distributions are clearly aggregate-based metrics. The flow transfer times depend on propagation and queueing delays, and since the latter are aggregate-based, the same holds for flow transfer times. In contrast, the distribution of the number of active flows of a particular group (per-group distribution) is clearly a group-based metric.

Theorem 2: Using DSCALEs, the distributions of aggregate-based metrics, e.g., the queue length distributions and the overall distribution of active flows, remain unchanged.

Proof: Without loss of generality, consider the link $R2-R3$ of the network in Fig. 1(i), and assume that the flows of all groups belong to a new "supergroup." As a result of merging two Poisson processes with rates λ_1 and λ_3, the overall arrival process at the original link is still Poisson with rate $\lambda=\lambda_1+\lambda_3$. (Notice that $grp1$ flows arrive as a Poisson process at the link $R2-R3$ since the first, uncongested link does not alter the traffic statistics by assumption.) Further, replace the propagation delay of each packet with the average propagation delay that equals $P=(\lambda_1(P_1+P_2)+\lambda_3P_2)/(\lambda_1+\lambda_3)$. (Note that while flows with small propagation delays grab a larger portion of the available bandwidth than flows with large propagation delays, here we are only concerned with aggregate-based metrics, which remain unchanged by the above operation.)

Now, consider the link $R2'-R3'$ of the scaled network in Fig. 1(iii). It is a simple property of the Poisson process that sampling a proportion a of the points of a rate λ Poisson process will yield a Poisson process with rate $a\lambda$. In addition, the independent sampling process does not destroy the independent and identically distributed (i.i.d.) nature of the flow sizes. Thus, $grp1$ flows arrive as a Poisson process with rate $a_1\lambda_1$ and $grp3$ of flows arrive as a Poisson process with rate $a_3\lambda_3$. Hence, the overall arrival process is still Poisson with rate $a_1\lambda_1+a_3\lambda_3$. Further, the propagation delay is P' for all packets, and note that $P'=((a_1\lambda_1P'+a_3\lambda_3P')/(a_1\lambda_1+a_3\lambda_3))=((\lambda_1(P_1+P_2)+\lambda_3P_2)/(a_1\lambda_1+a_3\lambda_3))$.

Hence, the total arrival rate λ, the capacity C_2, and the average propagation delay P, change to $s\lambda$, sC_2, and P/s, respectively. Thus, the link $R2'-R3'$ is a slower, by a factor of s, replica of the link $R2-R3$, and, according to the time-downscaling law, performance is preserved. ■

Theorem 3: Using DSCALEs, the distributions of group-based performance metrics, e.g., the per-group distribution of active flows, are altered, unless the end-to-end queueing delays are negligible compared to the total end-to-end delays.

Proof: Without loss of generality, consider again the link $R2-R3$ of the network in Fig. 1(i), and concentrate only on $grp3$ flows. The average arrival rate of this group at the original link equals λ_3, its propagation delay is P_2 and the link capacity is C_2. Now, look at the link $R2'-R3'$ of the scaled network in Fig. 1(iii). The average arrival rate of $grp3$ flows equals $a_3\lambda_3$, the propagation delay equals $P'=P_2/a_3$, and the link capacity is sC_2. Since $s\neq a_3$, the time-downscaling law does not apply.

Now, let us look at the case where the end-to-end queueing delay is insignificant compared with the end-to-end propagation delay. (Technically, let the capacity grow to infinity which makes queueing delay reduce to zero.) In this case, the RTT of each packet is dictated by the propagation delay only. Hence, the RTT of $grp3$ packets is stretched by a_3. Since the flow arrival rate and the packet RTT are stretched by the same factor a_3, from the $grp3$ packets' perspective the scaled system runs slower by a factor of a_3 from the original, and the time-downscaling law applies. Considering $grp1$ flows only, their rate changes to $a_1\lambda_1$ and their RTT is stretched by a_1 $P'=(P_1+P_2)/a_1$. Hence, from their perspective, the scaled system runs slower by a factor a_1, and the time-downscaling law applies. ■

VII. Computational Savings, Amount of Downscaling, and Accuracy

In this section, we address the following fundamental issue: What is the connection between the amount of downscaling used by the methods, the simulation time required to run an experiment, and the accuracy of the performance prediction? Tables I and II provide some answers.

For our experiments, we used a 2 GHz processor with 2 GB memory (RAM) and recorded the simulation time needed to complete the experiment for each scenario.[8] The accuracy is quantified using the average histogram similarity measure (HSM), where the average is taken over the HSMs obtained from comparing: 1) the overall distribution of active flows in the original and scaled network; 2) the end to end delay histogram of the groups of flows of interest in the two networks; and 3) the packet queueing delay histogram on the congested links of the two networks.

The amount of downscaling is quantified by the fraction of links and flows of the original network present in the scaled network. In particular, the size column in the tables indicates the fraction of the links of the original network that are part of the scaled one, and, the traffic column indicates the fraction of flows of the original network that traverse the scaled network. This fraction depends on two factors: 1) the number of flows we ignore due to ignoring the links they traverse and 2) the amount of sampling. We have used DSCALEd in conjunction with time downscaling which employs sampling, and the value of $a\leq 1$ (see the parenthesis in the tables) equals the probability of sampling. (In particular, after the smaller replica is constructed using DSCALEd, we also sample all incoming flows by a, multiply the capacities by a, and divide the propagation and the fixed delays imposed at the source by a, thus creating, in addition, a slower in time system.) For DSCALEs, the value of δ (see tables) dictates the intensity of sampling, since $\delta=\sum a_i$, where $a_i\leq 1$ is the sampling factor for each flow group. Let N be the total number of groups of flows in the original topology, $M\leq N$ be the total number of groups of flows in the scaled replica, and let each group have an equal number of flows. Then, without considering sampling, a fraction of flows equal to M/N is present in the scaled network. Taking sampling into account, under DSCALEd the fraction of flows in the scaled topology is $a(M/N)\leq 1$, and under DSCALEs the corresponding fraction equals $(\delta/M)(M/N)=(\delta/N)\leq 1$.

Table I shows how effective our methods are when applied to the simple tandem topology shown in Fig. 1(i). The total number of flows used in this experiment is 390 000 belonging to three different groups. Two of those groups are present in the scaled topology. Simulating that original topology takes 61 min, whereas using DSCALEd with $a=1$ takes 31 min. More impressive savings are obtained for $a<1$, i.e., when the system is also scaled in time. For example, when we use an a of 0.1 the simulation time is only 3 min, while the accuracy remains good.

[8]In order to facilitate large-scale simulations, we had to modify the built-in routines of ns-2 in order to allocate memory for the TCP sessions dynamically. The patch, along with more information, can be found at http:////www-scf.usc.edu/~fpapadop/.

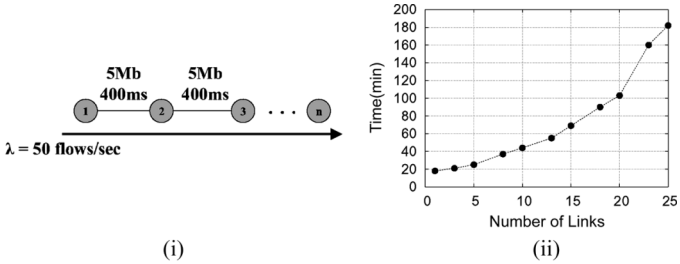


Fig. 12. (i) Experimental topology, and (ii) simulation time versus number of links on the experimental topology.

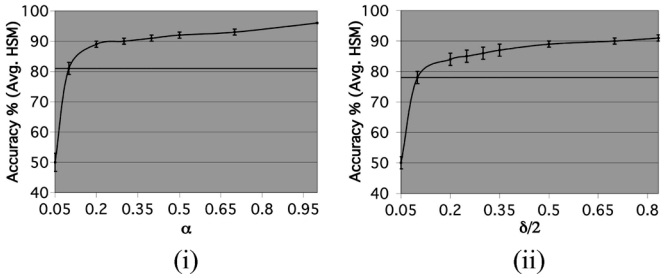


Fig. 13. Accuracy of the methods as a function of traffic sampling. (i) DSCALEd. (ii) DSCALEs. (Simple topology experiments.)

The gain we get by using DSCALEs is also notable. For example, for $\delta = 1$, the simulation takes only 15 min to terminate.

Table II shows the effectiveness of our methods when applied to the CENIC topology shown in Fig. 2. The total number of flows used in this experiment is 1 500 000 belonging to 600 different groups. 120 of those are present in the scaled topology. From the table, it is evident that the gain we get by using either of the methods is quite remarkable, close to two orders of magnitude, while the accuracy of the methods is quite good.

A. *Savings From Topology and Traffic Downscaling*

There are two reasons why we save time with our methods: 1) topological downscaling, that is, the act of removing uncongested links and nodes and 2) traffic downscaling, that is, the act of considering as input only a fraction of the original flows.

Let us first concentrate on topological downscaling. Recall that uncongested links have small but nonzero delays. Hence, the more uncongested links one ignores, the less accurate performance prediction is, for example, with respect to the end to end delay. Because of this, the average HSM of DSCALEd is higher in Table I, which corresponds to a scenario where only one out of two links is ignored, than in Table II, which corresponds to a scenario where 39 out of 41 links are ignored, for all values of a. Interestingly enough, our methods are very accurate even in the later scenario.

A comparison of these tables also reveals that the larger the number of links one ignores, the larger the time savings are, as expected. For example, simulations that use DSCALEd with $a = 1$ run two times faster than the original simulation in the case of the scenario of Table I, and 43 times faster in the case of the scenario of Table II. For $a = 0.1$, the simulation speedup equals 20 in the simple topology and 345 in the CENIC topology. The large gains associated with topological downscaling leads us to the following question: In what way does the time needed to run an experiment depend on the number of links/nodes that comprise a network topology? To answer this question, we use the experimental topology shown in Fig. 12(i). For that topology, we measure the time needed to run an experiment when we have n nodes and $n-1$ identical links, while varying n from 1 to 25. The total number of flows for this experiment is 100 000 and they have the same characteristics as before. Fig. 12(ii) shows that the time needed to run the experiment increases with the number of links, with the increase being between linear and exponential. Hence, computational savings are expected to increase dramatically as the size of the network under study increases. We believe this to be the case for any event-driven simulator, not just ns-2, since the number of events that a simulator needs to schedule increases rapidly as the length of the paths grows.

Now, let us take a closer look at traffic downscaling. Going through both Tables I and II, it is evident that as the amount of traffic decreases, the time savings increase. This comes as no surprise. As the number of flows decreases, the number of simulator events decreases, and the simulator terminates faster. From the tables, it is evident that the simulation time depends approximately linearly on the number of flows, which is also somewhat expected.

Recall that traffic downscaling occurs for two reasons. One reason is topology downscaling; when one ignores links, one also ignores the traffic that traverses them. The other reason is traffic sampling. In the rest of the section, we discuss some practical issues related to how traffic sampling affects accuracy.

In theory, random sampling captures the statistics of an infinite length input accurately. But, in practice, we work with a finite length input, that is, with a finite number of flows. For the sampled traffic to accurately represent the original traffic, the right proportion of short and long flows should exist on the sample. Because flow sizes are heavy-tailed, small deviations on the proportion of long flows may have a sizeable effect on the overall statistics of the sample. The smaller the sampling probability is, the larger the possibility that this is going to be an issue. Another practical consideration has to do with convergence. Clearly, if one keeps on reducing the sampling probability over a finite length input, after some point there are simply not enough arrivals for the observed histograms to converge to the actual distributions.

Fig. 13 shows how accuracy, measured by the average HSM, changes as a function of the intensity of traffic sampling in the simple topology scenario. (The vertical lines in these plots represent the 95% confidence intervals.) Recall that traffic sampling is employed on 2/3 of the original traffic, since two out of the three original groups of flows are present in the scaled network. As it is evident from the plot, for $a < 0.1$ and $(\delta/2) < 0.1$, the average HSM starts dropping fast, whereas for larger values of a and δ it is quite high. In general, in all our scenarios, 10% of the original flows are enough to achieve reasonably accurate performance prediction. (Recall that in the simple scenario, we started with hundreds of thousands of flows belonging to a handful of groups, and in the CENIC scenario, we started with millions of flows belonging to hundreds of groups.)

VIII. Conclusion and Future Directions

We proposed two methods, DSCALEd and DSCALEs, to scale down an arbitrary network topology that is shared by TCP flows and controlled by various AQM schemes. DSCALEd is applicable to any scenario. DSCALEs has some limitations: it cannot accurately predict some group-based metrics when queueing delays are large in comparison to the total end-to-end delays.

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 and theoretical arguments. 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.

There are some interesting yet challenging extensions to this work. This paper shows that downscaling the Internet while maintaining performance characteristics is possible. It studies, via simulations, the interplay between the amount of downscaling and the accuracy loss that might occur. What is of great interest is to study this tradeoff using rigorous analytical tools. For example, it would be useful to analytically quantify the relationship between a, the factor by which one samples flows, and the accuracy of performance prediction in practical situations, where the total number of flows is a large yet finite number. Similarly, it would be helpful to analytically quantify the relationship between the number of uncongested links that are ignored by topological downscaling and the achieved accuracy.

As a final note, topological downscaling is based on the assumption that uncongested links do not alter packet dynamics. A number of works have been cited that support this claim experimentally in a variety of scenarios. Our simulation results indirectly validate this assumption for the case of Internet TCP traffic, and our theoretical results support the assumption under a simplified view of uncongested links, according to which their effect is a fixed delay to each packet. What is of great interest is to theoretically verify this assumption when queueing dynamics of uncongested links are taken into account. While this has been done for the case of open-loop networks [23], no such analysis exists for close-loop systems that resemble the Internet. We believe this to be a very interesting future-work direction.

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 *Proc. ACM HOTNETS*, Oct. 2002.

[2] Y. Liu, F. L. Presti, V. Misra, D. Towsley, and Y. Gu, "Fluid models and solutions for large-scale IP networks," in *Proc. ACM SIGMETRICS*, Jun. 2003.

[3] C. Barakat, P. Thiran, G. Iannaccone, C. Diot, and P. Owezarski, "A flow-based model for Internet backbone traffic," in *Proc. ACM SIGCOMM Internet Measure. Workshop*, Nov. 2002.

[4] S. B. Fredj, T. Bonalds, A. Prutiere, G. Gegnie, and J. Roberts, "Statistical bandwidth sharing: A study of congestion at flow level," in *Proc. ACM SIGCOMM*, Aug. 2001.

[5] D. Wischik, "The output of a switch, or, effective bandwidths for networks," *Queueing Syst.*, vol. 32, no. 4, pp. 383–396, 1999.

[6] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: A simple model and its empirical validation," in *Proc. ACM SIGCOMM*, Aug. 1998.

[7] 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 Trans. Netw.*, vol. 5, no. 1, pp. 71–86, Feb. 1997.

[8] 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," *IEEE Network*, vol. 17, no. 6, pp. 6–16, Nov. 2003.

[9] 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 *Proc. IEEE INFOCOM*, Jun. 2002, pp. 535–544.

[10] J. Liu and M. Crovella, "Using loss pairs to discover network properties," in *Proc. ACM SIGCOMM Internet Measure. Workshop*, Nov. 2001.

[11] V. Paxson and S. Floyd, "Wide area traffic: The failure of Poisson modeling," *IEEE/ACM Trans. Netw.*, vol. 3, no. 3, pp. 226–244, Jun. 1995.

[12] J. C. Mogul, "Observing TCP dynamics in real networks," in *Proc. ACM SIGCOMM*, Aug. 1992.

[13] J. S. Ahn and P. B. Danzig, "Speedup and accuracy versus timing granularity," *IEEE/ACM Trans. Netw.*, vol. 4, no. 5, pp. 743–757, Oct. 1996.

[14] D. Nicol and P. Heidelberger, "Parallel execution for serial simulators," *ACM Trans. Modeling Comput. Simulation*, vol. 6, no. 3, pp. 210–242, Jul. 1996.

[15] A. Yan and W. B. Gong, "Time-driven fluid simulation for high-speed networks," *IEEE Trans. Inf. Theory*, vol. 45, no. 5, pp. 1588–1599, Jul. 1999.

[16] B. Liu, D. R. Figueiredo, Y. Guo, J. Kurose, and D. Towsley, "A study of networks simulation efficiency: Fluid simulation vs. packet-level simulation," in *Proc. IEEE INFOCOM*, Apr. 2001, pp. 1244–1253.

[17] R. Pan, B. Prabhakar, K. Psounis, and D. Wischik, "SHRiNK: Enabling scaleable performance prediction and efficient simulation of networks," *IEEE/ACM Trans. Netw.*, vol. 13, no. 5, pp. 975–988, Oct. 2005.

[18] K. Papagiannaki, "Provisioning IP backbone networks based on measurements," Ph.D. dissertation, Univ. College, London, U.K., Mar. 2003.

[19] C. Fraleigh, "Provisioning Internet backbone networks to support latency sensitive applications," Ph.D. dissertation, Stanford Univ., Stanford, CA, Jun. 2002.

[20] C. Fraleigh, F. Tobagi, and C. Diot, "Provisioning IP backbone networks to support latency sensitive traffic," in *Proc. IEEE INFOCOM*, Mar. 2003, pp. 375–385.

[21] D. Y. Eun and N. B. Shroff, "Simplification of network analysis in large-bandwidth systems," in *Proc. IEEE INFOCOM*, Mar. 2003, pp. 597–607.

[22] ——, "Analyzing a two-stage queueing system with many point process arrivals at upstream queue," *Queueing Syst.*, vol. 48, no. 1–2, pp. 23–43, Sep. 2004.

[23] ——, "Network decomposition: Theory and practice," *IEEE/ACM Trans. Netw.*, vol. 13, no. 3, pp. 526–539, Jun. 2005.

[24] ——, "Network decomposition in the many sources regime," *Adv. Appl. Prob.*, vol. 36, no. 3, pp. 893–918, Sep. 2004.

[25] "Intermapper Web Server." Jan. 2005. [Online]. Available: https://intermapper.engineering.cenic.org

[26] R. Pan, B. Prabhakar, K. Psounis, and D. Wischik, "SHRiNK: A method for scalable performance prediction and efficient network simulation," in *Proc. IEEE INFOCOM*, Mar. 2003.

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

[28] V. Krishnamurthy, J. Sun, M. Faloutsos, and S. Tauro, "Sampling Internet topologies: How small can we go?," in *Proc. Int. Conf. Internet Comput.*, Jun. 2003.

[29] T. Bu and D. Towsley, "On distinguishing between Internet power law topology generators," in *Proc. IEEE INFOCOM*, Jun. 2002, pp. 638–647.

[30] M. Faloutsos, P. Faloutsos, and C. Faloutsos, "On power-law relationships of the Internet topology," in *Proc. ACM SIGCOMM*, Aug. 1999.

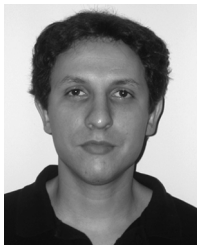
[31] V. Misra, W. Gong, and D. Towsley, "A fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED," in *Proc. ACM SIGCOMM*, Aug. 2000.

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

[33] W. Wei, B. Wang, D. Towsley, and J. Kurose, "Model-based identification of dominant congested links," in *Proc. ACM SIGCOMM Internet Measure. Conf.*, Oct. 2003.

[34] A. Akella, S. Seshan, and A. Shaikh, "An empirical evaluation of wide-area Internet bottlenecks," in *Proc. ACM SIGCOMM Internet Measure. Conf.*, Oct. 2003.

[35] Y. Liu, F. L. Presti, V. Misra, D. F. Towsley, and Y. Gu, "Scalable fluid models and simulations for large-scale IP networks," *ACM Trans. Modeling Comput. Simulation*, vol. 14, no. 3, pp. 305–324, Jul. 2004.
[36] A. Fei, G. Pei, R. Liu, and L. Zhang, "Measurements on delay and hop-count of the Internet," in *Proc. IEEE GLOBECOM Internet Mini-Conf.*, Nov. 1998.
[37] F. Begtasevic and P. V. Mieghem, "Measurements of the hopcount in the Internet," in *Proc. Passive Active Measure. Workshop*, Apr. 2001.
[38] G. de Veciana, G. Kesidis, and J. Walrand, "Resource management in wide-area ATM networks using effective bandwidths," *IEEE J. Sel. Areas Commun.*, vol. 13, no. 6, pp. 1081–1090, Aug. 1995.
[39] Y. Zhang, M. Roughan, N. Duffield, and A. Greenberg, "Fast accurate computation of large-scale IP matrices from link loads," in *Proc. ACM SIGMETRICS*, Jun. 2003.
[40] N. Hu, L. Li, Z. M. Mao, P. Steenkiste, and J. Wang, "Locating Internet bottlenecks: Algorithms, measurements and implications," in *Proc. ACM SIGCOMM*, Aug. 2004.
[41] "Performance measurement tools taxonomy." Apr. 2006. [Online]. Available: http://www.caida.org/tools/taxonomy/performance.xml
[42] W. Fang and L. Peterson, "Inter-AS traffic patterns and their implications," in *Proc. 4th Global Internet Symp.*, Dec. 1999, pp. 1859–1868.
[43] C. Fraleigh, C. Diot, B. Lyles, S. Moon, P. Owezarski, D. Papagiannaki, and F. Tobagi, "Design and deployment of a passive monitoring infrastructure," in *Proc. Passive Active Measure. Workshop*, Apr. 2001.
[44] "Rocketfuel: An ISP topology mapping engine." Aug. 2005. [Online]. Available: http://www.cs.washington.edu/research/networking/rocketfuel/interactive
[45] F. Papadopoulos, K. Psounis, and R. Govindan, "Performance preserving topological downscaling of Internet-like networks," Univ. Southern California, Los Angeles, CA, Tech. Rep. CENG-2005-7, 2005.
[46] V. Paxson, "End-to-end routing behavior in the Internet," *IEEE/ACM Trans. Netw.*, vol. 5, no. 5, pp. 601–615, Oct. 1997.
[47] L. Zhang, S. Shenker, and D. D. Clark, "Observations on the dynamis of a congestion control algorithm: The effects of two-way traffic," in *Proc. ACM SIGCOMM*, Sep. 1991.
[48] "Network simulator." [Online]. Available: http://www.isi.edu/nsnam/ns
[49] A. Feldmann, A. C. Gilbert, and W. Willinger, "Data networks as cascades: investigating the multifractal nature of Internet WAN traffic," in *Proc. ACM SIGCOMM*, Aug. 1998.
[50] C. J. Nuzman, I. Saniee, W. Sweldens, and A. Weiss, "A compound model for TCP connection arrivals," in *Proc. ITC Seminar IP Traffic Modeling*, Sep. 2000.
[51] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge, U.K.: Cambridge Univ. Press, 1992.
[52] "Passive measurement and analysis." Dec. 2004. [Online]. Available: http://pma.nlanr.net/Special



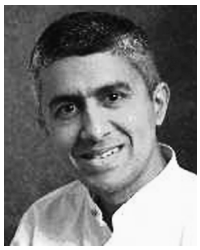
Fragkiskos Papadopoulos received the first degree from the Department of Electrical and Computer Engineering, National Technical University of Athens, Athens, Greece, in 2002 and the M.Sc. degree in electrical engineering, specializing in computer networks from the University of Southern California, Los Angeles, in 2004. He is currently working towards the Ph.D. degree in electrical engineering at the University of Southern California.

His research interests include modeling, simulation, and performance prediction/analysis of computer networks.

Konstantinos Psounis (S'97–M'02) received the first degree from the Department of Electrical and Computer Engineering, National Technical University of Athens, Greece, in 1997, and the M.S. and Ph.D. degrees in electrical engineering from Stanford University, Stanford, CA, in 1999 and 2002, respectively.

He is an Assistant Professor of Electrical Engineering and Computer Science at the University of Southern California, Los Angeles. He models and analyzes the performance of computer networks, sensor and mobile systems, and the web. He also designs methods and algorithms to solve problems related to such systems. He is the author of more than 30 research papers on these topics.

Dr. Konstantinos has received faculty awards from the National Science Foundation (NSF) and the Zumberge Foundation. He has been a Stanford University Graduate Fellow throughout his graduate studies, and has received the Best-Student National Technical University of Athens Award for graduating first in his class.



Ramesh Govindan received the B. Tech. degree from the Indian Institute of Technology, Madras, and the M.S. and Ph.D. degrees from the University of California at Berkeley.

He is an Associate Professor in the Department of Computer Science, University of Southern California, Los Angeles. His research interests include Internet routing and topology, and wireless sensor networks.