



### Scenario-Specific Topology Reduction in Network Simulations

Journal:	<i>International Symposium on Performance Evaluation of Computer and Telecommunication Systems</i>
Manuscript ID:	SPECTS-05-NTS-012.R1
Topic Area:	Networking and Telecommunication System
Date Submitted by the Author:	26-May-2005
Complete List of Authors:	Petit, Barath; Georgia Tech, College of Computing Ammar, Mostafa; Georgia Tech, College of Computing Fujimoto, Richard; Georgia Tech, College of Computing
Keywords:	scalable simulation , topology reduction

# Scenario-Specific Topology Reduction in Network Simulations<sup>1</sup>

Barath Petit (bpetit@cc.gatech.edu)

Mostafa Ammar (ammar@cc.gatech.edu)

Richard Fujimoto (fujimoto@cc.gatech.edu)

College of Computing, Georgia Institute of Technology  
Atlanta, GA 30332 USA

## Abstract

Our current ability to perform packet-level simulations of large-scale networks is limited by factors such as processing speed and memory resources. Many approaches have been proposed to reduce the complexity of simulation models in a manner that reduces the computing resources required while preserving the fidelity of the simulation. In this paper we consider *topology reduction* as a means to reduce the resource requirements of network simulations. We show that it is possible to reduce topologies of simulation models while preserving certain classes of metrics. We do this in the particularly challenging environment where closed loop traffic (e.g., TCP) is being simulated in the network. The reductions are *scenario specific* in the sense that the desired performance metrics of a network simulation experiment dictate the feasibility of particular reduction methods. We use a set of experiments representing simulation scenarios to demonstrate our approach.

## 1 Introduction

Current communication networks are extremely complex making theoretical analysis difficult or impossible. The bulk of the complexity lies in the often vast number of interacting agents including protocols, traffic sources and routers. Thus, simulation is the only recourse for the analysis of large networks. Simulation of packet switched networks involves the modelling of every packet in the network. Packet level simulation requires many events and thereby simulation of large scale networks is very resource intensive in nature. Thus, our current ability to simulate large networks is limited by factors such as processing speed and memory resources. Reducing simulation runtime often involves reducing the number of events to be processed.

A reduction in model size of the simulation directly translates to speeding up of the execution times. Reducing the topology of the simulation would lead to a decrease in the number of hops to be traversed by a packet thereby yielding a savings in the number of packet events. Further, the memory requirements are also reduced. In [4] the authors list various methods for reducing Internet topologies with the objective of preserving the graph-theoretic properties. The methods are essentially traffic and topology independent and, thus, it is not clear that preserving graph-theoretic properties would indeed preserve the experimental outcomes of the network simulations. In order to demonstrate the importance of bandwidth awareness in topology reduction, we perform the following experiment. We removed some of the bottleneck links of the

topology of one of our experiments (to be described in section 5) and compared the results of the simulation run of the reduced topology with the results obtained by running the simulation on the original topology. In this case the outcome of interest is the response time distributions of the web flows. As seen in figure 1, the results do not match.

Present approaches to reduce simulation complexity pertain to traffic reduction[1] or abstraction of the simulation model[3]. However, reducing the topology to be simulated (to reduce simulation cost) while preserving the traffic properties and metrics (in the context of TCP/IP traffic) has proven to be difficult. As will be elaborated in section 3, the difficulty in reducing the topology owes much to the closed-loop nature of transport protocols employed in current communication networks. It is inherently difficult to characterize traffic from closed-loop sources since this requires prior knowledge of measures such as packet loss ratios. The difficulty further increases in direct proportion to the number of bottleneck links in the network. On the other hand it has been shown in [4] that it is possible to remove high capacity nodes in an ATM network, while preserving queuing behavior at the low capacity nodes that are retained.

In this paper we show that it is possible to reduce topologies of networks when the traffic is closed-loop in nature with the caveat that only certain classes of metrics are preserved as a result of the reduction. Thus, the reductions are *scenario-specific* since the metrics to be observed dictate the feasibility of the reduction method. We define the reductions to be scenario-specific because unlike current schemes whose feasibility is determined by the traffic in the network and network parameters, our methods' feasibility is dictated by the scenario or more specifically the performance measures to be determined.

The outline of the paper is as follows. In section 2 we present related work on approaches to reduce simulation complexity. We elaborate on the model reduction framework and a classification of network simulation scenarios in section 3. We list our topology reduction methods in section 4. We present the applicability of the reduction methods on three example scenarios in section 5 with a discussion of the results in section 6. We conclude the paper with directions for future work in section 7.

## 2 Related Work

Current approaches to reducing run-time of simulations are hinged on abstracting or reducing some aspects of the simulation. For example, recently approaches have been sug-

<sup>1</sup>This work is supported in part by NSF under contract number ANI-0136936

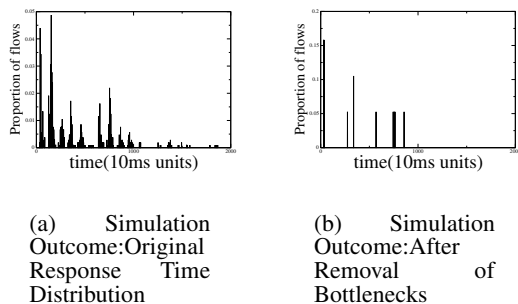


Figure 1: Sensitivity of Response Time Distributions to Bottlenecks

gested to reduce simulation run-time by scaling down the traffic fed to the simulator while keeping the network topology untouched [1]. In addition to the traffic, the network parameters such as link capacities and buffers are scaled by the corresponding factor but the number of nodes in the network remains unchanged. However, SHRINK[1] does not work for DropTail schemes in certain scenarios and its behavior in scenarios where the network composition is diverse remains unclear. Similarly RSM[2] scales down only the network parameters while leaving the network traffic and topology untouched. We have also observed that while RSM[2] preserves to some extent packet-level metrics such as queuing delay and link utilization, it fails for flow level metrics.

Another approach includes fluid simulations[3] that seeks to determine network characteristics by solving a set of closely coupled differential equations. Fluid simulation methods model packet transfer as fluid flows. Thus, each connection is assumed to have a certain amount of fluid to be transferred across the network. Event processing is required only when the fluid transfer is initiated or terminated and intermediate events that involve change of rate of fluid transfer. Thus, the fluid abstraction obviates the necessity of simulating every single packet. The fluid approach leads to event reduction in scenarios where network topology is moderate and dynamicity of traffic is limited. However, in cases where the topology is large or the traffic is dynamic the event rate exceeds that of the packet level simulation. This is due to the fact that dynamic traffic scenarios involve frequent rate change events when the transfer rate of a flow needs to be changed to accommodate a new flow arrival/departure or buffer overflow at the links. A rate change event at a link can trigger multiple rate changes in downstream links. This problem is exacerbated in large networks. This is referred to as the 'ripple' effect [3] and is one of the main limitations of fluid simulations. It is notable that fluid simulations are not scalable with respect to heavy web traffic and are not suitable for packet-level metrics. Note that the reduction methods discussed above *cannot be applied* to the simulation scenarios considered in this paper. More specifically, fluid simulation techniques fail to preserve packet-level measures and are prone to ripple effects (described above) in face of heavy web traffic. Similarly, as discussed above, SHRINK does not apply in scenarios where routers employ the DropTail scheme and RSM fails for flow-level measures.

### 3 Model Reduction Framework

A simulation experiment can be thought to be consisting of three main aspects viz. *the topology, traffic and outcome*. If the topology is a router-level topology then other factors such as the queuing scheme at the links come into play as well. The outcome of the experiment could be values or distributions of specific metrics or relative ordering among various metrics among others.

#### 3.1 Fine and Coarse Grained Metrics

Defining a metric as a measure that is a function of the traffic and topology of the simulation experiment, we could classify metrics as *fine-grained* or *coarse-grained*. Some examples of fine-grained metrics are packet-level metrics in a packet switched network such as the end-end delay (per packet), queuing delay and buffer occupancy at the routers. Some examples of coarse-grained metrics are:

- Response time distributions of web flows in a network with long-lived TCP flows as background traffic
- The distribution of response times, number of customers in servers in Content Delivery Networks (CDN), anycast delivery network scenarios.
- The query response time, rate of QueryHits in a Gnutella P2P network.
- Server metrics of web servers with finite job buffers

#### 3.2 Open and Closed Loop Traffic

With the aim of reducing router-level topologies, we classify metrics relative to the traffic used in the simulation experiment. Thus, the traffic would dictate the feasibility of reduction methods relative to the particular metric but the degree of reduction would be dictated by the topology. Within the context of the Internet we classify the traffic as *closed-loop*(elastic) and *open-loop*(inelastic). Before delving further, we outline the issues involved in reducing a topology for the above two traffic classes with respect to the corresponding metrics.

First we consider the open-loop traffic scenario. Prime examples are ATM networks with voice/video traffic. It is possible to characterize the traffic as a function of time for open-loop scenarios. For example, the traffic could be characterized by its peak rate or the envelope. It is possible to reduce these networks and still preserve fine-grained metrics such as buffer overflow probability or buffer length distribution. In [5] the authors demonstrate that in a network of ATM buffers fed with open-loop traffic removal of upstream nodes with high service rates leaves the queuing behavior at downstream nodes unchanged.

However, it is not possible to apply the same to a closed-loop traffic based network. For example, we consider now the case of closed loop traffic where the traffic is controlled by TCP-like congestion control algorithms. In these cases it is non-trivial to obtain traffic estimate without the prior knowledge of measures such as drop probability and round trip time. Furthermore, under this regime

sources modify their sending rates in response to network conditions. Thus, unlike the above case it is non-trivial to reduce the topologies while preserving fine-grained metrics such as throughput of flows, packet loss ratio and buffer occupancy at the routers.

### 3.3 Simulation Scenario Classes

With the above as background we now move on to classify simulation experiments into four main classes. The classification will be based on traffic and observable metrics assuming that we are dealing with router level topologies.

- *Class 1 scenario* consists of open-loop traffic cases. As pointed out in the discussion above, it is possible to reduce these networks while maintaining the fidelity of fine-grained metrics. Thus, the range of observable metrics under topology reduction includes both fine-grained and coarse-grained metrics (since we expect coarse-grained metrics to hold in the face of fine-grained metrics' fidelity).
- *Class 2 scenario* consists of networks that use closed-loop traffic and metrics to be observed are fine-grained like packet-level metrics.
- *Class 3 scenario* consists of networks that use closed-loop traffic but the metrics to be observed are coarse-grained besides being relatively insensitive to the closed-loop nature of the system.
- *Class 4 scenario* consists of networks that use a mixture of closed-loop and open-loop traffic. The observable metrics here are not only coarse-grained metrics but fine-grained metrics pertaining to the open-loop traffic as well.

The classification of simulation scenarios is summarized in Table 1. The classification also applies to QoS centric experiments. For example, if the metric of interest is an end-to-end packet delay bound determined as a function of link utilization, then assuming TCP based traffic, this experiment falls under class 2 scenarios. However, if the experiment pertains to evaluation of an admission control scheme based on the number of flows then this experiment will fall under class 3 scenarios. Similarly if the experiment pertains to packet scheduling schemes then since packet-level detail is central to the experiment, it falls under either class 1 or class 2 subject to the input traffic being TCP or open-loop.

Reduction methods for class 1 scenarios include [5] and [6]. To the best of our knowledge there are no known reduction methods covering class 2,3 and 4 scenarios.

Reduction methods for class 2 scenarios are non-trivial. The method would have to preserve fine-grained metrics such as packet level details. One plausible approach could be the fixed point approximation approach, whereby we aim to reduce the topology while preserving the operating point of the network. However, applying this approach seems non-trivial for networks with more than one bottleneck link[7].

In this work, we list several methods of topology reduction for class 3 and 4 scenarios as per the above classification. To preserve the fidelity of these metrics we reduce the topology represented by a graph  $G(V, E)$ , where  $V$  is the set of all nodes of the topology and  $E$  is the set of all the links in

the topology to a reduced graph  $G(V', E')$  where  $|V| > |V'|$  and  $|E| > |E'|$ .

Table 1: Classification of network simulation experiments

	Traffic	Observable metrics
Class 1	Open-loop	fine-grained,coarse-grained
Class 2	Closed-loop	fine-grained
Class 3	Closed-loop	coarse-grained
Class 4	Closed-loop and open-loop	coarse-grained and fine-grained(wrt open-loop traffic)

## 4 Model Reduction Methods

In this section, we outline different methods of reducing the topology of the network. We maintain the following invariants while reducing the topology.

- The addition of a new link in the topology should not induce a new bottleneck.
- The total propagation delay encountered by a packet of a flow remains unchanged.

**Notation** We use the following notation while describing the algorithms.

- Flows belonging to the same source destination pair are classified as belonging to the same *class*.
- $P(l) = \{S, N_1, N_2, \dots, D\}$  is the sequence of nodes traversed by the class  $l$  where  $S, D$  are the source and destination nodes.
- $C(m, n)$ : capacity of the bidirectional link between nodes  $m, n$
- $S_l(N_i)$ : for any node  $N_i \in P(l), N_i \neq S, D$  we refer to  $N_{i-1}$  as entry point of  $l$  through  $N_i$  represented by  $S_l(N_i)$
- $D_l(N_i)$ : for any node  $N_i \in P(l), N_i \neq S, D, N_{i+1}$  is referred to as the exit point of  $l$  through  $N_i$  represented by  $D_l(N_i)$ .
- $C_n$ : represents the set of all flows passing through node  $n$ .
- $L_n$ : for a given node  $n$  we define the set  $L_n$  as  $L_n = \{m : m = S_l(n) \forall l \in C_n\}$
- $R_n$ : for a given node  $n$  we define set  $R_n$  as  $R_n = \{k : k = D_l(n) \forall l \in C_n\}$
- $I_x$ : For each node  $x$  of the set  $R_n$  for a given node  $n$ , we define the set  $I_x = \{m : \exists l \in C_n \text{ s.t } m = S_l(S_l(x))\}$ .  $I_x$  represents the set of nodes that have flows coming into  $x$  through  $n$ .
- $M_x$ :  $\sum_{i \in I_x} C(i, n)$
- Throughout the reduction process, we denote the nodes that have been removed as *marked*. A set of nodes, all of whose members have not been removed is referred to as being *not marked*.

**Method-1** We remove a node from the topology if the outbound capacity exceeds the inbound capacity. The network is rewired by adding the propagation delays of the removed links. The new link has its capacity as the minimum of the previous two capacities and its link cost (used by the routing algorithm) is the sum of the previous two link costs. Before removing node  $n$ , we verify that the existing bottlenecks for the nodes in  $R_n, L_n$  are unchanged. In effect, we are removing links that induce negligible queuing delay. Thus, removing these nodes would have minimal effect on the class 3 and 4 scenario metrics. We remove nodes only if for all flows passing through the node, the inbound capacity is exceeded by the outbound capacity. More formally, method-1 can be described by the following pseudo-code.

```

for each  $n \in V$  do
  if  $L_n, R_n$  not marked  $\wedge L_n \cap R_n = \emptyset$  then
    for each  $x \in R_n$  do
      if  $M_x > C(n, x)$  then
        break
    if  $n$ 's removal modifies bottlenecks of  $R_n, L_n$ 
    then
      break
    else
      for each  $l \in C_n$  do
         $C(S_l(n), D_l(n)) = C(S_l(n), n)$ 
        Add propagation delays
        Modify link costs
      Remove  $n$  and all incident links
      Add newly formed links
      Mark  $n$  as marked

```

**Method-2** Since class 3 and 4 scenario metrics are insensitive to topology reductions, reduction of the capacity of new links to be added should not induce much distortion either. The reduced capacity on the other hand results in reduced number of packet events to be processed. Now let  $C_1$  and  $C_2$  be the inbound and outbound capacity of a flow through a node  $n$  and  $C_2 \geq C_1$ . We apply the same method as in method-1 to determine if a node  $n$  is to be removed. Then we replace the new link's capacity by  $C = (\frac{1}{C_1} + \frac{1}{C_2})^{-1}$  and its link cost (used by the routing algorithm) is the sum of the previous two link costs. This results in the new node providing the same transmission delay per packet as the previous two nodes in succession. As in method-1, the propagation delay of the new link is the sum of the delays of the links that it replaced. We apply this method as long as the new link does not induce a new bottleneck in the topology. The reduction in number of packet events is inversely proportional to capacity of the new link. Thus, since  $C < C_1$  and  $C < C_2$ , the method is likely to yield more reduction in terms of numbers of events to be processed. As will be seen below, the reduction in the number of events is reflected in the loss of throughput of the long-lived closed-loop traffic. More formally, method-2 can be described by the following pseudo-code.

```

for each  $n \in V$  do
  if  $L_n, R_n$  not marked  $\wedge L_n \cap R_n = \emptyset$  then
    for each  $x \in R_n$  do
      if  $M_x > C(n, x)$  then
        break
    for each  $l \in C_n$  do
       $C_{eff}^l = (\frac{1}{C(S_l(n), n)} + \frac{1}{C(n, D_l(n))})^{-1}$ 
      if  $C_{eff}^l$  modifies bottlenecks of  $R_n, L_n$  then
        break
    for each  $l \in C_n$  do
       $C(S_l(n), D_l(n)) = C_{eff}^l$ 
      Add propagation delays
      Modify link costs
    Remove  $n$  and all incident links
    Add newly formed links
    Mark  $n$  as marked

```

We examine the sensitivity of packet level metrics (class 2 scenarios) to the reduction methods described above. In a scenario where TCP long-lived flows share the network with web flows, the throughput of one of the long-lived TCP flows was observed to be 1.032 Mbps. After the removal of a single node (via method-2 described above), a difference of 13% was observed in the throughput. After a removal of another node by method-2, the difference in throughput increased to 27%. This reinforces the applicability of these methods to class 3 and 4 scenarios only.

#### 4.1 Reduction methodology

The reduction methodology is illustrated in figure 2. We use the ns-2 simulation script to extract the topology parameters. The parser processes the script and populates the data structures such as the adjacency matrix. In addition, the cost and delay matrices are also populated with the entries corresponding to the link costs (used by the routing algorithm) and the propagation delays of the links. These matrices, next-hop routing matrix and the traffic specification (source-destination tuples) are used by the reduction algorithm to mark edges and nodes for removal as detailed in the pseudo-code for method-1 and method-2. Note that we assume flat routing i.e., there is no hierarchical routing. This is necessary to ensure that the reduction algorithm preserves the propagation delay of each path in the topology.

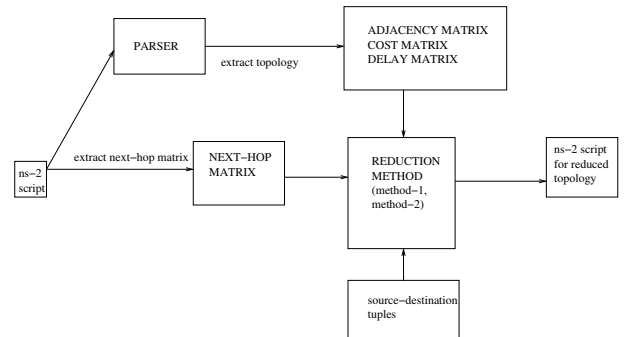


Figure 2: Reduction Methodology

## 5 Class 3 and Class 4 Scenarios

We elaborate on the effectiveness of the above reduction methods while dealing with class 3 and 4 scenarios. We consider three example scenarios noting that the metrics of interest are relatively insensitive to the the closed-loop nature of background traffic. The background traffic for all the scenarios consists of long-lived TCP flows. The scenarios are:

- Scenario A: This is a class 3 scenario where we study the response times of web requests.
- Scenario B: This is a class 3 scenario where we study web servers with limited job buffers. The metric of interest is the response time distributions.
- Scenario C: This is a class 4 scenario where we consider UDP sources in a network with background TCP traffic. The metric of interest is average one-way packet delay.

### 5.1 The Topology

We use the GT-ITM topology generator[8] to generate a 2420 node topology with 3442 links. We use the transit-stub model to generate the topologies. The router buffers of the links employ the DropTail policy. The propagation delay of the links were assigned uniformly in the range of 10ms to 50ms. The bandwidth of the links were uniformly assigned in the range of 1Mbps to 500Mbps. The foreground traffic consists of HTTP flows and the background traffic consists of long-lived TCP flows. The web traffic (for scenarios A and B) is generated using the webtraf module of the ns-2 network simulator. The foreground traffic for scenario C consists of UDP exponential on/off sources.

### 5.2 Scenario A: Web traffic

We study a scenario, where we are interested in the response times of the web requests in a network. In order to generate traffic for the experiment, 1000 source-destination pairs were chosen randomly. We refer to each of these source-destination pairs as a *flow*. The endpoints of each flow were designated as the server and the client. The background traffic consists of 100,000 long-lived TCP flows. For each flow, the requests for web objects is assumed to be Poisson at the rate of 20 per second. The size distribution for the requests is heavy-tailed with mean 3 packets and shape parameter 1.2.

**Results** We compare the response time distributions of original(unscaled) and reduced simulation runs by employing the Kolmogorov-Smirnov test[9]. The level of significance  $\alpha$  is assumed to be 0.05. While checking for the acceptance of the null hypothesis that the empirical distributions are the same, we also observe the value of the D-statistic. Note that the D-statistic gives us the absolute maximum difference in the empirical cumulative distribution functions of the samples. We list the results in Table 2 and figures 3 and 4. The run-time results correspond to 10 seconds of simulation time. As seen from Table 2, the application of the reduction methods distorts the mean response times by 0.45% (for method-1) and 0.62% (for method-2). The low values of the D-statistic indicate that the response time distributions obtained from the reduced topology simulation runs match the original response time distributions. This is reflected in

figures 3 and 4, where the response time distribution obtained by each method is superimposed against the original response time distribution. The difference in average response times of individual flows (between the original average response time of the flow and average response time of the flow in the scaled simulation) was also observed. The empirical probability distribution of the difference in average response time of each flow for method-1 is depicted in figure 5. From figure 5, we observe that most of the flows experience a distortion of less than 3% in the average response times.

Table 2: Scenario A: Results

	Unscaled	Method-1	Method-2
Mean response time(ms)	1762.19	1754.64	1751.05
D-statistic	N.A	0.0166	0.0191
Simulation Run-time(secs)	45900	35600	32400

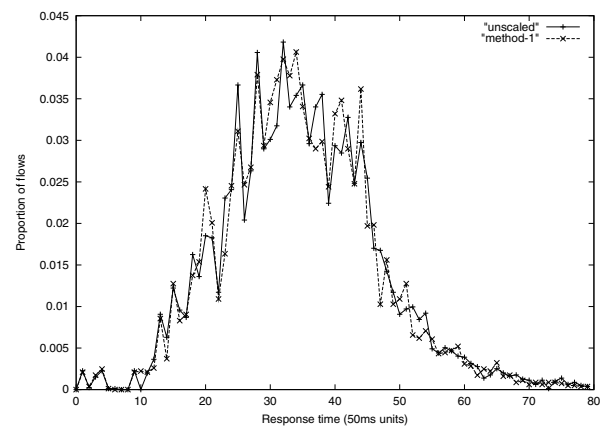


Figure 3: Scenario A: Response time distribution - Method 1

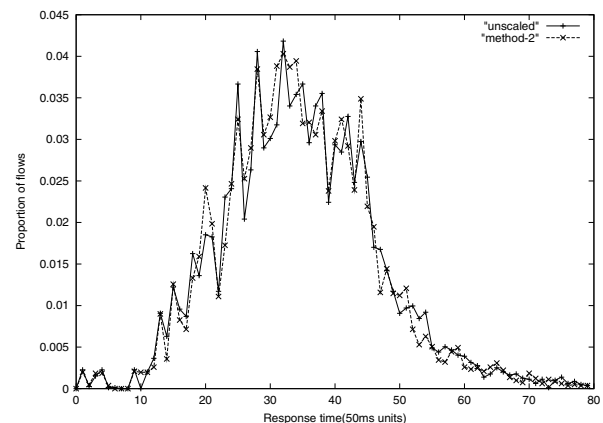


Figure 4: Scenario A: Response time distribution - Method 2

### 5.3 Scenario B: Servers with finite job queues

In scenario A, the web servers had zero processing overhead. In this section, we consider servers which induce finite pro-

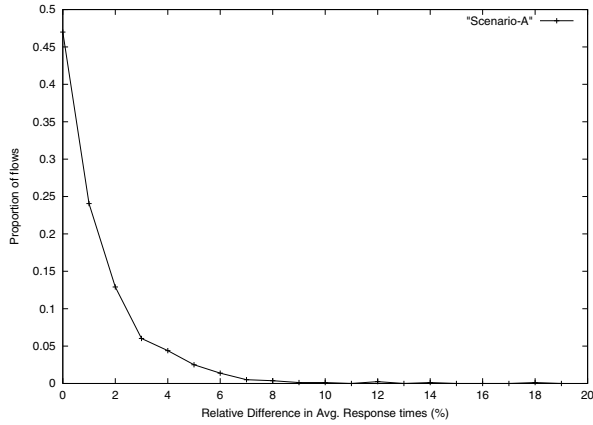


Figure 5: Scenario A: Relative Difference in Response Times

cessing delay for every incoming request, the delay being dependant on the size of the request. This entails a job buffer at the servers. Note that this queue resides at the application level and the processing latency of the server will depend on the CPU speed of the server. We denote the processing rate of the server by  $C$  expressed in packets per second. Thus, a request of size  $b$  packets entails a processing time of  $\frac{b}{C}$  secs. The size of the job buffer is denoted by  $B$ . An incoming request is enqueued if there is no buffer overflow. The request's *response time* is the sum of the queuing time, the processing latency and the transfer latency of the web object across the network. The *acceptance ratio* is defined as the ratio  $\frac{N-N_d}{N}$  where  $N$  is the total number of requests which arrive at the servers and  $N_d$  denotes the number of requests that are dropped due to buffer overflow at the job queue. Clearly factors such as  $B$ ,  $C$  and the distribution of the input arrival stream of the requests affect metrics such as distribution of response times and the acceptance ratio.

As in scenario A, we pick 1000 flows randomly and each flow is assigned a server and client. The background traffic consists of 100,000 long-lived TCP flows. The arrival rate for the requests for the web objects was assumed to be Poisson at the rate of 20 per second. The size distribution for the requests is heavy-tailed with mean 3 packets and shape parameter 1.2. The values of the parameters were  $B = 10$  and  $C = 220$ .

**Results** We list the results in Table 3 and figures 6 and 7. The run-time results correspond to 10 seconds of simulation time. As seen from Table 3, the distortion induced in the average response times and the acceptance ratios on the application of method-1 and method-2 is minimal. Similarly, the low value of the D-statistic for method-1 and method-2 is reflected in figures 6 and 7. The difference in average response times of individual flows (between the original average response time of the flow and average response time of the flow in the scaled simulation) was also observed. The empirical probability distribution of the difference in average response time of each flow for method-1 is depicted in figure 8. From figure 8, we observe that most of the flows experience a distortion of less than 3% in the average response times.

Table 3: Scenario B: Results

	Unscaled	Method-1	Method-2
Mean response time(ms)	1889.65	1885.67	1881.66
D-statistic	N.A	0.0124	0.0123
Acceptance Ratio	0.604	0.589	0.585
Simulation Run-time(secs)	46800	32400	31680

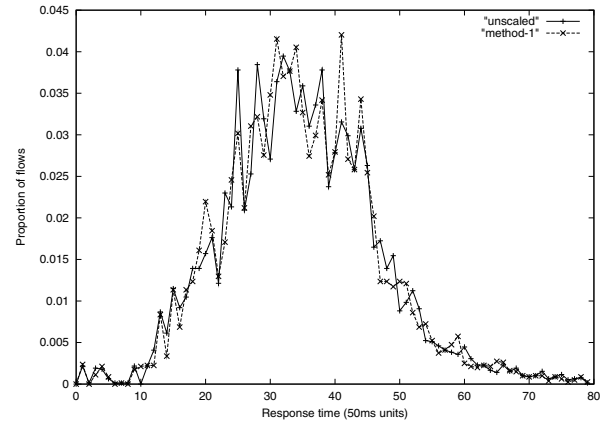


Figure 6: Scenario B: Response time distribution - Method 1

#### 5.4 Scenario C: UDP Sources

In this scenario, we consider a set of UDP sources with exponential on-off traffic. The mean on and off times were 10ms and 90ms respectively. We observe that topology reduction applied via method 1 preserves the mean one way per-packet delay. We consider two cases where the rate of each of the sources during ON time are 20 Mbps and 100 Mbps. The experiments included 1000 ON/OFF sources with the background traffic consisting of 1000 long-lived TCP flows. We present results for method-1 only since results for method-2 were not satisfactory.

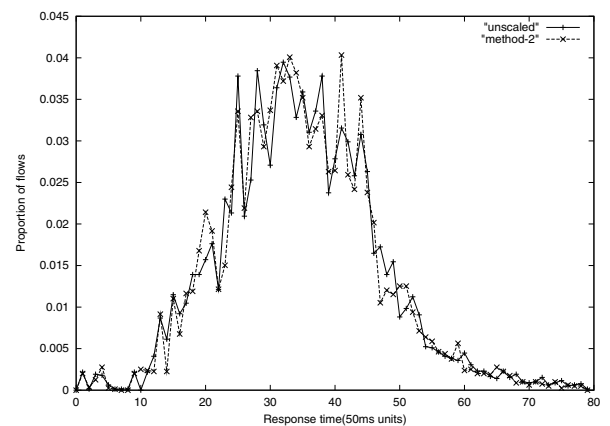


Figure 7: Scenario B: Response time distribution - Method 2

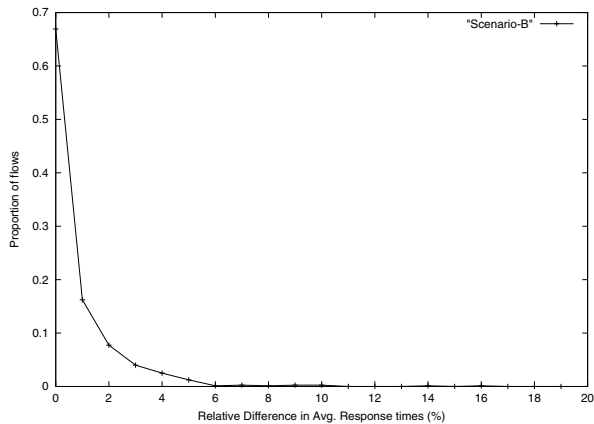


Figure 8: Scenario B: Relative Difference in Response Times

**Results** The results are listed in Tables 4 and 5. For the case where the rate of the sources is 100Mbps, we see from Table 4 that the distortion induced in the average one-way packet delay is 1.5%. For the case where the rate of the sources is 20 Mbps, we see from Table 5 that the distortion induced in the average one-way packet delay is 0.45%. The low values of the D-statistic in Tables 4 and 5 is reflected in figures 9 and 10 where we plot the empirical probability distribution of the one-way packet delays.

Table 4: Scenario C: Results (Rate=100Mbps)

	Unscaled	Method-1
Mean packet delay(ms)	177.841	180.52
D-statistic	N.A	0.020
Simulation Run-time(secs)	28600	16600

Table 5: Scenario C: Results (Rate=20Mbps)

	Unscaled	Method-1
Mean packet delay(ms)	194.421	193.532
D-statistic	N.A	0.0175
Simulation Run-time(secs)	14300	8403

## 6 Discussion

The results indicate the feasibility of the reduction methods for the above scenarios in particular and class 3 and 4 scenarios in general. The results for scenario A can be attributed to the inelasticity of web traffic. Since most of the web flows complete their transfer without exiting the slow-start phase of TCP, their behavior is inelastic in spite of the underlying transport protocol being TCP. Thus, as long as we preserve the core bottlenecks of the topology, bandwidth offered to the web traffic remains near constant. However, the behavior of the long-lived TCP flows is more sensitive to the closed-loop nature of the system, resulting in the loss of the throughput. This loss of throughput is reflected in the reduction of the

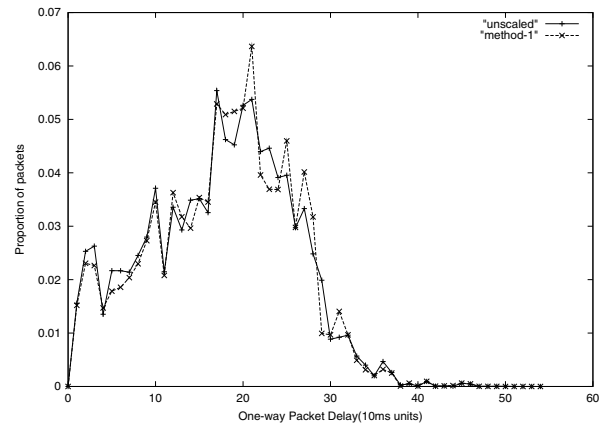


Figure 9: Scenario C: Packet Delay Distribution (Rate=100Mbps)

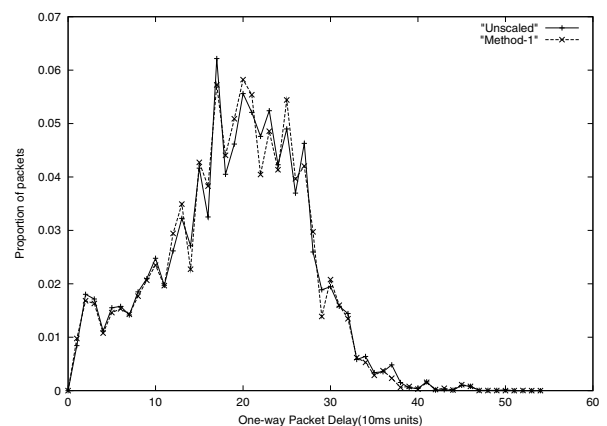


Figure 10: Scenario C: Packet Delay Distribution (Rate=20Mbps)

number of packet events to be processed, thus resulting in a reduction in the simulation run-time.

Results for the second scenario, where servers have finite job buffers could be attributed to the fact that the buffer size evolution depends on the arrival stream. To be more precise, the buffer size evolution is sensitive to the offered load defined by  $\frac{\lambda x}{C}$ , where  $\lambda$  is the arrival rate of the requests at the server,  $x$  the mean job size and  $C$  being the server processing rate ( $C$  is not the bandwidth of the link). However, since the arrival of the request stream is open-loop the model reduction does not change offered load on the server. In addition as in the previous scenario, the response time distributions are also preserved due to the inelastic nature of the web traffic.

In case of scenario C, we observe that the ON/OFF sources are inelastic in nature. Thus, as in previous scenarios, the reduction does not alter the offered bandwidth to the UDP sources.

## 7 Conclusion

In this paper we identified a class of network simulation experiments that are amenable to simplistic topology reduction methods. We listed two topology reduction approaches

and demonstrated their feasibility on three example scenarios. The example class 3 and 4 scenarios described in this paper typify the general class of scenarios that can be reduced by our methods. Our results show that despite the inherent complexity of closed-loop traffic, there exist some performance measures that are insensitive to the closed-loop nature of the system. This allows us to reduce the topology in such a way so as to preserve the metrics defined by the class 3 and 4 scenarios, though the fine-grained metrics of class 2 scenario incur loss in fidelity. Our future work hinges on developing topology reduction methods for class 2 scenarios.

1996 Conference on Computer Communications, INFOCOM 1996, (San Francisco, CA, Mar 24-28). IEEE, Piscataway, NJ, 594-602.

- [9] M. Hollander, D. Wolfe. 1999. "Nonparametric Statistical Methods." Wiley-Interscience, Hoboken, NJ.

### References

- [1] Rong Pan, Balaji Prabhakar, Konstantinos Psounis, Damon Wischik. 2003. "SHRiNK: A method for scaleable performance prediction and efficient network simulation." In Proceedings of the 2003 Conference on Computer Communications, INFOCOM 2003, (San Francisco, CA, Mar 20 - Apr 3). IEEE, Piscataway, NJ, 1943-1953.
- [2] Hwangnam Kim, Hyuk Lim and Jennifer C. Hou. 2004. "Network Invariant-Based Fast Simulation for Large Scale TCP/IP Networks." Research Report UIUCDCS-R-2004-2424, University of Illinois at Urbana-Champaign, April 2004.
- [3] Benyuan Liu, Daniel R. Figueiredo, Yang Guo, James F. Kurose, Donald F. Towsley. 2001. "A Study of Networks Simulation Efficiency: Fluid Simulation vs. Packet-level Simulation." In Proceedings of the 2001 Conference on Computer Communications, INFOCOM 2001, (Anchorage, Alaska, Apr 22-26). IEEE, Piscataway, NJ, 1244-1253.
- [4] Vaishnavi Krishnamurthy, Junhong Sun, Michalis Faloutsos and Sudhir Tauro. 2003. "Sampling Internet Topologies: How Small Can We Go?." In Proceedings of the International Conference on Internet Computing, (Las Vegas, NV, June 23-26), CSREA Press, 577-580.
- [5] Do Young Eun, Ness B. Shroff. 2003. "Simplification of Network Analysis in Large-Bandwidth Systems." In Proceedings of the 2003 Conference on Computer Communications, INFOCOM 2003, (San Francisco, CA, Mar 20 - Apr 3). IEEE, Piscataway, NJ, 597-607.
- [6] Costas Courcoubetis, Antonis Dimakis, George D. Stamoulis. 1999. "Traffic Equivalence and Substitution in a Multiplexer." In Proceedings of the 1999 Conference on Computer Communications, INFOCOM 1999, (New York, NY, Mar 21-25). IEEE, Piscataway, NJ, 1239-1247.
- [7] Tian Bu, Donald F. Towsley. 2001. "Fixed point approximations for TCP behavior in an AQM network." In Proceedings of the 2001 ACM SIGMETRICS international conference on Measurement and modeling of computer systems, (Cambridge, MA, June 16-20), ACM, New York, NY, 216-225.
- [8] Ellen W. Zegura, Ken Calvert, S. Bhattacharjee. 1996. "How to Model an Internetwork." In Proceedings of the