

Efficient Inter-domain Quality of Service Simulation Based on Time Series Information

S. Kardos, I. Ráncsik, A. Vidács

*Budapest University of Technology and Economics
Department of Telecommunications and Media Informatics
Magyar tudósok körútja 2.
Budapest, H-1117, Hungary*

E-mail: {kardos, rancsik, vidacs}@tmit.bme.hu

Abstract

The Intermon project focuses on the quality of service (QoS) issues of broadband inter-domain scenarios, as it is becoming more and more important to be able to analyze these large interconnected computer networks. Besides measurement and monitoring, simulation tools are required to assess their performance. However, traditional packet based simulators can be ineffective when simulating large backbone network traffic, because they have to update their state space with the arrival of every single packet. Therefore, as part of the Intermon simulation toolkit, we have developed a simulator architecture that handles network state descriptors in an aggregate, time series form. The simulator can be used to investigate load and packet drop rates at any point in the network, or delays and jitter along a specific path. As the state descriptors can come from real world measurements done by other tools of the Intermon toolkit, the simulator can be of high value when adding new traffic to the network, or rerouting traffic in certain parts of the network. At a slight cost in accuracy, the Time Series Simulator (TSSIM) performs much more efficiently than traditional simulators.

1. Introduction

While the size and the complexity of data communication networks have grown significantly recently, traditional packet-based simulation tools can hardly be used to analyze their behavior as these simulators spend too much time with state space maintenance caused by the huge amount of packets in the simulated network. The problem calls for new solutions, and many proposals have been made to speed up the simulation of these large, multi-domain networks.

One approach, proposed in [5], models a cluster of closely spaced packets as a single unit, a “packet train”. Another technique simplifying the simulation is the fluid model, first proposed by Anick *et al.* in [6]. The fluid simulation paradigm is analogous to natural liquids; it

operates on continuous amount of data rather than individual packet instances. Evenly spaced packets arriving close to each other may be modeled as a single fluid chunk with a constant fluid rate and small time-scale variations. A fluid simulator keeps track of the fluid rate changes at traffic sources and network queues, while an equivalent packet-level simulator would keep track of all packet instances in the network. The higher level of abstraction suggests that less processing might be needed to simulate the traffic. However, in [7] the authors found that the fluid simulation can sometimes be less efficient than packet-level simulation, which can be explained with the following. When the bandwidth sharing of fluid flows changes due to congestion, the flow rates need to be propagated throughout the downstream queues. This way a single rate change may affect a large amount of flows causing a so-called ripple-effect.

This paper describes the time series approach proposed for analyzing large multi-domain networks. It speeds up the simulation by evaluating the state space in large-scale discrete time points and by reducing the state space by handling only aggregate traffic.

The paper is organized as follows. Section 2 describes the approach in more detail; section 3 gives some comments on the implementation, section 4 gives validation results, and section 5 concludes the paper.

2. The time series approach

The Time Series Simulator (TSSIM) is a discrete time simulator, which handles network description information in the form of time series. This means the following. The timeline is divided into discrete time slots typically (but not necessarily) of equal size. Each time slot has certain traffic data assigned to it, such as the number of bytes arriving in that time slot, the number of bytes dropped in that time slot, or the average delay in it. The series of these data constitute a time series that the simulator can handle. The length of a time slot can be from a second to a minute, or even larger; it can be chosen freely. The longer the time slots, the larger the aggregation level.

Larger aggregation implies faster execution time, but smaller accuracy [1].

The following subsections describe how the network is modeled in the simulator.

2.1. Network model

The network model we use is similar to that described in [3]. The simulated network consists of network domains connected using inter-domain links (Figure 1).

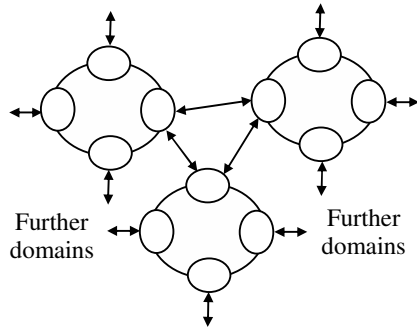


Figure 1. Network with domains

Each domain gets some input and produces some output that can be fed to other domains as input. Here, inputs and outputs are time series describing the aggregate load on the particular inter-domain link.

2.2. Domain model

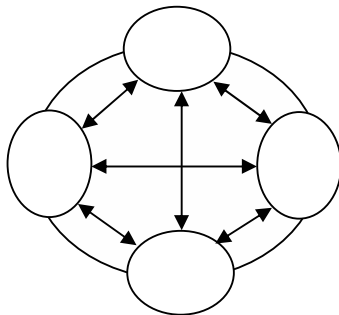


Figure 2. A single domain with border routers

Each domain of the network is represented by its border routers only. Core routers are not modeled, because it is supposed that the domain owner avoids congestion inside a domain. Thus the interior of a domain can be modeled as a constant delay at the border router, and all the border routers of a domain are connected directly to each other (Figure 2). Sources and sinks inside a domain are ignored. The simulator assumes that the output traffic of a given domain has no effect on its input traffic.

Each border router processes the time series that represents the input traffic and produces time series that are forwarded to the successive border routers. The input time series to a router can come from traffic sources or other border routers. Traffic sources generate traffic based on traffic source models or on some measurement done in a real network, while routers produce output based on their inputs.

When handling domain input traffic, each border router applies an input model specified for the router to the traffic, and then distributes the resulting time series of traffic among its neighbors in the same domain. Since the simulator handles the traffic as aggregated values in time series, the required routing information is given by a time dependent traffic distribution vector for each router. These traffic distributions tell the node how to divide the resulting traffic among the appropriate set of neighbors at the given time slot.

When generating output for the domain, the router merges the outputs of the input models of its domain neighbors, and applies an output model specified for the router to it. The result is distributed among the inter-domain neighbors similarly as it is done inside the domain (Figure 3).

In addition, the border routers compute QoS state information on the same aggregation level as the traffic is described in the input time series. This information includes the throughput, drops, delay and jitter of the traffic output by the router.

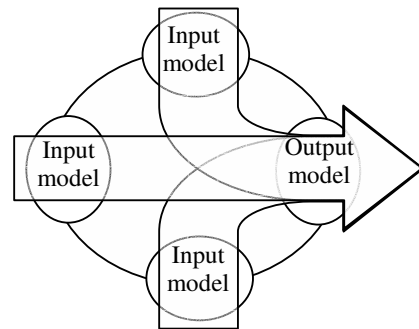


Figure 3. Models applied to the traffic at different border routers to determine domain output traffic

Since the simulator is based on aggregate load information, the simulated QoS values represent an average for the traffic as a whole. This state information can be used in the end to evaluate the simulation scenario. The more detailed the load information is the more accurate the QoS values will be with an increasing run time.

2.3. Router models

Because the border routers handle the traffic that enters the domain and the traffic that leaves the domain separately, two models should be specified for each router: an input model for the traffic that enters the domain and an output model for the traffic that leaves the domain. The models can be one of the following now:

- **NullModel:** This model simply copies the input to the output introducing only a static delay and distributing the output among the neighbours. It is primarily used as input model since the inside of a domain is supposed to be ideal except a static mean delay. This model has only one parameter: the mean delay to be introduced by the model.
- **LeakyBucket:** This models a simple FIFO queue with limited server capacity and queue length. It handles aggregate load information and produces throughput, drop, delay and jitter state information (the latter two at low accuracy). The LeakyBucket model requires the output rate, the initial length of the queue (`queueLen`), the queue size (`queueMax`) parameters to be set up (Figure 4).
- **Class4LeakyBucket:** This models the same queue as the LeakyBucket model but it handles aggregate and packet level load information too as described in [2]. It produces the same QoS state as the LeakyBucket model but at a higher accuracy. Class4LeakyBucket model takes the same parameters as the LeakyBucket model does plus the average length of packets modelled as background traffic (`backPacketLen`).

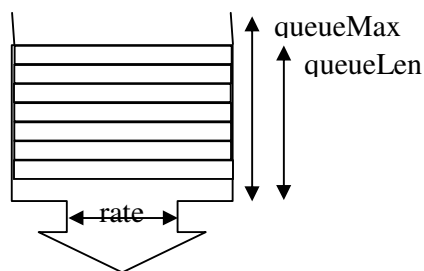


Figure 4. Leaky Bucket Model

2.4. Running time of Time Series Simulation

The main advantage of this simulator architecture is that the simulation time does not depend on the amount of traffic in the simulated network. During simulation the input time series are processed by each border router, so the simulation time depends on the length of the time series, the number of border routers and the complexity of the border router models.

3. Simulator implementation

3.1. Domain models

Models are pluggable modules that process the input of border routers they are assigned to. There is a well defined interface between the border router and the model through which the border router can initialize the model, can have its input processed by the model and can query the results from the model. Each model has the ability to distribute the output (in form of time series) among several output links according to a given distribution. The models are implemented as dynamically loadable libraries.

3.2. Execution of a simulation

After starting the simulation, the simulator first creates the border nodes, groups them into domains and defines the inter-domain links. Then it installs the input and output models and sets up the input traffic on the network edges.

The execution of a simulation job in TSSIM differs from that of a packet based simulator. It is divided into several runs and performed in an execution loop. In one run the simulator evaluates a whole time series (as input load) on a border router. Then it takes the next router, feeds in the previous output as input and does the computation again. The selection of the routers to evaluate is important. Certainly only those models can be evaluated that have every input available. Input can be given by sources or by the output of other routers.

In the node execution loop the simulator initializes a border node, lets it process its input, then propagates the output to the next nodes. Propagation means that the produced time series get set as input for the successive nodes. It repeats the loop till any unprocessed nodes exist. In the end it gathers the state information of the borders that are of interest.

3.3. Intermon toolkit integration

The TSSIM simulator has a graphical user interface as part of the Intermon simulation toolkit. The TSSIM GUI allows the selecting of input and output models along with setting their parameters to be used in the simulation. The models can be selected in a popup window activated by right-clicking on a border router. The model specific parameters can be set for each model on this same pane. Before starting a simulation, users can also select the link whose QoS state they are interested in by right-clicking on it, and checking the appropriate option. The simulator will only return results for the selected link.

The simulator can be accessed remotely through an XML-RPC interface. Simulation requests must specify the topology to investigate, and the description of the traffic.

TSSIM returns the time series of aggregate throughput, packet drops, delay and jitter in the unified XML format used by all intermon simulators. The simulator specific part of the results is stored as 'time load drop delay jitter' lines, where fields are separated by white space.

3.4. Simulator application

The Time Series Simulator evaluates the QoS state of the network in terms of throughput, drops, delay and jitter. This way, it can be used to analyze the effects of rerouting a part of the traffic on the QoS state of the network. additional traffic on the QoS state of the network.

4. Simulator validation

To validate the simulator, we use the results of measurements performed on a testbed at Salzburg Research. The investigated network has the following configuration. (Figure 5.)

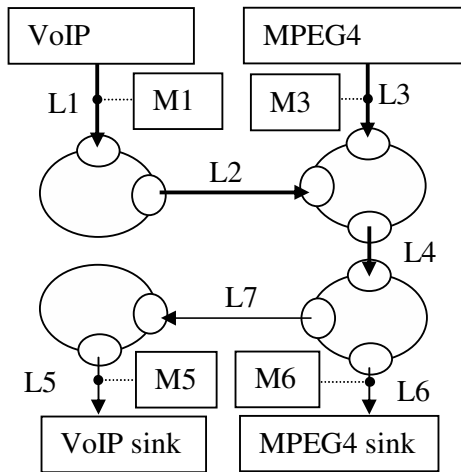


Figure 5. Investigated network

The network has four domains with Cisco 3600 and 7204 routers. Links L1-L4 have 100Mbps capacity, while links L5-L7 have 10Mbps capacity. The queue management mechanism is DropTail, the queue size is 100 packets (150000 bytes). There is only one class of traffic (best effort). Two traffic types are introduced: constant bitrate VoIP traffic from L1 to L5 and varying bitrate video traffic from L3 to L6. Flows share the capacity of L4. For VoIP, 10 flows (G.711 like) that produce 160 bytes every 20 ms are used. For video, high quality MPEG4 trace files from the Mr. Bean film are made use of. The traffic produced by the flows at the sources and destinations are measured using IPFIX meters [4], M3-M6. Meters are configured to report measurement data every 5 seconds. Seven measurements are performed,

10 minutes each. The average of the measurements is used in the following figures.

Simulations are performed using NullModel as input, and Class4LeakyBucket as output model for all border routers with parameters set as in the measurement scenario. The following figures compare the output of the simulator at the destinations to the data measured in the real network.

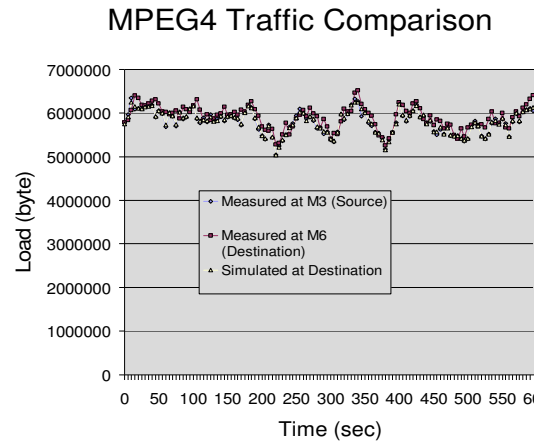


Figure 6. Comparison of measured and simulated results for the MPEG4 flows

As it can be seen in Figure 6, for the high volume MPEG4 streams, the output of the simulator almost exactly matches the data measured. For the VoIP traffic in Figure 7, however, it can be noticed that the simulated results show a much larger deviance than the measured data. This is caused by the low precision of the traffic distribution data that we used (only 2 decimal digits).

VoIP Traffic Comparison

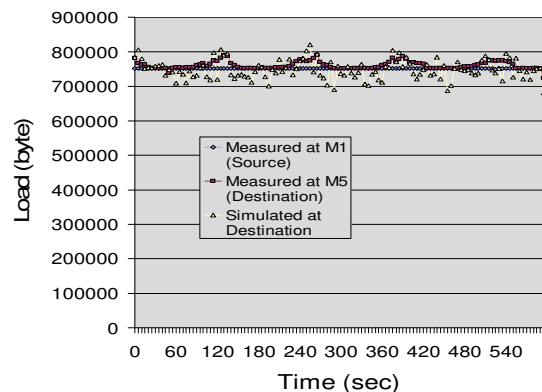


Figure 7. Comparison of measured and simulated results for the VoIP flows

5. Conclusion

Analyzing large inter-domain scenarios using packet based network simulators can be very time-consuming due to the high number of events that must be handled. TSSIM is an efficient inter-domain simulator, which works on the time series of aggregated traffic data, and evaluates the QoS state of the network in terms of throughput, drops, delay and jitter. As the tests have shown, the high level QoS values of the time series simulator match measurement data quite closely.

6. References

- [1] T. Máhr, T. Dreilinger, A. Vidács, "Time Series Based Simulation Architecture", pp. 31-36, IPS 2003, Salzburg, Austria, 21-22 February, 2003.
- [2] U. Hofmann, "Trace Based Traffic Modelling", pp. 57-60, IPS 2003, Salzburg, Austria, 21-22 February, 2003.
- [3] F. Baumgartner, M. Scheidegger, T. Braun, "Enhancing Discrete Event Network Simulators with Analytical Network Cloud Models", pp. 21-30, IPS 2003, Salzburg, Austria, 21-22 February, 2003.
- [4] F. Raspall, S. Tartarelli, M. Molina, J. Quittek, "Implementing an IETF IPFIX Meter", IPS 2003, pp. 85-86, Salzburg, Austria, 21-22 February, 2003.
- [5] J.S. Anh and P.B. Danzig, "Packet network simulation: speedup and accuracy versus timing granularity", IEEE/ACM Transactions on Networking, vol. 4, no. 5, pp. 743-757, Oct, 1982.
- [6] D. Anick, D. Mitra, M. M. Sondhi, "Stochastic theory of data handling system with multiple sources", The Bell System Technical Journal, col. 61, no. 8, pp. 1871-1897, Oct., 1982.
- [7] B. Liu, Y. Guo, J. Kurose, D. Towsley, W. Gong, "Fluid simulation of large scale networks: issues and tradeoffs", in Proc. of PDPTA '99, Las Vegas, NV, June, 1991, pp. 2136-2142.