An Integrated Simulator for Inter-Domain Scenarios

Matthias Scheidegger, Florian Baumgartner, Torsten Braun

Institute of Computer Science and Applied Mathematics
University of Bern, Neubrückstrasse 10, 3012 Bern, Switzerland
Email: (mscheid|baumgart|braun@iam.unibe.ch)
Tel: +41-31-63189557 Fax: +41-31-6313261

Abstract. The simulation of large-scale inter-domain networks is useful for various aspects of network planning and management. It is also a challenge due to its scalability problems, and the often incomplete knowledge about the network in question. Automatic configuration of the simulator based on network measurements may also be required. None of the many proposed approaches to network simulation are suitable for all possible simulation scenarios. A combination of several approaches provides much more flexibility. We describe a hybrid network simulator capable of combining packet-based simulation with a wide variety of other simulation approaches. A combination of packet-based simulation with analytical models is presented. This simulator is part of Intermon, an integrated system for the collection, processing, modeling, simulation and visualization of inter-domain data. The collected information is used for simulations allowing a "what-if" analysis of real networks. This paper describes the architecture and the underlying analytical models, and it evaluates the hybrid simulator.

1 Introduction

With the growing Internet and increased customer demands for quality of service, systems and tools for the monitoring and management of networks become more and more indispensable. This includes tools to monitor traffic within the network, and tools to analyze the gathered measurements in order to identify present and potential future problems in the network. Simulation is a powerful tool for predicting network performance but there are a number of problems that must be overcome to make it usable in the inter-domain context.

First and foremost is the problem of scalability. Classical network simulators like `ns2` do not scale to the network sizes and traffic volumes of today's inter-domain networks. Much research has gone into making simulators more scalable. Furthermore, simulation scenarios must be carefully configured if they should accurately predict the effects of changes in the real network. This should ideally be done automatically from measurement data. A simulator should thus provide a mechanism to support this task. Another problem is that many existing simulators assume perfect knowledge of the network's topology and traffic flows, which is rarely the case because of technical and political reasons. Even if an ISP has measurement data available it is very unlikely to publish it, as this could present a competitive advantage to other ISPs. Simulators should thus also be able to model parts of the network based on coarse-grained information obtained using network tomography tool, for example.

The "advanced architecture for INTER-domain quality of service MONitoring, modeling and visualization" (Intermon) project aims to develop an architecture for monitoring, modeling, simulation, prediction and the visualization of inter-domain quality of service. This includes tools for the monitoring process itself as well as the development of models and simulators to predict the behavior of inter-domain networks.

Within this project the Hybrid Simulator has been developed and implemented. It is based on the observation that none of the various simulation approaches that have been proposed in the literature fulfill all requirements from above. Instead, a combination of several approaches may provide much greater flexibility. The hybrid simulator extends the packet-based simulator `ns2` with other simulation approaches by using a hot-plug mechanism that can insert modules into the `ns2` topology. These modules simulate parts of the network using other, more adequate approaches.

A first plug-in implements an analytical modeling approach for multi-domain networks, which is based on the assumption that, at a given time, the Internet can be divided into areas where congestion is negligible, interconnected by bottleneck links. Congestion free areas are treated as black boxes with several ingress and egress points. Packet loss and delay caused by excessive queuing are only in the bottleneck links. Creating models for congestion free areas has the advantage that the simulation of packet losses and excessive queuing can be restricted to a small part of a simulation scenario (the bottleneck links), thus greatly reducing the complexity of the scenario as a whole. In fact it is sufficient to model congestion free areas using quasi-stationary delay distributions. Apart from its scalability advantage this approach may be useful to model network areas of which we do not know the exact topology. Moreover, a mechanism has been implemented that can automatically configure scenarios using measurement data provided by the Intermon architecture.

Suggested application areas for the hybrid simulator include end-to-end QoS evaluation of single flows – simulated using traditional packet-based models – over a complex backbone network, or the effect of changes in a backbone network (e.g. addition/removal of links, capacity changes, big changes of network load due to new SLAs, etc.) on flows traversing the network.

The remainder of this document is structured as follows: In Section 2 the Intermon architecture, the integration of the simulator in the system, and the hybrid simulator itself are described. Section 3 gives an overview about the analytical models used by the hybrid simulator. In Section 4 the simulation results are compared with measurements from a laboratory network, and Section 5 concludes the paper.

2 Architecture

The Intermon architecture (Fig. 1) has been designed as a highly distributed system. Measurements, data storage, simulation and visualization can all be hosted on individual systems. The communication elements are implemented in Java using the Java Messaging Standard API (JMS) as a communications middleware.

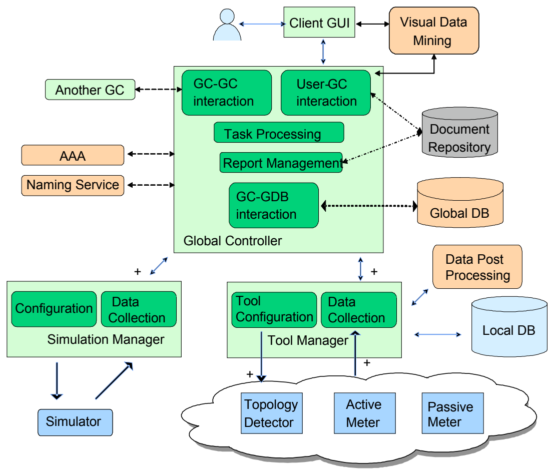


Fig. 1. A global view of the Intermon architecture

A central goal of the Intermon project is the integration of the different components into a single Intermon toolkit architecture. The integrated tools cover mechanisms for structure discovery of inter-domain topologies, as well as for measurement, modeling, simulation, and visual data mining of inter-domain traffic [1]. The key elements of the system are a global controller (GC), which accepts and forwards user requests, and the tool managers, which control the tools and convert messages between tool and GC into the appropriate format.

Since the Intermon system is distributed and depends on intense communication between the components, security issues have to be taken into account. Accordingly, all Intermon components communicate through a dedicated Virtual Private Network (VPN). Another advantage of the VPN is the simplified configuration of firewalls which may exist between the different hosts in the Intermon VPN.

2.1 Generating and Processing Simulation Jobs

Many applications of the Intermon toolkit, like "what-if" scenarios for example, require the use of simulation. A set of four distinct simulators, one of which is the hybrid simulator described below, has been implemented and integrated into the Intermon system in order to be able to adequately solve different simulation tasks and to study the advantages of different simulation approaches [2]. All integrated simulators can be interactively configured and controlled through a common graphical user interface, which can also be used to configure the meters deployed throughout the network gathering measurement data, and to visualize the simulation results.

A generic XML-based simulation job description format has been developed to control the different integrated simulators in a common way. This format separates the data into sections containing simulator-specific data and sections with simulator-independent data. Simulation job descriptions are structured as follows:

Topology information: The topology usually consists of data previously collected by the InterRoute tool [3], which queries BGP data and reconstructs the topology based on route advertisements and route withdrawals. Therefore this data reflects the current status of the inter-domain scenario to be simulated.

User applied changes: Any changes to the topology a user makes through the GUI are listed in this section. As the possible actions differ depending on the selected simulator, the content of the GUI dialogs is different for every simulator. Possible changes include removing a link, changing its capacity, or adding flows to the scenario, amongst others. Here, the user can also select and configure the analytical models used in the hybrid simulator described below.

Simulator specific parameters: Each simulator has some simulator specific parameters that are not related to the topology. They are encoded within this section. This allows the user to supply general instructions and parameters to the simulator. Examples of such parameters are the duration of the simulation and the granularity of the results.

The XML job description is generated by the graphical user interface and sent via the global controller to the tool manager of the appropriate simulator.

Since network simulations can require a lot of computing power and also can produce a huge amount of data, scalability was a central aspect during the design and the implementation of the hybrid simulator's tool manager and its underlying tool chain (see Fig. 2). The system supports the processing of simulation requests in parallel and thus supports multiprocessor computers as well as computer clusters.

The first steps in job processing consist of converting the XML simulation job description into configuration files for the hybrid simulator, and then starting the simulator. This is done by the demultiplexer component. After the simulation has finished, the multiplexer components takes care of processing the simulator tracefiles and stores the results in a data repository. Finally, it returns a message telling the system by which URL the results can be obtained.

The messages returned by the hybrid simulator's tool manager include some descriptive comments on the content of the reply and indicate the processing

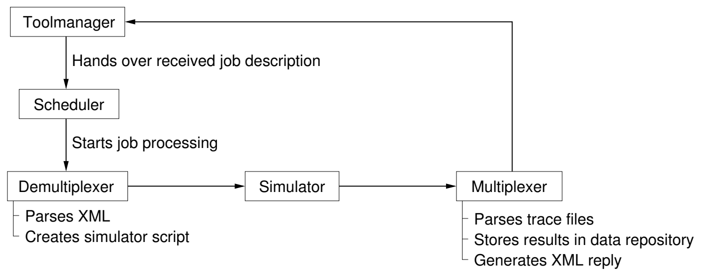


Fig. 2. The hybrid simulator's tool chain for the automatic processing of simulation requests

time and the status (e.g. running, ok, canceled, broken) of the simulation job. As the simulators may produce several different types of results, multiple result sections may be included in the reply, one for each desired measure (e.g. time series of throughput, or of end-to-end delay on a certain path). The sub-results have their own descriptive comment and a type identifier to help visualization.

2.2 Hybrid Simulator

The hybrid simulator aims to combine packet-based simulation with other simulation approaches to increase modeling flexibility and to overcome the limitations of the single simulation approaches used. In order to be able to integrate other simulation approaches with packet-based simulation, the `ns2` network simulator has been modified and extended. The main changes apply to the `ns2` node, whose structure is shown in Fig. 3. The typical node structure consists of an address classifier, a port classifier and a set of agents acting as traffic sources and sinks. The address classifier routes incoming packets either directly to outgoing links or to a port classifier forwarding the packets to an appropriate agent. We added a mechanism to the node structure to defer the delay and loss behavior of a node to loadable external modules, which can be implemented using arbitrary simulation approaches.

While the internal predictor module implements some minimal functionality to apply delay patterns to passing packets, its main function is to defer the respective computation to dynamically loadable external modules. In this case the predictor only provides an interface to the external module and takes care of delaying or dropping the packet inside the simulator.

Apart from the functions allowing for dynamic loading and unloading as well as initialization and configuration of the module, a predictor module only has to provide a single function `process_packet`, which is called for each arriving packet. Depending on the return value, the module interface within the node either delays or discards the packet (for values ≥ 0 or < 0, respectively). With each call to the this function, the predictor gets information about the packet itself, the previous and the next hop, and the simulation time. If the module requires another representation it has to generate it by itself, based on the events

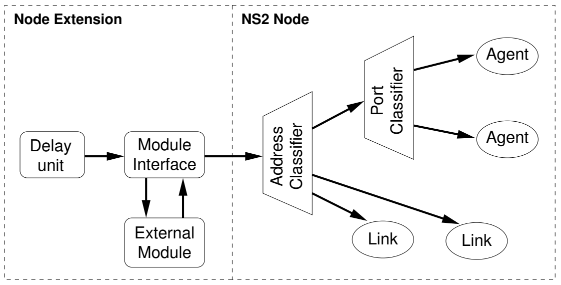


Fig. 3. Extension of the `ns2` nodenetwork simulator with an interface for delay and loss predictor modules

it receives. For example, if the module uses fluid flow representation internally, it will have to convert the packet arrival events into an estimate of current bandwidth.

3 An Analytical Modeling Extension for `ns2`

The extension mechanism described in Section 2.2 allows to load arbitrary modules into `ns2` nodes. A first fully useable extension module is based on a set of analytical models, which can be used to scalably model the behavior of whole inter-domain networks. In this section we introduce the ideas and concepts these models are based on and describe how they can be applied to improve the scalability of inter-domain simulation scenarios.

Scalability is often an issue when simulating large-scale computer networks. Especially packet-based simulators like `ns2` do not scale to scenarios with large topologies and high traffic volumes because of the large number of events to be processed. Many approaches to this scalability problem have been proposed. While parallel simulation [4,5] is arguably the most prominent one, alternatives include approaches such as fluid flow simulation [6,7,8], time stepped hybrid simulation [9] and packet trains [10], to name a few. Scalability in network simulation is generally achieved by reducing the level of detail (or accuracy) of the simulation scenario or of the simulation algorithm. Carefully chosen, such abstractions can considerably reduce the complexity of large-scale simulations without significantly distorting the results. However, an abstraction suitable for one task is not necessarily the right choice for other tasks. For the Intermon project a simulator was needed that can scalably simulate inter-domain networks while allowing for automatic measurement-based configuration, and which provides adequate abstractions for parts of the network where the topology cannot be determined.

We propose a model that is based on the assumption that, over certain time spans, networks like the Internet can be divided into areas where congestion is negligible, and which are interconnected by bottleneck links. We treat congestion free areas as black boxes, which we call *domain models*. Modeling congestion free areas has the advantage that we can neglect packet losses and excessive queuing

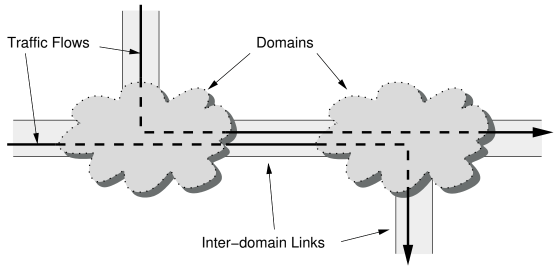


Fig. 4. The basic modeling view

in large parts of the network and restrict the model to quasi-stationary delay behavior. Apart from its scalability advantage this approach is primarily useful to model network areas of which we do not know the exact topology. Domain models can be based on empirical cumulative distribution functions (ECDFs) to simulate the delays of packets crossing the domain. The ECDF is chosen depending on the ingress and egress points on which the packet enters and leaves the domain, respectively. A big advantage of this concept is that delay measurements from a real network can be directly used to configure a domain model. It is important to note that ECDF models, while giving good reproductions of observed first and second-order moments in measurements, ignore any non-stationarity of the sample.

The bottleneck links between two domains of a simulation scenario are represented by *inter-domain link models*. Here, packet loss and queuing delay are simulated. The basic parameters of an inter-domain link model are similar to those of a link in a packet-based simulator. Nonetheless, inter-domain link models are not event-driven but rely on parameters like offered load and link capacity. Figure 4 shows this modeling view.

Traffic in the network is modeled using *application traffic models*, which serve as scalable models for large aggregates of application traffic like VoIP, Video, HTTP, etc. They take the form of a function $f(t)$ that yields the load generated by the traffic aggregate given a (monotonously rising) point in time. Also, traffic caused by `ns2` packets passing the extended node is also modeled by application traffic models.

By combining domain, inter-domain link and application traffic models we create a *multi-domain model*, which can then be inserted into the extended `ns2` simulator. We refer to this combination of analytical models and packet-based simulation as *hybrid simulation*. To determine how to handle a packet when it reaches the multi-domain model through the extension interface we have to update and inspect the model at the time of the packet arrival. This is done in three steps: First, the application traffic models must be updated, which includes the bandwidth estimators that convert between packet arrival events and load estimates. Second, any changes in traffic load have to be distributed through the multi-domain model. Then, we can decide whether to drop the packet or to delay it by a certain value by inspecting the domain and inter-domain link models along the path between the ingress and egress points of the packet.

One useful partitioning scheme for the above approach is to model autonomous systems (ASs) as domains, and their border links as inter-domain links. This partitioning is reasonable since the ingress routers of an AS may police flows to prevent congestion inside the AS. Moreover, the interior links usually have bigger capacities than inter-AS links, and internal routes may be changed to distribute traffic load. A possible application of this combination of fine grained packet-based simulation and coarse grained analytical models could be scenarios like a multi-site virtual private network. The local networks would then be modeled inside `ns2`, while the network in between would be modeled as a multi-domain model.

3.1 Modeling of Delay and Packet Loss

Packet loss and delay depends on the interaction of links and traffic flows in the network. We chose to model inter-domain links with the simple M/M/1/K queue, that is, an analytical queue with Poisson arrival and service processes, a single server (the physical link) and system capacity K. The arrival and service rates λ and μ depend on the offered load on the link and the link's capacity, respectively. The system capacity K, if unknown, can be set to a typical value (for example, 128-packet buffers are rather common in routers). Recent work [11] suggests that the arrival process would be better modeled as a Batch Markovian Arrival Process (BMAP). Also, sophisticated techniques like traffic-based decomposition [12] or the decomposition approach in Sadre et al. [13] could be used. These techniques also consider the effect of correlations in network traffic, which the M/M/1/K queue clearly ignores. However, while these approaches use traffic models that statistically describe traffic behavior over long time periods, the traffic models in our approach only describe the state of traffic sources at specific time instants – when a packet arrives at the multi-domain model. Correlations are thus only ignored on the small time-scale. The system's behavior in the long run is not modeled but rather simulated and hence also includes the effects of correlations.

In order to model the behavior of the inter-domain link we have to find the probability p_i of the system to be in state i, where state K means the queue is full, and state 0 means the system is empty and does not send. The M/M/1/K queue is a birth and death process with arrival and departure rates λ and μ, respectively. For a birth and death process of this kind the probabilities p_i are given by

$$p_i = \begin{cases} \frac{1-\lambda/\mu}{1-(\lambda/\mu)^{K+1}}\,, & i = 0 \\ (\lambda/\mu)^i p_0\,, & i > 0 \end{cases} \tag{1}$$

if $\lambda \neq \mu$, and

$$p_0 = p_1 = \ldots = p_K = \frac{1}{K+1} \tag{2}$$

if $\lambda = \mu$. As stated above, p_K is the probability of the system being full. Therefore, p_K is also the loss ratio of the link. The functional representation of the inter-domain link used in the section about multi-domain models above can thus be written as $L(\lambda) = (1 - p_K)\lambda$, with p_K calculated according to formulas 1 and 2. From the probabilities p_i we can further construct a discrete density function of the link's delay distribution. The number of bytes that are in the system when another byte arrives is proportional to the time this byte has to wait before it is sent to the link. δ_{pr} is the propagation delay on the link, which depends on physical properties of the link, e.g. its length. The discrete delay distribution looks like this

$$\begin{pmatrix} p_0 & \cdots & p_{K-1} & p_K \\ \delta_{pr} + \frac{1}{\mu} & \cdots & \delta_{pr} + \frac{K}{\mu} & \infty \end{pmatrix} \tag{3}$$

The infinite delay in the case of a full queue indicates that this packet is effectively lost.

Using the above results we can compute the loss ratio by multiplying the forwarding probabilities $(1 - p_K)$ of all the inter-domain links on the path. Similarly, the time it takes for a packet to traverse a path P can be described as a random variable δ_P with

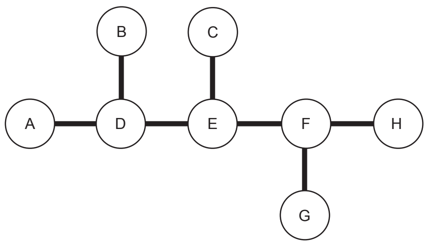
$$\delta_P = \sum_{i=1}^{n} \delta_{L_i} + \sum_{i=1}^{n-1} \delta_{L_i, L_{i+1}} \tag{4}$$

where δ_L is the random variable of the delay caused by inter-domain link L, and $\delta_{L,K}$ is the random variable of delay in the domain between the inter-domain links L and K ($\delta_{L,K}$ is only defined if L is a predecessor of K). Having a random variable of a path's delay further allows to easily calculate values like the path's mean delay or jitter.

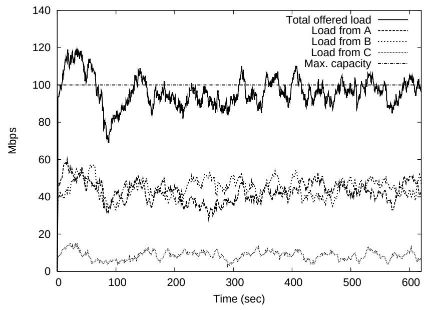
4 Evaluation

A first evaluation experiment for the hybrid simulator was done by comparing measurements from a testbed network to the results from the corresponding simulation scenario. The testbed was set up using the extended cross traffic topology shown in Fig. 5(a), which is useful to evaluate the performance of the inter-domain link and multi-domain models as well as the `ns2` toolchain. A comparison of domain models to real network behavior is less useful as these models are directly based on measurements from these networks. All nodes were Intel-based Linux systems, interconnected by 100 mbit ethernet links.

A reference flow was sent from A to H, and cross traffic was sent from B to G and from C to G. All three traffic sources consisted of multiple 1 mbit/s CBR streams with Pareto interarrival times and exponential hold times. The traffic load produced by these sources can be seen in Fig. 5(b). For reference, the theoretical maximum capacity of the the links is indicated as a line in the Figure.



(a) Topology



(b) Generated traffic

Fig. 5. Testbed setup

As the delay introduced by the nodes was expected to be minimal we chose to model them by domain models with zero delay. The ethernet links were modeled with inter-domain link models. Interfaces to `ns2` were attached at A and H, and a reference stream generated by `ns2` was sent along the path A-D-E-F-H to determine delay and loss ratio as well as to test the interface code itself.

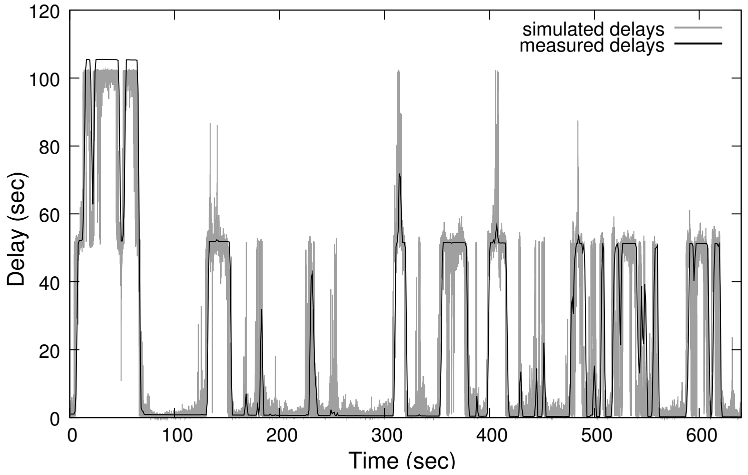


Fig. 6. Comparison of delays from testbed and simulation scenario

Figs. 6 and 7 show a comparison of the measurements and the simulation results. The delays in the testbed showed very little variance even with full queues, which is due to the CBR characteristic of the generated traffic. Consequently, the inter-domain link models, expecting Poisson arrivals, overestimated the traffic's burstiness. However, the mean of delay was similar in both, testbed results and simulator traces. As can be seen in Fig. 6 both graphs match rather well. Only when nodes E and F are under full load, packet forwarding in the routers begins to slow down slightly, which leads to the small gap between simulated and measured delays.

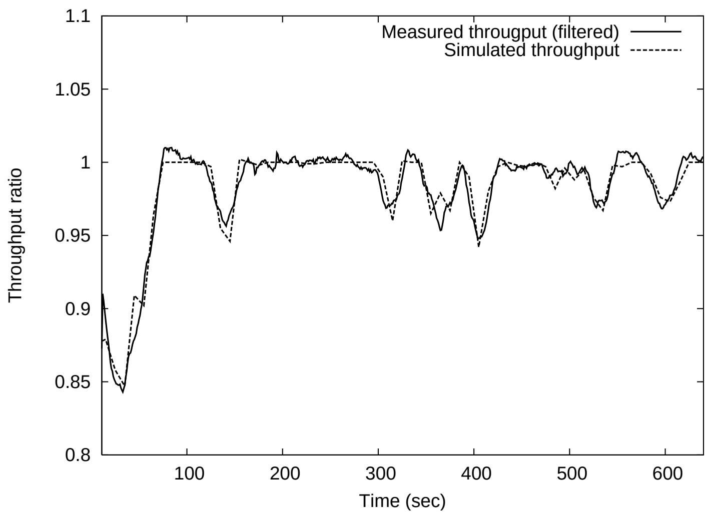


Fig. 7. Comparison of throughput from testbed and simulation scenario

In contrast to delay the throughput in the testbed proved to be rather bursty, probably because of interrupt timing in the routers. This didn't have much effect on the delay as the effect was hardly noticeable in comparison with queuing delay. However, the resulting small transient queues caused the throughput graph to be rather noisy. The graph in Fig. 7 was therefore smoothed using a box filter. Nonetheless, it can be seen that the simulated values closely match the testbed measurements.

5 Summary

In this document we presented a hybrid simulator for the scalable analysis of inter-domain networks. Measurements from other tools in the Intermon toolkit can be used to automatically generate scenarios, which makes the system suitable for "what-if" analysis. A tool chain handles the XML simulation job descriptions coming from the GUI, converts them to a local format, and generates an XML reply based on the simulation results.

The simulator itself is based on the combination of packet-based simulation with other simulation approaches to increase flexibility and scalability. A plug-in mechanism extension to the `ns2` simulator allows to attach arbitrary extension modules to a `ns2` node. A plug-in module enhances `ns2` by providing scalable analytical and stochastic models for the modeling of inter-domain networks. The evaluation showed that delay and throughput values obtained by the simulator are very similar to direct measurements in a laboratory test network.

Since the underlying network can be abstracted by an appropriate network model (e.g. the presented analytical), the size of the network does not have a direct impact on the complexity of the simulation. This increases the scalability of the simulation and, combined with the capability to simulate networks without an exact knowledge about their internal structure, makes the hybrid simulator a perfect tool for the measurement based simulation of large network scenarios.

A Acknowledgements

The work presented in this paper was carried out within the IST project "Advanced Architecture for INTER-domain Quality of Service MONitoring, modeling and visualization" (Intermon), funded by the European Community and the Swiss "Bundesamt für Bildung und Wissenschaft (BBW)."

References

1. Gutiérrez, P.A.A., Malone, P., Ofoghlu, M. et al.: Acquisition, modelling and visualisation of inter-domain routing data. IPS'04, Budapest, Hungary, 2004
2. Bartoli, M., Baumgartner, F., Scheidegger, M. et al.: The intermon simulation framework. IPS'04, Budapest, Hungary, 2004
3. Gutiérrez, P.A.A., et. al.: Integrating inter-domain routing analysis in novel management strategies for large scale ip networks. NEW2AN'04, St.Petersburg, Russia
4. Chandy, K.M., Misra, J.: Asynchronous distributed simulation via a sequence of parallel computations. Communications of the ACM **11** (1981) 198–205
5. Ammar, M.H., Riley, G.F., Fujimoto, R.M.: A generic framework for parallelization of network simulations. MASCOTS'99, College Park, MD (1999)
6. Yan, A., Gong, W.B.: Fluid simulation for high speed networks with flow-based routing. IEEE Transactions on Information Theory (1999) 1588–1599
7. Liu, B., Guo, Y., Kurose, J., Towsley, D., Gong, W.: Fluid simulation of large scale networks: Issues and tradeoffs. PDPTA'99, 2136–2142
8. Liu, B., Figueirido, D.R., et al.: A study of networks simulation efficiency: Fluid simulation vs. packet-level simulation. Proceedings of IEEE Infocom (2001)
9. Guo, Y., Gong, W., Towsley, D.: Time-stepped hybrid simulation (TSHS) for large scale networks. Proceedings of IEEE Infocom (2000)
10. Ahn, J.S., Danzig, P.B.: Packet network simulation: speedup and accuracy versus timing granularity. IEEE/ACM Transactions on Networking **4** (1996) 743–757
11. Klemm, A., Lindemann, C., Lohmann, M.: Modeling ip traffic using the batch markovian arrival process. Performance Evaluation (2003) 149–173
12. Heindl, A., Telek, M.: Output models of MAP/PH/1(/K) queues for an efficient network decomposition. Performance Evaluation (2002) 321–339
13. Sadre, R., Haverkort, B., Ost, A.: An efficient and accurate decomposition method for open finite and infinite-buffer queueing networks. Proceedings of the Third International Workshop on Numerical Solution of Markov Chains (1999) 1–20