Performance Modeling of Internet Nodes

Max Helm*, advised by Georg Carle*
*Chair of Network Architectures and Services, Department of Informatics
Technical University of Munich, Germany
Email: {helm, carle}@net.in.tum.de

***Abstract*—Recent changes in network architecture such as Software Defined Networking or Network Function Virtualization introduced more configuration and optimization vectors. To fully utilize that potential, accurate performance metrics of network elements are required. Those metrics can be acquired using different modeling techniques. We describe a high-level approach to automatically select the best fitting models based on reproducible and automatable measurements on real hardware.**

***Index Terms*—network modeling, performance, measurements**

1. Introduction

Recent advances in the architecture of networks and network components, such as Software Defined Networking (SDN) and Network Function Virtualization (NFV), allow for functionality to be physically decoupled from specialized networking hardware and for a central decision making entity with a complete view of the network. Consequently, routing decisions can be made with more available information and the method and physical location of packet processing can be chosen freely. Those degrees of freedom provide the opportunity for more optimization. For this optimization to be meaningful, we need access to key metrics of single nodes in the network as well as metrics of the network itself. Since doing live measurements of those metrics is infeasible and might interfere with the results, we need modeling techniques appropriate for the different metrics and use cases. There might even be a need for new modeling techniques or at least differently parameterized modeling techniques for some aspects of those networks (e.g. the SDN control plane).

Our approach for finding suitable models for the different components and use cases is divided into five steps. First, a set of models has to be identified that are applicable for the component type and for the level of granularity. Second, measurements on real hardware will be performed in order to have a real-life reference value for the components' performance. Third, the different models will be applied to the same components as the measurements to get predictions of their performances. Fourth, the parameters of the chosen model can be refined by a feedback loop using additional measurements. This step has to consider the complexity of the model, finding a trade-off between the amount of parameters and the quality of the results. Last, the measurement results and the model predictions are compared to each other to find the best fitting model.

2. Approach

This section details the five steps of our approach.

2.1. Model Identification

The approach for the initial selection of suitable modeling techniques is as follows. Models are categorized based on their metrics: what are needed inputs to the model, what are possible outputs of the model, what is the complexity of the model, and what is the granularity level of the model? Those information are then evaluated against the metrics obtainable from each component as well as what results the use case requires from the model. For example, do we want to optimize some metric? Or do we want to predict performance? Or do we need a hard upper bound on a metric? The results of this selection should be a set with elements of the form $\langle component, \{model_0, \ldots, model_n\}\rangle$.

A subset of possible modeling techniques is: Network Calculus, queuing models, resource models, and machine learning-based models. Additionally, simulations and emulations can be considered (e.g. sn-3[1], mininet[2]).

2.2. Modeling

The modeling of components should be automated. The goal is to have a single script that invokes a tool for each modeling technique. Additionally it should preprocesses the input data (component details) to fit the input format of each tool.

2.3. Measurements

The next step are the measurements. Measurements should be reproducible and easily automatable. To this end, we have the *Baltikum* testbed [2] at our chair. The testbed contains, among others, 1G, 10G, and 40G Intel and Mellanox Network Interface Cards (NICs). This allows for measurements under different circumstances, giving results which are valid for a greater amount of use cases. The testbed allows easy automation because of

1. nsnam.org
2. mininet.org

a built-in orchestrating service for measurements. Measurements on different hosts in the testbed can be started, stopped, repeated, and evaluated from a central controller.

The general setup of the measurements is as follows. There is one load generator and one device under test. The load generator produces and sends packets to the device under test, which processes the packets and sends them back to the load generator, which in turn calculates the required metrics and uploads them to the controller. The load generator in use is *Moongen* [1], since it allows for high throughput, timestamping, and equally sized inter frame spacing. The device under test can run different packet processing software based on the use case. Promising candidates for a broad set of applications are *Snabb*[3] and *Vector Packet Processing (VPP)*[4]. They can, for example, be used to implement the behavior of a forwarder, a switch, a firewall, or a traffic shaper.

2.4. Best Fit

The goal is to combine the measurements and the potential models into a single, best-fitting model. This can either be done using conventional methods, such as the Symmetric Mean Absolute Percentage Error or it can be done using machine learning. Additionally, we have to consider the complexity of the model, since a model with more parameters will almost always be more precise. A tradeoff between complexity and precision can be found using the Akaike Information Criterion.

2.5. Model Improvement

Continuous model improvement can be reached by supplying a feedback loop of measurements, model error calculation, and model parameter adjustments. This process should be automated as much as possible. Model error calculation uses the same methods as described in Subsection 2.4.

3. Conclusion

This short paper detailed our approach to finding suitable modeling techniques and model parameters for networking components based on active, reproducible, and automatable measurements. The need for this is outlined by explaining the impact of SDN and NFVs on network architectures.

4. Acknowledgements

This work has been performed in the context of the Model-based Analysis of Next-Generation Networks project – Deutsche Forschungsgemeinschaft (DFG) - Projekt number 397973531. The project is a cooperation between Technical University of Munich's Chair of Network Architectures and Services, and the Chair of Communication Networks.

3. github.com/SnabbCo/snabb
4. wiki.fd.io/view/VPP

References

[1] EMMERICH, P., GALLENMÜLLER, S., RAUMER, D., WOHLFART, F., AND CARLE, G. MoonGen: A Scriptable High-Speed Packet Generator. In *Internet Measurement Conference (IMC) 2015, IRTF Applied Networking Research Prize 2017* (Tokyo, Japan, Oct. 2015).

[2] GALLENMÜLLER, S., SCHOLZ, D., WOHLFART, F., SCHEITLE, Q., EMMERICH, P., AND CARLE, G. High-Performance Packet Processing and Measurements (Invited Paper). In *10th International Conference on Communication Systems & Networks (COMSNETS 2018)* (Bangalore, India, Jan. 2018).