# Predicting SLA Conformance for Cluster-based Services

Rafael Pasquini*†§, Farnaz Moradi‡, Jawwad Ahmed‡, Andreas Johnsson‡, Christofer Flinta‡, Rolf Stadler†§
* Faculty of Computing (FACOM/UFU), Uberlândia – MG, Brazil – Email: rafael.pasquini@ufu.br
† ACCESS Linnaeus Center, KTH Royal Institute of Technology, Sweden – Email: stadler@kth.se
‡Ericsson Research, Sweden – Email: {farnaz.moradi, jawwad.ahmed, andreas.a.johnsson, christofer.flinta}@ericsson.com
§Swedish Institute of Computer Science (SICS), Sweden

***Abstract*—The ability to predict conformance or violation for given Service-level Agreements (SLAs) is critical for service assurance. We demonstrate a prototype for real-time conformance prediction based on the concept of the capacity region, which abstracts the underlying ICT infrastructure with respect to the load it can carry for a given SLA. The capacity region is estimated through measurements and statistical learning. We demonstrate prediction for a key-value store (Voldemort) that runs on a server cluster located at KTH.**

***Index Terms*—Capacity Region, Feasible Region, Real-time Prediction, Statistical Learning, Service-level Agreement (SLA).**

## I. Background/Concepts

Understanding and predicting the performance of telecom services is intrinsically hard. Such services involve large and complex software systems that run on general-purpose platforms and operating systems, which do not provide real-time guarantees. Our approach to performance prediction is based upon statistical learning whereby the behavior of the target system is learned from observations. It has been described in our previous work [1], [2].

Fundamental to this demonstration is the concept of the capacity region. Given a service and an SLA, the capacity region characterizes the load that the underlying ICT infrastructure can carry while conforming to the SLA.

Figure 1 shows the capacity region for read and write operations performed on a key-value store (Voldemort [3]) running on our testbed at KTH. In this case, the load space has two dimensions, comprising the rate of read operations (horizontal axis) and the rate of write operations (vertical axis), respectively. The (carried) load of the system at any time is thus a vector in this space. The capacity region is shown in green and describes all possible load vectors the system can support while conforming to the SLA.

The red area in Figure 1 describes all possible load vectors that violate the SLA. The green and red areas combined make up the feasible region for the service on the given infrastructure and capture all possible load vectors for which the offered load is equals to the carried load.

The boundary of the capacity region is the line (manifold) that separates the green and the red parts of the load space. It has been learned through measurements, using a classifier.

ISBN 978-3-901882-94-4 

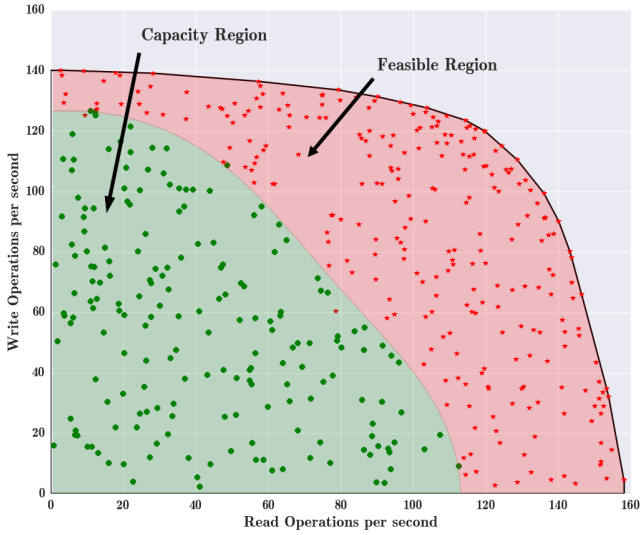



Fig. 1. The capacity region and the feasible region for a key-value store on our testbed. The green area defines the capacity region with the load vectors (read rates, write rates) that the testbed can carry while conforming to a given SLA. The red area describes the load vectors that violate the SLA. The green and red areas combined make up the feasible region.

Green dots in the figure show load vectors that conform to the SLA, red dots show vectors that violate the SLA.

The demonstration shows the behavior of the system under dynamic load patterns. When the load vector is within the capacity region, the prototype will predict SLA conformance, otherwise, SLA violation.

## II. Testbed

The demonstration includes a platform that implements the above approach in an on-line setting illustrated in Figure 2. The platform is an extension of the setup described in [1]. A management station provides access to the KTH testbed and displays measurements and predictions from the platform running on the testbed.

The testbed is deployed on a server rack in our laboratory at KTH. It includes ten high-performance machines interconnected by Gigabit Ethernet. Nine of them are Dell PowerEdge R715 2U servers, each with 64 GB RAM, two 12-core AMD Opteron processors, a 500 GB hard disk, and four 1 Gb network interfaces. The tenth machine is a Dell PowerEdge

R630 2U machine with 256 GB RAM, two 12-core Intel Xeon E5-2680 processors, two 1.2 TB hard disks, and twelve 1 Gb network interfaces. All machines run Ubuntu Server 14.04 64 bits, and their clocks are synchronized through NTP.

The Key-value (KV) store is deployed on six PowerEdge R715 machines, all of which act as KV store nodes in a peer-to-peer fashion, running Voldemort version 1.10.22 [3]. Device statistics are extracted from the kernel of the Linux operating system that runs on the servers executing the KV store. To access the kernel data structures, the $X$ sensor accesses the System Activity Report (SAR), a popular open-source Linux library [4]. The store is first populated with 10 million unique keys, selected uniformly at random from a 32-bit key-space. The size of the stored values is 40960 bytes. Each key-value pair is stored on three machines in the cluster. Consistent hashing is used to identify these machines. The KV clients run the benchmark tool from Voldemort and are deployed on a separate PowerEdge R715 machine. This machine measures the response times through the $Y$ sensor and produces load patterns for experimentation.

The Real-time Analytics Engine is written in Python and makes use of the scikit-learn package [5]. The classifiers for both the Load Space and Resource Space are also written using the scikit-learn package.

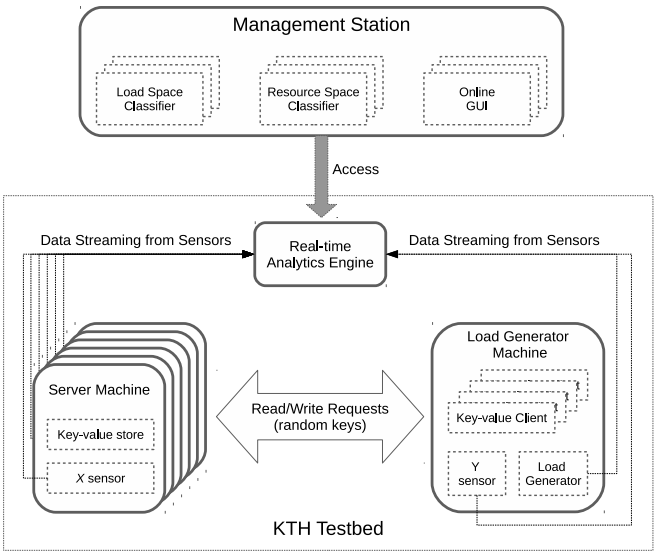
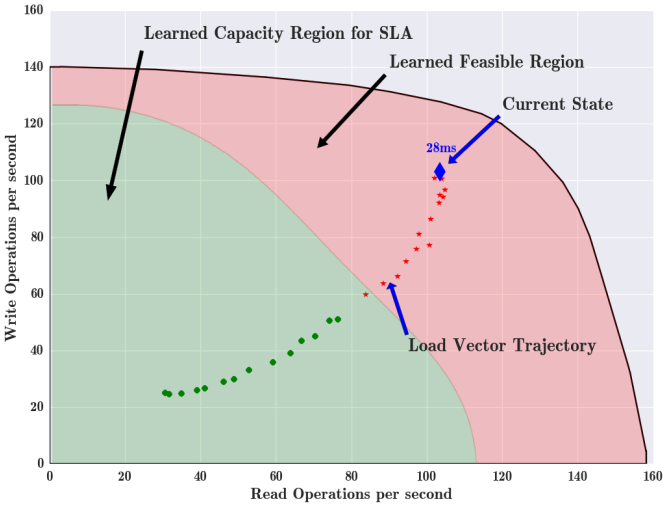


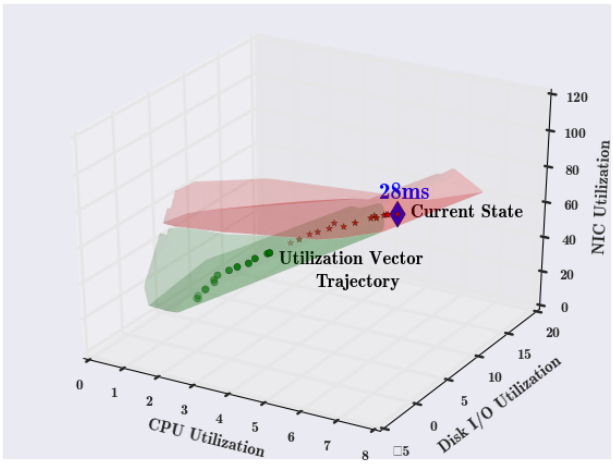

Fig. 2. Setup for the demonstration.

## III. Demonstration

The demonstration shows real-time predictions of SLA conformance or violation and the accuracy of those predictions for dynamic load patterns, as illustrated in Figure 3. The management station displays: 1) the learned feasible region in load space; 2) the learned capacity region in load space and its mapping into resource space; 3) the trajectories of the system state in load and resource space; 4) the predictions regarding SLA conformance or violation; 5) real-time measurements of read/write response times from a KV client, for both load space and resource space, used to validate the predictions.




(a) Trajectory of the load vector in the load space. The current response time is displayed for illustration purposes.




(b) Trajectory of the resource utilization vector in the resource space.

Fig. 3. Two windows of the management station.


## Acknowledgements

This research has been supported by the Swedish Governmental Agency for Innovation Systems, VINNOVA, through project SENDATE-EXTEND and by the Swedish Research Council through the ACCESS Linnaeus Centre.


## References


[1] R. Yanggratoke, J. Ahmed, J. Ardelius, C. Flinta, A. Johnsson, D. Gillblad, and R. Stadler, "Predicting service metrics for cluster-based services using real-time analytics," in *Network and Service Management (CNSM), 2015 11th International Conference on*. IEEE, 2015, pp. 135–143.

[2] ——, "Predicting real-time service-level metrics from device statistics," in *Integrated Network Management (IM 2015), 2015 IFIP/IEEE International Symposium on*, April 2015.

[3] Project voldemort - a distributed database. http://www.project-voldemort.com/voldemort/.

[4] S. Godard. SAR. http://linux.die.net/man/1/sar.

[5] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.