Towards a Cloud Performance Research Repository

Ian Gorton, Shravanthi Rajagopal
Khoury College of Computer Sciences
Northeastern University, Seattle Campus
Seattle, WA, USA
e-mail: i.gorton@neu.edu

Yan Liu
Faculty of Engineering and Computer Science
Concordia University
Montreal, CA
e-mail: yan.liu@concordia.edu

***Abstract*— In this paper, we describe our goals and initial efforts to create a community wide infrastructure for empirical research on cloud systems performance modeling. Performance analysis and prediction has long been an active research area in software engineering. Recent work in this area has focused on cloud-deployed applications that are characteristic of an ever growing percentage of modern software systems. The drivers for cloud deployment are improved quality attributes such as performance and scalability, access to advanced cloud-based technologies, and reduced costs. In terms of research, efforts include optimizing resource utilization for cloud systems under various workloads, understanding the influence on performance of various cloud-based infrastructure components, and architecture design trade-offs. To support these efforts and energize the creation of new models and methods, our goal is to build a repository that stores both raw and analyzed performance results from a collection of benchmarks characterizing different types of cloud applications. This data can be used by researchers to explore, calibrate and validate higher fidelity performance models and identify new research directions. Results from benchmarks can also help architects understand the effects of their architectural decisions and cloud services selections and configurations on application quality attributes.**

Keywords—performance, scalability, benchmarks, community research repository

I. INTRODUCTION

As contemporary systems continue to grow in complexity and scale, performance has become a driving software architecture quality attribute. Many examples demonstrate that Internet-based systems that are unresponsive or unable to scale see rapid loss of their user base. For example, a Google study [1] found that "53% of mobile users abandon sites that take longer than 3 seconds to load". This makes performance a key practical concern for software architects, who routinely strive to deliver 99th percentile response times of the order of a few hundred milliseconds from their services [19].

Software performance analysis and prediction is an established area of component-based software engineering research [7]. The research uses models to either predict the performance of a system before it is built, or analyze existing system performance usage models to inform the evolution of the system. The challenge of performance modeling is that the behavior of a component and system depends upon both its usage profile at runtime and the context in which it is deployed [28]. These challenges, despite a multitude of mathematically mature modeling approaches [2][3][4], have proven significant. In fact, no methods proposed from the research community have made widespread industrial impact [7].

Cloud computing has profoundly changed the landscape of software systems in the last decade [17]. For this reason, interest in measuring, analyzing and modeling cloud system performance has grown in the performance engineering research community [16]. Clouds however exacerbate the problems of performance prediction by exploiting virtualization, offering a multitude of platform services, and providing a shared set of resources with variable performance depending on collective load [6].

Accurate performance prediction of cloud applications requires the performance-relevant properties of cloud environments to be reflected in performance models [16]. Several current efforts utilize design patterns and metamodeling-based approaches [8][18][20][21] to derive performance models, or provide simulators of cloud platforms [10][24]. However, clouds are complex environments, and several researchers, for example [14][22][23], have reported counter-intuitive performance from cloud services and applications. This means that without detailed performance profiling of the cloud services to be modeled, useful and accurate predictions are unlikely to be produced.

Several approaches exemplify the need for detailed cloud platform measurements to calibrate and parameterize predictive models. For example, the PARIS approach [14] utilizes a two-level data collection approach, one for infrastructure and one for application characteristics. This data is used to parameterize a random forest model to analyze cost-performance tradeoffs for candidate VM specifications. Efficient autoscaling is the focus of [5][6][9], which utilize observed and predicted workload profiles to manage provisioning of new resources. Managing performance-based Service Level Agreements (SLAs) is the focus of Cerebro [15]. This work exploits static analysis and on-going response time monitoring with a time series analysis method to predict an upper-level 95th percentile for latencies on Cloud-based APIs. Others focus on predicting the performance of analytical applications [27], Apache Spark [13], Hadoop [25], NoSQL [26], and cloud hosted databases [11].

Underpinning these modeling approaches is the requirements to collect performance measurements. As [16] emphasizes and our experience supports, gathering and analyzing this performance data involves considerable effort. We believe this effort imposes considerable inertia on

researchers investigating new modeling and prediction approaches for cloud systems. Extensible benchmarking tools such as YCSB [12] or JMeter can be used to generate performance profiling data for an application. Regardless of the benchmark client however, the data must still be analyzed for anomalous effects, and tests run multiple times and for sufficient durations to ensure they are representative. Finally the raw data, which can be many MBs or more in size, must be processed and transformed into a form suitable for modeling.

In the remainder of this paper, we describe our proposal and early efforts for creating a curated community-wide repository for performance modeling of cloud based applications. We also briefly describe three results from our initial benchmarking efforts that show how the data we generate and curate can help researchers create higher fidelity, more comprehensive performance prediction models.

II. Cloud Performance Research Repository

Our aim is to build and deploy an online data repository of curated performance results for cloud application benchmarks. The repository would support the software performance engineering community by providing both the code for multiple benchmarks and workloads, and curated data sets obtained from running the benchmarks. The overall community repository design is shown in Figure 1.

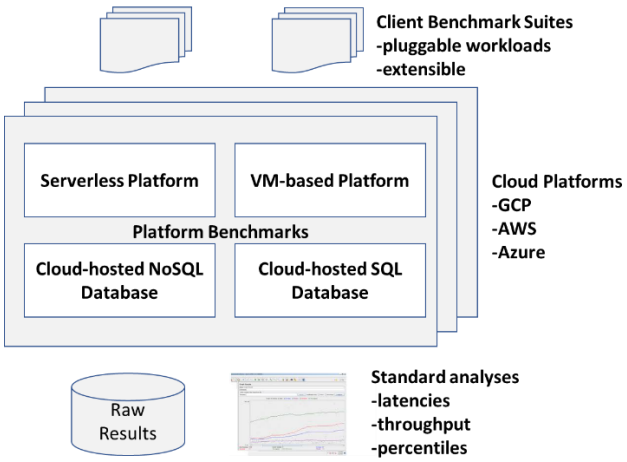

Figure 1 Cloud Performance Research Repository

Our initial work focuses on Web application benchmarks. These can be built in multiple programming languages and deployed on cloud platforms using either serverless or VM-based applications. They can store data in cloud hosted NoSQL or SQL databases. Each benchmark has a client and server component. The results of running the benchmark, along with metadata describing the test configuration, will be stored in the repository in raw form. We have also built a results processing pipeline that will generate standard analyses of the results, which will also be stored in the repository.

We are working with Google Cloud Platform (GCP) and Amazon Web Services (AWS) to build and execute our initial benchmark using different programming languages and storage platforms. We plan to expand to include Azure in the near future.

Once online, researchers will be able to search our repository based on benchmark metadata that characterizes the test environment. Researchers can then download both raw and processed data sets to support their creation, calibration and testing of performance models. They will also be able to download the benchmark code and run it in cloud environments and configurations that are not covered by the repository data.

The advantages for the software performance research community from this repository will be considerable. For example:

- There will be an unprecedented volume of performance data about cloud platforms, services and deployments that can be used for the creation of new models. For example, result files from our initial benchmark runs are ~700MB for each run. These represent time-series based latencies for tests of around 45 minutes duration.
- There will diverse a collection of data covering more cloud platforms, distinct services and application programming languages than any single project has so far investigated. This diversity will enable models to be refined to have greater utility and fidelity.
- The repository contents will support model comparisons. With a trusted, curated set of data for model calibration, researchers will be able to compare their model's predictive capabilities directly with others to assess benefits.
- The repository will reduce the cost and effort required by researchers to obtain data for their models. Obtaining high quality performance results requires considerable expertise in software design and familiarity with cloud service configuration. It is also expensive – many of our tests cost upwards of $50 in cloud costs per run.

We have commenced work on GCP and AWS and our initial set of results will be stored in a github repository along with the code and test metadata. Once we have completed our initial set of benchmarks, we will move the repository to a Web site and build custom search, filtering and simple analysis tools for the research community to exploit.

In the next section we briefly describe some initial results. These give an indication of the data and findings our repository will eventually hold, and the new areas of investigation they will create for modelers.

III. Example Findings

Our initial benchmark is designed to mimic a common Web use case where uploads of data are predominant, and reads are less frequent, requesting aggregated results. The application scenario is based on capturing and analyzing data from a wearable device such as a FitBit. Data on a user's step count is continually uploaded to a server that is deployed in a cloud, and queries ask for total step counts on a given day. Detailed information about the application API, workload and the implementations can be found in our git repo.

We have implemented the server on GCP's App Engine (GAE) in Java, Go and Python. All versions use the GCP's

NoSQL Datastore. On AWS we have built a Java implementation for both a load balanced virtual machine deployment and AWS Lambda, with the data stored in an EC2 MySQL hosted database. We followed a rigorous experimental approach, running each test configuration at least 3 times to average results, and performed statistical checks to reject outlier runs (eg unusual network congestion).

The experimental method and a more complete results analysis is fully described in the github repo. In the following we briefly present 3 results that to the best of our knowledge have not been presented in the literature, and we believe have implications for cloud systems performance models.

A. *Language Comparison – Go and Java*

To investigate the effect of server language selection, tests were run against the Go and Java servers, with both accessing the same logical database. Client workloads varied from 128 to 1024 concurrent threads. All GAE tuning parameters were left at default settings.

In Figure 2, we compare the mean throughput of the Go and Java servers. We see that the Go implementation consistently outperforms the Java code. At best, with 512 clients, the Java code delivers 84% of the throughput of the Go server. At worst, with 256 clients, the Java server lags at 61% of the Go server performance. We hence suggest that cloud performance models may need to be sensitive to the programming language utilized.

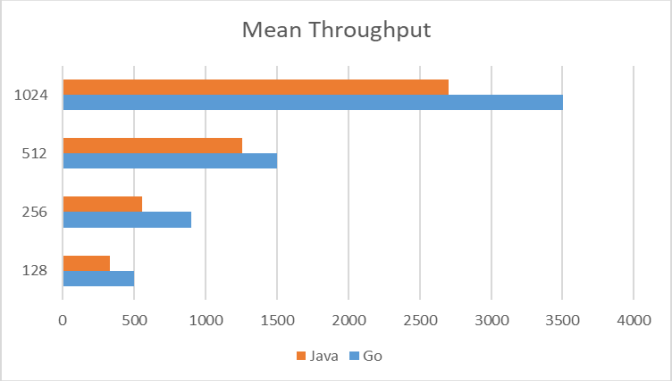

Figure 2 Comparison of Mean Throughput (requests/sec) for 128, 256, 512 and 1024 Clients

B. *Platform Comparison – AWS VM and Lambda*

To investigate the effects of VM-based versus serverless platforms, we ran a Java implementation of our benchmark on EC2. Both Java servers accessed the same MySQL AWS instance. As Figure 3 depicts, a load balanced EC2 configuration (with micro instances, 4916 requests/sec mean throughput) considerably outperformed AWS Lambda (with default configuration, 2989 requests/sec mean throughput) for 256 clients. Interestingly, while slower average. AWS Lambda provided a lower 99th percentile response time (166ms versus 210ms), indicating highly stable performance. We suggest this result indicates that cloud performance models should be sensitive to the deployment platform that is utilized.

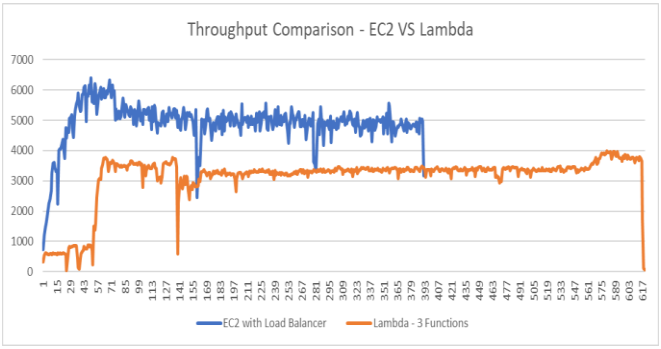

Figure 3 Throughput Comparison (Requests/sec) for Java server on EC2 and AWS Lambda

C. *Configuration Parameter Study – GCP*

In order to study the effects of GAE auto-scaling parameters, three tests were run for each of the six different configurations in Table 1. The three parameters, *Max_CPU*, *Max_Latency* and *Min_Latency*, govern the maximum CPU usage for an instance and how rapidly GAE scales up and down instances as the request load grows and decreases. The experiment utilized our Python server implementation.

Configuration Label	Max CPU	Max Latency	Min Latency
Default	0.6	30	30
Latency tuned	0.6	20	20
CPU relaxed	0.7	30	30
Full Tune	0.7	20	20
Full Tune+	0.8	20	20
CPU SuperRelaxed	0.8	30	30

Table 1 Configurations for Scalability Parameter Testing

As Figure 4 shows, we observed highest throughput with the *FullTune+* configuration. This has *Max_CPU* set to 80% as opposed to the 60% default setting, and also lowers the latency settings to 20ms from their 3oms default. In addition, this configuration also provides the lowest cost, with average test costs approximately 10% lower than other configurations. Hence we suggest that cloud performance models should be sensitive to the application configuration parameter settings.

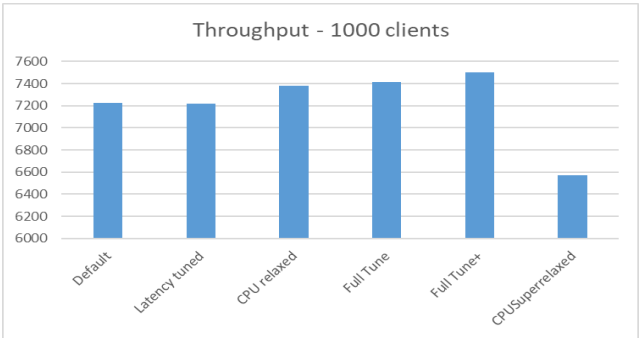

Figure 4 Configuration Comparison for 1000 Clients – Mean Throughput in Requests/sec

IV. Conclusions and Further Work

Modern cloud platforms present significant challenges for the software performance modeling community. In this paper we describe our proposal and initial work on building a community data repository focused on cloud application performance. The repository will provide comprehensive data on the performance implications of a range of architectural and platform choices. This will enable the software performance modeling community to experiment and create higher fidelity models with much greater applicability to future cloud architects.

Acknowledgment

We would like to acknowledge the GCP Research Credits Team for generously donating resources on the Google Cloud Platform to enable us to carry out this research. We'd also like to acknowledge AWS Educate for making AWS resources available for this project.

References

[1] https://developers.google.com/web/fundamentals/performance/why-performance-matters/ (retrieved January 2nd 2019)

[2] E.D. Lazowska, J. Zahorjan, S. Graham, K. C. Sevcik
Quantitative System Performance, Prentice Hall (1984)

[3] F. Bause, P.S. Kritzinger, Stochastic Petri Nets—An Introduction to the Theory (2nd ed.), Vieweg Verlag (2002)

[4] H. Hermanns, U. Herzog, JP. Katoen, Process algebra for performance evaluation, Theoretical Computer Science, 274 (1-2) (2002), pp. 43-87

[5] R.N. Calheiros, R. Ranjan, R. Buyya, Virtual machine provisioning based on analytical performance and QoS in cloud computing environments. InParallel processing (ICPP), 2011 international conference on 2011 Sep 13 (pp. 295-304). IEEE.

[6] S. Islam, J. Keung, K. Lee, A. Liu. Empirical prediction models for adaptive resource provisioning in the cloud. Future Generation Computer Systems. 2012 Jan 1;28(1):155-62.

[7] H. Koziolek. Performance evaluation of component-based software systems: A survey. Performance Evaluation. 2010 Aug 1;67(8):634-58.

[8] D. Franceschelli, D. Ardagna, M.Ciavotta, E. Di Nitto. Space4cloud: A tool for system performance and costevaluation of cloud systems. InProceedings of the 2013 international workshop on Multi-cloud applications and federated clouds 2013 Apr 22 (pp. 27-34). ACM.

[9] N. Roy, A. Dubey, A. Gokhale. Efficient autoscaling in the cloud using predictive models for workload forecasting. In Cloud Computing (CLOUD), 2011 IEEE International Conference on 2011 Jul 4 (pp. 500-507). IEEE.

[10] RN Calheiros, MA Netto, CA De Rose, R Buyya. EMUSIM: an integrated emulation and simulation environment for modeling, evaluation, and validation of performance of cloud computing applications. Software: Practice and Experience. 2013 May;43(5):595-612.

[11] B Mozafari, C Curino, S Madden. DBSeer: Resource and Performance Prediction for Building a Next Generation Database Cloud. InCIDR 2013 Jan 6.

[12] BF Cooper, A Silberstein, E Tam, R Ramakrishnan, R Sears. Benchmarking cloud serving systems with YCSB. InProceedings of the 1st ACM symposium on Cloud computing 2010 Jun 10 (pp. 143-154). ACM.

[13] K Wang, MM Khan. Performance prediction for apache spark platform. InHigh Performance Computing and Communications (HPCC), 2015 IEEE 7th International Symposium on Cyberspace Safety and Security (CSS), Aug 24 (pp. 166-173). IEEE.

[14] NJ Yadwadkar, B Hariharan, JE Gonzalez, B Smith, RH Katz. Selecting the best vm across multiple public clouds: A data-driven performance modeling approach. InProceedings of the 2017 Symposium on Cloud Computing 2017 Sep 24 (pp. 452-465). ACM.

[15] H. Jayathilaka, C. Krintz, and R. Wolski. 2015. Response time service level agreements for cloud-hosted web applications. In Proceedings of the Sixth ACM Symposium on Cloud Computing (SoCC '15). ACM, New York, NY, USA, 315-328

[16] M Hauck, J Happe, RH Reussner. Towards Performance Prediction for Cloud Computing Environments based on Goal-oriented Measurements. In CLOSER 2011 (pp. 616-622).

[17] D. Ardagna, E. Di Nitto, G. Casale, D. Petcu, P. Mohagheghi, S. Mosser, P. Matthews, A. Gericke, C. Ballagny, F. D'Andria, C. Nechifor, and C. Sheridan. 2012. MODAClouds: a model-driven approach for the design and execution of applications on multiple clouds. In Proceedings of the 4th International Workshop on Modeling in Software Engineering (MiSE '12). IEEE Press, Piscataway, NJ, USA, 50-56.

[18] G. Brataas, E. Stav, S. Lehrig, S. Becker, G. Kopčak, and D. Huljenic. 2013. CloudScale: scalability management for cloud systems. In Proceedings of the 4th ACM/SPEC International Conference on Performance Engineering (ICPE '13), Seetharami Seelam (Ed.). ACM, New York, NY, USA, 335-338.

[19] JF Pérez, G Casale. Assessing sla compliance from palladio component models. InSymbolic and Numeric Algorithms for Scientific Computing (SYNASC), 2013 15th International Symposium on 2013 Sep 23 (pp. 409-416). IEEE.

[20] F Brosig, N Huber, S Kounev. Architecture-level software performance abstractions for online performance prediction. Science of Computer Programming. 2014 Sep 15;90:71-92.

[21] J Kroß, A Brunnert, H Krcmar. Modeling big data systems by extending the Palladio component model. Softwaretechnik-Trends. 2015;35(3):1-3.

[22] M Hajjat, R Liu, Y Chang, TE Ng, S Rao. Application-specific configuration selection in the cloud: impact of provider policy and potential of systematic testing. InComputer Communications (INFOCOM), 2015 IEEE Conference on 2015 Apr 26 (pp. 873-881). IEEE.

[23] K He, A Fisher, L Wang, A Gember, A Akella, T Ristenpart. Next stop, the cloud: Understanding modern web service deployment in ec2 and azure. In Proceedings of the 2013 conference on Internet measurement conference 2013 Oct 23 (pp. 177-190). ACM.

[24] MC Filho, RL Oliveira, CC Monteiro, PR Inácio, MM Freire. CloudSim plus: a cloud computing simulation framework pursuing software engineering principles for improved modularity, extensibility and correctness. InIntegrated Network and Service Management (IM), 2017 IFIP/IEEE Symposium on 2017 May 8 (pp. 400-406). IEEE.

[25] X Wu, Y Liu, I Gorton. Exploring performance models of hadoop applications on cloud architecture. InProceedings of the 11th International ACM SIGSOFT Conference on Quality of Software Architectures 2015 May 4 (pp. 93-101). ACM.

[26] VA Farias, FR Sousa, JG Maia, JP Gomes, JC Machado. Machine learning approach for cloud nosql databases performance modeling. InCluster, Cloud and Grid Computing (CCGrid), 2016 16th IEEE/ACM International Symposium on 2016 May 16 (pp. 617-620). IEEE.

[27] S Venkataraman, Z Yang, MJ Franklin, B Recht, I. Stoica Ernest: Efficient Performance Prediction for Large-Scale Advanced Analytics. InNSDI 2016 Mar 16 (pp. 363-378).

[28] R Heinrich, E Schmieders, R Jung, K Rostami, A Metzger, W Hasselbring, R Reussner, K Pohl. Integrating run-time observations and design component models for cloud system analysis. CEUR Workshop Proceedings, 2014