Online Performance Modeling for NoSQL Databases using Extreme Learning Machines

Victor A. E. Farias [1], Pedro R. A. Pinheiro [1], Flávio R. C. Sousa [1], João P. P. Gomes [1], Javam C. Machado [1]

[1]Computer Science Department
Federal University of Ceara
Fortaleza, Brazil

{victor.farias, pedro.ramyres, flavio.sousa, joao.pordeus, javam.machado}
@lsbd.ufc.br

Abstract. *NoSQL databases rise as a solution to manage large amounts of data in the cloud. Mechanisms to guarantee Quality of Service in can significantly benefit from performance predictability. Building an accurate predictive model to estimate a DBMS performance in a cloud environment is challenging since i) workload and resources allocation change dynamically; ii) concurrency and distribution introduce nonlinearity on performance metrics and iii) predictive models should be trained and updated online to capture unseen workloads. This paper presents an online performance modeling approach for NoSQL databases using extreme learning machines. Experimental results confirm that our performance modeling can accurately predict throughput under several scenarios.*

1. Introduction

Cloud computing is a paradigm of remarkable success for service-oriented computing. Scalability, elasticity, pay-per-use pricing, and economy of scale are the major reasons for this achievement. Since most of cloud applications are data-driven, database management systems powering these applications are critical components in the cloud software stack [Elmore et al. 2011].

Cloud NoSQL database systems are leading to data processing environments that concurrently execute heterogeneous query workloads. At the same time, these systems need to satisfy diverse performance expectations defined in Service Level Agreements (SLAs). In these newly emerging settings, avoiding potential SLA violations rely on performance predictability, i.e., the ability to estimate the impact of concurrent query execution on the performance of the system in a continuously evolving workload [Duggan et al. 2011].

This work discuss a data-driven approach which applies machine learning to construct accurate performance models by generalizing training data. Performance models are able to predict system's performance metrics based on a given workload and a resource allocation configuration. Thus, monitoring system collects workload and systems configuration metadata from production environment to feed machine learning techniques.

Workload and resources allocation change dynamically, thus predictive models should update to capture the behavior of newly unseen workloads and system's settings

to improve accuracy. Retraining a new model on every update is impracticable since the number of workloads configurations may grow indefinitely which entails in a high computational cost. Thereby, in this scenario, a predictive model should update online in real-time with only most recent workloads while preserving accuracy.

Many NoSQL databases employ replication to improve reliability and performance, but it implies in an additional cost of maintaining consistency. Traditionally, consistency mechanisms are based on distributed locks which add waiting periods on queries execution. Authors in [Gray et al. 1996] suggests that the number of distributed locks grows quadratically, in the worst case, with the number of resource nodes and to the rate of transactions. Thus, the concurrent interaction of distributed queries execution is theses systems can degrade the performance in a non-linear fashion[Gray et al. 1996], which is a challenge for performance modeling approaches.

Modeling the performance impact of complex interactions of concurrent execution of a heterogeneous workload share computing resources and data is difficult albeit critical. Particularly, it supports many mechanisms used to guarantee Quality of Service (QoS) manning of cloud-based database platforms as efficient resource allocation and admission control. Performance prediction has been widely studied in the literature. This problem is also tackled by analytical approaches which analyzes each single factor that impacts the performance on a specific database system. However, these approaches can become very complex and system-specific.

Learning-based offline approaches were proposed to predict isolated queries performance in single node DBMSs [Ganapathi et al. 2009] and to predict concurrent workload performance for single node DBMSs [Duggan et al. 2011, Mozafari et al. 2013] and for distributed DBMSs [Farias et al. 2016b, Didona and Romano 2014, Farias et al. 2016a]. However, these approaches require a costly retraining when faced to new workloads and queries. An online approach is explored in [Sheikh et al. 2011] where the authors leverage gaussian models for predicting only single queries' response time under a concurrent workload in a single node DBMS. Our

This paper presents a machine learning approach for online performance modeling of NoSQL databases. The major contributions of this paper are as follows:

- We present a performance modeling approach for NoSQL database systems exploring machine learning techniques to predict system's performance metrics considering nonlinear effects of concurrency and distribution;
- We propose a strategy for online update in real-time of the predictive models;
- We present a full prototype implementation of the proposed approach and we evaluate models constructed by our prototype to assess their predictive accuracy.

Organization: This paper is organized as follows. Section 2 explains our approach. Section 3 describes the experimental evaluation. Conclusions and future research steps are shown in Section 4.

2. Our Approach

Our approach aims to construct, in a online manner, an accurate performance model M which predicts system's performance metrics based on a given workload configuration and on a resource allocation. The workload is composed by a set of request templates

$T_1, T_2, \ldots, T_n$. A workload configuration is the rate (request per second) $R_1, R_2, , R_n$ of each template, respectively.

NoSQL systems are designed to run in a distributed manner by taking advantage of a large resource pool. The data partitioning and management are database-specific. Thus, resource allocation is described by one parameter: the number of working nodes in the NoSQL cluster DB. This parameter is generic to any kind of NoSQL database system and it represents the capacity of the system.

The performance metric addressed to assess the accuracy of M is queries' mean response time (MRT) which is intuitive from the point of view of the database manager. We observe that our work is generic to other performance metric as throughput since the discussion in the remainder of this paper is not tied to MRT. So, formally, M is defined

$$M(DB, R_1, R_2, \ldots, R_n) \mapsto MRT$$

This setup of input variables allows us to capture several factors that can affect the performance of a distributed NoSQL database system. Particurlarly, three aspects: i) **Requests complexity**. Each single workload has a different access pattern and requires a different amount of disk operations. Each R_i variable independently accounts the complexity of template T_i; ii) **Concurrency**. Concurrent execution of two distinct workload classes may generate interference caused by lock resource sharing [Mozafari et al. 2013]; and iii) **Replication**. Most NoSQL databases distribution strategies employ replication to improve availability and reliability. However, this implies on an overhead to maintain consistency. Some consistency strategies use distributed locks leading to waits and deadlocks, and, consequently, degrading the performance in a non-linear fashion [Gray et al. 1996]. For factor ii and iii, We employ machine learning models capable of capturing interactions between inputs variables.

Our approach relies on machine learning algorithms and we show, by experimentation, that it can capture the aforementioned aspects with an acceptable error. In particular, since the performance metrics are continuous and non-finite, we address the performance modeling problem as a regression problem. Regression methods are capable of producing predictive models for estimating a system's performance given a set of training data under several scenarios of workload configuration and resource allocation.

2.1. Online Learning

Traditional regression methods require a training data set *a priori* to construct predictive models. In a cloud scenario, workload configuration and resource allocation change frequently. Machine leaning models tends to be more inaccurate when making predictions of a point $(DB, R_1, R_2, \ldots, R_n)$ distant from the samples on the training data set. Thus, a online continuous learning is necessary to capture newly unseen workload configuration and resource allocation to preserve accuracy.

Our approach is based on a feedforward neural network model with randomly assigned hidden weights as proposed in [Schmidt et al. 1992]. More recently, such model has gained much attention and is often referred as Extreme Learning Machine (ELM, [Huang et al. 2004]). This network is a standard feed forward single hidden layer network where the input layer is fully connected to hidden layer and the output layer is composed

by a single unit connected to all units in the hidden layer. The output of the network is given by:

$$O(x) = \sum_{1}^{n} w_i F(M_i^T X)$$

Each unit in the hidden layer of size n calculates a weighted ($M_i = [m_1, m_2, \ldots, m_n]$) sum of the inputs and applies a activation function F (sigmoid). Particularly, all M_i are uniform random values in $[-1, 1]$. Finally, the output of the network is the weighted ($W = [w_1, w_2, \ldots, w_n]$) sum of the output of the hidden layer. Weights W are adjusted to fit the data according to L_2 criterion.

In this work, we apply an online sequential approach for computing W proposed in [Liang et al. 2006]. Data arrives sequentially (one-by-one or chunk-by-chunk of varying size), it is learned and discarded at the end of the learning process. The algorithm learns in real time and does not have any adjustable hyper-parameter for tuning requiring least user intervention.

3. Experimental Evaluation

Our experiments were designed considering the goal of assessing learning effectiveness by measuring the accuracy in the learning online. We quantify accuracy by mean percentage error (MPE) metric so that the smaller the error, the more accurate is the model.

3.1. Experimental setup

We implemented a prototype of our approach using Python and Bash Script languages. We implemented ELM algorithm in python on top of Numpy library [Walt et al. 2011]. The model is update online in batches of 120 samples (120 seconds).

To generate the workload, we use the *de facto* NoSQL benchmark YCSB [Cooper et al. 2010]. This benchmark has four types of requests templates: point and range query, update and insert where we can simulate workloads by changing the rate of each template independently. In our experiments we fixed the following settings on the benchmark: 40 client threads, 1000000 records on database, 1-byte size record, uniform access pattern.

The test database in our experiments is MongoDB [Inc 2015] set on Master/Slave replication mode. The *ReadPreference* parameter is set to *Nearest* which means that MongoDB driver sends read for a randomly chosen node while writes are always issued to the master. This prototype was deployed in a Openstack private cloud environment. Each virtual machine holds 4 vCPUs and 4 GB main memory.

3.2. Experimental Results

We evaluate our performance model under a scenario that the workload configuration and the resource allocation vary over time. We explore two request templates of YCSB: point queries (T_1) and inserts (T_2). Thus, the experimental scenario is defined as the following ordered tuple representing (DB, R_1, R_2): $(7, 6000, 0)$,$(7, 4200, 1800)$,$(7, 5400, 600)$,$(7, 4800, 1200)$,$(7, 6000, 0)$,$(3, 6000, 0)$,$(5, 6000, 0)$ and $(6, 6000, 0)$. Each tuple is executed for 120 seconds.

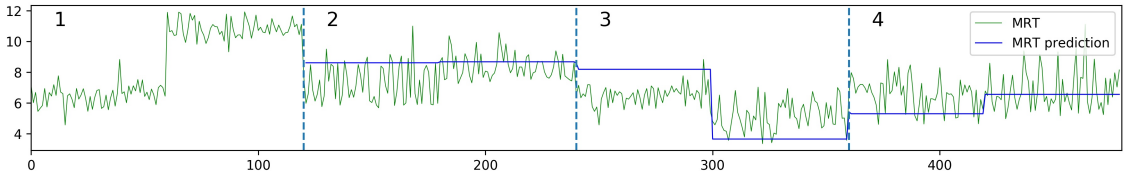

Figure 1. Experimental scenario demonstrating the online learning of our approach

Figure 1 plots the MRT for the experiment (in green) against the predicted MRT from the network. We divided the experiment in 4 windows separated by a vertical dashed line. In each windows, the monitoring data of previous windows were given for the network learning algorithm.

In the first 2 windows, the system the workload configuration changes and, in the last 2 windows, the resource allocation changes. We observe that in windows 2 the network could generalize the the behavior from windows 1 and make accurate predictions ($MPE = 16.5\%$). From second 300 to 360, the system had the first change of resource allocation so it obtained a inaccurate estimate ($MPE = 29.2\%$) since it has no information about resource allocation so far. However, in window 4, the network has learned from window 3 and made better estimates for resource allocation changes ($MPE = 17.1\%$). The MPE obtained for the whole experiment is 20.9%.

4. Conclusion

This paper presents a novel approach for online performance modeling for NoSQL databases. This approach can create performance models with production monitoring data. We conducted experiments to show that such model is accurate and can learn online to preserve accuracy. Experimental results confirm that this model predicts with 20.9% of error for a new workload configuration and resource allocation showing that this approach can generalize from the past experience. This work opens discussion for new research opportunities. There is a number of research opportunities that derive from this work, including: guided online learning where we can adjust system's knobs (as DB parameter) to obtain a better data set to improve model's accuracy; and concept drift handling, e.g., the cloud manager substitute the hard disk drive for a faster device as solid state disk, thus the performance changes abruptly making the trained models inaccurate.

Acknowledgment

This Research was partially supported by LSBD/UFC.

References

Cooper, B. F., Silberstein, A., Tam, E., Ramakrishnan, R., and Sears, R. (2010). Benchmarking cloud serving systems with ycsb. In *Proceedings of the 1st ACM symposium on Cloud computing*, pages 143–154. ACM.

Didona, D. and Romano, P. (2014). On bootstrapping machine learning performance predictors via analytical models. *arXiv preprint arXiv:1410.5102*.

Duggan, J., Cetintemel, U., Papaemmanouil, O., and Upfal, E. (2011). Performance prediction for concurrent database workloads. In *ACM SIGMOD*, pages 337–348. ACM.

Elmore, A. J., Das, S., Agrawal, D., and El Abbadi, A. (2011). Zephyr: live migration in shared nothing databases for elastic cloud platforms. In *SIGMOD '11*, pages 301–312.

Farias, V. A. E., Sousa, F. R. C., Maia, J. G. R., Gomes, J. a. P. P., and Machado, J. C. (2016a). Elastic provisioning for cloud databases with uncertainty management. In *ACM SAC*, pages 390–397.

Farias, V. A. E., Sousa, F. R. C., Maia, J. G. R., Gomes, J. P. P., and Machado, J. C. (2016b). Machine learning approach for cloud nosql databases performance modeling. In *CCGrid*, pages 617–620.

Ganapathi, A., Kuno, H., Dayal, U., Wiener, J. L., Fox, A., Jordan, M., and Patterson, D. (2009). Predicting multiple metrics for queries: Better decisions enabled by machine learning. In *ICDE*, pages 592–603. IEEE.

Gray, J., Helland, P., O'Neil, P., and Shasha, D. (1996). The dangers of replication and a solution. In *ACM SIGMOD Record*, volume 25, pages 173–182. ACM.

Huang, G.-B., Zhu, Q.-Y., and Siew, C.-K. (2004). Extreme learning machine: a new learning scheme of feedforward neural networks. In *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*, volume 2, pages 985–990. IEEE.

Inc, M. (2015). *MongoDB*. `http://www.mongodb.com`.

Liang, N.-Y., Huang, G.-B., Saratchandran, P., and Sundararajan, N. (2006). A fast and accurate online sequential learning algorithm for feedforward networks. *IEEE Transactions on neural networks*, 17(6):1411–1423.

Mozafari, B., Curino, C., Jindal, A., and Madden, S. (2013). Performance and resource modeling in highly-concurrent oltp workloads. In *ACM SIGMOD*, pages 301–312. ACM.

Schmidt, W. F., Kraaijveld, M. A., and Duin, R. P. (1992). Feedforward neural networks with random weights. In *Pattern Recognition, 1992. Vol. II. Conference B: Pattern Recognition Methodology and Systems, Proceedings., 11th IAPR International Conference on*, pages 1–4. IEEE.

Sheikh, M. B., Minhas, U. F., Khan, O. Z., Aboulnaga, A., Poupart, P., and Taylor, D. J. (2011). A bayesian approach to online performance modeling for database appliances using gaussian models. In *Proceedings of the 8th ACM international conference on computing*, pages 121–130. ACM.

Walt, S. v. d., Colbert, S. C., and Varoquaux, G. (2011). The numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, 13(2):22–30.