Performance Prediction for Burstable Cloud Resources[*]

Daniel J. Dubois, Giuliano Casale
Imperial College London, Department of Computing, London, United Kingdom
{daniel.dubois, g.casale}@imperial.ac.uk

ABSTRACT

We propose ForeBurst, an open source tool for performance prediction for complex cloud-based applications. ForeBurst leverages queueing network models for predicting performance metrics such as resource utilizations, request response times, and credit usage in burstable resources, such as the Amazon EC2 T-family instances.

CCS Concepts

•Computing methodologies → Simulation evaluation; •Software and its engineering → Software performance;

Keywords

Burstable cloud; performance prediction; cloud simulator.

1. INTRODUCTION

Cloud computing success derives from the possibility for organizations to provision computing resources over the Internet and pay only for the ones actually used. The most common pricing models are: (*i*) *on-demand* resources with fixed price and fixed performance; (*ii*) *spot* resources in which the prices fluctuate; (*iii*) *preemtible* resources that do not have a guaranteed availability; and finally (*iv*) *burstable* resources, in which the resource performance changes based on utilization history.

One of the main problems cloud users face is the minimization of the costs for provisioning the computational requirements while still fulfilling their performance needs. Current research efforts are tackling this problem by providing means to predict the performance of a cloud application from a model of the application itself, which may be, for example, in the form of an annotated UML model or a queueing network (QN) model. Most existing works focus on static on-demand resources only [3, 5], while more recent ones also consider additional types of pricing models, such as spot and preemptible pricing models [2, 1]. For what concerns the burstable resources, some works that study their behavior already exist [7], but an usable tool to predict the performance from the application model in this situation is currently missing.

The purpose of this paper is to propose ForeBurst, a performance prediction tool, which supports the analysis of burstable cloud resources. In Section 2 we describe the system model, the prediction problem, and the solution. Section 3 describes the tool and shows some results. Finally, Section 4 concludes the paper.

2. MODEL, PROBLEM, AND SOLUTION

The Model. We consider cloud applications modeled as a queueing network (QN), which is a model that can be automatically derived from other commonly used models such as annotated UML diagrams [5]. In particular, we consider closed multi-class and multi-queue QN models with exponentially distributed services times. Each queue represents a cloud resource running an application component (e.g., a DB server). The *service rate* μ of each queue, defined as the average number of jobs that can be processed per time unit, is assumed scaled proportionally to the speed of the cloud resource in which the application component is deployed.

The Problem. The main problem we want to address is the prediction of performance information, such as – for example – the utilization, the response time distribution, etc. This problem has already been solved by several existing tools (e.g., [4]) under the assumption that μ is constant, which unfortunately is not true for application components deployed on burstable resources. In such resources the μ of every queue can change over time depending on the jobs processed in the past. A real implementation of a burstable cloud resource is the Amazon EC2 T2.micro resource. This resource encodes past history in the form of a number credits k that increases when the resource utilization is under a certain threshold, and decreases when the utilization is over such threshold. Low credits result in a slower resource (lower μ), while high credits result in a faster resource (higher μ).

The Solution. We have solved this problem in the specific case of the T2.micro resources. However, the same method can be used for other types of burstable resources as well. In our approach we had first to measure through experiments how μ varies with respect to the number of credits, thus obtaining a $\mu(k)$ function that is not currently given by the cloud providers. Second, we had to determine the period in which the actual μ is updated as the number of k varies: in the case of the T2.micro resources, updates happen every 5 minutes. Since μ is periodically constant for each queue, we can solve the QN until the next μ is expected to change. To solve the QN we used LINE [4], which, thanks to the fact that it solves the QN using a system of ODEs, it is able to predict the evolution of the QN after a customized period of time. This result can then be used to estimate the utilization and therefore any change in k and μ. At

[*] Daniel J. Dubois has been partially funded by the People Programme (Marie Curie Actions) SPANDO FP7 project (FP7-629982). Giuliano Casale has been supported by the European Unions Horizon 2020 research and innovation programme under grant agreement No. 644869 (DICE). The data generated in this paper are available at: http://dx.doi.org/10.5281/zenodo.49068.

this point we can solve again the QN using the final ODEs state of the previous evaluation as starting state for the new evaluation, but with the new updated service rate. Finally, by iterating this process it is eventually possible to predict the future performance with no limitations.

3. THE TOOL

We have developed ForeBurst as a MATLAB module downloadable from [6]. It uses LINE [4] as its backend QN solver, and expands its functionality with the possibility to mark some queues as burstable. Each burstable queue requires the specification of the initial number of credits, the maximum credits, how credits are accrued or deducted based on previous utilization (amount and periodicity), the service rate function $\mu(k)$, and the utilization discount value L. L is a factor that reduces the utilization measured by the cloud provider with respect to the real utilization. All the parameters are given by the cloud provider, except for $\mu(k)$ and L, which – in the case of Amazon EC2 – had to be measured from experiments in the real system.

Fig. 1 shows some plots of the prediction data produced by ForeBurst in the case of a system of 100 jobs and two queues: a Web server and a DB server, both deployed on Amazon EC2 T2.micro instances with 30 initial credits. The Web server can process 1000 jobs/s at full credits, while the DB server can process 500 jobs/s. By looking at the evolution of the available credits for 7200 seconds, we can see that the DB server depletes its starting credits, while the Web server eventually accrues new credits. The reason is that the DB server loses its credits first because of the high utilization. Once the service rate of the DB server is scaled down, most jobs are stuck in its queue, thus leaving the web server severely underutilized with few jobs to process.

4. CONCLUSIONS

In this paper we have presented ForeBurst as a tool for predicting the performance of cloud applications deployed on burstable cloud resources, whose performance can change based on past utilization. The tool can be downloaded from [6]. We envision that this tool will make possible to study and better understand the performance–cost tradeoff that this type of resources actually offers. In the future we plan to integrate ForeBurst in an optimization framework so that the decision whether to use a certain type of resources or not can be automated.

5. REFERENCES

[1] D. J. Dubois and G. Casale. OptiSpot: minimizing application deployment cost using spot cloud resources. *Cluster Comp.*, 2016.

[2] D. J. Dubois, C. Trubiani, and G. Casale. Autonomic Provisioning and Application Mapping on Spot Cloud Resources. In *IEEE CLOUD '16*, 2016.

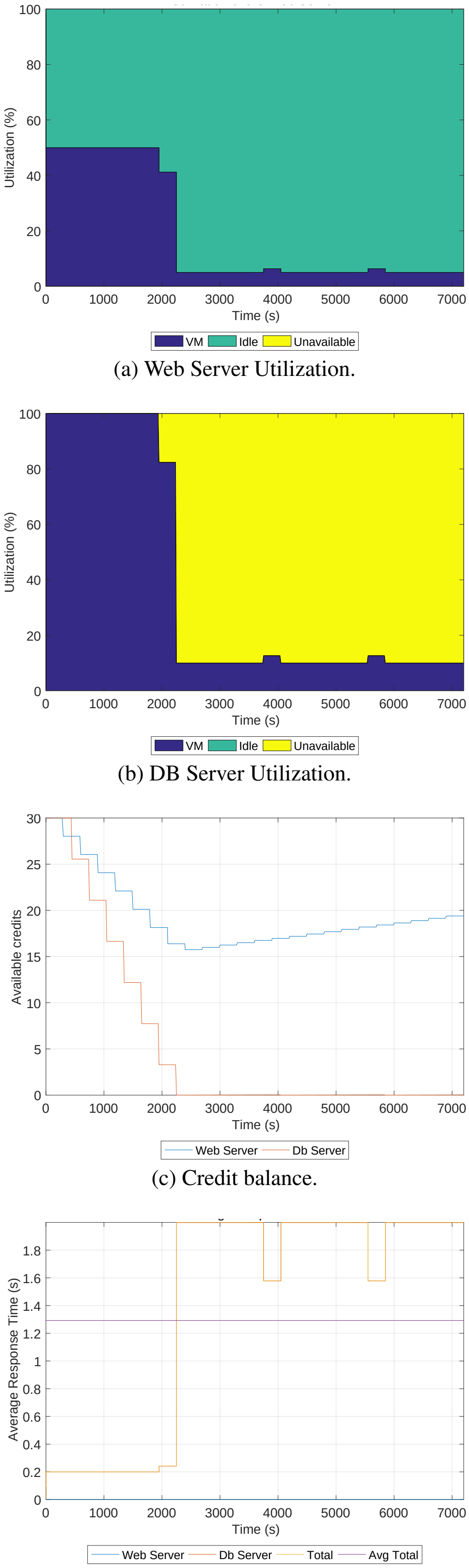
[3] D. Franceschelli, D. Ardagna, M. Ciavotta, and E. Di Nitto. SPACE4CLOUD: A Tool for System Performance and Cost Evaluation of Cloud Systems. In *MultiCloud '13*, pages 27–34, 2013.

[4] Line solver http://line-solver.sourceforge.net.

[5] J. F. Perez and G. Casale. Assessing SLA Compliance from Palladio Component Models. In *MICAS '13*, pages 409–416, 2013.

[6] SPANDO EU Project. Software download: http://www.spando.org/software.

[7] J. Wen, L. Lu, G. Casale, and E. Smirni. Less can be more: Micro-managing vms in amazon ec2. In *IEEE CLOUD '15*, pages 317–324, 2015.



(a) Web Server Utilization.

(b) DB Server Utilization.

(c) Credit balance.

(d) Average Response Time.

Figure 1: Results produced by the ForeBurst tool.