An Overview of Elastic Cloud Applications

Technical University of Vienna,
Advanced Internet Computing Lecture,
(Jan 2014)

Soodeh Farokhi[*]
1228800

Gajo Gajic
0828150

Martin Kalany
0825673

Jia Wei
0035204

ABSTRACT

Elastic resource provisioning is one of the main advantages of using the Cloud paradigm and makes it different to traditional outsourcing solutions. This feature allows users to scale up or down resource allocation for their applications at run-time. However, there are various challenges from the Cloud providers perspective in order to materialize this feature such as predicting the elasticity requirements of a given application. Several approaches have tackled this issue in various ways so far. In this paper, we investigate the state-of-the-art of elastic Cloud applications and describe the requirements for supporting elasticity in the Cloud environment.

1. INTRODUCTION

Nowadays more and more enterprises have been encouraged to migrate their business applications to the Cloud infrastructures in order to benefit from its advantages. Some of these benefits are the ability to dynamically increase or decrease computing power on demand, using a more flexible pay-as-you-go pricing model in order to reduce costs of maintaining a private data-center. These costs are significant especially if a high availability and short response times are required.

Ideally, a Cloud platform is infinitely and instantaneously elastic, meaning that infinite computing resources are available and the scaling up of an application can be done instantaneously. Based on this assumption, an application can be scaled out indefinitely with increasing load without increasing response times [1]. However, supporting this scenario is not easy to be solved by Cloud Infrastructure-as-as-Service (IaaS) providers or Platform-as-as-Service (PaaS) providers. Namely, they face serious challenges in order to understand the elasticity characteristics of applications and workloads while having to consider the required capacity of their Cloud platforms.

In this paper, we introduce the current research and commercial solutions for the elasticity issue in Cloud computing. Since the authors had a working experiment with the CloudScale, a middleware to build elastic applications on top of Cloud IaaS, advantages and disadvantages of utilizing it in comparison with other work, will be explained in this paper. The goal of the CloudScale experiment, which was investigated as a course project, was comparing the process of deployment an application, twitter-based sentiment analysis, on top of an IaaS (Amazon EC2[1]) by using CloudScale and without using it directly on a PaaS (Google AppEngine [2]. It is worth mentioning that, although elasticity is interpreted as the capability of both scaling up and down, while scalability is used more for scaling up, in this paper we use both interchangeable.

The rest of the paper is organized as follows. Section 2 discusses the possible ways to provide elastic applications on Cloud as well as the essential features to support it. Then, in Section 3, which is the main focus of the paper, the state-of-the-art of elastic Cloud applications is presented in two categories, research work and commercial technologies. In Section 4, the features of CloudScale is introduced briefly. Finally, Section 5 concludes the paper.

2. ELASTICITY REQUIREMENTS

In general, IaaS or PaaS automatic elasticity in a Cloud environment is typically achieved by using a set of provider-defined rules that govern how and when the service should scale up or down to adapt to a varying application load [11]. These rules are a set of conditions that when met trigger some actions on the infrastructure or platform in order to support dynamic scaling. Existing approaches differ greatly in the abstraction level of this process, e.g. the customization of rules and the degree of automation.

While some approaches allow the user to specify only simple conditions by using fixed predefined set of metrics such as CPU and memory usage, other approaches offers service level metrics (e.g., cost-to-benefit ratio) in order to allow the user to specify more complex conditions that may further be combinations of simple rules. Furthermore, existing approaches differ in the way they behave when the supported conditions are met. Figure 1 depicts possible mechanisms of elasticity support on the level of Cloud IaaS or PaaS [11].

As represented in Figure 1, IaaS scaling can roughly be divided in two categories, horizontal and vertical. Horizontal scaling is done by either adding new server replicas and

[*] In alphabetical order

[1] http://aws.amazon.com/ec2/

[2] https://developers.google.com/appengine/

load balancers to distribute the load among more servers, or through dynamic bandwidth allocation by supporting network scaling. Vertical scaling can be achieved by changing the instances on-the-fly[3] either by re-sizing (e.g. dedicating more physical resources such as CPU and memory to a running virtual machine (VM)) or replacing this instance . However, on-the-fly changes of resources dedicated to a VM are not supported by the most common operating systems. Some work like [10] tries to facilitate this process by proposing a new abstraction layer closer to the lifecycle of services, which allows for the automatic application deployment and escalation depending on the service status (not only on the infrastructure). Their proposed abstraction layer sits on top of various Cloud providers, hence avoiding a potential lock-in and allowing Clouds federation transparently.

Apart from VM scalability, load balancing is also a major elasticity issue that needs to be investigated. A load balancer is required to distribute load as evenly as possible among VMs. For example, Amazon as a commercial public Cloud IaaS provider, already has some strategies for load balancing by replicating VMs via the Elastic Load Balancing feature[4]. To sum up, having several servers and the mechanisms to distribute load among them is a necessary step towards scaling a Cloud application [10].

However, network scalability is a nearly neglected element that should also be considered [13] for Cloud data-centers in order to be able to support application elasticity. It is because as several VMs share the same network, potentially producing a huge increase in the required bandwidth, the network also has to be scalable.

3. RELATED WORK

In the following, we provide an overview of significant work dealing with elastic Cloud applications, in which both scientific research and commercial products, technologies and tools are covered.

3.1 Scientific Approaches

Keller et. all [6] propose a framework contributes by describing necessary interfaces, features, and data exchanges to deploy complex applications across several Cloud infrastructure services, such as Amazon EC2 in a dynamic and adaptive way . In the other word , the proposed framework in this work uses high-level interfaces for an adaptation plug-in in order to support elastic applications deployment. These interfaces simplify the retrieval of necessary input data for all placement algorithms and support state-full, or complex application architectures.

Work presented in [1] introduces an elasticity mechanism of a typical Cloud IaaS platform (inspired by Amazon EC2) and presents a Service Oriented Performance Modeling method and tool to model and predict the elasticity characteristics of three realistic applications and workloads on this Cloud platform. The authors compare the pay-as-you-go instance costs and end-user response time for these elasticity scenarios. Their proposed model is able to predict the elasticity requirements (in terms of the maximum instance spin-up time) for the working scenarios .

The OSGi[5]-inspired component framework COSCA [5] automatically manages the elastic deployment of component-based applications by isolating components of different applications and hides distribution using a virtualized and distributed OSGi-like framework. It eases the usage of Cloud resources and scalability for component-based applications.

Han et. all [4] adopt a lightweight approach along with its algorithm to enable cost-effective elasticity for Cloud applications. The proposed approach operates fine-grained scaling at the resource level (CPUs, memory, I/O) in addition to VM-level scaling to efficiently scale resources up and down in order to meet the given Quality of Service (QoS) requirements while reducing the cost of Cloud providers.

Orleans [7, 2] is a software framework developed at Microsoft Research to build reliable, scalable and elastic Cloud applications. It includes a programming model that encourages the use of simple, easy to understand and employs concurrency patterns. It is based on distributed actor-like components called grains, which are isolated units of state and computation that communicate through asynchronous messages. *Orleans* enables a developer to concentrate on application logic, while the *Orleans* runtime provides scalability, availability, and reliability.

3.2 Commercial Approaches and Tools

Several application provisioning solutions exist, enabling developers and administrators to declaratively specify deployment requirements and dependencies to support repeatable and managed resource provisioning. Some examples are *Opscode Chef*[6], *Puppet*[7] and *juju*[8]. In juju basic services are described as predefined charms and it can fall into the category in which it supports elastic applications by providing help, hints, or triggers .

Aneka [12] as a .NET-based platform focuses on enabling hybrid Cloud applications, as a composition of two or more Clouds mostly private and public, by employing a specialized programming model. It is able to deploy containers and run user applications on several IaaS providers. Aneka similar to Grid computing middleware, provides a relatively low-level abstraction based on the message passing interface (MPI). In general, Aneka seems more suitable for building scientific computing applications than enterprise applications.

AppScale [3] as an open source extension to the Google AppEngine (GAE) PaaS allows users to build their own GAE compliant PaaS on top of any private or public IaaS service. Namely, it provides a framework to investigate the interaction between PaaS and IaaS systems. To provide elasticity, it scales the VMs used to host containers depending on actual application demand, automatically configuring the load balancers. It targets Online Transaction Processing (OLTP) style enterprise applications.

Carina Environment Manager[9] automates and speeds up the deployment of services onto the OpenNebula IaaS platform. It supports the automated creation and run-time scaling of multi-VM application environments according to some policies, which can be defined in a way to control how VMs are added or removed based on manual, time of day,

[3] without rebooting the machine

[4] http://aws.amazon.com/autoscaling/

[5] Open Services Gateway initiative

[6] www.opscode.com/chef/

[7] http://puppetlabs.com

[8] http://juju.ubuntu.com

[9] https://github.com/blackberry/OpenNebula-Carina

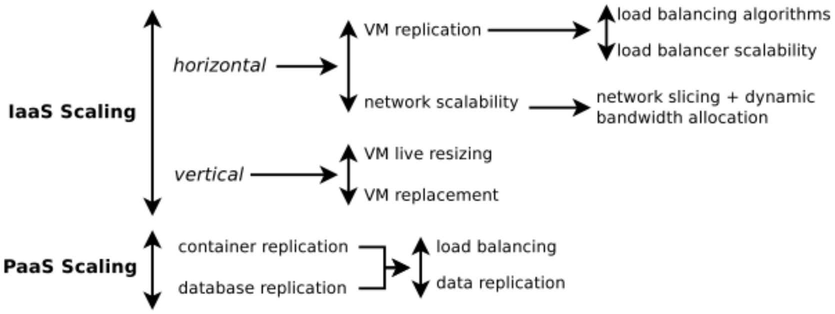

Figure 1: Possible mechanisms to support elasticity on Cloud IaaS and PaaS [11].

or application load-based triggers.

CA AppLogic[10] is another commercial tool to automate complex application deployment. It scales applications without changing code or architecture. It provides on demand scaling by assigning resources to the service as a single entity, rather than a collection of components.

4. CLOUDSCALE FEATURES

In this section, we briefly enumerate the features of CloudScale and compare it with the previous introduced approaches.

CloudScale is used as a middleware to build applications on top of Cloud IaaS by providing an abstraction that makes elastic applications running on top of an IaaS seem like regular, non-distributed Java applications. More precisely, it places a middle layer between IaaS offerings, which provide great control over the application, but with high deployment effort, and PaaS offerings, which are easy to use, but with limited control [9].

In the other word, it allows application developers to have full control over the deployed applications which can not been possible by using a PaaS. Another advantage of CloudScale is in providing an abstract layer so that applications are not bound to any specific Cloud providers. Thus, they are easy to migrate, and work well in private or hybrid Clouds while still providing an abstraction comparable to commercial PaaS solutions.

Compared to similar approaches for elastic application deployment in Clouds, CloudScale endeavors to be a more general tool to support a wide range of elastic application types, such as data-intense, processing-intense and OLTP style web applications, while other work mostly restrict the types of applications that are supported or usage of proprietary APIs [8].

The main difference between CloudScale and the other similar approaches, as mentioned previously, is providing full control over the applications for the developers. For this purpose, although some scalability-related issues are hidden from developers, they are still free to customize how CloudScale works based on their requirements. This is achieved by implementing custom scaling policies or managing so called Cloud-Objects (regular program-level objects that are abstractions of application logics, and should be distributed over a Cloud) in the application manually [9].

5. CONCLUSION

In this paper we presented an overview of approaches in elastic Cloud applications which include both current scientific research as well as commercial tools. We also elaborated necessary requirements to support elasticity in the Cloud environment. Advantages and disadvantages of CloudScale were discussed, based on the experience gained by the authors related to deploying a twitter-based sentiment analysis application with and without using CloudScale in the Amazon EC2 Cloud (IaaS) and Google AppEngine PaaS.

6. REFERENCES

[1] P. C. Brebner. Is your cloud elastic enough?: performance modelling the elasticity of infrastructure as a service (iaas) cloud applications. In *Proceedings of the third joint WOSP/SIPEW international conference on Performance Engineering*, pages 263–266. ACM, 2012.

[2] S. Bykov, A. Geller, G. Kliot, J. R. Larus, R. Pandya, and J. Thelin. Orleans: cloud computing for everyone. In *Proceedings of the 2nd ACM Symposium on Cloud Computing*, page 16. ACM, 2011.

[3] N. Chohan, C. Bunch, S. Pang, C. Krintz, N. Mostafa, S. Soman, and R. Wolski. Appscale design and implementation. 2009.

[4] R. Han, L. Guo, M. M. Ghanem, and Y. Guo. Lightweight resource scaling for cloud applications. In *Cluster, Cloud and Grid Computing (CCGrid), 2012 12th IEEE/ACM International Symposium on*, pages 644–651. IEEE, 2012.

[5] S. Kächele and F. J. Hauck. Component-based scalability for cloud applications. In *Proceedings of the 3rd International Workshop on Cloud Data and Platforms*, pages 19–24. ACM, 2013.

[6] M. Keller, M. Peuster, C. Robbert, and H. Karl. A topology-aware adaptive deployment framework for elastic applications. In *Intelligence in Next Generation Networks (ICIN), 2013 17th International Conference on*, pages 61–69. IEEE, 2013.

[7] J. R. Larus. Look up!: your future is in the cloud. In *Proceedings of the 34th ACM SIGPLAN conference on Programming language design and implementation*, pages 1–2. ACM, 2013.

[8] P. Leitner, Z. Rostyslav, W. Hummer, C. Inzinger, P. Leitner, W. Hummer, and C. Inzinger. CloudScale :

[10] http://www.ca.com/us/Cloud-platform.aspx

Efficiently Implementing Elastic Applications for Infrastructure-as-a-Service Clouds. Technical report, 2013.

[9] P. Leitner, B. Satzger, W. Hummer, C. Inzinger, and S. Dustdar. Cloudscale: a novel middleware for building transparently scaling cloud applications. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, pages 434–440. ACM, 2012.

[10] L. Rodero-Merino, L. M. Vaquero, V. Gil, F. Galán, J. Fontán, R. S. Montero, and I. M. Llorente. From infrastructure delivery to service management in clouds. *Future Generation Computer Systems*, 26(8):1226–1240, 2010.

[11] L. M. Vaquero, L. Rodero-Merino, and R. Buyya. Dynamically scaling applications in the cloud. *ACM SIGCOMM Computer Communication Review*, 41(1):45–52, 2011.

[12] C. Vecchiola, X. Chu, and R. Buyya. Aneka: a software platform for .net-based cloud computing. *High Speed and Large Scale Scientific Computing*, pages 267–295, 2009.

[13] H. Wu and B. Kemme. A unified framework for load distribution and fault-tolerance of application servers. In *Euro-Par 2009 Parallel Processing*, pages 178–190. Springer, 2009.