PERFORMANCE, RESOURCE AND COST AWARE VIRTUAL MACHINE ADAPTATION

Lajanugen Logeswaran

148055X

Thesis submitted in partial fulfillment of the requirements for the degree Master of Science in Computer Science and Engineering

Department of Computer Science & Engineering

University of Moratuwa

Sri Lanka

August 2015

DECLARATION

I declare that this is my own work and this dissertation does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also, I hereby grant to University of Moratuwa the non-exclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Candidate

....

Lajanugen Logeswaran Date

The above candidate has carried out research for the Masters thesis under my supervision.

Supervisors

....

Dr. H. M. N. Dilum Bandara Date

....

Dr. A. S. Perera Date

ABSTRACT

Performance, Resource and Cost aware Virtual Machine Adaptation

Cloud Computing has increasingly become an attractive paradigm for computing during recent years. In the current Infrastructure as a Service (IaaS) cloud landscape users pay for statically configured Virtual Machine sizes irrespective of usage. Although the auto-scaling features offered by current cloud providers enable cloud hosted applications to dynamically scale the amount of resources allocated, the adopted configurations are often sub-optimal owing to the lack of flexibility involved in resoure provisioning. This results in higher costs and difficulty in meeting performance targets for clients.

It would be more favorable for users to consume (and be billed for) just the right amount of resources necessary to satisfy the performance requirement of their applications. Although prior work have suggested a variety of approaches to the auto-scaling problem, the benefits of these approaches remain restricted to applications that mainly depend on CPU and memory. The reason is partly due to cloud operators not providing guarantees on resource types that are difficult to partition such as IO and networking performance in their typical VM offerings (although specialized instances for these types of resources are available).

We take a novel perspective in addressing this problem, where we assume that the cloud operator exposes a small, dynamic fraction (for security and privacy reasons) of its infrastructure and the corresponding resource specifications and constraints to each application. Assuming such a scenario we propose a dynamic VM reconfiguration scheme which comprises an Application Performance Model, a Cost Model and a Reconfiguration algorithm. The performance model helps estimate the performance of an application given specific resources. The Cost model assigns a numerical cost value to resource candidates made available to the application considering the lease expense, reconfiguration penalty and operating income. A reconfiguration algorithm assisted by the cost model makes optimal reconfiguration decisions. Simulation results for the RUBiS and filebench-fileserver applications and the worldcup workload show significant cost savings can be achieved while meeting performance targets compared to rule-based scaling systems.

Our proposed framework has the advantages of being simple, generic and computationally efficient. This framework is also attractive from a cloud operator's perspective as it indirectly assists the operator with the problem of efficient datacenter utilization.

Keywords: Auto-scaling; Cloud Computing; Cost Model; IaaS Cloud;

ACKNOWLEDGEMENTS

First and foremost, I express my sincere gratitude to my advisor Dr. Dilum Bandara. His continuous support, patience, guidance and advice made the successful completion of this research possible. I am thankful to him for regular meetings and productive discussions despite his busy schedule. I highly appreciate his tolerance during times of my slow progress due to health problems or other issues. It has been a wonderful experience working with him, during the course of which I have acquired and improved new knowledge and skills that will be useful to my career in future.

I would like to thank members of my review committee Dr. Srinath Perera and Dr. Chinthana Wimalasuriya for their helpful feedback during my progress reviews.

My sincere thanks goes to the Computer Science Department of the University of Moratuwa for facilitating me with the necessary resources throughout the course of my research. I am thankful to Dr. Dilum Bandara and Prof. Sanath Jayasena for providing me with office space necessary to carry out my research without any hindrance. Thanks are in order to the System Engineers of the Department, especially Mr. Sujith Fernando, for helping me acquire, setup and manage hardware resources. I also thank the department staff in general for their friendly interaction, making my time at the department a fruitful and pleasant one.

I thank the Senate Research Committee of the University of Moratuwa for supporting this research under Senate Research Grant award number SRC/LT/2014/01. I also thank the LK Domain Registry for supporting this research through the Prof. V. K. Samaranayake top-up grant.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Amazon EC2	Amazon Elastic Compute Cloud
CDF	Cumulative Distribution Function
IaaS	Infrastructure as a Service
IOPS	IO operations per second
KCCA	Kernel Canonical Correlation Analysis
MI	Millions of Instructions
MIPS	Millions of Instructions per second
PaaS	Platform as a Service
PDF	Probability Distribution Function
PM	Physical Machine
RL	Reinforcment Learning
RUBiS	Rice University Bidding System
SaaS	Software as a Service
SLA	Service Level Agreement
SVC	Support Vector Clustering
SVR	Support Vector Regression
VM	Virtual Machine
VMWare DRS	VMware Distributed Resource Scheduler

Chapter 1

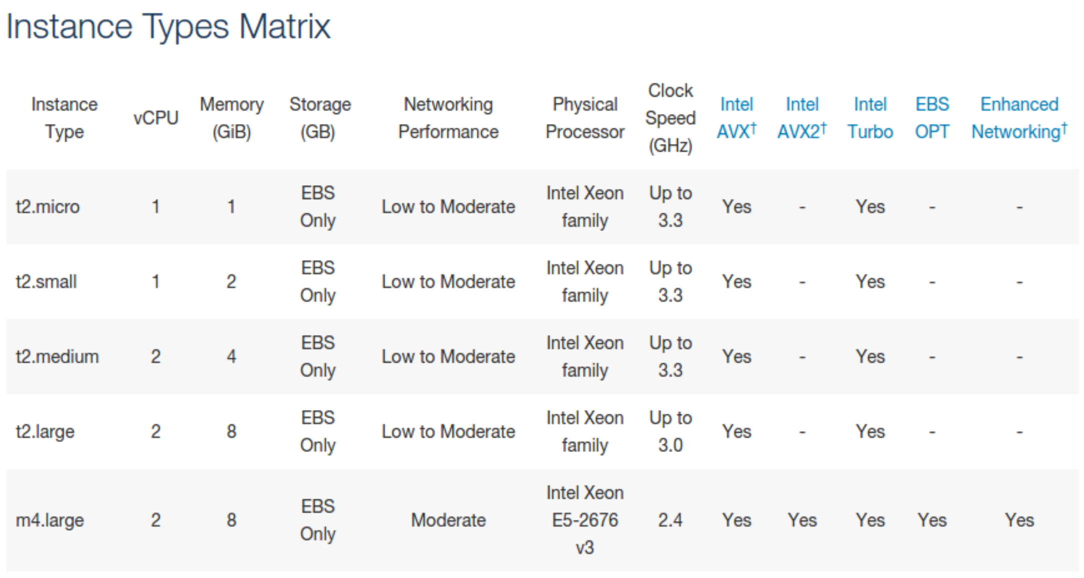
INTRODUCTION

1.1 Cloud Computing

Cloud computing has become an attractive paradigm for computing in the recent years. Cloud computing refers to the applications delivered as services over the Internet and the hardware and systems software in the datacenters that provide those services [1]. This model of computing offers numerous advantages, some of the most attractive ones being no upfront investment, elasticity and the pay-as-you-go model. It saves us the hassle of purchasing and maintaining hardware and software for personal/enterprise use, and instead provides compute capability as a utility.

There are three service models associated with cloud computing - Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). *IaaS* provides computing infrastructure (physical or virtual machines) and other resources such as storage and networking for lease. Major providers of IaaS include Amazon EC2, Windows Azure, Rackspace and Google Compute Engine. *PaaS* provides computing platforms which typically include an operating system, programming environments and tools to build and deploy applications, database, web server, etc. Major players in this market segment include Windows Azure, Heroku, Google App Engine, Apache Stratos, etc. *SaaS* provides access to application softwares often referred to as on-demand softwares. In this case the client does not need to be concerned about installation, setup and running of the application, which are done by the service provider and the user simply uses the software and pays for it. Examples include Google Apps, Microsoft Office 365 and SalesForce.

We concern ourselves with the IaaS service model. As mentioned above, in the IaaS setting users of the cloud may request for compute and other resources,



Instance Types Matrix

Instance Type	vCPU	Memory (GiB)	Storage (GB)	Networking Performance	Physical Processor	Clock Speed (GHz)	Intel AVX†	Intel AVX2†	Intel Turbo	EBS OPT	Enhanced Networking†
t2.micro	1	1	EBS Only	Low to Moderate	Intel Xeon family	Up to 3.3	Yes	-	Yes	-	-
t2.small	1	2	EBS Only	Low to Moderate	Intel Xeon family	Up to 3.3	Yes	-	Yes	-	-
t2.medium	2	4	EBS Only	Low to Moderate	Intel Xeon family	Up to 3.3	Yes	-	Yes	-	-
t2.large	2	8	EBS Only	Low to Moderate	Intel Xeon family	Up to 3.0	Yes	-	Yes	-	-
m4.large	2	8	EBS Only	Moderate	Intel Xeon E5-2676 v3	2.4	Yes	Yes	Yes	Yes	Yes

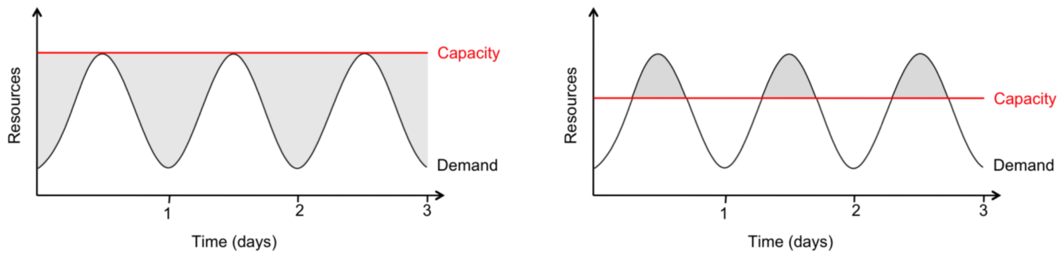
Figure 1.1: A few instance types offered by Amazon EC2.

for which they are billed based on usage. Figure 1.1 shows a few Virtual machine (VM) instance types offered by Amazon EC2 [2].

A cloud computing infrastructure is a complex system with a large number of shared resources and users. The cloud provider is faced with the challenge of accomodating and managing the large number of its user VMs effectively and efficiently on its hardware. The user is faced with such problems as how to determine the performance needs of his application, choose a particular VM size, deal with applications sensitive to networking and IO performance, etc. We discuss these challenges associated with resource provisioning in IaaS clouds next.

1.2 Resource Provisioning in the Cloud

In the current setting, users pay for statically configured VM sizes irrespective of the actual resources consumed by the hosted application. This makes it necessary for clients to choose a particular VM resource configuration in advance, which also requires a good understanding of the kind of workload to expect. As Figure 1.2 shows, underprovisioning leads to performance requirements not being met, and overprovisioning leads to higher operating costs to the user. It is desirable for user VMs to adapt based on actual performance needs. This benefits users because



(a) Provisioning for peak load (b) Underprovisioning

Figure 1.2: Overprovisioning and underprovisioning.

of the ability to meet performance needs at lower costs, as well as the provider because of increased customer willingness to use the cloud.

Although cloud providers do provide scaling mechanisms to cope with workload variations, these schemes are based on rules such as adding/removing VMs based on resource utilization [3]. They provide limited flexibility to the application in adapting different resource configurations over time and often produce suboptimal configurations.

Prior work have suggeted a variety of approaches to the auto-scaling problem beyond rule-based systems [4]. However, the benefits of these approaches remain restricted to applications that mainly depend on CPU and memory. The reason is partly due to cloud operators not providing guarantees on resource types that are difficult to partition such as IO and networking performance in their typical VM offerings.

Another main drawback of statically configured VMs is that applications co-located on the same hardware can have unpredictable impacts on each other [5]. This is typically a problem for types of resources which are hard to partition. As a result cloud providers are unable to provide guarantees for IO and network resource availability to offered VM's. Although some providers do provide specialized network/IO optimized instances with guaranteed discrete levels of service such as low, moderate and high [2]. The availability of these resources can vary over time depending on the applications running on the cloud. This situation would be better if it were possible for users to request and for providers to provide specific quantities of many types of resources.

1.3 Problem Statement

Consider the scenario of a client application deployed in an IaaS (Infrastructure as a Service) cloud. The client is interested in spending less for leasing resources while making sure that the application performance requirements are met. This requires the ability to accurately estimate resource requirements of an application, and a framework to provision the required amount of resources dynamically over time based on the workload.

The cloud being a highly dynamic environment with co-located applications exerting unpredictable impact on each other, a significant challenge involved is the ability to provision required amount of resources along resource dimensions that are hard to partition such as IO and networking performance. Furthermore, any such scheme has to be mutually agreeable between the client as well as the provider without conflicts of interest.

In summary, we attempt to *enable an application deployed in an IaaS cloud to consume just the right amount of resources to fulfill its performance needs while minimizing the cost of resources leased.*

1.4 Contributions

We propose a novel dynamic resource reconfiguration framework for cloud hosted applications. The proposed scheme enables applications to meet performance targets and maximize profits while minimizing the cost of resources leased from the cloud provider.

Prior approaches have looked at the resource scaling problem from the perspectives of the user as well as the provider. User space solutions exist such as scaling schemes driven by application performance models. Rule-based scaling is a provider perspective solution where the provider determines resource requirements and allocates VMs appropriately. We take a hybrid approach in which the user and cloud operator both provide some information regarding the application and the cloud infrastructure respectively. This leads to a resource provisioning scheme that is beneficial to both the user and the provider.

Our specific contributions are as follows:

- We break away from prior approaches which limit themselves to the resource management capabilities of current cloud providers, and consider a scenario where the cloud provider makes available a relatively small pool of machines as choices (VMs/PMs) to each client application hosted by the provider. The main advantages of this scenario are:
  - It provides applications the flexbility to adopt favorable configurations along resource dimensions that are difficult to partition (such as IO and networking performance). This makes it possible for applications to make more accurate scaling decisions and offers room for cost savings.
  - It naturally exploits the co-hosted application interference problem in shared resources to the benefit of the client as well as the cloud vendor. Instead of determining and controlling performance interference due to co-located applications, the applications themselves figure out which of the vacant resource options are suitable to meet performance needs.

- Within this proposed framework, we propose a novel dynamic reconfiguration scheme for cloud hosted applications in an attempt to fill the gaps in prior work discussed in Section 1.1, which is comprised of the following components:
  - *Application Performance Model*

    Predicts the performance of an application given the workload and resources allocated to application.
  - *Cost Model*

    Assigns a numerical value to resource configurations considering performance predictions, monetary costs and reconfiguration penalties.
  - *Reconfiguration Algorithm*

    Makes use of the cost model to make favorable scaling decisions.

- The proposed framework is generic and offers the flexibility to plug in specific choices of models for each of the components. We propose specific choices of models for these components, including a novel analytical cost model.

- We build a discrete event simulator and use it to validate our approach. Simulation results of different scenarios indicate that the proposed approach offers significant cost advantages while meeting application performance requirements compared to scaling schemes currently used by cloud providers.

1.5 Organization of the Thesis

The rest of the thesis is organized as follows. Chapter 2 reviews prior related work. In Chapter 3 we describe the proposed approach. We evaluate our approach in Chapter 4 - Section 4.1 describes the experimental setup, Section 4.2 describes the simulation setup and performance results are discussed in Section 4.3. We conclude and discuss future work in Chapter 5.

Chapter 2

LITERATURE SURVEY

We set out with the goal of devising a resource provisioning and management scheme that is desirable from both the client as well as cloud providers perspective. We reviewed several related problems addressed by previous work, as well as their approaches to the problem considered. Several ideas from these prior work formed the basis of our work. We discuss these ideas under the following major class of problems in an attempt to place our work in the context of previous studies. In Section 2.1 we discuss literature on the *Auto Scaling* problem. Sections 2.2 and 2.3 discuss the *VM Placement* and *Application Placement* problems. In Section 2.4 we discuss prior work on *Application Performance Modeling*. Section 2.5 describes cost models proposed in the literature. Finally, in Section 2.6 we discuss relevant simulation tools.

2.1 Auto Scaling

The *auto-scaling problem* has been widely studied. Auto-scaling refers to adding/removing resources to the operating pool of resources of an application dynamically depending on the workload. These scaling actions can be of two types; *horizontal scaling*: adding or removing a number of VM instances, and *vertical scaling*: adding more resources (CPUs/memory) to existing VMs on the fly, while they are running.

Approaches to auto-scaling dominantly fall under the following techniques: Rule-based systems, Reinforcement Learning, Queueing Theory and Control Theory. We give a brief overview of these techniques and discuss their pros and cons in the following sections. A comprehensive review of these techniques and related bibliography can be found in [4].

2.1.1 Rule Based Systems

Rule based systems are the most intuitive approach to auto-scaling [6, 7, 8]. Scaling decisions are made based on a set of rules for scaling up and down. Rules dictate adding or removing VMs based on a performance metric such as CPU load, request rate or average response time. Scaling actions are controlled by upper and lower thresholds, which correspond to scaling up and down respectively. To prevent frequent reconfigurations and system instability inertia durations are introduced during which scaling actions are forbidden. An example rule could be something like *If CPU utilization exceeds 90% throughout a duration of 100s add a new VM.* Further parameters have also been introduced for more flexibile rule-based systems.

An apparent difficulty associated with these systems is choosing a good set of parameters for a target application. However, usually adopted best practices are useful [9].

Despite all the sophisticated techniques for auto-scaling proposed in the past (which we discuss briefly below), cloud providers dominantly use rule-based autoscaling systems. The main reason being the simplicity and intuitive nature of rule-based systems and the skepticism about these methods due to the unrealistic models they employ and their lack of robustness [10].

2.1.2 Reinforcement Learning

Approaches to the auto-scaling problem based on reinforcement learning have been quite common [11, 12, 13]. A Reinforcement Learning (RL) formulation contains an intelligent agent that automatically learns from an environment. The main elements of the RL framework are states, actions and rewards. The agent interacts with the environment by applying an action and learning from the reward (positive or negative) awarded by the environment. The agent chooses actions based on a *policy*. The objective of the agent is to learn the optimal policy to achieve maximum reward in the long run.