1-1-2012

Pascalyptus: a power-aware scheduler for eucalyptus framework

Soumyasudharsan Srinivasaraghavan
Wayne State University,

PASCALYPTUS: A POWER-AWARE SCHEDULER FOR EUCALYPTUS FRAMEWORK

by

SOUMYASUDHARSAN SRINIVASARAGHAVAN

THESIS

Submitted to the Graduate School

of Wayne State University,

Detroit , Michigan

In partial fulfillment of the requirements

for the degree of

MASTER OF SCIENCE

2012

MAJOR: COMPUTER SCIENCE

Approved by:

_____

Advisor Date

DEDICATION

To My Family and Friends

ACKNOWLEDGMENTS

I am heartily thankful to my advisor, Dr. Weisong Shi, who not only served as my advisor but also encouraged and challenged me throughout my academic program.He and the other faulty members, Dr. Monica Brockmeyer and Dr. Nathan Fisher,guided me through this process, never accepting less than my best efforts. Dr.Weisong Shi and his LAST lab has been working in the field of distributed systems, and has established versatile fruits and international fame. Dr.Brockmeyer and Dr. Fisher have conducted fruitful research on computer systems and realtime systems, respectively. Their expertise and suggestions are undoubtedly invaluable. I thank them all. Last but not least, I am thankful to my current labmates Grace Metri , Guoxing Zhan, Tung Nguyen, Hui Chen, Dajun Lu, and Youhuizi Li for their friendly support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

Cloud computing [1, 2] is a new term for a long-held dream of computing as a utility, which has recently emerged as a commercial reality. It has become an attractive computing platform offering on-demand computing power and storage capacity. The datacenters hosting the cloud are built out of cheap unreliable heterogeneous machines. These machines are under constant strain of large scale computation and are considered mega data centers because they house over tens of thousands of servers consuming tens of mega watts of energy during peak hours adding up to 9.3 million dollars a year. Hence improving energy consumption of data centers through energy efficiency can be extended beyond the actual savings in energy consumption of the servers. It also has added savings through the energy consumption for cooling because cooling equipments can consume between one half and one watt for every watt of node power consumed [3].

1.1 Motivation

Most of today's well known datacenter are heterogeneous in nature. Due to the technological advancement in processor there is always a quench for maintaining the performance by replacing the old servers with latest servers. These datacenters hosts thousands of applications and each consume resources in various proportions. Hence it is necessary to host the application in the right server which consumes less energy with better

performance. This can be achieved only by scheduling the application to host on the server based on their energy efficiency. Past work such as [4, 5, 6] provides black box methods and profiling tools to measure the energy consumption of individual virtual machines running on the server. But there is an ardent necessity for a framework which provides high overall energy efficiency of the system in a heterogeneous environment. Even internet Giants like Facebook launched Open compute project [7] in order to provide energy efficient computing infrastructure by re-modifying its homogeneous hardware servers. Though this provides 38% more energy efficiency, it will be too expensive for a well established datacenter to again remodel and adapt them. Hence software level optimization by specifying the energy characteristic in heterogeneous environment is a promising solution. This work provides a model for an energy aware heterogeneous cloud environment without reducing the performance using EUCALYPTUS [8] framework.

The real beneficiary of a cloud is to utilize all the computing resources of all kinds and we realized that it is not only important to reduce the power consumption but also to reuse the older machines which could be a huge environmental hazard if under-utilized and trashed. Hence we consider an environment where there are machines with different configurations and are a part of cloud. The PASCALYPTUS framework consists of cloud controller, cluster controller, PASCAL scheduler and node controller. Thus making sure that the framework abides the real beneficiary of cloud.

Our contributions are three-folds. First, as far as we know the model we proposed is the first of its kind in a heterogeneous environment. Second, we utilize the EUCALYPTUS framework to extend the flexibility of the system and provide efficient means to conserve energy. Finally, we utilized standard Roundrobin and Greedy schedulers to evaluate the proposed PASCAL scheduler.

1.2 Definition of energy efficiency

Energy efficiency of datacenters is very important topic as their energy consumption keeps doubling every year. Since our goal is to increase the energy efficiency by realizing a method to consume less amount of energy while processing the same load, it is necessary to have consistent definition of energy efficiency.

Hence energy efficiency can be defined quantitatively where energy efficiency of one server can be compared to another

$$Energy\ efficiency = \frac{Load\ of\ the\ application}{Energy\ consumption\ of\ application}$$

Based on the type of application the definition of *Load* can be different. It can be defined as throughput of the application in case of web application. Thus, the energy efficiency becomes the energy consumed in order to fulfill a single web interaction request and response. The lower the ratio of energy efficiency, the greater is the server's energy efficiency.

For granularity the Load can also be defined as the size of data processed in case of data processing application. Thus, energy efficiency becomes the energy required to process a single unit of data. Also, the lower the ratio of energy efficiency, the greater is the server's energy efficiency value.

One of the standard metric for measuring datacenter efficiency is Power Usage Efficiency (PUE). It is defined as the ratio of total amount of power used by a computer data center facility to the power delivered to computing equipment. Another metric, Data Center Infrastructure Efficiency (DCIE), is a performance improvement metric used to calculate the

energy efficiency of a data center. DCIE is the percentage value derived, by dividing information technology equipment power by total facility power. As the power consumption of various applications differ from each other, therefore neither PUE nor DCIE provide efficiency information of individual servers. Also the power consumption depends on the load factor that server is acting upon. Hence PUE and DCIE cannot provide fine grained efficiency of a datacenter hosting cloud.

The remainder of this paper is organized as follows. The chapter 2 provides a basic overview of private cloud architecture followed by the system design of PASCALYPTUS framework and PASCAL scheduler. Chapter 3 presents the experimental setup and methods used. The chapter 4 presents the experimental results and discussions. In Chapter 5, the existing approaches and previous research are briefly discussed. Finally we summarize the paper and describe future work and implications in chapter 6.

Chapter 2

The private cloud architecture and overview

The cloud computing technology has seen a rapid growth due to the growing acceptance of innovative and cutting edge technologies. In this chapter we discuss what cloud computing implies, various types of cloud and discuss the architecture of PASCAL scheduler.

2.1 What is cloud computing?

Decades of research in virtualization, utility computing, distributed computing, and the increase in network bandwidth lead to the constructive development of cloud computing due to which the possibility of reduction in information technology overhead for the end-user, and total cost of ownership is made evident. It basically lead to great flexibility, service oriented architecture on-demand services and many other things.

The computing resources can be customized and managed remotely by various nuances of cloud computing technology. To get started with deploying the application one has to establish an account with cloud providers like Microsoft [9] or Amazon [10] or Google and build their application on it. These applications can be, but certainly are not restricted to being, simplistic. They can be web applications that require only http services [11] with relational database. It can be web service infrastructure and message queues which will interoperate with e-commerce application services. The storage used for the application can be of persistent that might never have to be replicated as they require reliability [12].It might

require the use of custom 3rd party software's and capability to programmatically increase of decrease computing resources on demand using virtualization [13]. Not all of these capabilities are provided by the cloud providers but a good portion of them can be provisioned. There are various type of cloud based on the usage of it. The next section describes each of them in detail.

2.1.1 Public cloud

Public cloud is the traditional way of describing a cloud computing network. In a public cloud, resources are dynamically provisioned by a third party provider to general public on a fine grained, over the internet or through web services/web application. The third party provider charges on the basis of utility of the resources.

Figure 2.1 shows the public cloud architecture. This being the most popular embodiment of the cloud is widely implemented by many businesses and individuals. As it requires a huge capital to set up a public cloud, companies which are well reputable like Microsoft, Amazon and Google can afford it. Being implemented on about thousands of servers which run across hundreds of data centers established in various locations around the world, public clouds allow customers the facility to select a location for their application; thereby reducing latency time when the application is accessed[14].

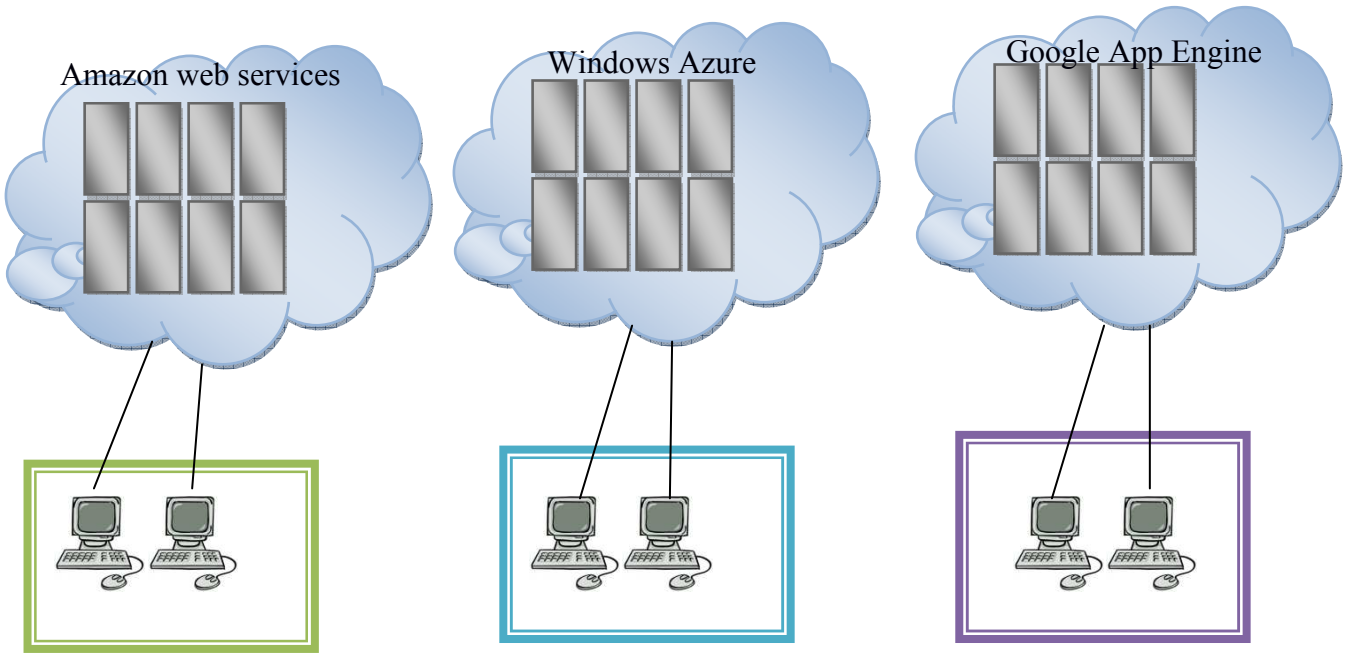


Figure 2.1: Public cloud architecture

2.1.2 Community cloud

Organizations from a specific community, which have common concerns like security, compliance, jurisdiction etc, share their infrastructure and these constitute the community cloud. It is either managed internally or by a third party. It can be hosted internally or externally. Billing for the resource utilization is spread over fewer users when compared to a public cloud, but it is more than a private cloud; hence only few of the benefits of cloud computing are recognized [15, 16]. Such a kind of cloud is established when a set of businesses share similar requirement and same kind of framework; hence making it available to a set of selected organization. Figure 2.2 depicts the basic community cloud

architecture. As an example, let's consider that the federal government decides to setup a government specific community cloud which can control all the states. Therefore, the state government will be free from investing, maintaining and managing their local data centers.

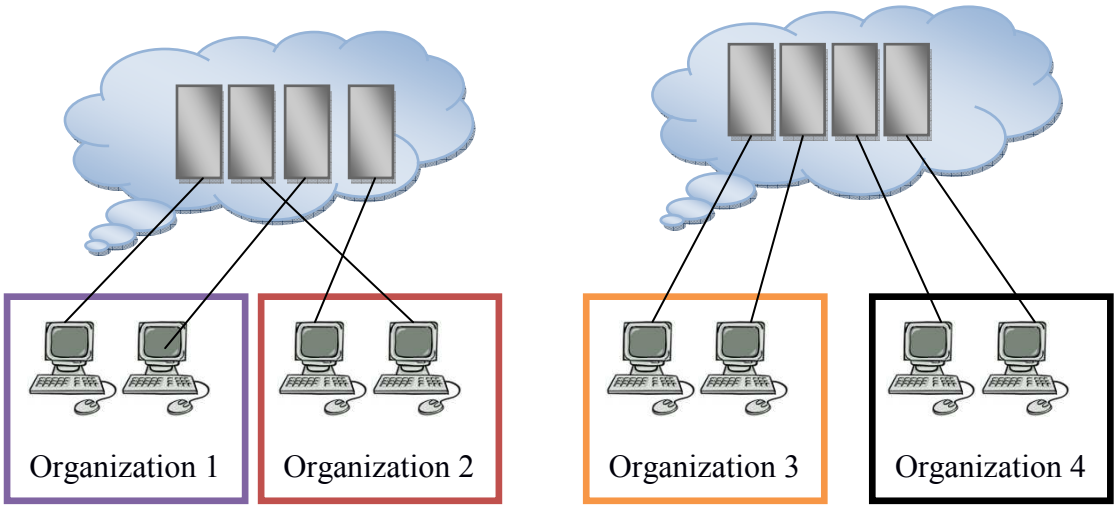


Figure 2.2:Community cloud architecture.

2.1.3 Hybrid cloud

Two or more clouds (private, public or community) are combined to form the hybrid cloud and it offers the benefits of multiple deployment models. In a hybrid cloud, multiple cloud systems are connected in such a way that it allows easy movement of programs and data from one deployment system to another. Figure 2.3 shows the architecture of Hybrid cloud. As it allows easy movement, they offer scalability by moving some of the on-premise and private cloud application to the public cloud. When connecting the private cloud to the public cloud, security plays a vital role [17]. Understanding the importance of hybrid cloud, Amazon

Web Services has launched Virtual Private Cloud (VPC). It actually bridges private cloud and Amazon Web Services(AWS) in a secure manner. Hybrid cloud allows extending the organization's infrastructure beyond its boundary and firewall, in a safer and secure way.

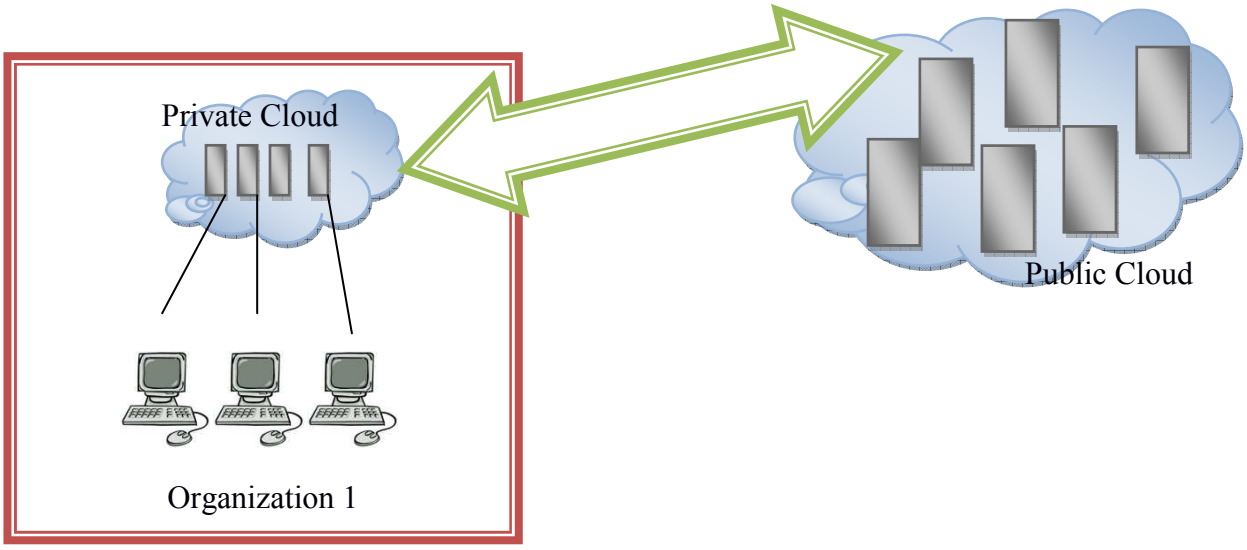


Figure 2.3: Hybrid cloud architecture.

2.1.4 Private cloud

When infrastructure is operated only by a single organization, it is termed as private Cloud. Such cloud can be either managed internally or by a third-party and can be hosted internally or externally. Private clouds can also be defined as normal data centers within an enterprise which follows all the 4 attributes of cloud namely – Elasticity, self Service, pay-By-use and programmability [18]. Enterprises require lesser IT personnel to manage the data center as their IT infrastructure is consolidated by establishing a private cloud. Establishing private cloud reduces power consumption and hence lowers power bills [19]. Employees can be quickly assigned to project teams as provisioning new machines is easy when the

infrastructure is powered by private cloud. In other words, when public cloud is limited to an organizational boundary it is private cloud. Figure 2.4 describes the basic architecture of private cloud. Open source implementations like EUCALYPTUS, OpenNebula and Ubuntu enterprise cloud are some of the most popular private cloud offerings.

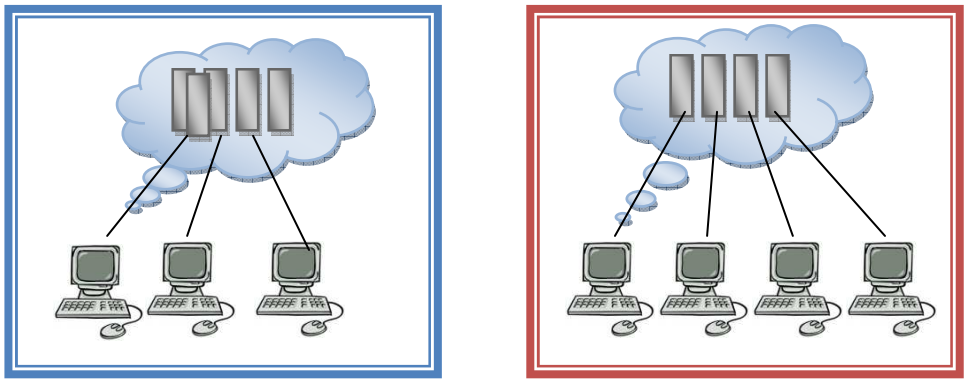


Figure 2.4: Private cloud architecture

2.2 Why now?

With the advent of Web 2.0 the “low-margin and low-commitment” of provisioning service has been transformed from high-margin and high-commitment self service. As an example in Web 1.0, an agreement with a payment processing service such as VeriSign is required to accept credit card payments. This requires a partnership or a large business relationship which makes small business a burdensome to accept credit cards online [20]. Today any individual can accept credit card payments with no contracts and only modest pay-as-you-go transaction fees using PayPal. The DoubleClick(now acquired by Google) are being replaced

by AdSense by which an individual can make revenue from their web page without setting up a relationship with an ad placement company. Also they can use amazon cloudfront to distribute the web content without establishing a relationship with a content distribution network such as Akamai.

Another possible breakthrough is the concept of selling virtual machine cycles on hardware-level. This allows customers to choose their own OS without disrupting each other while sharing the same hardware and reducing the infrastructure investment further [21]. The innovation of virtual machines further enhanced the possibility of much awaited cloud technology. Further the increased network bandwidth which offers seamless transfer of data eased the migration of OS images and data in cloud. This is the major breakthrough which made the old concept of cloud computing a reality now.

2.3 PASCALYPTUS architecture

PASCALYPTUS is built on the EUCALYPTUS framework an open source software tool which can convert a pool of servers into a private cloud. It allows the users to build, deploy, and modify a pool of resources under their own control, and provides the ability to program a specified fraction of resource. It is one of the leading Infrastructure As A Service (IAAS) provider which enables user to dynamically scale up and scale down the infrastructure with respect to the request volume and performance. This framework is likely to have the same impact on software that foundries have had on the chip production [22].

Small clusters, pools of workstations, and various server/desktop machines are inherently accessed by academic research group. Due to the scarce availability of public IP addresses

and security complexity, it would be daunting for allowing complete access from public Internet [23]. Hence system administrators routes the traffic between slave pools and a public network using a “head node”. Usually the cluster is a pool of “Slave” machines on private networks which are interconnected to each other and the head nodes. So with this configuration most machines can initiate connections to external hosts but external hosts cannot connect to machines running within each cluster [24, 25, 26], thus making it more secured using publicly routable addresses. For example, an administrator might configure two Linux clusters each having a single front-end machine with a publicly accessible IP address, a small server pool in which the nodes are connected via private network such that they can inter-communicate to each other and their respective front-ends, and a collection of computer lab workstations having public IP addresses.

The workstations are behind a firewall and cannot be contacted from the outside world. Hence it is clear that many of the machines can only initiate connections to external host and are isolated from outside networks. As their networks are completely private and cannot be routed, the two sets of cluster nodes can even have overlapping IP addresses. The figure 2.5 reflects the hierarchical nature of this configuration in the architecture of EUCALYPTUS framework and makes all of the types of resources as a single cloud. Figure 2.6 shows an example where the hierarchical components are sufficiently general on networks found within many institutions.

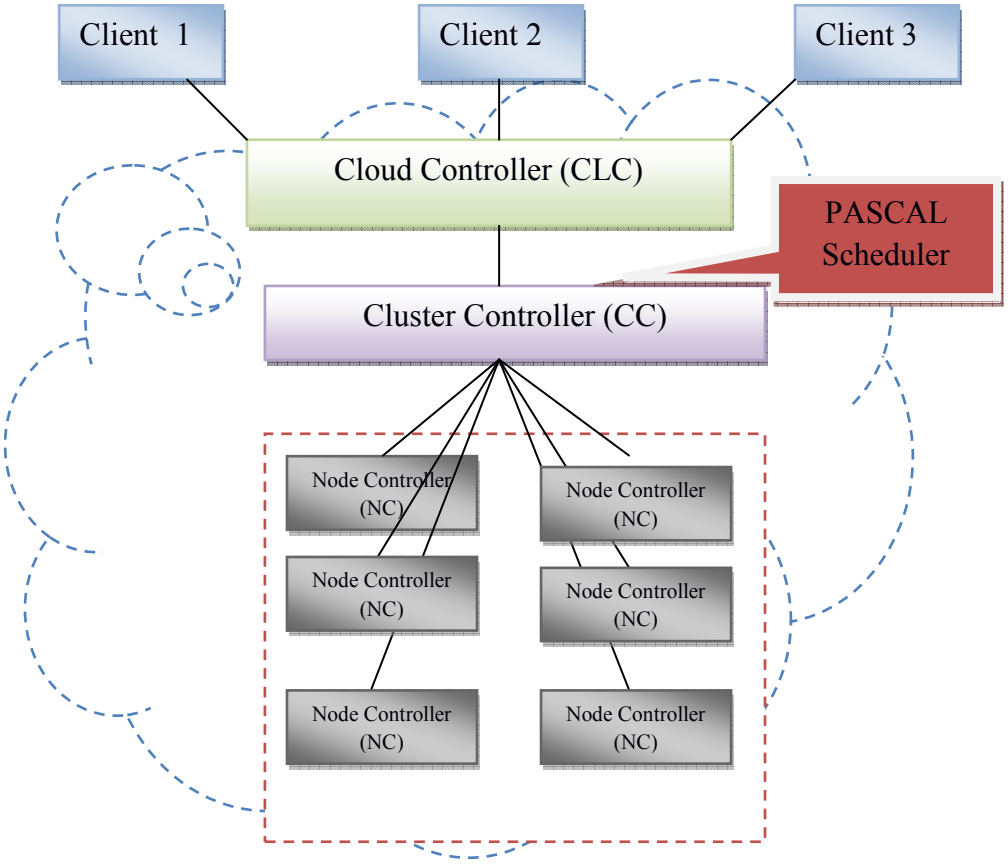


Figure 2.5: Architecture of PASCALYPTUS

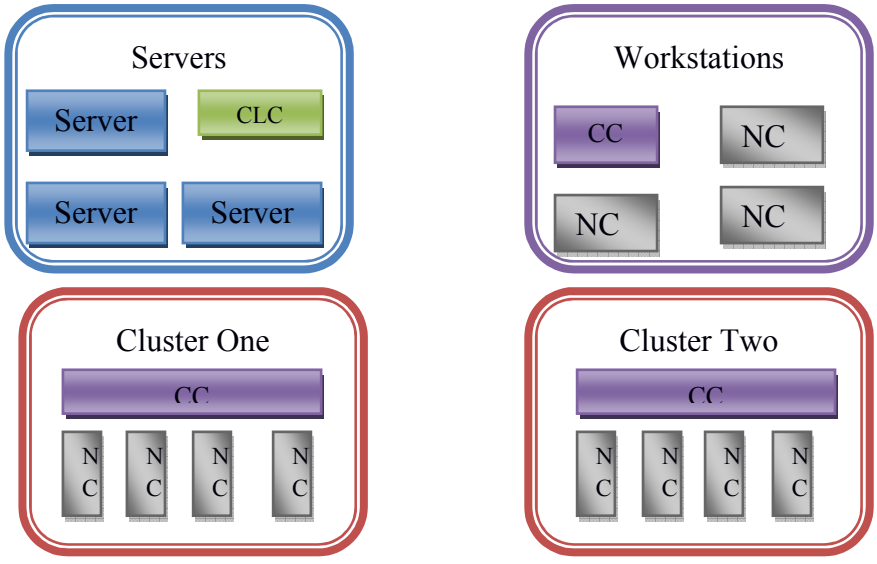


Figure 2.6: Hierarchical view of PASCALYPTUS architecture

2.3.1 Node controller

The Node Controller (NC) is the component responsible to start, inspect, shutdown and cleanup the instances and executes on the physical resources which hosts VM instances. The EUCALYPTUS framework installation can have many NCs. As a single NC can manage multiple VM instances on a single machine , only one NC is required per physical machine. The instance control operation and data structures are defined by WSDL document which describes the NC interface. It supports sustainable operations like describeInstance, runInstance, terminateInstance, startNetwork and describeResource. The system setup, calls to the hypervisor (Xen in the current implementation) and running instance inspection are performed by the run, describe, and terminate operations on an instance.

2.3.2 Cluster controller

A single Cluster Controller (CC) executes on cluster head node and has access to both private and public networks. The Node controller reports their status to these cluster controllers. The status of NC includes state information, VM instance scheduling and managing the configuration of public and private instance networks. Like the interface of NC the CC interface is also described by WSDL and the operation being plural includes runInstances, terminaInstances, describeResources. The CC determines which NC's can be assigned to start an instance by querying each NC through describeResource and schedules to start an instance. The resource which best fits the criteria of the instance are determined first based on the information it gathers from the NC as a heartbeat signal. Based on the describeResource operation the CC returns how many instances of that type can be

emulated simultaneously on the NCs. Most of the schedulers are implemented in CC and PASCAL scheduler is also implemented here.

2.3.3 Cloud controller

The Cloud Controller CLC is the global decision-making component of a EUCALYPTUS framework installation. It is the user-visible entry point to the cloud. It makes high-level instance scheduling decisions. It is responsible for managing persistent and metadata of the users. This component is responsible for processing user or administrative request. As shown in Figure-2.6 the cloud controller consists of several services that handle user requests, authentication and monitoring of VM instances. For instance, when a user initiates to start an instance the CLC authenticate the user using ssh key pairs and handles the high level service of scheduling the instance. The service implementations are separated from message routing and are handled by the Enterprise Service Bus (ESB) [27]. This design makes sure simplicity and transparency to aid further experimentation and extension .Due to the design transparency and simplicity it is easier to include fascinating features in all the domains of cloud. One among the implementation is PASCAL scheduler and the next section describes the details about PASCAL scheduler.

2.4 PASCAL-Scheduler

The EUCALYPTUS framework has three types of schedulers roundrobin, greedy and explicit. The scheduler schedules the instances based on the types and policies. The roundrobin policy schedules the instance based on the way the nodes are registered and in roundrobin fashion. In greedy policy, the instances are scheduled based on the nodes which are immediately available to the cluster controller [28, 29]. This policy follows the greedy fashion of allocating the node. The explicit policy is used only if the user explicitly provides the node IP address to where the instance has to be started. In PASCALYPTUS there is an inclusion of another scheduler which schedules the instance based on the energy efficiency of the node and based on the application it hosts. The Figure 2.7 shows the algorithm of PASCAL scheduler. The algorithm first checks for the application type and based on the application type it pulls the priority table. The priority table consists of the nodes in the prioritized order of energy efficiency. This is done in the schedule_instance function which further scrutinizes the application and is given to the schedule_instance_pascal function. For instance, in case of TPCW which has 3 types of mixes with different types of Emulated Browser's (EB).Therefore, if the application is of type mix 1 and EB of 750 then this function will scrutinize this using the policy configuration and the table is passed on to the schedule_instance_pascal function . The schedule_instance_pascal checks for the availability of the first priority node by comparing against the users requested resource and the available resource in the node. If the available resource in the slave node is more than the requested resource then, the scheduler assigns the node to start the instance. If the criteria are not met then the scheduler looks for the next node in the priority table and continues to check for the availability of the nodes.

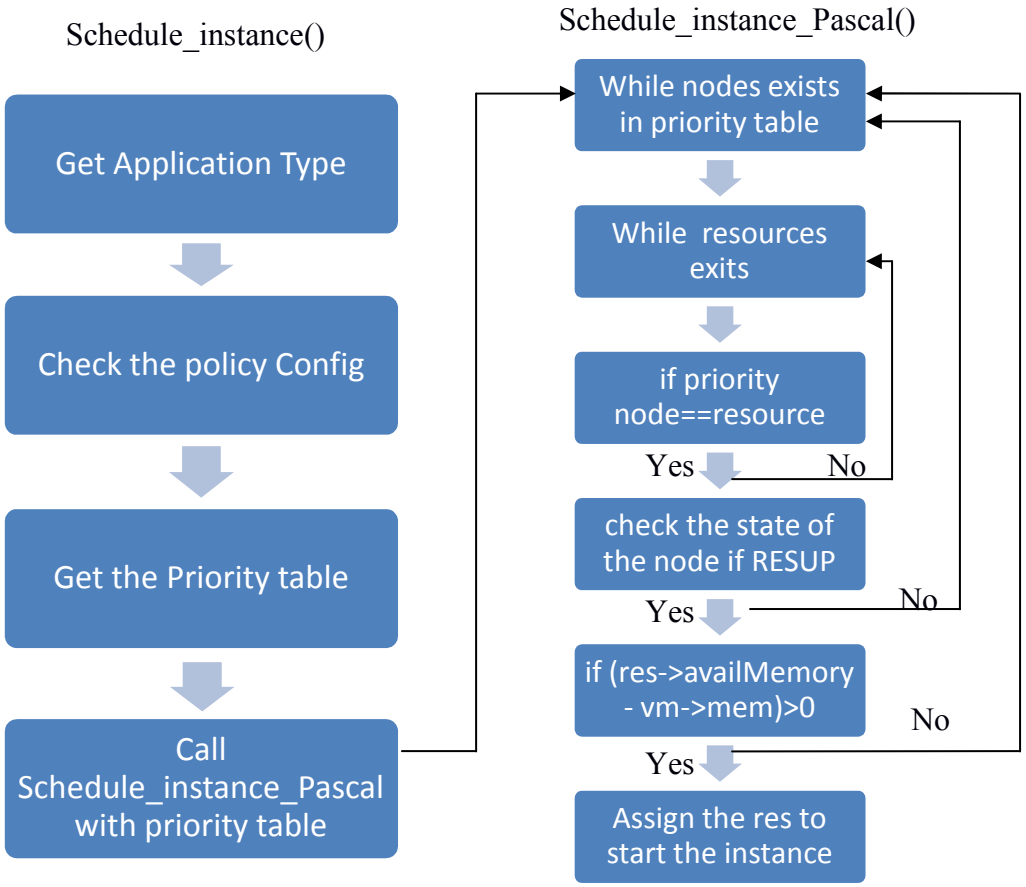


Figure: 2.7: Algorithm of PASCAL scheduler

Chapter 3

Experimental methods

3.1 TPC-W

Our test bed includes an online bookstore TPC-W which serves as a transactional web e-commerce benchmark [30]. We deployed a java TPC-W implementation from the University of Wisconsin-Madison [31] and are based on TPC-W specification 1.0.1. TPC-W ships Remote Browsing Emulator (RBE) by which we generated the workload. We tested with different number of concurrent clients which were controlled by the number of Emulated Browsers (EBs). All our tests included 10,000 items in the database. There are three different mixes in TPC-W: Mix 1 is the browsing mix, Mix 2 is the shopping mix and Mix 3 is the ordering mix and their request composition is displayed in Table 3.1. Mix 1 consists of mainly disk accesses and consists of requests stressing the database server whereas mix 3 has the least stress on database [32]. Four different workloads such as 250, 500, 750, and 1000 EBs were generated with TPC-W for each mix. We then calculated the energy consumed for each test and extracted the throughput which is the total number of web interactions requested and completed successfully.

	Mix 1: Browsing	Mix 2: Shopping	Mix 3: Ordering
Browsing request	95%	80%	50%
Ordering request	5%	20%	50%

Table 3.1: Request Composition of TPCW

3.2 BS Seeker

BS Seeker [33] is a bio-informatics application for mapping bisulfate-treaded reads in genome-level granularity of DNA methylation at single nucleotide resolution and are CPU intensive in nature. We deployed a python BS Seeker implementation from the University of California, Los Angeles [34] in our test bed which takes an input file containing the genome reference and converts into three-letter alphabet. It uses bowtie [35] to align the reads to reference genome. A file of size 100k is provided as input to all of our testing with BS Seeker.

Chapter 4

Experimental results

4.1 Experimental setup

We used EUCALYPTUS [36] framework to create a heterogeneous cloud. We used EUCALYTPTUS 2.0 source version and used Euca2ools of 1.3.1 version which was obtained from http://open.eucalyptus.com/downloads . We used xen 3.0 which are obtained as an in-built package of CentOS 5.5. Table 4.1 shows the nodes in our cloud .All of the servers ran Redhat version of CentOS 5.5 operating system, a linux flavor, which has Xen 2.0 hypervisor. Node 1 was used as our head node. Table 4.2 describes the types of Virtual Machines (VM) within our cloud. We used CentOS 5.5 image from EUCALYTPTUS repository for all of our Virtual Machines (VM). Apache Tomcat [37] is used application server and for database server MYSQL [38] is used. We did all our power consumption measurements using an electronic watt meter manufactured by Electronic Educational Devices Inc, Denver, CO [39]. The model used is Wattsup [40] which uses voltage of 120 VAC, 60 HZ and the max wattage is 1800 Watts and outlet rating of 120 VAC/15 amps.

Machine	Architecture	Cores	RAM	Disk space	Speed per Core
Node 1	64 bit Intel (R) XEON (TM) E5620	16	12GB	855GB	2.4 GHz
Node 2 & 3	64 bit Intel (R) XEON (TM)	1	2GB	28 GB	2.8 GHz

Node 4	64 bit Intel (R) XEON (TM)	1	2GB	97 GB	2.8 GHz
Node 5	64 bit Intel (R) XEON (TM) E5620	16	12GB	855GB	2.4 GHz

Table 4.1: Cloud node types

Virtual Machine	Number of Cores	RAM	Hard Disk
Small	1 core	128 MB	10 GB
Medium	1 core	256 MB	10 GB
Large	1 core	512 MB	10 GB
XLarge	2 cores	2 GB	40 GB
XXLarge	4 cores	4 GB	50 GB

Table 4.2: Types of available instances

The three types of mixes ran on the physical nodes and this is the first experiments we conducted. We ran the experiment for 250,500,750 and 1000 EBs for each type of mix. Figure 4.1 and Figure 4.2 shows the throughput and the energy consumption of each test respectively. Based on our experiments, we observed that the identical nodes had similar throughput and energy consumption values. Node-5 had the highest throughput and the least energy consumption when compared to the other nodes. We also observed that the increase in energy consumption is not proportional to the increase in throughput as the number of EBs increase, the energy consumption increase as well. For instance, when changing the number of EBs from 250 to 750 by running mix 3 on Node-2, there was 16% increase in energy consumption and 135.5% increase in the throughput.

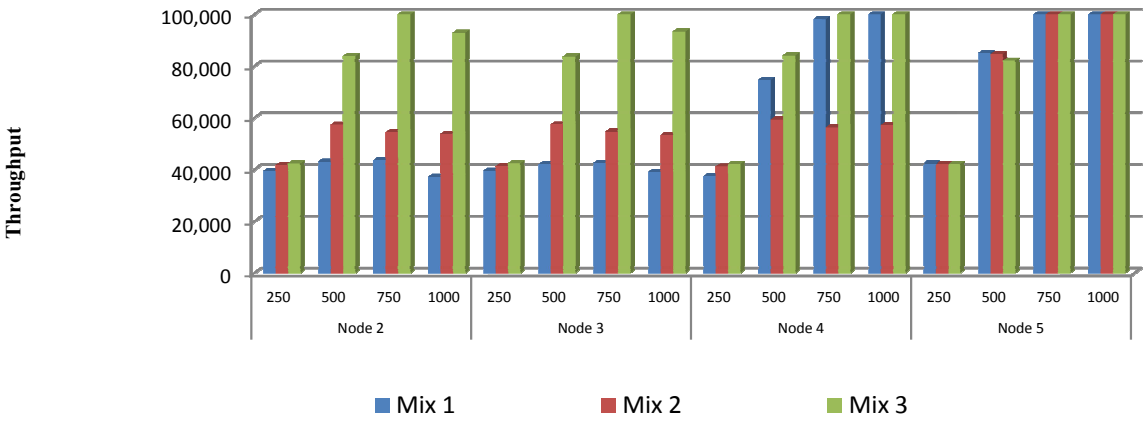


Figure 4.1: Comparing the throughput when running TPC-W on the physical nodes.

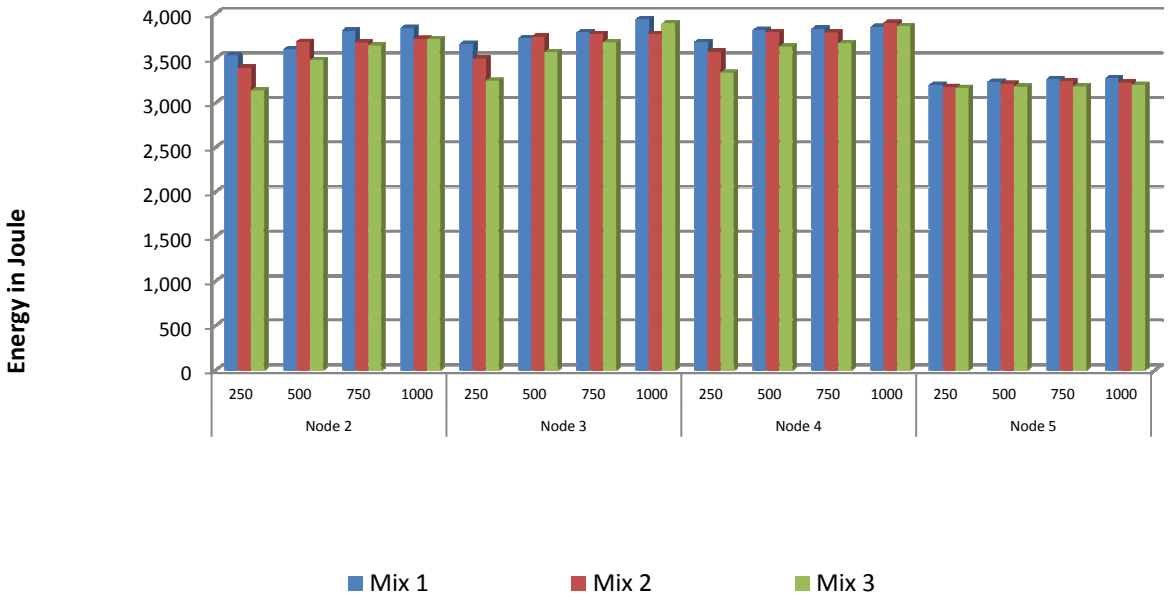


Figure 4.2: Comparing the energy consumption running TPC-W on the physical nodes.

Our second experiment consists of running the three types of mixes for 250,500,750 and 1000 EBs within the emulated VMs on all the nodes. Figure 4.3, Figure 4.4 and Figure 4.5 shows the results of throughput of our experiments for mix1, mix2 and mix3 respectively.

Figure 4.6 and Figure 4.7 shows the energy consumption results of our experiments for mix1 and mix2 respectively. Figure 4.8 shows the energy consumed when running mix3. It has been noted that on all of the experiments we conducted node-5 consumed much less energy and with higher performance which is due to the fact that node 5 has a very high configuration on all of their resources.

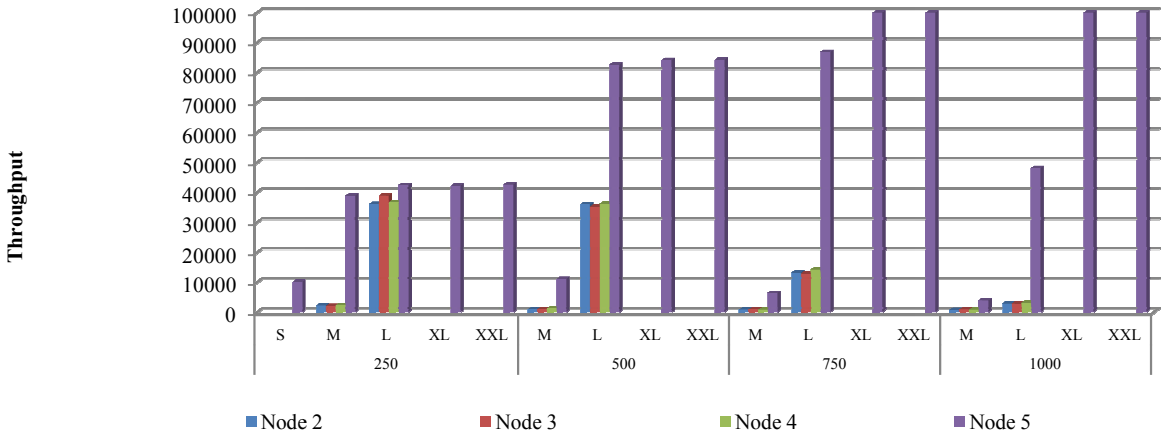


Figure 4.3: Comparing mix 1 throughput for heterogeneous nodes.

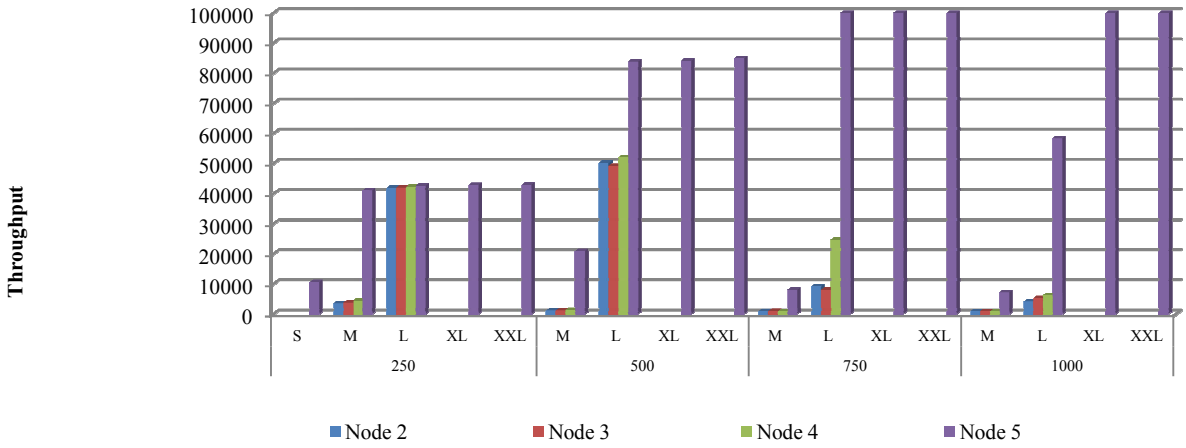


Figure 4.4: Comparing mix 2 throughput for heterogeneous nodes.

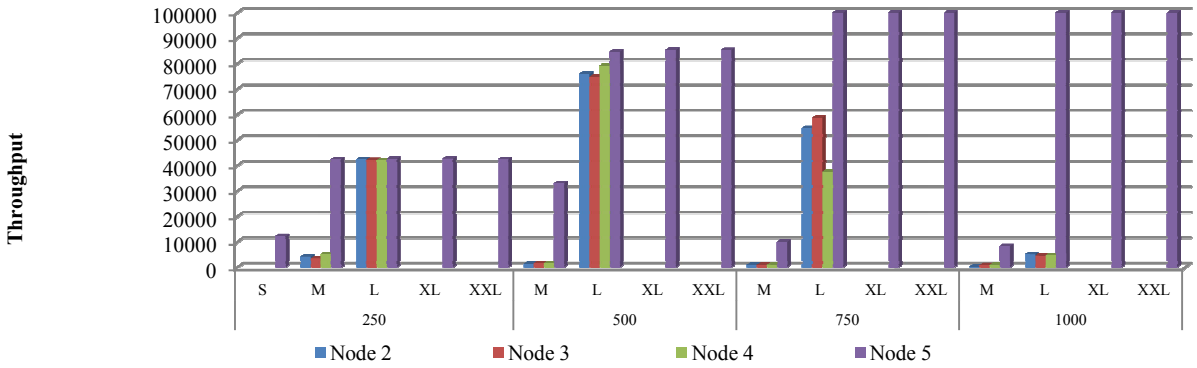


Figure 4.5: Comparing mix 3 throughput for heterogeneous nodes.

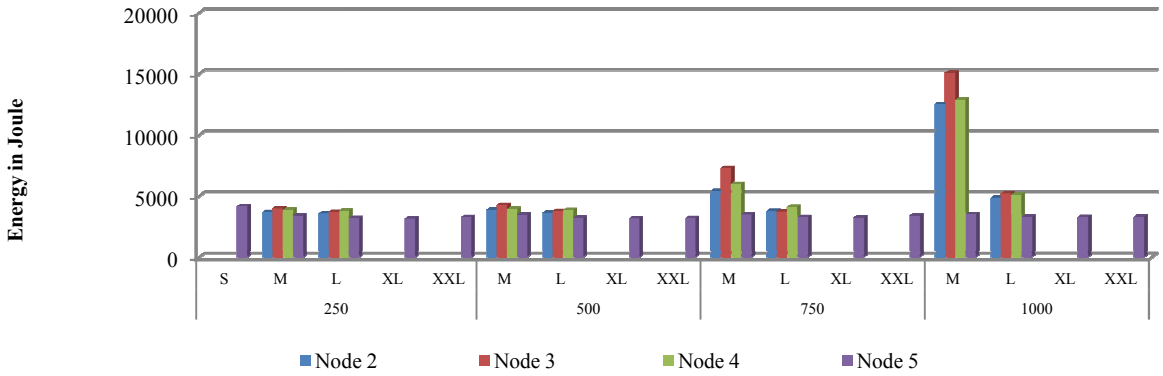


Figure 4.6: Comparing mix 1 energy consumption for heterogeneous nodes.

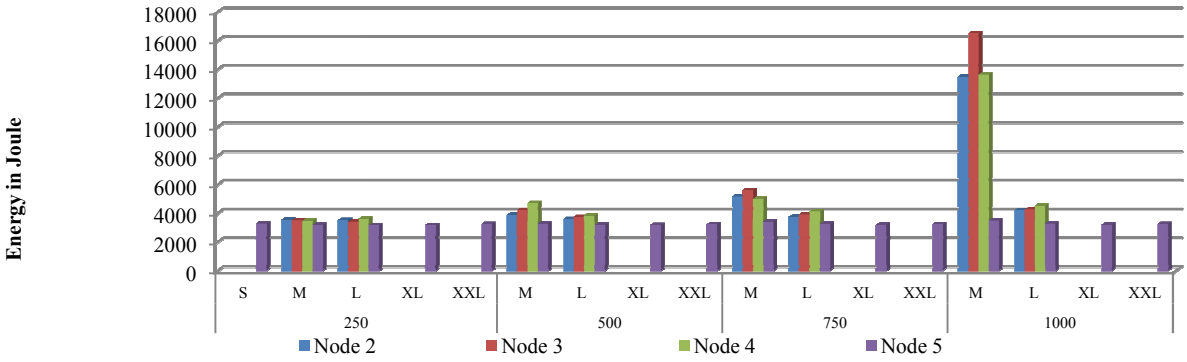


Figure 4.7: Comparing mix 2 energy consumption for heterogeneous node

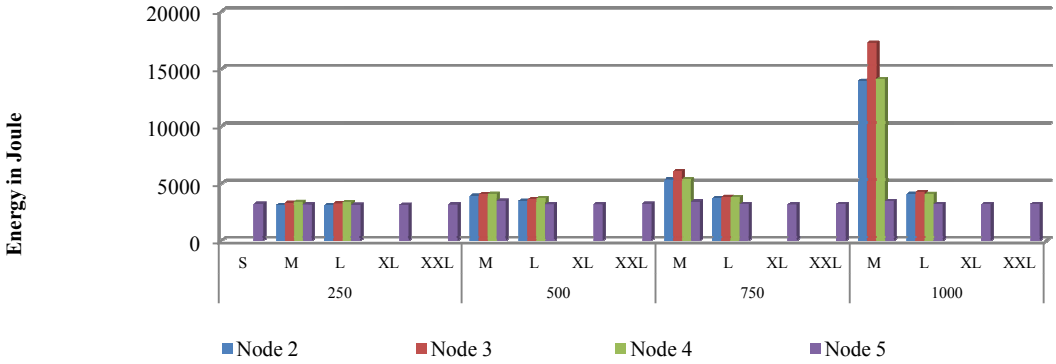


Figure 4.8: Comparing mix 3 energy consumption for heterogeneous nodes.

The third set of experiment which we conducted was running the three types of mixes for 250, 500, 750, and 1000 EBs on VM's using EUCALYTPTUS framework and PASCALYPTUS framework. The energy efficiency of roundrobin and greedy scheduler are compared with PASCAL scheduler. The Figure 4.9, 4.10, 4.11 shows the throughput, energy efficiency and the energy consumption of Mix-2 using roundrobin, greedy and PASCAL scheduler respectively.

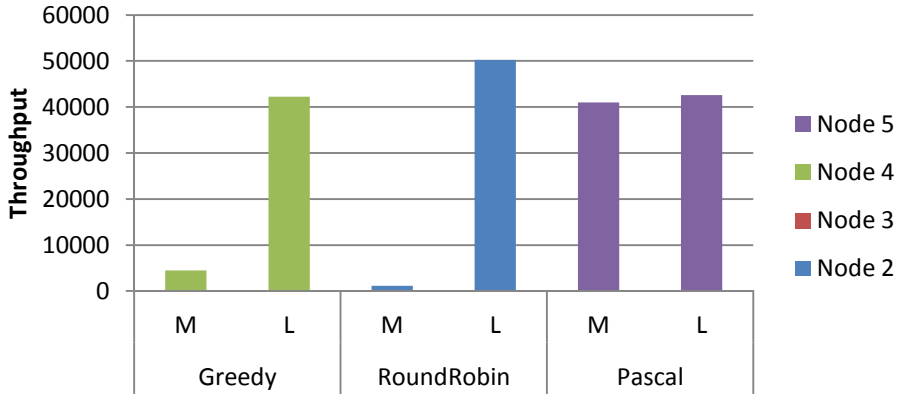
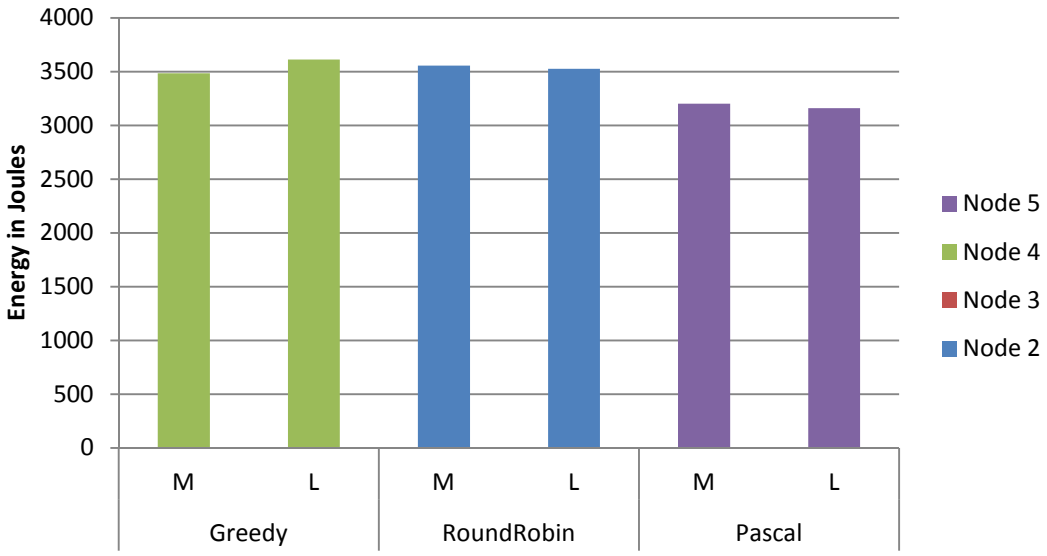


Figure 4.9: Comparison of schedulers based on throughput of mix-2 on priority nodes



. Fig 4.10: Comparison of schedulers on energy consumption of mix-2 on priority nodes

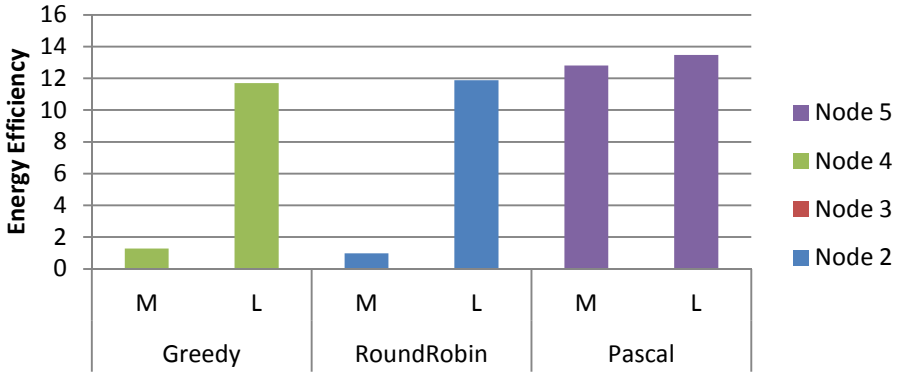


Fig 4.11: Comparison of schedulers on energy efficiency of mix-2 on priority nodes

4.2 TPCW results

As we increased the instance size from small to large, the throughput increased consistently when keeping the node type and EB size constant. Whereas its energy consumption decreased. Since different parts of a computer consume different power rates where memory and network devices consume a negligible amount of power when compared to the CPU and disk accesses [41,42,43], we can attribute our results to the fact that these types of instances have the same number of cores but different memory sizes. The energy consumption decreases and the energy efficiency increases as we increase the memory size and we require fewer accesses to the disk. We also noticed an increased throughput and an improvement in the energy efficiency, when increasing the VM size from L to XL in node -5. However, the energy efficiency of an XLarge VM is better than the energy efficiency of the XXLarge. The XLarge has two cores whereas the XXLarge has four cores. Having four cores would have reduced the latency to approach zero, however, the throughput gets to a point where it levels off because the user think time becomes dominant of the possible request generation rate [44].

We noticed an increase in throughput until the VM reaches its capacity of peak throughput and drops sharply after reaching the peak value, when keeping the node type and instance size constant, and comparing the throughput and energy efficiency based on the EB size. When in case of smaller instances like “small, medium and large”, the VM reaches the highest value of throughput it is the most energy efficient. But when the number of concurrent connections exceeds the peak, the number of throughput sharply drops due to dropped connections [45, 46], and also the energy consumption rises sharply leading to a very low energy efficient [47, 48] VM. We also noticed that when changing the EB size from

750 to 1000 , the throughput increases sharply and energy efficiency decreases when considering larger instances like XLarge and XXLarge.

When the EB size instance and mix type are kept constant, we observed that node-5 is more energy efficient than all the other nodes. The above comparison is made against the energy efficiency to each other. This proves that having the same VM type running on different nodes can have different energy efficiencies even when running the same application with the same input. The EB size, instance size, and mix type are kept constant and the energy efficiency is compared against the different schedulers. We noticed that we obtained more than 80% increase in efficiency with the PASCAL scheduler when compared with roundrobin and greedy scheduler. The Table 4.3 shows the priority of nodes assigned by greedy, roundrobin and PASCAL scheduler for medium and large instance with EB-250 of mix-2. The greedy scheduler schedules the instance to the nodes randomly and roundrobin scheduler schedules instance in roundrobin fashion by which the nodes are registered to the cloud.

Priority	Greedy		RoundRobin		Pascal	
	M	L	M	L	M	L
Priority 1	Node 4	Node 4	Node 2	Node 2	Node 5	Node 5
Priority 2	Node 2	Node 2	Node 3	Node 3	Node 4	Node 3
Priority 3	Node 3	Node 3	Node 4	Node 4	Node 3	Node 2
Priority 4	Node 5	Node 5	Node 5	Node 5	Node 2	Node 4

Table 4.3: Priority table of different schedulers

The Figure 4.12 shows the efficiency attained by PASCAL Scheduler with greedy and roundrobin scheduler for mix2. We observed that PASCAL scheduler outperforms other

schedulers on all the types of mixes. The percentage of efficiency gained by PASCAL scheduler is more on roundrobin scheduler than the greedy scheduler. It also shows the standard deviation of each scheduler based on the instance. Therefore, the hypothesis considering scheduling the VMs based on their energy efficiency in order to reduce the overall energy consumption of a data center and increasing its energy efficiency is a valid hypothesis.

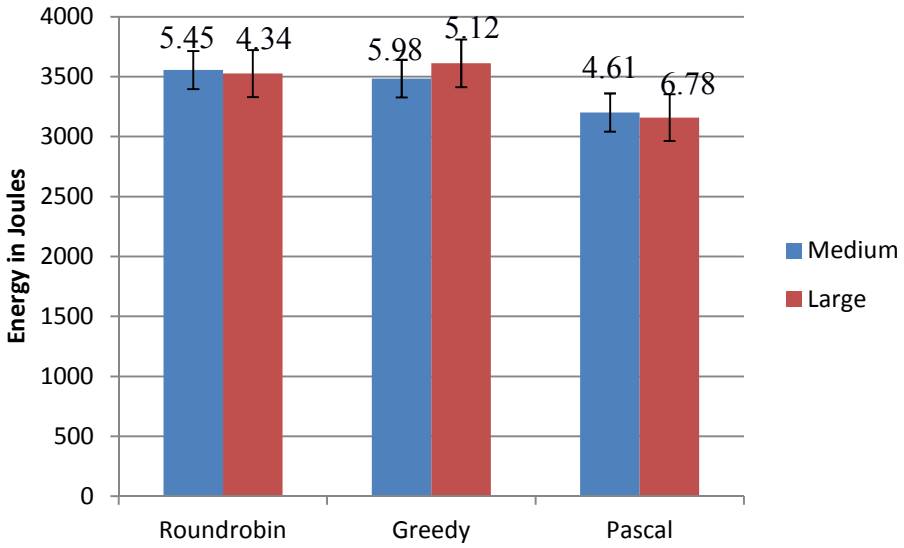


Figure 4.12: Mix 2 energy standard deviation of PASCAL over greedy and roundrobin

The Table 4.4 shows the Energy of mix 2 with various EB's on different nodes. It also describes the energy efficiency on various types of instances.

		Node 2	Node 3	Node 4	Node 5
250	S				0.408656
	M	1.598541	1.866174	1.615985	0.087412
	L	0.098886	0.094867	0.103575	0.075458
	XL				0.075088
	XXL				0.076821
500	M	3.944095	3.930414	2.965026	0.313249
	L	0.101432	0.107	0.10642	0.03888
	XL				0.037801
	XXL				0.037871
750	M	5.887756	7.3399	6.680077	0.542611
	L	0.286287	0.289886	0.289725	0.037696
	XL				0.032445
	XXL				0.033931
1000	M	14.82494	16.09498	14.52868	0.897761
	L	1.665526	1.800069	1.594771	0.068722
	XL				0.032811
	XXL				0.032911

Table 4.4: Energy efficiency for mix 2

Our next sets of experiments determine how the scalability of a node can affect its throughput, power consumption, and energy efficiency. We noticed when increasing from one VM running on a single node to more VM running on the same node, the energy efficiency and throughput decreases slightly as the scalability degree increases until it reaches a scalability threshold where the energy consumption spikes, and throughput and

energy efficiency drops sharply. In addition, the connections which complete successfully will have the added latency of 1 ms for one availability zone and 5.5 ms for two availability zones [49, 50].

We observed a reverse relationship between throughput and energy variation. Figure 4.13 shows the relationship between throughput and energy for 250 EBs of mix1 (similar results were witnessed with combination of mixes and concurrent requests).We reduced the values of throughput by 10% for all of our VM test cases for the purpose of the graph. Based on the graph, there is a spike in energy consumption when the throughput drops which is due to failed requests. However, as the number of throughput remains consistent, so does the energy consumption.

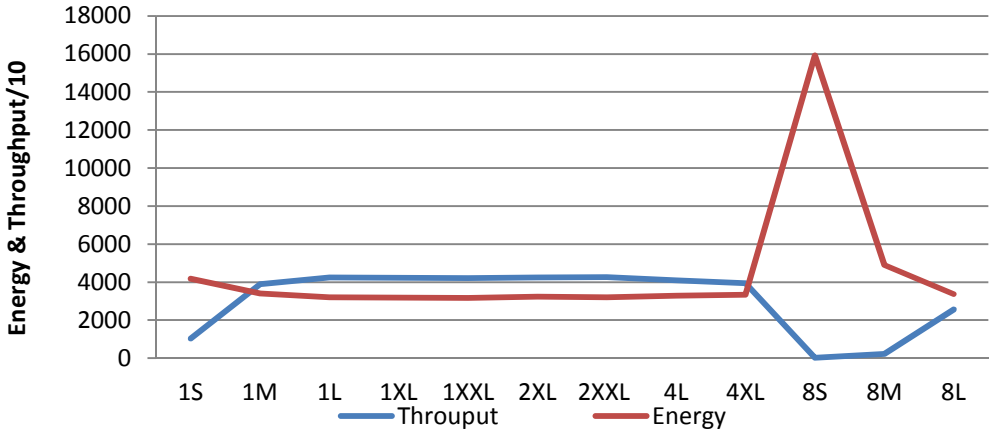


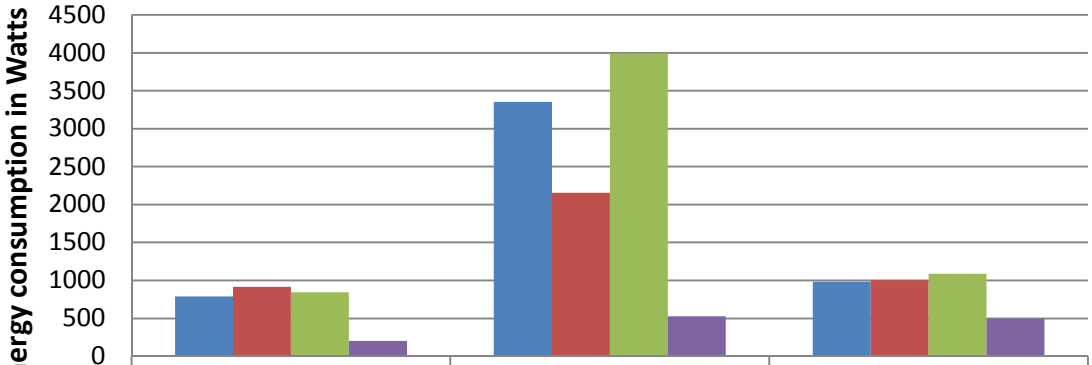
Figure 4.13: Relationship between throughput and energy for mix 1 with 250 EBs

4.3 BS Seeker results

We collected the energy consumption of bare metal when running BS seeker. We then collected the energy consumed when starting the cloud and ran our benchmark on all the

nodes with different instance sizes. The energy consumed by VM's and node type is shown in Figure 4.14. The improvement of BS seeker's energy efficiency is directly proportional to the energy consumption improvement since we used the same file size for all our testing.

Based on our results, node-5 is the most energy efficient whether BS seeker is running on the bare metal or within the same instance type as the other nodes within our cloud. In addition, we observed that energy efficiency varies with different nodes and we also observed that we significantly improve the energy efficiency with more resources. We also noticed that we can improve the energy efficiency between 1% and 7% when running the benchmark on a large instance as opposed to a medium instance. But this is a low improvement when compared to the improvement from small to a medium instance. Figure 4.15 shows the comparison of energy consumption of nodes running BS seeker using different schedulers. We observed that PASCAL scheduler outperforms the other schedulers by more than 80%.



	Bare	Small	Medium
Node 2	790.23	3355.36	981.25
Node 3	915.03	2154.08	1003.12
Node 4	845.03	3990.95	1089.14
Node 5	201.08	527.61	500.16

Figure 4.14: Comparing the energy consumption based on the virtual machine type

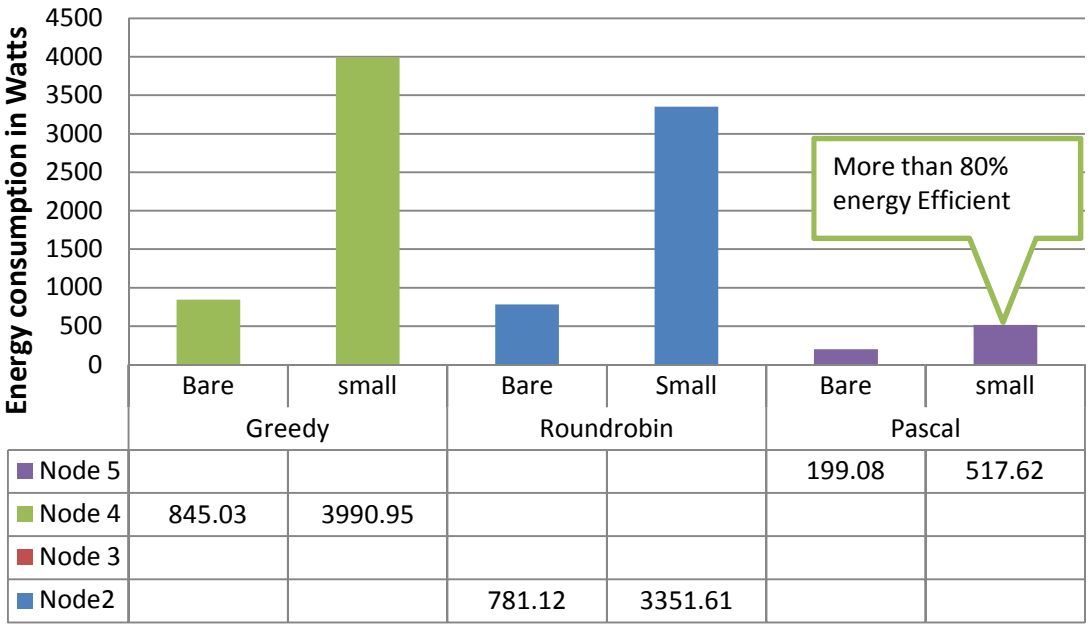


Figure 4.15: Comparing the energy consumption based on the schedulers

Chapter 5

Related work

The need for application-aware power meter for shared data centers is discussed in WattApp [46] where they considered the application parameters like throughput when building their power modeling framework. They found a linear relationship between marginal power and marginal application throughput. But their model differs from ours, since they did not consider the heterogeneity of servers as we did even though they dealt with heterogeneity in application.

Krishnan et.al [47] explores the feasibility and challenges of black box monitoring of the power usage of VM and discusses a VM-level power utilization metering. They experimentally observed that there is a substantial rise in power consumption when increasing the cores. They did not consider the impact of energy efficiency even though the paper deals with modeling the power for VM.

Bellosa et.al [48] provides energy distributed accounting on vertical structured OS with virtual machines and is one of the earlier works in this related field. They provide a frame work for managing energy in multilayered OS and accounts recursive energy consumption spent in virtualization layer of driver components. However they do not consider the fact that different application consumes different level of resource under various constraints. Also they do not consider the need for resource predictability.

Some works performed by Lefevre et.al [49] deals with evaluating energy efficient cloud on a multicore platform. They consider only the impacts of energy consumption during VM migration and consider only evaluation of CPU intensive benchmarks with many cores. Other related works such as done by Lee et.al [50] are based on the fact that the energy consumption scales linearly with the processor. They did not consider the impact of memory associated with it. A typical cloud service provider provides resources which are not only depending on the types of cores but also on various memory types and size needed by the customer. Hence it is evident that they did not consider the performance hit of task even though energy consumption can be reduced to some extent when two or more tasks are consolidated.

Zhao et al [28] studied the inter-relationship between energy consumption, resource utilization, and performances of consolidated workloads are. They only considered the consolidation that allows amortizing the idle power costs more efficiently. However they do not consider the fact that different portions of resources are used by different applications. They primarily focused only on a manageable subspace spanned by CPU and disk resource combinations and did not consider the energy efficiency of an individual machine based on the application.

Chapter 6

Implications

To summarize we ran two different application types within a heterogeneous datacenter and determined the energy efficiency of each node and its available virtual machine types. Based on these values the priority table is fed to the PASCAL scheduler which is implemented based on the energy efficiency of each node and its VM types. We were able to get several implications based on the observation in the previous chapters. The following sections discusses about the implications and future work.

6.1 Resource predictability

Based on our result we observed that the more resources used for a single application, the better its energy efficiency and its performance and are consistent. However, the law of diminishing returns applies where the addition of resources can negatively affect the energy efficiency, once reaching a certain VM size. There should be an equilibrium between the resources available and energy efficiency which is due to the fact that there is a finite number of a physical resources, when building energy efficiency model for scheduling resources. In addition, accurate prediction of resources needed can affect the energy efficiency. We observed the importance of accurate predictability of resource needs. For an e-commerce application our observation is valuable. Having Service Level Agreements (SLAs) which guarantee low number of failed requests is not only in the best interest of the

clients hosting the application in the data center but also for the data center service provider. The operating cost of the provider can suffer spikes which in turn can cost loss to the client's business when there are high request failures. Hence it is essential for algorithms which are independent of application types and which can automatically scale VM dynamically. Also application specific performance metrics can monitor energy spikes in order to determine scaling needs and perform them accordingly.

6.2 Scalability issue

Scalability plays a vital role to realize the concept of mass computing and cannot be separated from the principles of computer systems. For example Internet solved the problem of naming which is more common in building large scale system by IP protocol. It is important to necessitate a mechanism which distinguishes each user and makes sure that the information is delivered to the correct individual. Though this problem has not reached a point of being an urgent obstacle to the current research community, issues related to scalability will become a primary factor to ensure such systems to benefit the society.

6.3 Extensibility of PASCAL scheduler

As the present framework requires the energy consumption of individual server to be measured manually, it would be an interesting implication if the work done by Koller et.al [46] is combined with PASCAL scheduler which automates the measurement of energy consumption of individual server. Based on the application and various version of it, the

priority table can be automated by the PASCAL. This makes sure to get the energy rank of the servers prior to scheduling the instance.

Krishnan et.al [47] provides a method to obtain the energy consumption of each VM instances. More fine grained energy management can be obtained when the PASCALYPTUS framework is coupled with this work. This is an interesting direction which sounds promising.

Chapter 7

Conclusions and future work

7.1 Conclusions

In we developed energy aware scheduler (PASCAL) built on EUCALYTPTUS framework which schedules the allocation of VMs based on their energy efficiency. We performed experimental analysis while running two different application types within a heterogeneous datacenter in order to determine the energy efficiency of each node and its available VM types. We found difference in the efficiencies between applications. We noticed a reverse relationship between throughput and energy variation with web based applications. We compared the PASCAL scheduler with the other standard schedulers like roundrobin and greedy. The experiments show that PASCAL is 80% more efficient than the other schedulers. In the future we plan to attribute the scheduler with more powerful heterogeneous servers. There is an ardent need of scalability of multiple instances on a single powerful machine. Since different applications have different demands it is important to study the magnitude of various system components from service provider point of view.

7.2 Future works

Though the PASCAL scheduler is robust in its kind we could get insight information when extending it and potentially can pave more research opportunities. The following section discusses about our future works.

7.2.1 Improving the granularity of PASCAL scheduler on powerful servers

As the system under consideration has only few powerful machines, there is not much information about the scalability of instance on individual powerful server. It is an interesting direction to extend the PASCAL scheduler to schedule many instances on a single server by consolidating the resources. As the usage of resources is different for different applications, it is necessary to observe the scalability factor under more powerful machines. This is important when considering the data centers where there is not much machines but with powerful cloud.

A model which manages power in a cluster-wide by switching on and off based on clusters overall load to attain a better energy efficiency is discussed in [50]. Such mechanism can be paired with our energy efficiency ranking of the nodes in order to use the most efficient nodes and power off the least efficient ones. Thus optimizing the overall energy efficiency of the data center.

7.2.2 User interface

Due to the flexibility of cloud computing any person can be a potential user of the system. Hence it is an important design consideration that the interface of the system is more user friendly and straightforward and hide the complexity from the end user. Hence we plan to make the interface more effective and simple which can draw more people joining the project.

BIBLIOGRAPHY

[1] R. Buyya, C.S. Yeo and S. Venugopal . Market-oriented Cloud computing : vision ,hype and reality for delivering IT services as computing utilities . In *Proceedings of the 10th IEEE International conference on High Performance Computing and Communications (HPCC-08)*, pages 5-13,doi 10.1109/HPCC.2008.172, Oct 2008.

[2] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. Above the Clouds : A Berkley View of Cloud Computing. *In proceedings of Communications of ACM 2010* Vol 53, pages 23-30, doi:10.1145/1721654.1721672, April 2010.

[3] A. Berl, E. Gelenbe, G. Giuliani, H. Meer, M.Q. Dang and K. Pentikousis. Energy Efficient Cloud Computing. In *The Computer Journal* Vol. 53 , pages 32-39, doi: 10.1093/comjnl/bxp080 , Aug 2010.

[4] Hughes and Ron. The data center of the future-Part 1-Current trends. In *The Data Center Journal '05*, Vol. 32, pages 21-28, doi:10.2313/dcjnl, May 2005.

[5] A. Dunlap, R. Kevin, and P. Rasmussen, K. Neil. The Advantages of Row and Rack-Oriented Cooling Architectures for Data Centers. In *Proceedings of American Power Conversion, TCO '06.* Vol. 12 , pages 45-52,doi:10.1523/APC.2006.12, May 2006.

[6] J. McCune, S. Berger, R. Caceres, T. Jaeger, and R. Sailer. Shamon. A system for distributed mandatory access control. In *Proceedings of ACSA '06*, Vol 2, pages 23-30, doi:10.1109/ACSAC.2006.47, Dec 2006.

[7] V. Mulay. http://opencompute.org/2011/11/17/learning-lessons-at-the-prineville-data-center, Nov 2011.

[8] D. Nurmi , R. Wolski, C. Grzegorczyk, S. Graziano, S. Soman, K. Lamia and D. Zagorodnov. Eucalyptus: an open-source cloud computing Infrastructure. In *The Journal of Physics:conference SciDAC '09,* doi:10.1088/1742-6596/180/1/012051, Dec 2009.

[9] Microsoft windows azure cloud computing. http://www.davidchappell.com/writing/white_papers/introducing_windows_azure_v1-chappell.pdf ,March 2011.

[10] Amazon simple storage service . http://aws.amazon.com/s3/ , March 2011.

[11] L. A. Barroso, J. Dean, and U. Holzle. Web search for a planet: The Google cluster architecture. In *Proceedings of Micro IEEE '03*, vol 23, pages 22-28,doi:10.1109/MM.2003.1196112, Mar 2003.

[12] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels. Dynamo: Amazon's highly available key-value store. In *Proceedings of SOSP '07*, Vol 41, pages 205–220, doi:10.1145/1323293.1294281, Dec 2007.

[13] C. Pautasso, O. Zimmermann, and F. Leymann. Restful web services vs. big web services: Making the right architectural decision. In *Proceeding of WWW '08* ,Vol 52, pages 212-220, doi:10.1145/1367497.1367606, Jan 2008.

[14] A. Messer, I. Greenberg, P. Bernadat, D. Milojicic, D. Chen, T. Giuli, and X. Gu. Towards a distributed platform for resource-constrained devices. In *Proceedings of SOSP '07*, Vol 42 , pages 132-142, doi:10.1143/1324293.1294245, Dec 2007.

[15] P. Padala, K. Shin, X. Zhu, M. Uysal, Z. Wang, A. Merchant, and K. Salem. Adaptive Control of Virtualized Resources in Utility Computing Environments. In *Proceedings of EuroSys '07*, Vol 41, pages 121-129, doi:10.1145/1272996.1273026, June 2007.

[16] D. Skillicorn. Motivating Computational Grids.In *Proceedings of International,IPDPS '02*, pages 247–251 , doi:10.1109/CCGRID.2002.1017168 , March 2002.

[17] H. Lim, S. Babu, J. Chase, and S. Parekh. Automated Control in Cloud Computing: Challenges and Opportunities. In *1st Workshop on Automated Control for Datacenters and Clouds*, pages. 34-44, doi:10.1145/1555271.1555275, June 2009.

[18] J. Chen and L. K. John. Efficient program scheduling for heterogeneous multi-core processors. In *Proceedings of the 46th Annual Design Automation Conference DAC '09*, pages 927–930, doi:10.1145/1629911.1630149,April 2009.

[19] H. Lim, S. Babu, J. Chase, and S. Parekh. Automated Control in Cloud Computing: Challenges and Opportunities. In *1st Workshop on Automated Control for Datacenters and Clouds*, pages 34-44, doi:10.1145/1555271.1555275, June 2009.

[20] Y. Song, Y. Zhang, Y. Sun, W. Shi. Utility analysis for internet-oriented server consolidation in VM-based data centers. In *Proceedings of IEEE international conference on cluster computing (Cluster '09),* pages 1-10, doi:10.1109/CLUSTR.2009.5289190, Oct 2009.

[21] K. Fichera, P. Richard. Power And Cooling Heat Up The Data Center. Forrester Research, Inc. March 8, 2006.

[22] T. Stathopoulos, D. McIntire, W. J. Kaiser. The energy endoscope: Real-time detailed energy accounting for wireless sensor nodes. In *Proceedings of IPSN '08*, pages 383–394, doi: 10.1109/IPSN.2008.36, May 2008.

[23] F. Bellosa. The Benefits of Event-Driven Energy Accounting in Power-Sensitive Systems. In *Proceedings of the 9th workshop on ACM SIGOPS European workshop 2002*, pages 37-42,doi:10.1145/566726.566736, March 2007.

[24] W. Huang, J. Liu, B. Abali and D.K. Pand .A case for high performance computing with virtual machines. In *Proceedings of the 20th annual international conference on Supercomputing,* pages 125–134 doi:10.1145/1183401.1183421, June 2006.

[25] M. Curtis-Maury, F. Blagojevic, C.D. Antonopoulos, and D.S. Nikolopoulos, Prediction-based power-performance adaptation of multithreaded scientific codes. In *Proceedings of IEEE Transactions on Parallel and Distributed Systems*, vol. 19, pages 1396–1410, doi:10.1109/TPDS.2007.70804, Oct 2007.

[26] Y. Wang, K. Ma, and X. Wang. Temperature-constrained power control for chip multiprocessors with online model estimation. In *Proceedings of ACM SIGARCH Computer Architecture,* vol. 37, pages 314–324, doi:10.1145/1555815.1555794 June 2009.

[27] M. Schmidt, B. Hutchison, P. Lambros and R. Phippen . The Enterprise Service Bus: Making service-oriented architecture real . In *Proceedings of IBM Systems Journal* Vol 44 ,pages 781–797, doi:10.1147/sj.444.0781 ,April 2010.

[28] S. Srikantaiah, A. Kansal, F. Zhao.Energy aware consolidation for cloud computing. In *Proceedings of USENIX workshop on power aware computing and systems in conjunction with OSDI, 2008*, pages 1–5, doi:10.1148/s.424.1341 ,April 2008.

[29] J. Torres, D. Carrera, K. Hogan, R. Gavalda, V. Beltran, N. Poggi .Reducing wasted resources to help achieve green data centers. In *Proceedings of 4th workshop on high-performance, power-aware computing (HPPAC '08)*,doi: 10.1109/IPDPS.2008.4536219 , Jun 2008.

[30] D. McCutcheon, C. Orange,K. Bhagyam http://www.tpc.org/tpcw/spec/tpcw_V1.8.pdf, Aug 2005.

[31] T. Bezenek, T. Cain, R. Dick http://pharm.ece.wisc.edu/tpcw.shtml , Sep 2006.

[32] R. Nathuji, K. Schwan. VirtualPower: coordinated power management in virtualized enterprise systems. In *Proceedings of twenty-first ACMSIGOPS symposium on operating systems principles (SOSP '07),* Vol 41, pages 265–278, doi:10.1145/1294261.1294287, Dec 2007.

[33] P.Chen, S.J. Cokus, M.Pellegrini. BS Seeker: precise mapping for bisulfite sequencing. In *Journal of BMC* , vol 11, pages 123-133, doi:10.1186/1471-2105-11-203, April 2010.

[34] P. Chen, S.J. Cokus, M. Pellegrini. http://pellegrini.mcdb.ucla.edu/BS_Seeker.html, June 2010.

[35] J. Rao, C. Xu. CoSL: a coordinated statistical learning approach to measuring the capacity of multi-tier websites. In *Proceedings of IEEE Symp.on Parallel and Distributed Processing* '08, pages 1–12, doi:10.1109/IPDPS.2008.4536232, April 2008.

[36] D. Nurmi et al .The EUCALYTPTUSOpen-Source CloudComputing System. In *Proceedings of IEEE Symp.on Cluster Computing and Grid (CCGRID '09)*, pages 124-131, doi:10.1109/CCGRID.2009.93, June 2009.

[37] Apache Tomcat version 5.5.20. http://tomcat.apache.org/ ,April 2002.

[38] MySQL 5.5. http://dev.mysql.com/doc/refman/5.5/en/index.html, March 2010.

[39] http://moss.csc.ncsu.edu/~mueller/cluster/arc/wattsup/usb/watts-up-meters manual.pdf , June 2010.

[40] Wattsup?/PRO/ES/.Net. http://www.powermeterstore.com/p5674/eed_watts_up_net.php , June 2010.

[41] B. Langmead et al. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. In *The Journal of Genome Biology,* vol 25, pages 423-433,doi:10.1186/gb-2009-10-3-r25 , Mar 09.

[42] J.S. Chase, C. Darrell, K. Anderson, P. N. Thakar and A.M. Vahdat. Managing Energy and Server Resources in Hosting Centers. In *Proceedings of IEEE Symp.on Operating System Principles* , pages 103-116, doi:10.1145/502034.502045, October 2001.

[43] R.Bradshaw, P.Zbiegiel. Experiences with eucalyptus: deploying an open source cloud. In *Proceedings of IEEE Symp.on large installation system administration,(LISA'10), USENIX Association*, pages 1–16, doi:10.1145/202037.202049, March 10.

[44] E. Caron, F. Desprez, D. Loureiro, A. Muresan. Cloud computing resource management through a grid middleware: A case study with diet and eucalyptus. In *Proceedings of IEEE Symp.on Cloud Computing*, pages 151–154, doi: 10.1109/CLOUD.2009.70, Feb 2009.

[45] C. Rusu, A. Ferreira, C. Scordino, A. Watson, R. Melhem andD. Mosse. Energy-Efficient Real-Time Heterogeneous Server Clusters. In *Proceedings of IEEE Symp.on RTAS'06*, pages 418-428,doi: 10.1109/RTAS.2006.16, Jan 2006.

[46] R. Koller, A. Verma, A. Neogi. WattApp: An Application Aware Power Meter for Shared data Centers. In *Proceedings of IEEE Symp. On Autonomic computing (ICAC '10)*, pages 31-40, doi:10.1145/1809049.1809055, Jun 2010.

[47] B. krishnan, J. Hrishikesh and A. Karsten. VM Power Metering:Feasibility and Challenges. In *Proceedings of ACM SIGMETRICS'10* ,vol 38, pages 56-60, doi:10.1145/1925019.1925031,Dec 2010.

[48] J. Stoess ,L. Bellosa. Energy Management for Hypervisor-based Virtual Machines. In *Proceedings of IEEE Symp.on USENIX Annual Technical Conference (ATC '07)*, pages 28-37, doi:10.1110/ATC.2007.85, Jun 2007.

[49] L. Lefevre , K. Acecile ,"Design and evaluating an energy efficient Cloud", Journal of Supercomputing ,Vol 51, pp. 352-373, doi: 10.1007/s11227-010-0414-2, Mar 2010.

[50] Y. Lee, A.Y. Zomaya. Energy efficient utilization of resources in cloud computing systems. In *Journal of Supercomputing'10* ,Vol 51, pages 374-387, doi: 10.1007/s11227-010-0421-3, Mar 2010.

ABSTRACT

PASCALYPTUS: A POWER-AWARE SCHEDULER FOR THE EUCALYPTUS FRAMEWORK

by

SOUMYASUDHARSAN SRINIVASARAGHAVAN

August 2012

Advisor: Dr.Weisong Shi

Major: Computer Science

Degree: Master of Science

With the advent of cloud computing, scalable resource utilization has become the ultimate reality. The energy consumption in a data center hosting cloud yields many serious issues including carbon emissions and system reliability. Therefore energy efficiency plays a crucial role in reducing the energy consumption and operation cost of datacenter hosting millions of application per day. This paper presents a power aware scheduler-PASCAL on EUCALYTPTUS framework to provide an energy efficient heterogeneous cloud environment. In order to demonstrate the effectiveness of the PASCAL we compare it with the other schedulers of EUCALYPTUS framework using two benchmark applications- TPCW and BS-seeker. The experimental result shows that PASCAL is 80% more energy efficient than greedy and roundrobin scheduler respectively when running mix2 of TPCW benchmark. Also PASCAL is 10 times more energy efficient than greedy and roundrobin scheduler when running BS-seeker benchmark.

AUTOBIOGRAPHICAL STATEMENT

SOUMYASUDHARSAN SRINIVASARAGHAVAN

Soumyasudharsan received B.E in Electronics & Communications Engineering degree from Anna University in 2009. He is currently a Master's candidate in Computer Science Department, Wayne State University, Detroit, MI. His research interest includes Cloud computing, Mobile computing and Computer system power profiling and management.