A JOINT RESOURCE ALLOCATION SYSTEM FOR CLOUD COMPUTING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

HÜSEYİN SEÇKİN DİKBAYIR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

SEPTEMBER 2014

Approval of the thesis:

A JOINT RESOURCE ALLOCATION SYSTEM FOR CLOUD COMPUTING

submitted by **HÜSEYİN SEÇKİN DİKBAYIR** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen
Dean, Graduate School of **Natural and Applied Sciences** _______________

Prof. Dr. Gönül Turhan Sayan
Head of Department, **Electrical and Electronics Eng.** _______________

Assoc. Prof. Dr. Cüneyt F. Bazlamaçcı
Supervisor, **Electrical and Electronics Eng., METU** _______________

Examining Committee Members:

Prof. Dr. Semih Bilgen
Electrical and Electronics Engineering Dept., METU _______________

Assoc. Prof. Dr. Cüneyt F. Bazlamaçcı
Electrical and Electronics Engineering Dept., METU _______________

Prof. Dr. Uğur Halıcı
Electrical and Electronics Engineering Dept., METU _______________

Assoc. Prof. Dr. Ece Güran Schmidt
Electrical and Electronics Engineering Dept., METU _______________

Prof. Dr. Ahmet Coşar
Computer Engineering Dept., METU _______________

Date: **11/09/2014**

ABSTRACT

A JOINT RESOURCE ALLOCATION SYSTEM FOR CLOUD COMPUTING

Dikbayır, Hüseyin Seçkin
M.S., Department of Electrical and Electronics Engineering
Supervisor: Assoc. Prof. Dr Cüneyt F. Bazlamaçcı

September 2014, 72 pages

Cloud computing is a new trend in computing, where resources such as servers, storage devices and software applications are provided to customers over the Internet. It is typically based on a pay-per-use model similar to renting a car or taking a taxi in our daily life.

The primary purpose of a cloud system is to utilize available resources effectively to provide an economic benefit to customers. To succeed in this, jobs initiated by consumers are allocated to a set of virtual machines (VM) that run in big datacenters. These VMs, which differ in their features such as number of processors (CPUs), amount of main memory and storage capacity, are created by cloud providers.

Depending on actual demand, some jobs may be rejected due to over-crowding on VMs, which may result in business loss. Effective resource management processes are needed to prevent such losses and to avoid under-utilization or over-utilization of resources.

In this thesis, we propose a joint optimization model that aims to satisfy both cloud consumers and cloud providers, simultaneously. We first analyze the requirements of cloud providers to improve their services and the requirements of cloud consumers to increase their use of cloud services and to protect their rights. Our literature survey includes related work that focuses mainly on improving cloud efficiency. We identify the main parameters in describing cloud providers’ and cloud consumers’ needs and the cloud topology. Afterwards, a novel joint resource optimization model, which combines provider and customers perspectives, formed. The problem is formulated as a simple generalized assignment problem and is solved by employing a suitable heuristic algorithm. All in all, an alternative allocation system for cloud computing is created. Our approach is then evaluated and demonstrated to be able to achieve effective allocations satisfying both cloud providers and cloud consumers’ needs, simultaneously.

Keywords: Cloud computing, resource allocation, generalized assignment problem.

ÖZ

BULUT BİLİŞİM İÇİN BİR BİRLEŞİK KAYNAK ATAMA SİSTEMİ

Dikbayır, Hüseyin Seçkin
Yüksek Lisans, Elektrik Elektronik Mühendisliği Bölümü
Tez Yöneticisi : Doç. Dr. Cüneyt F. Bazlamaçcı

Eylül 2014, 72 sayfa

Bulut bilişim sunucuların, depolama aygıtlarının ve yazılım uygulamalarının internet üzerinden kullanıcılara sağlandığı bilişim alanındaki yeni bir eğilimdir. Genellikle, günlük hayattaki araç kiralama veya taksi kullanma gibi, kullandığın kadar öde modeline dayanır.

Bulut bilişimin birincil amacı kaynakları etkili bir biçimde kullanarak kullanıcılara ekonomik açıdan fayda sağlamaktır. Bunu başarmak için, kullanıcılar tarafından tanımlanan işler büyük veri merkezleri üzerinde çalıştırılan sanal makine kümelerine atanmaktadır. Farklı işlemci sayısına, farklı miktarda hafıza ve farklı depolama kapasitesine göre değişen özellikte olan bu sanal makinalar bulut bilişim sağlayıcıları tarafından oluşturulur.

Gerçek talebe bağlı olarak bazı işler sanal makineler üzerindeki aşırı kalabalıktan dolayı reddedilmektedir ki bu iş kaybı olarak sonuçlanabilir. Etkili kaynak atama süreçleri bu tür kayıpları önlemek ve kaynakların az veya aşırı kullanımını önlemek için gereklidir.

Bu tez çalışmasında, aynı anda hem bulut kullanıcısını hem de bulut servis sağlayıcısını memnun etmeyi amaçlayan ortak bir optimizasyon modeli önerilmiştir. İlk olarak bulut bilişim servis sağlayıcısının servislerini geliştirecek isterler ve bulut bilişim kullanıcısının bulut servislerini kullanmasını arttıracak ve haklarını savunacak isterler çözümlenmiştir. Literatür incelemelerimiz bulut bilişimi iyileştirmeye odaklanan çalışmaları içermektedir. Bulut topolojisinin, bulut servis sağlayıcılarının ve kullanıcılarının ihtiyaçlarını anlatan ana parametreler belirlenmiştir. Daha sonra, servis sağlayıcının ve kullanıcıların bakış açılarını birleştiren yeni bir ortak kaynak atama modeli oluşturulmuştur. Problem basit genel atama problemi şeklinde ifade edilmiş ve uygun sezgisel algoritma çalıştırılarak çözülmüştür. Sonuç olarak, bulut bilişim için alternatif bir atama sistemi oluşturmuştur. Yaklaşımımız daha sonra değerlendirilmiş ve aynı anda hem bulut bilişim servis sağlayıcısının hem de bulut bilişim kullanıcısının isterlerini karşılayabildiği gösterilmiştir.

Anahtar Kelimeler: Bulut bilişim, kaynak ataması, genel atama problemi.

To my family

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor Assoc. Prof. Dr. Cüneyt F. Bazlamaçcı for his guidance, support, encouragement, trust, patience, kindness and valuable contributions throughout the preparation of my thesis.

I would like to thank the other members of my examination committee, Prof. Dr. Semih Bilgen, Prof. Dr. Uğur Halıcı, Assoc. Prof. Dr. Ece Güran Schmidt and Prof. Dr. Ahmet Coşar for their review and evaluation.

I am grateful to my family, friends and colleagues who have given me encouragement and support.

I would like to acknowledge the support of Republic of Turkey Ministry of Customs and Trade for the realization of this thesis.

Finally, I would like to thank all of those who helped me while working on this thesis and throughout my Master of Science education.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

GAP	: Generalized Assignment Problem
IaaS	: Infrastructure as a Service
I/O	: Input/output
OS	: Operating System
PaaS	: Platform as a Service
QoS	: Quality of Service
SaaS	: Software as a Service
SLA	: Service Level Agreement

CHAPTER 1

INTRODUCTION

Amount of information and number of users has dramatically increased in Internet in recent decades. Cloud computing is a new technology trend, which can provide a diversity of services to consumers over the Internet and other networks. In cloud computing, consumers can hire information technology (IT) utilities such as infrastructure and software applications like databases for a limited time usage. According to consumers' needs, cloud architectures are designed in the form of multiple services as Infrastructure as a Service (IaaS), Platform as a Service (PaaS) or Software as a Service (SaaS).

Cloud computing aims to utilize resources more effectively in order to provide a sizeable economic benefit to consumers. In a cloud system, the provider uses virtual machines (VMs) in order to improve the utilization of the servers. Virtual machine pools are generated in datacenters, differing for example in the number of processors, main memory sizes and storage capacities. With the concept of virtualization, a provider serves more users. However, some jobs may be rejected due to over-crowding of the virtual machines which may result in a business loss. To prevent such business losses, cloud providers need effective resource management processes.

In cloud computing, cloud consumers directly affect a provider's business value. Before joining a cloud system, a consumer signs a service level agreement, which ensures the service quality of associated cloud. This agreement protects consumers against providers that might abuse the consumer by lowering the system quality.

Due to high demand and system failures, sometimes consumers cannot use their cloud services effectively. Moreover, some consumers cannot get the quality of service described on their service level agreements. From customer point of view also an efficient resource allocation model is needed to satisfy their requirements.

In the present work, we propose a joint optimization model that provides satisfaction both for consumers and providers in a cloud, simultaneously. In creating such a model, in addition to consumer parameters that might affect cloud usage, provider parameters that might increase resource utilization while decreasing cost of services are identified first. The model is in the form of a generalized assignment problem and hence easily solved by the auction algorithm. A simple assignment manager, which effectively assigns clients to suitable servers, is created by using this joint model and the auction algorithm.

1.1 PROBLEM DEFINITION

There exists a set of consumers each with a given level of demand for resources and a set of servers each with a given level of capacity where each customer must be served by at least one of the servers. The server and consumer parameters are to be merged suitably to satisfy both provider and consumer needs. A single data model is to be constructed using the merged parameters. Our aim is to minimize the total cost while satisfying consumer and providers need in a soft manner (no hard guarantee) and to achieve this within a very quick response time period.

1.2 MOTIVATION

Within the scope of this thesis, we first carried out an extensive analysis on cloud providers' requirements and cloud consumers' needs. We eliminated some of the parameters while researching for the most effective ones. Our joint model is formulated in the form of a generalized assignment problem. Well known approaches are applicable for solving this problem. Some examples are LP-relaxation, Lagrangian relaxation, Surrogate relaxation and Lagrangian decomposition. Techniques such as subgradient optimization, multiplier adjustment and dual ascent, which can be applied to a wide range of problems, are potentially applicable. However, these techniques are not guaranteed to reach to global optimum in a short time period in all cases. We therefore resort to heuristic solutions and choose the auction algorithm for solving our model because of its advantage in reaching a good sub-optimal solution in a short period of time.

1.3 CONTRIBUTION

In our study, attributes associated with the provider side VMs include number of processors (CPUs), main memory size and storage capacity. Attributes associated with the consumer side include priority, processing capacity and location.

We merged server and consumer attributes in a single model and analyzed the formulation. To make an effective resource allocation, a piece of software that assigns clients to suitable servers using auction algorithm is developed.

The present study demonstrates that our model may be used if both consumers’ and providers’ needs are required to be satisfied simultaneously without employing any hard constraint.

In 98% of our tests, we observed that our joint model chose computers that consume less power compared to other models. We also observed that, in 90% of our tests, depending on the scaling factor, our proposed model achieved a higher load imbalance level ratio compared to other models. Besides these advantages, our model is compatible in its average bandwidth utilization when compared to other models found in the literature.

1.4 THESIS ORGANIZATION

The rest of the thesis is organized as follows. Chapter 2 covers general background information about cloud computing. Chapter 3 presents the related works on cloud computing exiting in the literature. In Chapter 4, our proposed joint optimization model and the developed assignment manager is given. Chapter 5 presents an evaluation of the method in comparison with existing models in the literature. Chapter 6 summarizes and concludes the present work also pointing out some future directions.

CHAPTER 2

BACKGROUND

2.1 CLOUD COMPUTING

Cloud computing is used to describe the delivery of hardware or software resources over the Internet. The term ‘cloud’ is used to denote the set of hardware, software, network, storage and services, which all combine to deliver aspects of computing as services.

The National Institute of Standards and Technology (NIST) characterizes cloud computing as “… a pay-per use model for enabling available, convenient, on-demand network access to a shared pool of configurable computing resources (e.g. networks, servers, storage, applications, services) that can be rapidly provisioned and released with minimal management effort or service provider interaction” [1]. This definition covers cloud security, cloud architecture and cloud deployment strategies. This definition particularly articulates the following five essential elements of cloud computing [26]:

On-demand Self-service: Cloud computing is on-demand and self-service. Consumers who need computing resources (such as powerful CPU, high valued memory, network, storage, etc.) can request them at run time and can use them whenever they need.

Broad Network Access: In a cloud system, all resources are delivered over the network or the Internet. These services are used by different platforms such as mobiles phones, laptops and personal digital assistants (PDAs) [26].

Resource Pooling: In cloud computing, to increase server utilization, virtualization is used. The aim of virtualization is to multiply one computer to serve more consumers. In this technology, a provider's computing resources are pooled together and these pools are made available to any subscribing user.

Rapid Elasticity: Cloud users' computing resources are generally immediate and short-term. Users may need to have more resources at any time and with such elasticity they can use their resources after scaling up and release them when finished. With such elasticity, all resources seems to be infinite for consumers.

The main benefit is that if there is a peak requirement at any time it could be handled without any capital expenditure.

Metered Billing: Cloud has a metered billing model, i.e., consumers pay only for what they use. However resources are pooled and shared by multiple consumers and the cloud infrastructure enables this model for each individual consumer [26].

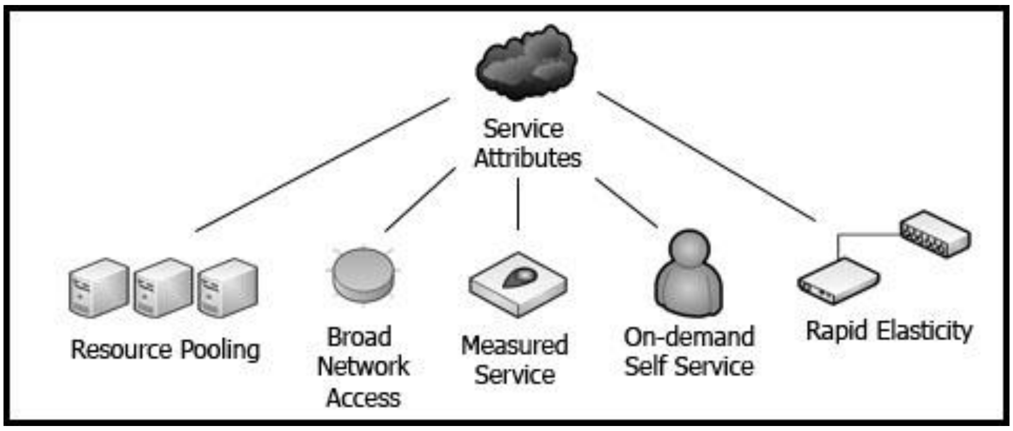


Figure 2-1 Cloud Computing Service Attributes

When looking at these features cloud computing seems to be the latest step in the evolution of IT [1]. In the history of IT evolution we following main steps are observed:
Between 1960s – 1980s since main frame computing was costly one user's idle time was used by other users as a service. Then automatic time-sharing was generated. In 1990s, this was followed by the client-server model in which tasks are distributed among client systems initiating requests and server systems responding over a computer network. In 2000s utility computing (Grids and Software as a Service models) became popular in which large arrays of hardware products are utilized for big computational tasks. Complex applications were accessed over the Internet using web browsers. After 2005 utility model is reborn. On-demand model as a service accessible via a browser over an Internet connection is made available by huge data centers [2]. Then this led to cloud computing which seems today as a model for the intersection of distributed computing, hardware utilization, internet technologies and system management.

2.1.1 CLOUD COMPUTING SERVICE MODELS

Cloud consumers get services from cloud providers according to three fundamental models; infrastructure as a service (IaaS), platform as a service (PaaS) and software as a service (SaaS).

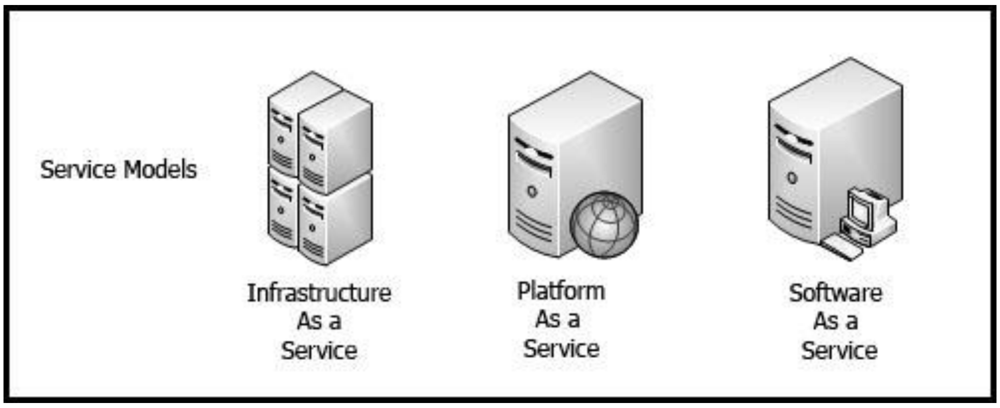


Figure 2-2 Cloud Computing Fundamental Service Models

In addition to these models some providers offer services using the following models also: database as a service (DaaS), security as a service (SECaaS), network as a service (NaaS), unified communications as a service (UCaaS), everything as a service (XaaS), which includes strategy as a service (STaaS), collaboration as a-service (COaaS) and business process as a service (BPaaS).

When looking at the scope of these models, IaaS is the most basic one. This model is more suitable for network architects whose main task is the management of infrastructure. PaaS is the upper level of IaaS, which is more suitable for application developers. The last one is SaaS and is suitable for end users who care mostly in using the service.

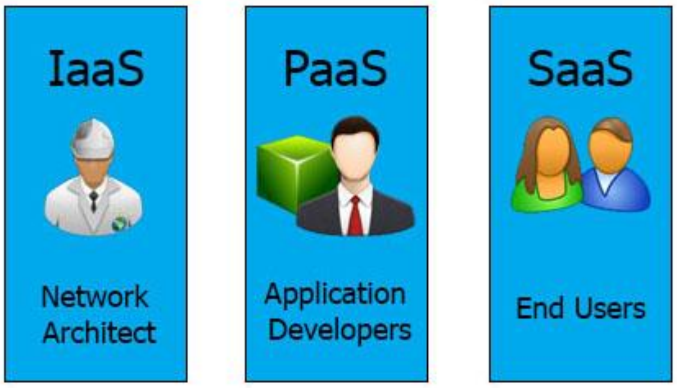


Figure 2-3 Service Based Cloud Users

Software as a Service (SaaS): In SaaS, a cloud provider publishes available applications on a hosting environment, which can then be accessed through the network by various different clients such as web browsers, mobile phones and PDAs. In this model, consumers prefer to use ready and customized applications. They do not have control over the cloud infrastructure such as hardware, operating system, network (routers, switches), bandwidth, etc.

Examples of SaaS include SalesForce.com, Google Mail, Google Docs and GoToMeeting, NetSuite, Workday and so forth.

Platform as a Service (PaaS): PaaS model is a development environment where cloud consumers can develop services, software flows and applications using this platform. PaaS model can be development infrastructure, programming environment, programming tools or configuration management.

Examples of PaaS service providers are Force.com, GoGrid Cloud Center, Google AppEngine, Windows Azure Platform and so forth.

Infrastructure as a Service (IaaS): In IaaS model cloud consumers directly use IT infrastructures such as processing capability, storage, memory, network or any other computing resource. In IaaS model, virtualization technique is used. The main strategy of using virtualization is to setup independent machines that are isolated from both the main hardware environment and other virtual machines [26]. In IaaS model consumers can create a virtual machine by specifying its processing power, storage capacity and network parameters. They have control over the operating system and the application environment.

Examples of IaaS service providers are Amazon Elastic Compute Cloud (EC2), Eucalyptus, GoGrid, FlexiScale, Linode, RackSpace Cloud, Terremark and so forth.

Data storage as a Service (DaaS): DaaS model is a special type of IaaS. In this model only the storage system is virtualized and served to consumers as a data storage service. Enterprise database systems have prohibitive upfront costs, which include software licenses, server licenses and IT maintenance. In DaaS, consumers are to pay just for what they use rather than the site license the for entire database system. Besides, there are no restricted data structures such as relational data base management systems (RDBMS) and file systems. Depending on cloud provider, these services could be custom data structures such as XML, JSON, etc. Users can use table-style abstractions, which can be designed to scale out to store and retrieve huge amount of data within a very compressed time frame [3].

Examples of DaaS include Amazon S3, Google BigTable, and Apache HBase, etc.

Security as a Service (SECaaS): In SECaaS, cloud provider integrates security services such as antivirus software, internet security systems and firewalls into the related infrastructure. It is used as a business model because many providers integrate this model to related cloud services such as SaaS, IaaS, etc. with additional costs. Consumers use the service on a subscription basis. This security model often includes authentication, anti-virus, anti-malware/spyware, intrusion detection and security event management over all operations.

The main advantage of SECaaS is that it is much more cost effective for most individuals or corporations.

Examples of SECaaS include Coiro, NetIQ and Ping Identity.

Network as a Service (NaaS): In NaaS, the provider offers network services virtually over the Internet based on pay-per-use subscription. In this model, network hardware and network infrastructure becomes a utility such as electricity or water. It is used as a business model because many providers integrate this model to related cloud services such as SaaS and PaaS similar to SECaaS.

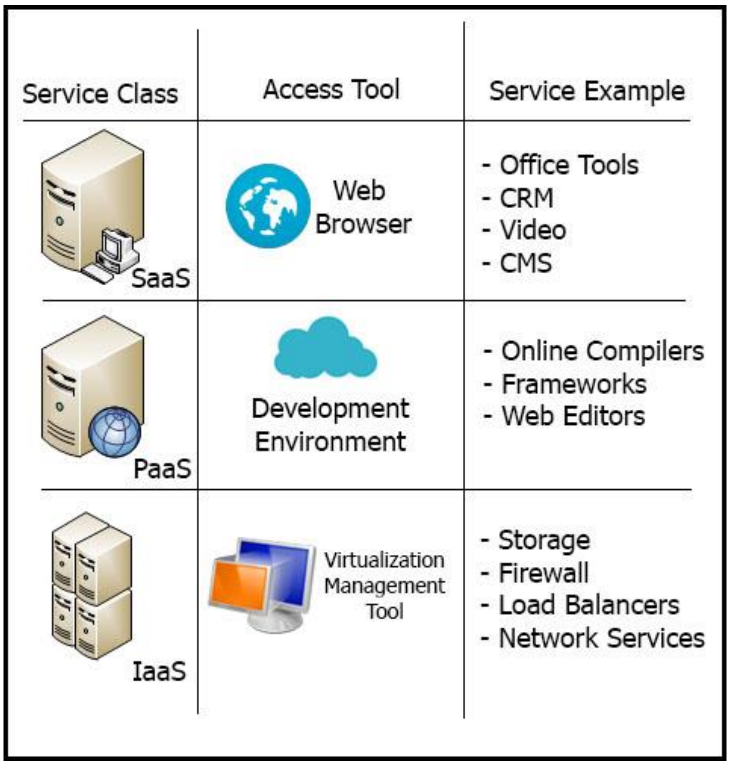
In this model, the only thing a customer does is to create a workspace on one computer. After the creation of this workspace, the provider creates Internet connection and accesses to the related network portal.

The main advantage of this model is that it saves customers from spending money on network hardware and the qualified IT staff which who manage the network in house.

Examples of NaaS include OpenNaaS, FENICS, K3Hosting, etc.

Unified Communications as a Service (UCaaS): In UCaaS, different communication and collaboration applications and services are outsourced to third-party providers and delivered over an IP network. UCaaS is used as a delivery model, which includes enterprise messaging, online meetings, telephony and video conferencing. This model offers high levels of availability, flexibility and scalability for core business tasks. Many companies, primarily small businesses, use UCaaS to avoid the capital and operational expenses associated with deploying a unified communications solution on their own [4].

Examples of this kind of UCaaS include ShoreTelSky, NexWave Telecoms, etc.



Service Class	Access Tool	Service Example
SaaS	Web Browser	- Office Tools - CRM - Video - CMS
PaaS	Development Environment	- Online Compilers - Frameworks - Web Editors
IaaS	Virtualization Management Tool	- Storage - Firewall - Load Balancers - Network Services

Figure 2-4 Cloud Computing Service Class Examples

2.1.2 CLOUD COMPUTING DEPLOYMENT MODELS

In cloud, providers offer their services depending on the size of consumers' demands. For example if privacy and security is more important than the other criteria for a consumer, provider serves a minimized model with the full security and ensured privacy. But if the consumer does not care about locality and security too much, the provider may change the plan and serve wider services such as sharing portals, joint databases, etc.

Basically cloud deployment models, which vary by consumers' requirements, are categorized as private, public and hybrid cloud. In addition, a provider may offer a communication cloud to a specific community or a company.

Figure 2-5 Cloud Computing Deployment Model

Community cloud: Different organizations may join and share same policies, requirements, values and concerns. In community cloud form, collective datasets and platforms are generated and shared by related organizations. The cloud infrastructure may be hosted by a third-party vendor or within one of the organizations in the community.

The main advantages of this form are its smaller cost compared to constructing the whole platform and its democratic equilibrium provided to stakeholders.

Private cloud: Consumers that care about their security and privacy most construct a smaller infrastructure in their local and restricted area. This type of cloud infrastructure is generally operated solely within a single organization such as academics for teaching purposes, etc.

The main advantage of a private cloud is that it is more reliable in data security and data privacy issues. In addition to reliability and security, a consumer may specify their data and server location. Consumers also handle mission-critical activities behind their firewalls. Optimizing the utilization of existing resource may be handled in-house.

Public cloud: Public cloud is the most extensive deployment model among cloud systems. In this deployment model, a provider has the full ownership of its services with its own policy, value, profit, costing and charging models [26].

It is used by the general public consumers who care less about the security and privacy issues compared to private cloud consumers.

Popular public cloud services include Amazon EC2, S3, Google App Engine and Force.com.

Hybrid cloud: Hybrid cloud is a combination of two or more of the above cloud deployment models. It bounds associated services together by standardized technology, which enables data, software and application portability. Organizations that want to control their core operations more securely in private cloud but at the same time want to share their applications and resources to increase their business value on the public side use the hybrid cloud deployment model.

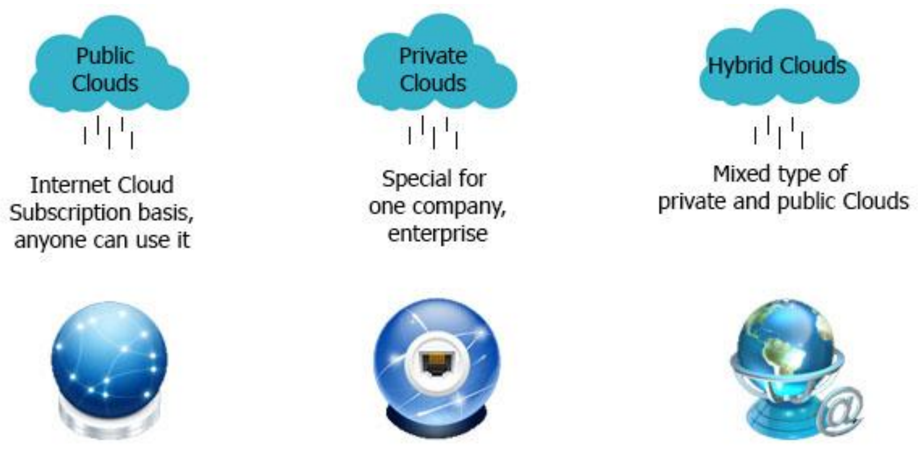

Public Clouds
Internet Cloud Subscription basis, anyone can use it
Private Clouds
Special for one company, enterprise
Hybrid Clouds
Mixed type of private and public Clouds

Figure 2-6 Use of Deployment Models

2.1.3 BENEFITS AND OBSTACLES OF CLOUD COMPUTING

Cloud computing services offer the capability of hiring infrastructure, software, application, etc. For example, for a short term usage, a consumer may provision required computer resources without the need to interact with a personnel. A consumer may create his/her own computer cluster with his/her specific attributes in an on-demand and self-service manner.

Cloud is platform independent, i.e., any consumer may access to resources in the cloud by using a standard network. It includes broad network access with different operating systems and different devices such as laptops, mobile phones and tablets.

Cloud has an elastic system mechanism. For varying consumer needs in time, resources can be rapidly and elastically provisioned.

Consumers can add more resources to scale up their systems. Scaling may be optional and automatic.

Cloud system is based on a metered billing system, which is audited and reported to the consumer. A consumer may be charged based on a known metric such as the number of transactions in the system, the amount of storage used, the application, or the count of network input/output or the amount of processing power used [27].

Cloud systems provide limitless resources to its consumers with a higher level of efficiency and utilization. In addition, consumers do not require hardware or software licenses to implement their services. Cloud provides ease of utilization for everyone.

Cloud computing systems provide resources to its consumers over a wide area network. Since the system incorporates load balancing and failover mechanisms, it becomes more reliable compared to an individual organization.

A cloud system generally has an outsourced IT management. For example in SaaS, software deployment is done by someone else while consumers manage their business only. With this opportunity consumers and businesses achieve considerable reductions in their IT staffing costs.

A cloud system is always up-to-date. In a private cloud, the main system is centralized meaning that consumers apply patches and upgrades in a simplified maintenance environment. Hence a user always gets access to the latest versions of applications and software.

To become subscribers of cloud computing systems consumers sign service level agreements (SLA). With SLA the Quality of Service (QoS) to be obtained from the provider is agreed upon.

In addition to client benefits, cloud providers have some benefits too. A service provider can create pooled resources supporting multi-tenant usage. A provider can create its own network and computer architecture to increase the utilization level of available servers. With the ability of using physical and virtual systems together, dynamic allocation and reallocation is easily managed. A provider takes the benefits of hiding the location of resources by using virtual machines that differ in their processing capability, memory, storage, network bandwidth and connectivity attributes. Being the main controller of the cloud, a provider can create its own security protocols by using virtual machines and roots [5].

Besides the above mentioned advantages cloud computing also faces some obstacles.

In cloud computing, sometimes a system could be unavailable. For example large internet service providers use multiple network providers, so the failure of one of these providers can affect customers' cloud access and decrease the availability rate of cloud [6].

A cloud includes network based services so it is open to criminal threats such as DDoS (Denial-of-Service-Attack) attacks. Criminals can cut off these systems making the service unavailable and causing business losses. Consumers may lose their data. For example, if a cloud provider uses a real computer instead of using the virtualization technique, losing a server causes a consumer to lose its infrastructure.

In cloud, some SLAs include Data Lock-In. In this agreement cloud customers cannot extract their data and move it from one provider to another [6].

Data confidentiality and auditing is another obstacle in cloud computing. Generally current cloud systems work on public networks which opens the system to attacks. Consumers do not want to share their personal data and their privacy with others. Although encrypted storage, virtual local area networks, firewalls, packet filters are generated for security concerns, privacy is still a big concern.

In cloud, performance unpredictability is a problem. Real server features such as CPUs, main memory, storage etc. are shared by multiple virtual machines. However by using virtualization more consumers take advantage of the cloud, hence performance may considerable be low compared to accessing a real computer. More consumer means more input/output interference also.

In cloud, using virtualization increases the utilization of the main system and increases the total client size. Sharing the CPU, memory and power is easier but sharing the storage is relatively harder. Whenever a virtual machine is created, it is tough to increase/decrease the storage size attribute for one customer; it is even harder for all customers, too. Scalable storage management is an ongoing struggle area and a difficult issue compared to scaling up the CPU, memory, etc.

Error management in such large scale distributed systems is a difficult challenge. Bugs usually cannot be reproduced in smaller configurations hence debugging should be done at production scale data centers [6].

Cloud is a pay-as-you-go model and a consumer can scale the resources up or down quickly. Scaling up seems easy whereas computation is slightly different depending on the virtualization level.

In cloud one customer’s bad behavior can affect the reputation of the cloud as a whole, which is named as reputation fate sharing. For instance, blacklisting of EC2 IP addresses by spam-prevention services may limit which applications can effectively be hosted.

Many cloud computing providers originally relied on open source software in part because the licensing model for commercial software was not a good match to utility computing. The licensed programs consumer needs could not be made available on a cloud platform. But nowadays, for example Microsoft and Amazon is offering pay-as-you-go software licensing for Windows Server and Windows SQL Server on EC2.

In cloud, a provider should offer its service with 7/24 availability. At the provider side a resource allocation management process is needed to avoid under-utilization or over utilization of the resources. Depending on actual demand some of the jobs should be rejected due to overcrowding of the virtual machines, which at the end means business loss. To prevent such business loss and to improve the service quality, efficient resource allocation models are needed [6].

2.1.4 VIRTUALIZATION

In cloud computing, a provider does not offer real resources to their clients directly. Dedicating one computer to one client is much more expensive compared to offering it to many clients. To offer the same services to more clients by using one computer, the provider uses a computer architecture technology called as virtualization. The purpose of a virtual machine (VM) is to enhance resource sharing among many users and improve computer performance in terms of resource utilization and application flexibility. Depending on the application, either hardware resources (CPU, memory, I/O devices, etc.) or software resources (operating system and software libraries) may be virtualized in various functional layers.

The main idea in virtualization is to separate the hardware layer from the software layer to yield better system efficiency. Besides, virtualization techniques can be applied to enhance the use of compute engines, networks or storage.Using virtualization, any computer platform can be created or installed in another host computer even if they use processors with different instruction sets and run different operating systems [25].

Consumers generally use a traditional computer with a host operating system that is suitable for the corresponding hardware. With virtualization, a traditional computer may offer different user applications requiring their own operating system (called as guest operating systems) running on the same hardware independent of the host OS. To have this model of operation a virtualization layer is available in many host operating systems. This layer is called as hypervisor or virtual machine monitor (VMM) [25].

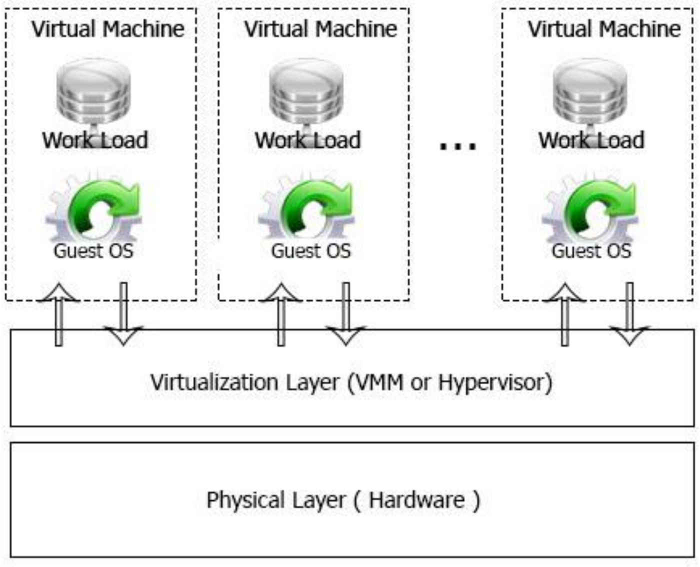


Figure 2-7 Layered Virtualization Technology

Virtualization architectures include three main characteristics; portioning, isolation and encapsulation.

Each VM is isolated from other VMs and also from its host physical system. The main benefit is that if one VM crashes it does not affect other VMs or its host. VMs act like different real computers so data in one VM cannot be seen on other VMs. Due to isolation, data cannot be shared or reached by another computer or user.

Virtualization uses an encapsulation process, which has a complete control of the system resources. To create and use a VM, the hypervisor provides an environment for the programs, which is essentially identical to the original machine environment.

There are five different levels of virtualization: (i) application level (ii) library level (iii) OS level (iv) hardware abstraction layer level and (v) instruction set architecture level. Each level of implementation has a different purpose. Application level virtualization is used for the separation of an installation of an application from the client computer that is accessing it. Library level virtualization is used to hide the operating system related nitty-gritty details to keep it simple for normal programmers. OS level virtualization is a server virtualization method where the kernel of an operating system allows for multiple isolated user space instances instead of just one. Hardware abstraction layer level is used for generating a new architecture such as an emulator, simulator, etc., which uses the host platforms. Instruction set architecture level is all about instruction set emulation, which is the technique of interpreting instructions completely in software [7].

2.2 GENERALIZED ASSIGNMENT PROBLEM

The generalized assignment problem (GAP) examines the minimum cost assignment of n jobs to m agents (machines) such that each job is assigned to exactly one agent subject to capacity restrictions on the agents [8].

The formulation of the problem is as follows:

$$Min \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}\, x_{ij}$$

subject to

$$\sum_{i=1}^{m} x_{ij} = 1\,, \qquad j = 1, \dots., n$$

$$\sum_{j=1}^{n} a_{ij}x_{ij} \;\leq\; b_i\,, i = 1, \dots., m\,;\; x_{ij} = 0 \; or \; 1$$

where

$$x_{ij} = \begin{cases} 1 & if\ agent\ i\ is\ assigned\ to\ task\ j \\ 0 & otherwise \end{cases}$$

$$c_{ij} = the\ cost\ of\ assigning\ agent\ i\ to\ task\ j$$

$$a_{ij} = amount\ of\ agent\ i's\ used\ capacity\ if\ i\ is\ assigned\ to\ task\ j$$

$$b_i = total\ available\ capacity\ of\ agent\ i$$

The generalized assignment problem is an NP-hard combinatorial optimization problem [8]. Considerable research has been done over the last ten years to find effective algorithms to solve various real life applications such as resource scheduling, project scheduling, storage space allocation in designing communications networks with node capacity constraints, routing problems, fixed-charge plant location models, scheduling of payments, assigning software development tasks to programmers, assigning jobs to computers, and so on [8].

2.2.1 SOLUTIONS FOR GENERALIZED ASSIGNMENT PROBLEM

Many algorithms exist to solve GAP. Most algorithms are based on branch-and-bound techniques. These methods mainly differ in the way lower and upper bounds are computed. Other algorithms mainly use knapsack constraints or relaxation of the assignment. Sometimes valid inequalities are added to strengthen the bounds in a relaxation. These methods mainly differ in the type of relaxed constraints. For lower bounds one can simply relax the hard constraints or use lagrangean or surrogate relaxation instead [8].

The main up-to-date solution techniques for GAP include LP-relaxation, lagrangean relaxation, surrogate relaxation, lagrangean decomposition and other heuristics [8] [9].

Experiments show that the computation time for LP-relaxation based algorithms tend to grow fast with increasing problem dimension. Hence heuristic strategies such as Hungarian method and the auction algorithm are preferred most [10].

In this thesis, we prefer to solve our GAP formulation using the auction algorithm and therefore, it is presented and explained in detail in the following subsection as background material.

2.2.2 AUCTION ALGORITHM

The assignment problem is important in many real life contexts, the most common ones being resource allocation such as assigning workers to jobs, servers to clients etc. Most linear programming problems can also be reduced to the assignment problem by means of a simple reformulation such as the linear network flow problem.

The auction algorithm is a heuristic technique used to solve the classical assignment problem. It performs better in time than its competitors and it is well suited for parallel computation.

In the classical assignment problem there are *n* agents (workers, machines, etc.) and *n* jobs that have to match one-to-one. In this algorithm a_{ij} is defined as the cost for matching agent *i* with job *j*. Main aim is to maximize the total benefit $\sum_{i=1}^{n} a_{ij}$ while finding a one-to-one assignment [a set of agent-object pairs (1, j_1), . . . ,(n, j_n), such that the objects j_1, . . . , j_n are all distinct] [11].

Most of the solution methods for the assignment problem are based on iterative improvement of the related cost function, either the primal cost function as in primal simplex methods or dual cost function as in Hungarian like methods, dual simplex methods and relaxations [11].

Auction algorithm is based on a notion of approximate optimality, called Ɛ-complementary slackness. At any iteration, auction algorithm can disrupt both the primal and the dual cost but it finds a good solution the end. With this property auction differs from others which have an ever improving successive cost point of view [11].

Auction Process

The auction algorithm acts like a real life auction system. Suppose that there are *n* historical objects and *n* collectors, each collector acting in his/her own best interest. Suppose that object *j* has a price p_j as a bid so far. If a_{ij} is defined as the cost that collector *i* is prepared to pay for acquiring the object *j*, then the net value of object *j* for collector *i* becomes $a_{ij} - p_j$ and each collector *i* wants to be assigned to an object *j* with maximizes $a_{ij} - p_j$ value, i.e.,

$$\arg max_{j=1,\dots,n} \{a_{ij} - p_j\} \qquad (1)$$

Similar to a real life auction, if this condition is true for a collector in the auction, it is said that the collector is satisfied. When all collectors are satisfied, this is said to be an optimal assignment and the set of prices are at equilibrium [11].

An optimal (or equilibrium) assignment offers maximum total benefit and is the solution for the assignment problem while the corresponding set of prices being the solution for an associated dual optimization problem. This is a consequence of the celebrated duality theorem of linear programming. To find this equilibrium assignment auction algorithm proceeds in iterations starting with any assignment and any set of prices. At the beginning of each round, a new assignment and a set of prices are generated. If all collectors are satisfied, the process terminates. Otherwise, an unsatisfied collector *i* is selected and this collector *i* finds an object j_i such that

$$j_i \in \arg max_{j=1,\ldots,n} \{ a_{ij} - p_j \} \tag{2}$$

Then the algorithm

(i) exchanges objects with the collector *i* assigned to j_i at the beginning of the round,

(ii) sets the price of the best object j_i to the level at which *i* is indifferent between j_i and the second best object as

$$p_{j_i} = p_{j_i} + \gamma_i \tag{3}$$

where

$$\gamma_i = v_i - w_i \tag{4}$$

and v_i and w_i being the best and the second best object values as follows:

$$v_i = max_j = \{ a_{ij} - p_j \} \tag{5}$$

$$w_i = max_{j \neq j_i} \{ a_{ij} - p_j \} \tag{6}$$

w_i is the best value over objects other than j_i. γ_i is the largest increment by which the best object price p_{j_i} can be increased with j_i still being the best object for collector *i*. It is actually the same in a real life auction. At each round, bidder *i* raises the price of its preferred object by the bidding increment γ_i. Like real life incrementing, this cannot be negative since $v_i \geq w_i$, hence object prices always increase [11].

When more than one object offers maximum value for bidder i then the bidding increment γ_i becomes zero. After this several collectors encounter a smaller number of equally desirable objects without raising their prices, which creates a never ending cycle.

In real life to break such cycles each bid for an object must raise its price by a minimum positive increment and bidders take risks to win their preferred objects. The same mechanism is used in the auction algorithm by fixing ε and saying that a collector *i* is almost happy with an assignment and its associates set of prices if the value of its assigned object j_i is within ε of its maximum. In other words *i* satisfied if

$$a_{ij_i} - p_{j_i} \geq max_{j=1,\dots,n} \{ a_{ij} - p_j \} - \varepsilon \qquad (7)$$

Using this fixed positive scalar ε, a solution is said to be an assignment and the associated set of prices are almost at equilibrium when all collectors are satisfied. This method is known as ε-complementary slackness and plays a central role in several optimization contexts.

ε = 0 case reduces to ordinary complementary slackness as described in Eq. (1).

For ε-complementary slackness, the bidding increment becomes

$$\gamma_i = v_i - w_i + \varepsilon \qquad (8)$$

Smaller increments of γ_i should also work as long as $\gamma_i \geq \varepsilon$ but using the largest of possible increments accelerates the algorithm similar to real life where bidding terminates if the bidding is aggressive.

For ε-complementary slackness, once an object receives a bid for the first time then the collector assigned to the object at every subsequent round is almost satisfied because the other object prices cannot decrease during the course of the algorithm. In the algorithm, collectors which are not satisfied should be assigned to objects that have never received a bid yet. When each object receives at least one bid, the algorithm should terminate [11].

If an object receives a bid in r rounds, its price must exceed its initial price by at least $r\varepsilon$. For sufficiently large r, the object will become expensive enough to be judged inferior to some object that has not received a bid so far. After r rounds two possible scenarios are possible. The auction either terminates with all collectors being satisfied before every object receives a bid or it continues until all objects receive at least one bid where the auction now terminates. After termination the assignment is almost at equilibrium. The maximization of the total benefit depends on the magnitude of ε. An assignment and its associated set of prices that are almost at equilibrium may be viewed as being at equilibrium for a slightly different problem where all benefits a_{ij} are the same as before except for n benefits of the assigned pairs, which are modified by an amount no more than ε. Suppose that the benefits a_{ij} are all integers. Then the total benefit of any assignment is also an integer hence if $n\varepsilon<1$ then a complete assignment within $n\varepsilon$ neighborhood of the optimal must be optimal [11].

It follows that if $\varepsilon = 1/n'$ and the benefits a_{ij} are all integers then the assignment obtained upon termination of the auction algorithm is optimal. The computation time depends on the value of ε and on the maximum absolute object value. The number of bidding rounds up to termination is proportional to $max_{i,j} \mid a_{ij} \mid / \varepsilon$. Sometimes termination time depends on initial prices also. If these prices are close to optimal values, the number of rounds becomes relatively small.

To get better quality solutions ε-scaling must be used by applying the algorithm several times starting with a large value of ε and reducing ε up to a critical value such as ${}^{1}\!/_{n}$ [11].

CHAPTER 3

LITERATURE REVIEW

In cloud computing resource arbitration and resource allocation are critical management issues. IT services are to be provisioned on subscriptions based on consumer computing requirements where any improper allocation may cause business loss. Defining the optimal utilization model and enabling the requested resources by the consumers is still a challenge in cloud computing. Any failure in this allocation may also lead to a serious performance degradation of the cloud.

Cloud computing is relatively a new trend hence many research tracks in this field are at their initial stages. While some papers about effective client-server allocation focus on generating optimal models, others focus on finding an effective algorithm [28] [29] to make the allocation. The main purpose of all is to provide optimal utilization in the cloud.

The present section presents a brief overview of existing cloud computing related literature on resource allocation.

3.1 MODELING AND APPROACHING COST TRANSPARENT SPECIFIC DATA CENTER POWER CONSUMPTION

Cloud computing uses virtual machines to increase utilization of servers. In [12], the authors aim to utilize servers in a data center so that the power consumption of the entire data center can be held at a specific level. To achieve this goal they focus on VMs and create a model that can estimate data center and VM power consumption. In their model data centers' total CPU utilization is used in predicting the power of a server. The authors estimate each VM's power demand as a fraction of the entire server system’s power consumption. To measure each server’s power consumption Voltcraft Energy Logger 4000 is used. To obtain the required CPU power measure VMware vSphere client tool is used.

The total CPU utilization $C_{server}(t)$ of the physical host is calculated as

$$C_{server}(t) = \sum_{i=1}^{n} C_{VMi}(t)$$

where *n* is the total number of virtual machines on a given server and $C_{VMi}(t)$ is the CPU utilization of VM *i*,

Then linear regression model is used to calculate the server’s power consumption at time *t* as

$$P_{model}(t) = a + b * C_{server}(t) + c * C_{server}(t)^2$$

where $C_{server}(t)$ denotes the CPU utilization of the entire server at time *t* and a, b and c are model parameters characterizing the hardware used.

3.2 EFFICIENT RESOURCE ARBITRATION AND ALLOCATION STRATEGIES IN CLOUD COMPUTING THROUGH VIRTUALIZATION

Resource allocation management is required in cloud in order to avoid underutilization or overutilization of resources. The authors in [13] focus on effective resource utilization generating an economic benefit in cloud. To decrease business losses, a queuing algorithm is employed where jobs generate requests at a random fashion for resources in the cloud. The following parameters are used in this algorithm;

Λ: Rate of resource requests from all subscribers

μ: Rate with which the resource is allocated to the subscribers

TP: Task priority given to the jobs based on their criticality.

BP: Priority estimated by the priority manager based on customer relationship factors and the cost of the current job.

The proposed queue based algorithm uses TP and BP as a decisive factor and to check the stable operation use the value of $\mu > \lambda$. In the beginning of a cycle, scheduler puts jobs into a FIFO system. Afterwards, job priority manager assigns the jobs to resources based on their TP and BP values.

To check for efficiency

$$n = No.of\ Processors\ on\ Supply - No.of\ processors\ on\ demand$$

is used as a measure.

3.3 A DYNAMIC AND INTEGRATED LOAD-BALANCING SCHEDULING ALGORITHM FOR CLOUD DATA CENTERS

Traditional load-balancing and scheduling algorithms proposed for cloud computing in the literature generally consider CPU load of physical servers or VMs. However in [14], resources are chosen to be CPU, memory and network bandwidth, which are integrated for both the physical and the virtual machines. Authors in [14] created a dynamic and integrated resource scheduling algorithm, in which the following parameters are considered:

CPU_i^U: Average CPU utilization of a single server *i*

Average utilization of all CPUs in a cloud datacenter:

$$CPU_u^A = \frac{\sum_i^N CPU_i^U * CPU_i^n}{\sum_i^N CPU_i^n}$$

where CPU_i^n is the total number of CPUs of server *i* and *n* is the total number of physical servers in the cloud datacenter.

Similar to average utilization of CPU, the authors compute average utilization of memory and network bandwidth for server *i*, and the overall ones and obtain MEM_i^U, NET_i^U, MEM_u^A, NET_u^A.

The variance is widely used in statistics as a measure of how far a set of numbers are spread out from each other. Having related average values and variance, an integrated load imbalance value (ILB*i*) for each server *i* is defined as follows:

$$\text{ILBi} = ((\text{Avgi} - CPU_A^U)^2 + (\text{Avgi} - MEM_A^U)^2 + (\text{Avgi} - NET_A^U)^2)/3$$

where;

$$\text{Avgi} = (CPU_i^U + MEM_i^U + NET_i^U)/3$$

ILBi indicates load imbalance comparing the utilization of CPU, memory and network bandwidth for a single server. Authors in [14] use these values as an input to their scheduling algorithm, which aims to consider CPU, memory and bandwidth, together.

3.4 BLACK-BOX AND GRAY-BOX STRATEGIES FOR VIRTUAL MACHINE MIGRATION

Similar to [14] Wood *et. al.* introduced virtual machine migration techniques in [15]. The authors focus on load imbalance level of servers to maximize the following integrated resource utilization model:

$$V = \frac{1}{(1 - CPU_u)(1 - MEM_u)(1 - NET_u)}$$

where CPU_u, MEM_u, NET_u denotes average utilization of CPU, memory, network bandwidth during each observation period, respectively.

In this model a high V value is expected for a high integrated utilization. The authors also aim to use each VM with full utilization. They propose allocation and migration algorithms based on their measurements.

3.5 GREEN CLOUD COMPUTING: BALANCING ENERGY IN PROCESSING, STORAGE, AND TRANSPORT

With the emergence of cloud computing, green computing became important. Green computing is the study and practice of environmentally sustainable computing or information technology. With cloud computing, many computers serve many clients. 'Many' term therefore refers to huge power consumption also. The literature has a lot of research work that aims to decrease power consumption and maintain green computing ideals in cloud.

One example is [16]. The developed model considers the power consumption of servers. A different model is presented for each cloud deployment model in [16] but only SaaS model in reviewed below since the present thesis work is limited SaaS. In cloud, consumers using SaaS service are charged a monthly or yearly fee to access the latest versions of available software. All computations are performed in cloud. A consumer PC is used only to send commands and get results. Typically, a consumer is free to use any computer connected to the internet. A consumer accesses the software service with a terminal that communicates with its server using simple commands transmitted through the Internet. The following power consumption model for servers is proposed in [16] considering many factors:

$$P_{sf} = P_{sf,\,PC} + \frac{1.5\,P_{sf,SR}}{N_{sf,SR}} + 2\,B_d\;\frac{1.5\,P_{SD}}{B_{SD}} + AET$$

where

P_{sf} is the per-user power consumption of the software service,

A is the bit rate,

PC is the power consumption corresponding to the user's terminal,

$P_{sf,SR}$ is the power consumption of the server,

P_{SD} is the power consumption of the hard disk arrays,

$N_{sf,SR}$ is the number of users per server,

B_{SD} is the capacity of the hard disk arrays,

AET is used for refreshing rates,

B_d is used for redundancy

Factor 2 is used to model the power requirements for redundant storage and factor 1.5 is used to model the energy consumption in cooling as well as other overheads.

3.6 INDEPENDENT TASK SCHEDULING IN CLOUD COMPUTING BY IMPROVED GENETIC ALGORITHM

In [17], authors combine min-min and max-min algorithms within a genetic algorithm framework. In min-min algorithm, unassigned tasks are taken. Algorithm computes the minimum execution times of tasks among available resources. Then minimum of these values is selected and the corresponding task is scheduled on the related resource. Max-min is almost the same as the min-min algorithm except the following: in this after finding the maximum execution times of tasks among available resources, the minimum times of these values is selected. Genetic algorithm combines these two to select the best solution. The authors aim to minimize at the end the scheduling time and to obtain a better scheduling.

In the literature up to now, we observe that server side attributes are generally used in search for better resource allocations. Among these attributes there are server power, CPU, load imbalance level and consumer and task priority. To the best knowledge of the author, consumer side attributes such as consumer priority, bandwidth, quota, etc. are never considered in previous research works. Most of the previous works used soft constraints in creating their models. To satisfy both the consumer and the provider, a mixed data model and a related solution method is required. In the present thesis work, such a model is developed, which aims to satisfy providers and consumers' needs in a joint fashion but without any hard constraints.

CHAPTER 4

JOINT RESOURCE ALLOCATION SYSTEM FOR CLOUD COMPUTING

In cloud computing, cloud process starts with the client request. A client generates request through network towards related cloud services. Virtual machine on cloud gets data from storage cloud. A virtual machine derived from real servers on cloud aggregates information for client then responds with the results to client through network. After the response, virtual machine on compute cloud puts data into storage cloud. To successfully complete this operation a cloud provider deals with many issues in their systems.

4.1 OPERATIONAL ISSUES FROM PROVIDER PERSPECTIVE

Towards creating our joint allocation model in cloud computing we first analyze main problems of providers.

When a provider decides to create a cloud computing business, the first issue is to find a suitable data center location. Geographical locations and physical sizes are decision variables in location planning. Subject to service levels, a provider wants to minimize its infrastructure investment, operations costs and non-renewable energy consumption [18].

When establishing a new cloud computing business, a provider should consider its data center capacity. If the cloud center is to serve many users, bandwidth requirements, computational requirements, storage array type and communication array type, etc. should also be considered and subject to service level agreements and consumer priorities operational costs should be minimized [18].

After creating a data center in a suitable location with a predefined capacity, data center layout planning is the next issue. Data centers consume huge energy while working and hence there exists serious cooling requirements. While designing the layout, a provider should consider hardware layout, rack space requirements, power requirements and cooling costs.

When a data center is up and ready to be used, resource scheduling should be done effectively. Some cloud providers choose power on/off and hardware on/off strategy to minimize energy consumption and cooling cost. Some use software based scheduling algorithms to assign clients to servers.

In cloud, many providers use virtualization technology to serve more consumers. If a provider wants to maximize hardware utilization subject to demand variability then hardware load balancing is also needed.

Some providers serve their services by using partners. The main aim in having a partner is to increase the quality of service but having a partner is expensive and cost of partnerships should be considered carefully [18].

Cloud services are internet based services hence any user from any computer can connect these services, which are open for intrusion also. So a cloud provider should get a good security system to avoid attacks subject to its QoS attributes and security budget.

Cloud computing is a pay-as-you-go service model so a provider should clearly define its product and service pricing as per processor unit price, per memory unit price, etc. A provider should satisfy all its customers while maximizing its own profit.

4.2 OPERATIONAL ISSUES FROM CONSUMER PERSPECTIVE

Towards creating our joint allocation model in cloud computing we secondly analyze main problems of customers and their needs.

In cloud computing, a consumer joins the cloud by signing an SLA first. In such aggreements topics such as service quality, service availability, disaster recovery, system stability, etc. are generally addressed.

In joining a cloud, a consumer wants its resources to be easily scalable subject to its budget. Scalability means increasing or decreasing service attributes such as CPU, memory, storage or process count. Moreover, cloud services should be feasible for consumers and consumer sould be allowed to migrate depending on various requirements.

A cloud consumers can be a person or an entity such as a company. A small company as a customer should decide carefully about its machine and storage requirements and machine configuration before becoming a cloud member. Subject to company’s demands, capacity planning is gerenally done and infrastructure subscription cost is mimized.

For larger companies, virtual machine scheduling should be done effectively. Because of variability in operational demand, consumer should at least consider virtual machine active/inactive schedule or use some more complicated software to manage demand variability.

Service availability plays an important role for consumers. In addition to effective scheduling, maximizing the utilization is also important. Hence a consumer should be able to decide about the extent virtualization is used and about the physical locations to be used in the system [18].

4.3 JOINT MODEL

In cloud computing, we observe that both providers and consumers have decision variables to satisfy their needs. In our work, we aim to create a joint model, which covers both consumers and providers' needs, simultaneously. To form such a model, similar to existing literature, both consumer and provider attributes are mostly used in the form of soft constraints.

A service level agreement signed by the provider and consumers in cloud is a part of a service contract where the service is formally defined. In this agreement contracted delivery time of the service or performance attributes of the service are described. These agreements ensure clients' rights against cloud providers. Consumer want to have a 7/24 available cloud service in general because consumers could be small business offering their systems to their own customers via the cloud. Unavailability may result in serious business loss. Another important issue for customers is stability in performance of the service. Unstable performance may also cause business loss. When works are uploaded to the cloud, consumers start worrying about privacy and security issues also. Cloud services may become unsuitable or infeasible for consumers in the course of time and therefore a consumer might want to quit or stop the subscription. Change management process should also as easy as possible. While working on local computers, consumers take backup of their systems themselves to prevent data loss. In cloud, consumers expect providers to have the necessary means and disaster recovery mechanisms to prevent corruption or loss in their private data. For security concerns consumers may not want to locate their data on certain countries. Some consumers want to reach their data or their software by using other devices such as PDAs, mobile phones, etc. Data access and data portability are important issues for mobile users.

Since a metered billing system is employed in the cloud many providers use quotas for various resources. A quota may be used for storage, process count, usage count per user, etc. In a local workspace, a user can change the storage, ram, CPU and other attributes of computers. Many consumers generally want to act in cloud in the same manner also. Therefore, elasticity plays an important role for such users. Elasticity allows users to change their SLA level specifications. In many cloud operations services are provided with different prices and QoS levels corresponding to different customer priority levels.

In cloud computing, providers want to increase their profit and quality of service. For this they purchase and install powerful computers to form services to serve consumers in an efficient manner. Providers try reducing computational load on their service and minimize virtual machine CPU and memory usage. Besides computational load they try to reduce storage load also to increase the quality of services.

Since cloud operates over the internet, a network load is generated. Providers also want to reduce network load, which is known during the assignment of clients to servers. Besides computational, storage and network load, some providers pay more electricity cost compared to other providers that can generate their own energy. Due to high cost of electricity some providers may need to emphasize reducing total power consumption of servers to generate more profit [18].

As was presented in Chapter 3, many models and algorithms in the literature, find optimal utilization for cloud computing resources while satisfying some stakeholder. But in these models either the providers' needs or the clients' needs are considered. We observe that there is a need for a joint model that satisfies both consumers and providers, simultaneously but in a soft fashion.

In creating such a model, we integrate the following consumer side attributes:

- consumer priority,
- total individual quota,
- total used individual quota,
- bandwidth of consumer variables

Besides the above consumer needs, we integrate the following provider side attributes:

- CPU value of servers,
- memory value of servers,
- storage value of servers,
- load imbalance level,
- bandwidth value of servers,
- power consumption value of servers

The model is generated by using the above attributes and a linear mapping of each attribute separately.

$$C_i = P_i * \left[\frac{Q_{ui}}{Q_{Ti}}\right] * \frac{1}{BW_i}$$

is the consumer part of the model where

C_i : cost for consumer *i*

P_i : priority for consumer *i*

Q_{ui} : quota used for consumer *i*

Q_{Ti} : total quota for consumer *i*

BW_i : bandwidth for consumer *i*

Similarly,

$$S_j = \alpha * \frac{LI_j}{BW_j} + (1 - \alpha)PW_j$$

is the provider part of the model where

α : scaling parameter

LI_j : load imbalance level for server *j*

BW_j : bandwidth value for server *j*

PW_j : power consumption value for server *j*

S_j : cost for server *j*

In the above load imbalance level (LI_j) is computed as

$$LI_j = \frac{(AVG_{CPU} - CPU_j)^2 + (AVG_{MEM} - Mem_j)^2 + (AVG_{STR} - Storage_j)^2}{3}$$

where

AVG_{CPU} : average CPU value for servers

AVG_{MEM} : average memory value for servers

AVG_{STR} : average storage value for servers

Consumer and provider models are constructed separately but can be combined to form a single objective function to be suitable for an assignment problem formulation. Hence a weight matrix is computed by simply multiplying the two developed cost models:

$$W_{ij} = C_i * S_j$$

The formulation of the problem then is as follows:

$$Min \sum_{i=1}^{m}\sum_{j=1}^{n} W_{ij}\, x_{ij}$$

subject to

$$\sum_{i=1}^{m} x_{ij} = 1\,, \qquad j = 1, \dots., n$$

$$\sum_{j=1}^{n} a_{ij}x_{ij} \;\le\; b_i\,, i = 1, \dots., m\,;\; x_{ij} = 0 \; or \; 1$$

where

$$x_{ij} = \begin{cases} 1 & if\ customer\ i\ is\ assigned\ to\ task\ j \\ 0 & otherwise \end{cases}$$

$$W_{ij} = cost\ of\ assigning\ customer\ i\ to\ server\ j$$

$$a_{ij} = amount\ server\ i's\ used\ capacity\ if\ i\ is\ assigned\ to\ customer\ j$$

$$b_i = total\ available\ capacity\ of\ server\ i$$

In our model each server can serve at most only one client and the above problem can easily be solved by the auction algorithm described as an overview in the background chapter.

CHAPTER 5

COMPARATIVE EVALUATION

In this chapter, our proposed joint model is evaluated in terms of power consumption, bandwidth efficiency, load imbalance level of servers and priority based assignments in comparison to existing cloud optimization models. We did our best to choose the most competitive models in terms of efficiency or cost as our benchmark models.

5.1 EXPERIMENTAL SETUP

To construct a meaningful experiment, 10 real computers are used on the cloud provider side. 30 virtual machines, which have different properties such as number of CPUs, size of memory and storage, are generated by using these ten real computers. 30 different client (cloud consumer) instances are generated at the consumer side.

An assignment manager program is built using C++. While creating this program to simplify matrix operations, boost library (version 1.47.0) is utilized. The assignment manager uses the auction algorithm as its allocation method. It uses a weight matrix, which is defined according to our joint model, as an input. Following the assignment process, our program outputs a 0-1 matrix as the assignment result and all analysis that follows is then based on such outputs.

5.1.1 INSTANCE CREATION

While creating our problem instances, we assumed that there is a provider, which has ten real computers, that establishes a cloud computing service to its clients. To maximize the utilization of real servers, virtualization technique is assumed to be employed.

50 different virtual machine pools are generated, which can afford a maximum of 30 clients. For each virtual machine pool, 4 different size example cases are considered: 5 clients versus 30 servers, 10 clients versus 30 servers, 15 clients versus 30 servers and 20 clients versus 30 servers.

Real server properties such as CPU, memory and storage are generated in accordance with [19] and [20]. Bandwidth values are average bandwidth values chosen in accordance with [21] and [22]. Table 5-1 lists the chosen characteristics of servers used in our test case.

Table 5-1- Main Characteristics of Real Servers

No	CPU	No of Cores	Memory	No of Memories	Hard Drive	Size (TB)	No of Hard Drives	Total Power (Watt)	Bandwidth (Mbps)
1	Core i7-950	4	4GB DDR3	2	10.000RPM 3.5	2,0	2	408	25,78
2	Core i7-960	4	4GB DDR3	4	7200RPM 3.5	2,0	4	497	19,34
3	Core i5-680	2	4GB DDR3	3	7200RPM 3.5	2,0	3	421	19,34
4	AMD Phenom x4	4	4GB DDR3	4	10.000RPM 3.5	2,0	4	473	12,89
5	AMD Phenom II x6	6	4GB DDR3	4	10.000RPM 3.5	1,0	8	1197	26,00
6	AMD Phenom II x6	6	2GB DDR3	4	7200RPM 3.5	1,5	3	428	26,00
7	AMD Phenom II x6	6	1GB DDR3	4	5400RPM 3.5	1,0	2	346	26,00
8	AMD Phenom II x6	6	4GB DDR3	4	7200RPM 3.5	2,0	8	558	14,63
9	Core i7-860	4	4GB DDR3	4	7200RPM 3.5	1,0	2	385	19,50
10	Core i7-880	4	4GB DDR3	4	7200RPM 3.5	2,0	6	481	19,50

Different manufacturers and different load imbalance properties are used while constructing these servers. In addition to these main properties, different hardware equipments and attributes are also assumed such as video cards, cooling options etc. In constructing these real server computers, we aim to be able to create later different virtual machine pools having different characteristics. Table 5-2 lists those additional characteristics chosen for servers used in our test case.

Table 5-2 – Additional Properties of Real Servers

No	MotherBoard	Video Card	Optical Drive	No of Disk Drives	PCI	Fan	Number of Fans
1	Server MotherBoard	Integrated	DVD-RW	1	Network	120mm	2
2	Server MotherBoard	Integrated	DVD-RW	2	Network	140mm	3
3	Server MotherBoard	GeForce 9600GT	Not installed	0	Network	92mm	2
4	Server MotherBoard	Ati Radeon 9600	Not installed	0	Network	140mm	4
5	Server MotherBoard	Ati Radeon 6990	DVD-RW	1	Network	140mm	2
6	Server MotherBoard	Integrated	DVD-RW	1	Network	140mm	1
7	Server MotherBoard	Integrated	Not installed	0	Network	60mm	2
8	Server MotherBoard	Integrated	Not installed	0	Network	140mm	4
9	Server MotherBoard	Integrated	Not installed	0	Network	120mm	2
10	Server MotherBoard	Integrated	DVD-RW	1	Network	140mm	1

To maximize utilization and to serve more clients, we generated a number of virtual machines (VMs) using the above real computer properties. VMs are generated in accordance to host machine properties and employed VM management tool's system requirements [24]. While creating VMs, we used Joule meter tool [23] to calculate the power consumption of the corresponding VM. We generated 30 VMs from 10 real computers to serve at most 30 clients.

Clients are generated randomly. Each client has a different priority (1-highest, 5-lowest), a different quota and a different average bandwidth value depending on the network infrastructure of the connected country. Bandwidth values are generated in accordance with [22]. In cloud computing, quota values are generally defined in SLAs or contracts. In our model, quota value is used as the number of transactions, which is chosen proportional to a client's priority. Table 5-3 lists properties of clients used in our test case.

Table 5-3 – Client Properties

No	Priority	Country	Bandwidth (Mbps)	Quota(number of transactions)
1	1	Norway	29,80	1000
2	2	Turkey	11,20	800
3	3	Russia	22,70	600
4	4	Germany	26,00	400
5	5	Turkey	11,20	200
6	5	Germany	26,00	200
7	3	Italy	8,40	600
8	2	France	33,60	800
9	1	Iraq	4,40	1000
10	5	Iran	3,00	200
11	4	Ukraine	21,50	400
12	2	Romania	56,70	800
13	1	Austria	20,61	1000
14	3	Romania	56,75	600
15	2	Austria	20,61	800
16	1	Ukraine	21,50	1000
17	3	China	18,80	600
18	3	China	18,80	600
19	1	Switzerland	52,41	1000
20	4	Spain	25,65	400
21	5	Ukraine	21,50	200
22	5	Germany	26,00	200
23	2	Turkey	11,20	800
24	2	Italy	8,40	800
25	3	Russia	22,70	600
26	4	Czech	25,78	400
27	2	France	33,65	800
28	1	Greece	8,95	1000
29	2	Bulgaria	31,75	800
30	3	Belarus	11,30	600

5.1.2 EVALUATED METRICS

We evaluated our technique on four key metrics with the goal of measuring the impact and efficiency of our joint model on the cloud. Following each assignment process in our test cases, the assignment result is analysed accordingly after computing the following metrics:

- Total power consumption (TPC): the sum of selected server power consumption
- Total bandwidth value (TBW): the sum of selected server bandwidth
- Priority based load imbalance ratio (LIBR): the ratio of the sum of the selected high priority (1 and 2) based load imbalance levels to the sum of selected low priority (4 and 5) based load imbalance levels
- Priority based bandwidth ratio (BWR): the ratio of the sum of high priority (1 and 2) based bandwidth values to the sum of low priority (4 and 5) based bandwidth values

5.1.3 MODELS USED IN COMPARISON

To evaluate our proposed joint cloud model, the following cloud optimization models found in the literature are chosen for comparison:

- Comparison model (CM1): DAIRS algorithm in [14] aims to select an average level of CPU, memory and bandwidth from the server pools to serve clients.
- Comparison model (CM2): Load balance measurement model in [15] aims to increase the utilization of virtual servers by selecting high CPU, memory and bandwidth.
- Comparison model (CM3): Data center power consumption model in [12] aims to decrease power consumption of virtual machines by selecting lower CPU counts.

5.1.4 TEST RESULTS

In our tests, 30 different virtual machines are generated from 10 real computers. 50 different VM pools are generated. Each VM pool consists of 30 VMs that differ from each other in their CPU size, memory, storage, bandwidth, and power consumption. For each optimization model 200 different assignments are generated.

In our joint model the scaling parameter α is used to set and customize the balance between provider and consumer requirements. Therefore, three different set of tests are conducted corresponding to three levels of α and the assignment results are compared with CM1, CM2 and CM3.

In our test results presented below, color green is used to represent servers in use whereas color red is used to represent servers out of use. The x-axis represents servers and the y-axis represents the associated quantity.

In bandwidth comparison graphs, unit is Mbps, in load imbalance level comparison graphs ratios are compared and in power consumption comparison graphs, unit is Watt.

5.1.5 RESULTS FOR α = 0.2 CASE

Scaling parameter α = 0.2 means that the provider prefers to select those computers that consume the least power among its available virtual machine pool. In other words, the provider does not care about the load imbalance level and the bandwidth to the same extent as power consumption. This kind of selection might be useful for those providers that cannot create their own energy or that have to pay higher unit costs for the energy.

Although other compositions are also tested, only results for 15 customers 30 VMs case is presented in the following.

Figure 5-1 shows the comparison of bandwidth use in different assignments obtained when different models are employed for resource allocation.

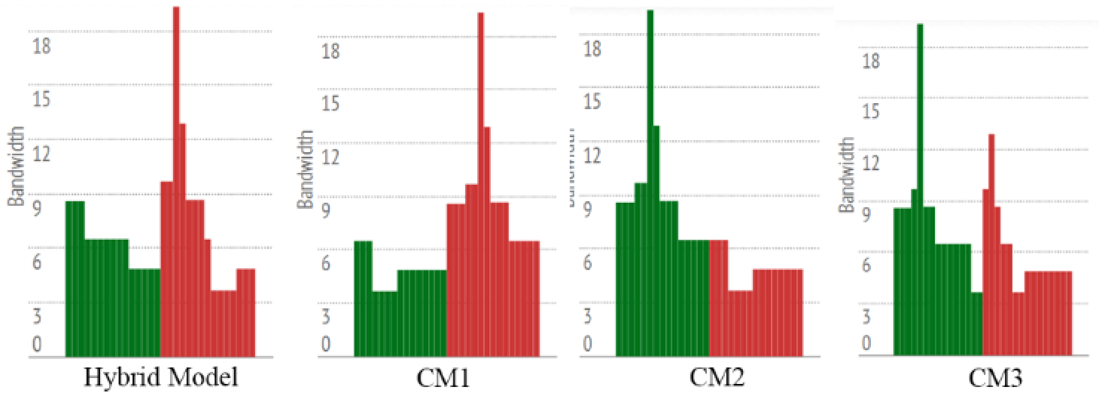


Figure 5-1 – Bandwidth Comparison for α = 0.2

As was stated earlier α = 0.2 means that bandwidth maximization is not given a high preference compared to minimization of power consumption. Therefore we observe that CM2 and CM3 achieve higher bandwidth values compared to our joint model and CM1. CM1 is known to aim an average level of bandwidth. For this case it seems that our joint model also makes assignments also achieving an average level of bandwidth selection.

Figure 5-2 shows the comparison of load imbalance level in different assignments obtained when different models are employed for resource allocation.

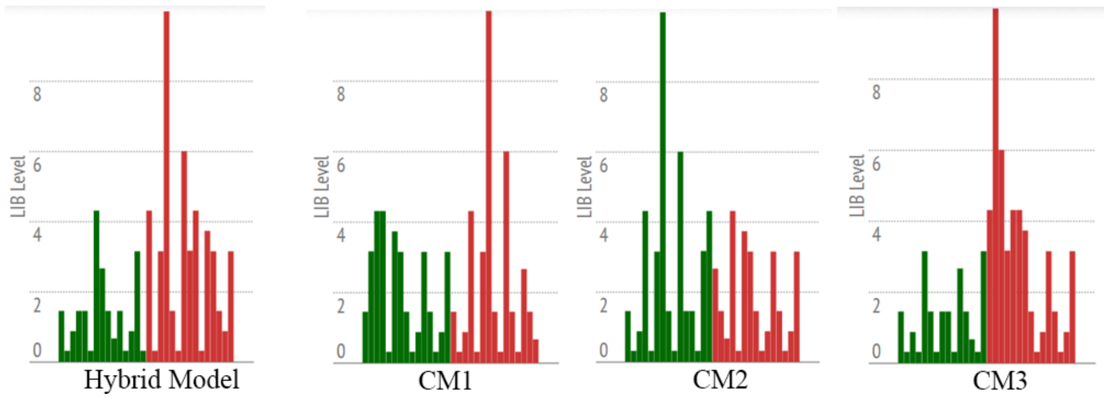


Figure 5-2 – Load Imbalance Level Comparison for α = 0.2

For α = 0.2, load imbalance level is not considered much similar to bandwidth utilization hence load imbalance level is selected depending on power consumptions of the computers. Therefore, any kind of LIB level can be chosen. CM1 aims to use average utilization levels for CPU, memory and bandwidth and it is observed that this is realized. CM2 aims to increase the utilization of VMs. In the following graph, we observe that CM2 attains higher load imbalance levels, which means highest CPU, memory and storage. CM3 considers only power consumption and in the following graph we observe that CM3 uses an average load imbalance levels in return.

Figure 5-3 shows the comparison of power consumption in different assignments obtained when different models are employed for resource allocation.

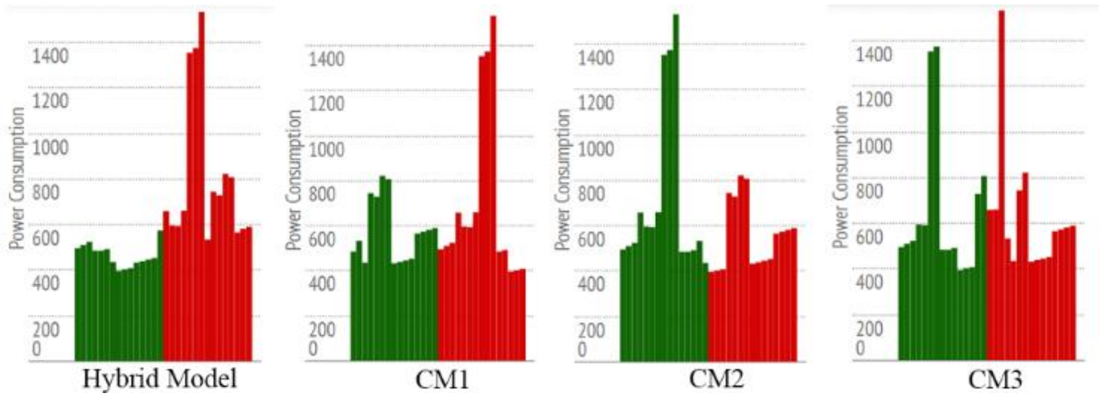


Figure 5-3 – Power Consumption Comparison for α = 0.2

Inspecting the power consumption values we observe that our joint model selects computers that consume less power. Because of having CM1 aiming average utilization, it also selects computers that consume an average level of power. Although CM3 aims to select computers with lower power consumption values, we observe that CPU is not the only factor for power consumption. Power consumption of course increases with memory size, storage and other hardware utilities such as graphic cards, etc. In CM2, power consumption values are higher compared to other models. CM2 does not care about power consumption anyway hence we observe that it is among the worst.

In addition to above, we checked the metrics given in section 5.1.2. for a detailed comparison. Table 5-4 tabulates these comparison metrics for α = 0.2.

Table 5-4 – Comparison Metrics for Scaling Factor 0.2

	TBW	TPC	LIBR	BWR
Joint	95.67	6976.19	2.08	1.59
CM1	73.13	8635.14	2.02	1.41
CM2	135.84	10728.57	0.63	1.00
CM3	118.44	9651.40	1.26	0.93

In Table 5-4 we observe that in our joint model, total bandwidth usage is at an average level when compared to others. Also total power consumption is the lowest. However load imbalance level and bandwidth values are not the best, but this expected due to using scaling factor as 0.2. We observe that LIBR has the highest value, which means high priority clients have more powerful computers assigned to then the low priority clients. For BWR we also observe that high priority clients have more bandwidth then lower priority ones. CM1 consumes the lowest power and CM2 uses the highest bandwidth as expected. Because of not using client priorities as a parameter, we cannot conclude that any of the models consider client priorities better by analysing the available values.

5.1.6 RESULTS FOR α = 0.5 CASE

Scaling parameter α = 0.5 means that the provider prefers to select those computers that consume average power, average load imbalance level and average bandwidth among its available virtual machine pool. In other words, the provider cares about the load imbalance level and the bandwidth to the same extent as power consumption. This kind of selection might be useful for those providers, which use a more economical plan for energy cost and has an average cost network infrastructure.

Although other compositions are also tested, only results for 15 customers 30 VMs case is presented in the following.

Figure 5-4 shows the comparison of bandwidth use in different assignments obtained when different models are employed for resource allocation.

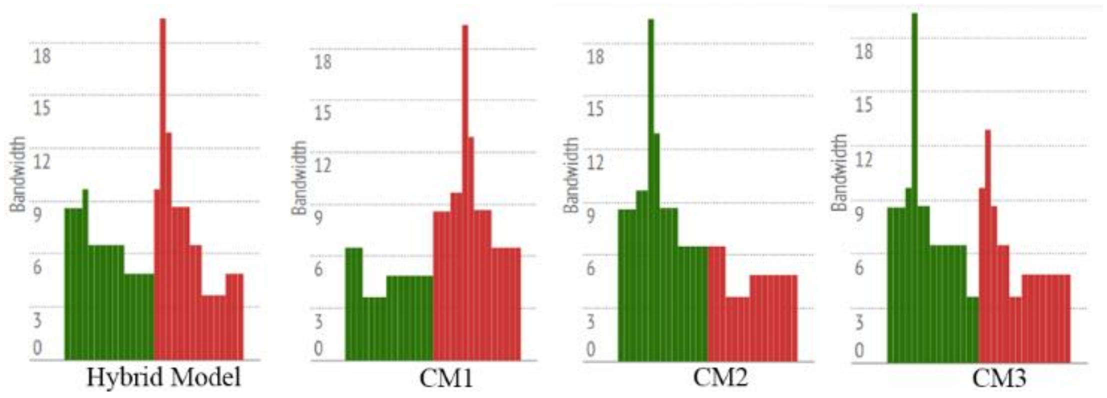


Figure 5-4 – Bandwidth Comparison for α = 0.5

As was stated earlier α = 0.5 means that average bandwidth utilization is aimed. Increasing the scaling parameter increases bandwidth utilization. We also observe that CM2 and CM3 achieve higher bandwidth values compared to our joint model and CM1. Although CM1 is known to aim an average level of bandwidth, we observe that our model achieves better bandwidth values compared to CM1.

Figure 5-5 shows the comparison of load imbalance level in different assignments obtained when different models are employed for resource allocation.

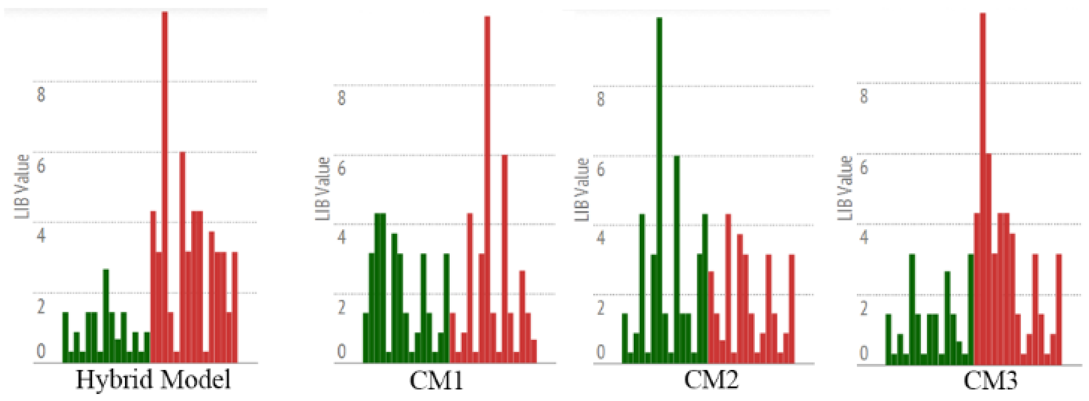


Figure 5-5 – Load Imbalance Level Comparison for α = 0.5

For α = 0.5, load imbalance level is considered much similar to bandwidth utilization. Although an average LIB level is preferred by our model, CM1 achived better average utilization levels for CPU, memory and bandwidth. We observe that CM2 attains higher load imbalance levels, which means highest CPU, memory and storage. CM3 uses an average load imbalance levels in return.

Figure 5-6 shows the comparison of power consumption in different assignments obtained when different models are employed for resource allocation.

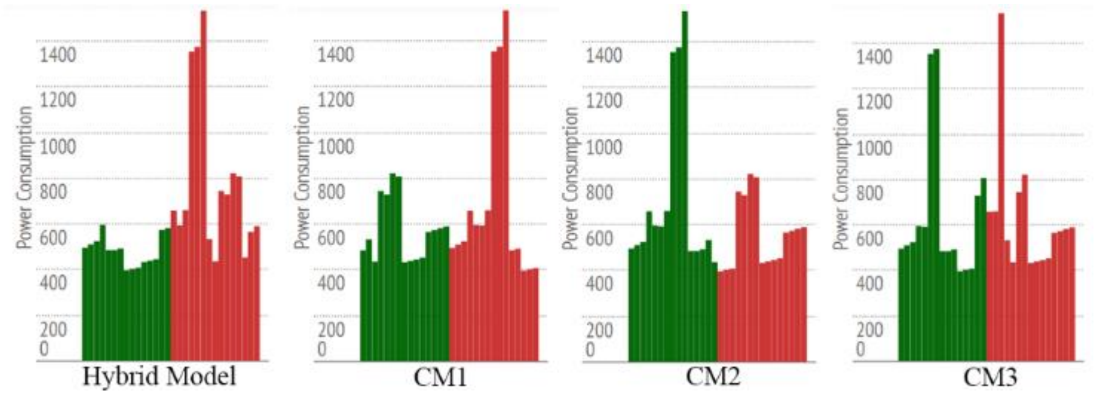


Figure 5-6 – Power Consumption Comparison for α = 0.5

We observe that by increasing α power consumption increases. Despite this increase, our joint model still selects those computers that consume less power. CM1 selects computers that consume an average level of power. Although CM3 aims to select computers with lower power consumption values, we observe that it is still not the lowest power consumption. CM2 does not care about power consumption anyway hence we observe that it is among the worst.

In addition to the above, we checked the metrics given in section 5.1.2. for a detailed comparison. Table 5-5 tabulates these comparison metrics for α = 0.5.

Table 5-5 - Comparison Metrics for Scaling Factor 0.5

	TBW	TPC	LIBR	BWR
Joint	98.83	7264.65	3.66	1.49
CM1	73.13	8635.14	2.02	1.41
CM2	135.84	10728.57	0.64	1.00
CM3	118.44	9651.40	1.26	0.93

In Table 5-5 we observe that in our joint model, total bandwidth usage is at an average level when compared to others. Total power consumption is still the lowest. However expecting average load imbalance level and and bandwidth rate, we observe the best. We observe that LIBR has the highest value, which means high priority clients have more powerful computers assigned to then the low priority clients. For BWR we also observe that high priority clients have more bandwidth then lower priority ones. CM1 consumes the lowest power and CM2 uses the highest bandwidth as expected. Because of not using client priorities as a parameter, we cannot conclude that any of the models consider client priorities better by analysing the available values.

5.1.7 RESULTS FOR α = 0.8 CASE

Scaling parameter α = 0.8 means that the provider prefers to select those computers that has lowest load imbalance level and bandwidth among its available virtual machine pool. In other words, the provider does not care about the power consumption to the same extent as load imbalance level and the bandwidth. This kind of selection might be useful for those providers that can create its own energy, or pay less cost to it than other expenditures.

Although other compositions are also tested, only results for 15 customers 30 VMs case is presented in the following.

Figure 5-7 shows the comparison of bandwidth use in different assignments obtained when different models are employed for resource allocation.

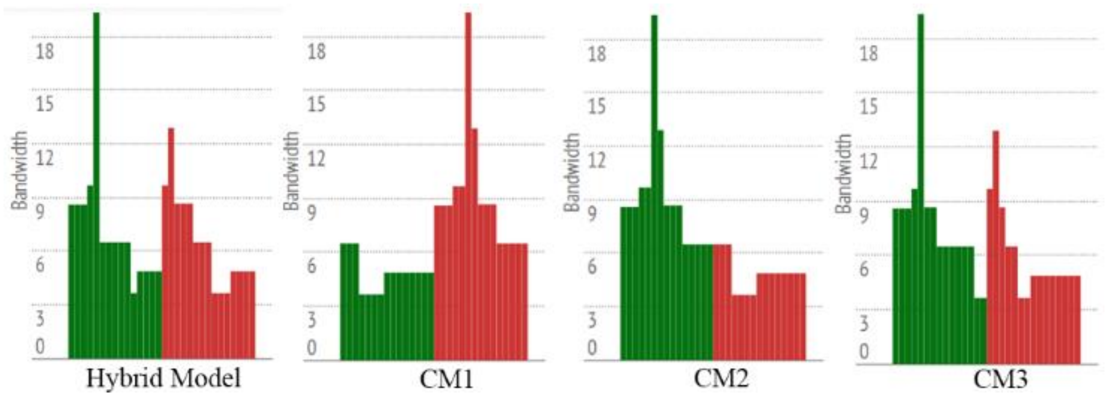


Figure 5-7 – Bandwidth Comparison for α = 0.8

As was stated earlier α = 0.8 means that highest bandwidth utilization is given. Increasing scaling factor increases bandwidth utilization. Although bandwidth utilization increases in all, we still observe that CM2 and CM3 achieve higher bandwidth values compared to our joint model and CM1. CM1 use lowest level of bandwidth.

Figure 5-8 shows the comparison of load imbalance level in different assignments obtained when different models are employed for resource allocation.

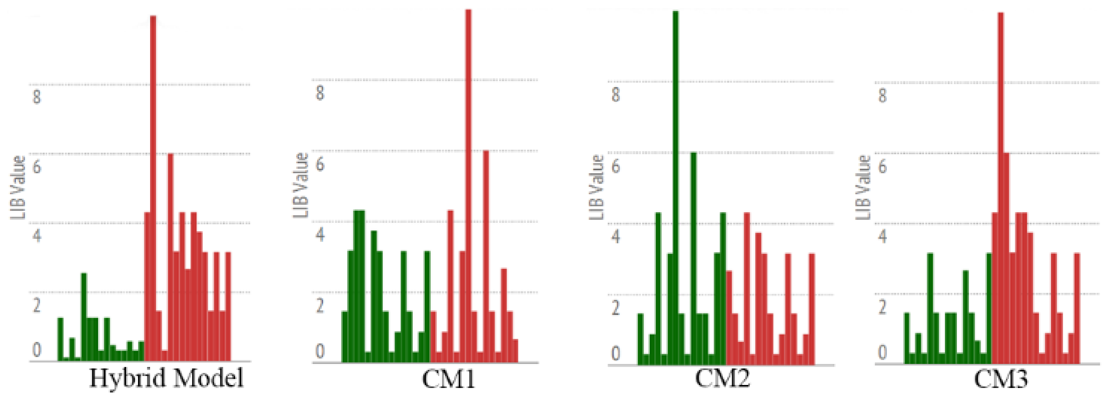


Figure 5-8 – Load Imbalance Level Comparison for α = 0.8

For α = 0.8, load imbalance level is considered as the lowest priority compared to other models. CM1 choose better average utilization levels for CPU, memory and bandwidth. CM2 attains higher load imbalance levels, which means highest CPU, memory and storage. CM3 uses an average load imbalance levels in return.

Figure 5-9 shows the comparison of power consumption in different assignments obtained when different models are employed for resource allocation.

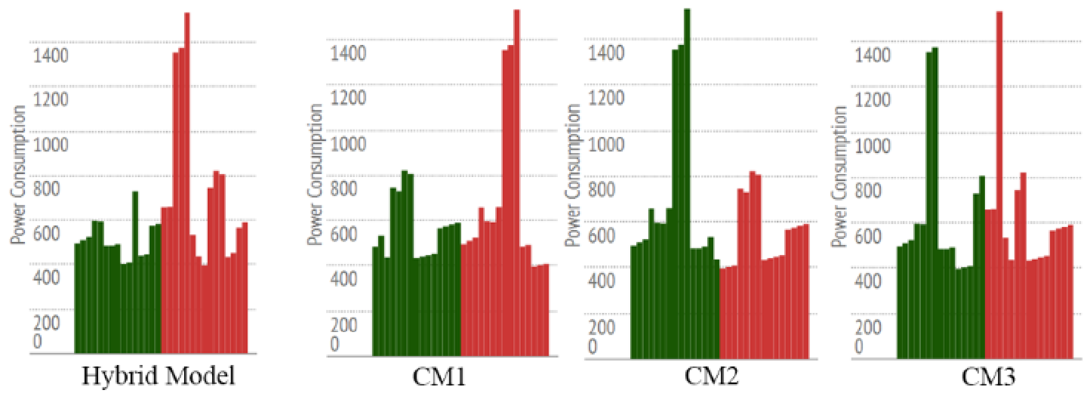


Figure 5-9 – Power Consumption Level Comparison for α = 0.8

We observe that increasing α increases power consumption. For this case, despite the increase, our joint model still selects computers that consume less power. CM1 selects computers that consume an average level of power. Although CM3 aims to select computers with lower power consumption values, we observe that it is still not the lowest power consumption values. CM2 does not care about power consumption anyway hence we observe that it is among the worst.

In addition to above, we checked the metrics given in section 5.1.2. for a detailed comparison. Table 5-6 tabulates these comparison metrics for α = 0.8.

Table 5-6 - Comparison Metrics for Scaling Factor 0.8

	TBW	TPC	LIBR	BWR
Joint	110.44	7757.39	4.99	1.57
CM1	73.13	8635.14	2.02	1.41
CM2	135.84	10728.57	0.63	1.00
CM3	118.44	9651.40	1.26	0.93

In Table 5-6 we observe that in our joint model, total bandwidth usage is at an average level when compared to others. However increasing α to 0.8, total power consumption is still the lowest and we observe the best load imbalance level and bandwidth rate. LIBR has the highest value, which means high priority clients have more powerful computers assigned to then the low priority clients. For BWR we also observe that high priority clients have more bandwidth then lower priority ones. CM1 consumes the lowest power and CM2 uses the highest bandwidth as expected. Because of not using client priorities as a parameter, we cannot conclude that any of the models consider client priorities better by analysing the available values.

We summarize our work by comparing scaling factors.

Figure 5-10 shows the comparison of bandwidth use in different scaling factors.

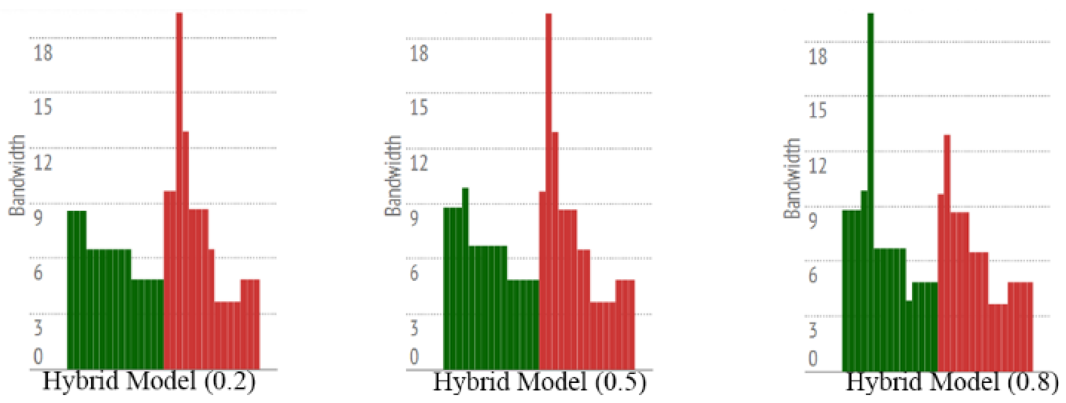


Figure 5-10 – Bandwidth Comparison of Scaling Factors

α = 0.2 means that bandwidth maximization is not given a high preference compared to minimization of power consumption. Compared the others, lowest bandwidth values is selected. We observe that average bandwidth utilization is given in α = 0.5. The highest utilization of bandwidth is observed when α = 0.8. Increasing α, increases the bandwidth utilization.

Figure 5-11 shows the comparison of load imbalance level in different scaling factors.

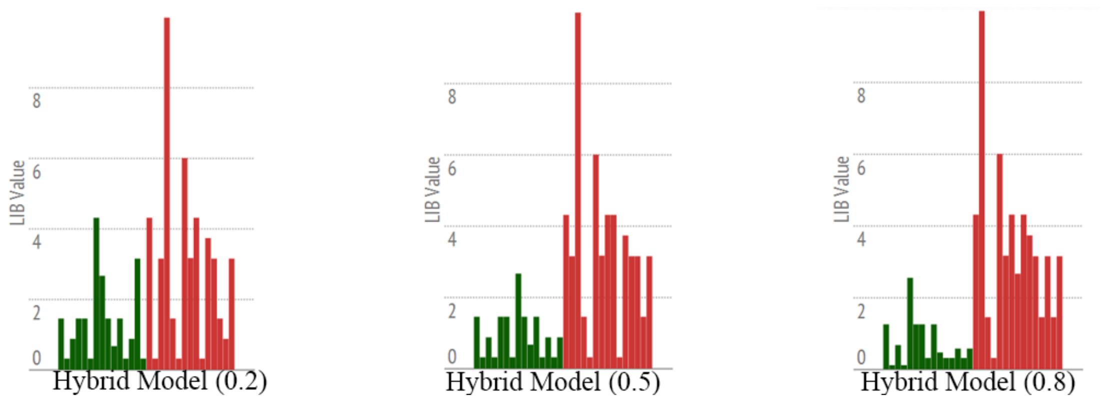


Figure 5-11 – Load Imbalance Level Comparison of Scaling Factors

For α = 0.2, load imbalance level is not considered much hence load imbalance level is selected depending on power consumptions of the computers. We observe highest load imbalance level compared the others. For α = 0.5, average load imbalance level is observed. For α = 0.8, load imbalance level is considered to the lowest one compared to other models. Increasing α decreases load imbalance level.

Figure 5-12 shows the comparison of power consumption in different scaling factors.

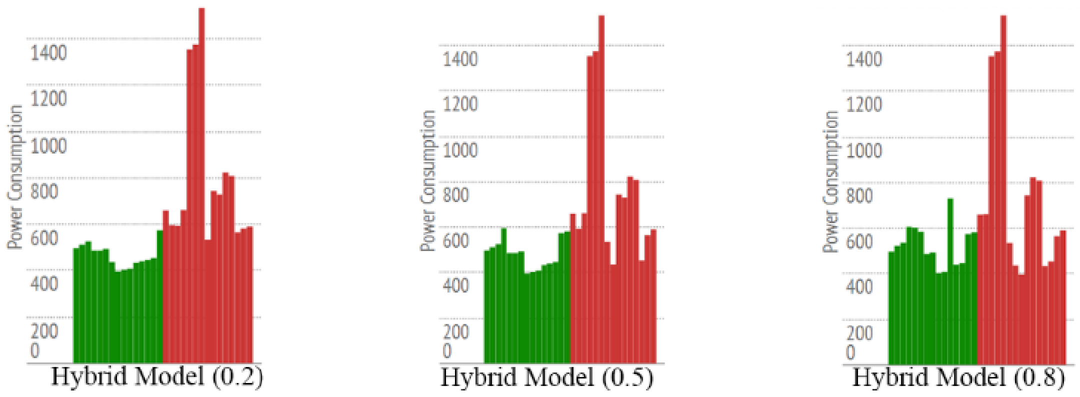


Figure 5-12 – Power Consumption Comparison of Scaling Factors

For α = 0.2, we observe that our joint model selects computers that consume least power. For α = 0.5, we observe an average level of power consumption. For α = 0.8, we observe the highest power consumption values. Increasing α increases power consumption.

In addition to above, we checked the metrics given in section 5.1.2 for a detailed comparison. Table 5-7 tabulates these comparison metrics for different α values.

Table 5-7- Comparison Metrics for Different Scaling Factors

	TBW	TPC	LIBR	BWR
α = 0.2	95.67	6976.19	2.08	1.59
α = 0.5	98.83	7264.65	3.66	1.48
α = 0.8	110.44	7757.39	4.99	1.57

In Table 5-7 we observe that in our joint model for α = 0.2, total power consumption is the lowest. For α = 0.5, average bandwidth values and load imbalance level is preferred. In additional to this, average power consumption is observed. We observe that in our joint model for α = 0.8, total bandwidth usage is increased when compared to others. We observe the best load imbalance level. However in this case, BWR is not the highest one. Because of not considering BWR as much as power consumption in α = 0.2, higher BWR is observer compared to BWR in α = 0.2.The same tests are also evaluated for other pools. In 98% of the tests, we observed that our joint model chose computers that consume less power compared to other models. We also observed that, in 90% of our tests, depending on the scaling factor, our proposed model has a higher load imbalance level ratio compared to other models. Besides these advantages, our model is compatible in its average bandwidth utilization when compared to other models existing in the literature.

CHAPTER 6

CONCLUSION

Cloud computing is a new trend in computing, which can provide a diversity of services to consumers over the Internet. In cloud computing, consumers may hire IT utilities such as infrastructure, software, application or databases for a limited amount of time.

In cloud computing, providers aim to utilize resources more effectively to provide economic benefits to consumers. To improve the utilization of servers, virtual machines are used. With virtualization, cloud data centers have the ability to serve more users than one real computer. However, depending on the actual demand, some jobs may be rejected due to over-crowding of virtual machines, which may result in business loss. To prevent such losses and to satisfy consumers' needs, cloud providers need effective resource management processes.

Several optimization models and algorithms have been proposed in the literature for effective resource management in cloud computing. The solutions can be implemented in hardware or software platforms. Some of these use server power consumption as an optimization criteria, some others use load imbalance level. Some of them use consumer priority and some of them use task priority. Consumer side factors such as consumer priority, bandwidth, quota, etc. are not considered in earlier works. Hence a joint optimization model, which can combine both the consumer and provider needs, is thought to have great potential to improve cloud computing.

In the present thesis work, we focus on proposing a joint data model and an effective allocation algorithm and aim to satisfy both consumer and providers' needs simultaneously. We first analyze the characteristics of real life cloud computing applications and identify consumer and provider requirements.

In our experiments, we use 10 real computers to create virtual machine pools. To get all kinds of solutions, we created different types of virtual machine pools. Our experimental analysis on the test virtual machines has shown that our proposed joint model has better results in terms of the defined comparison metrics. Because of having soft constraints in these models, we have examined our results from different perspectives such as power consumption, bandwidth, load imbalance level, etc.

In 98% of our tests, we observed that our joint model chose computers that consume less power compared to other models. We also observed that, in 90% of the tests, depending on the scaling factor, our model achieves a higher load imbalance level ratio compared to other models. Besides these advantages, our proposed model is compatible in its average bandwidth utilization when compared to other models existing in the literature.

Our joint optimization model and allocation algorithm selects computers that consume less power, and depending on the scaling factor considers a priority based load imbalance level selection and more or less an average level of bandwidth utilization.

A future direction of this work can be implementing a dynamic resource allocation system depending on change management in the client side. Changes may mean changing client priority, client quotas and other similar requirements. In addition, dynamic virtual machine client migration and reallocation can also be an important topic for the near future.

REFERENCES

[1] P. Mell, T. Grance, “The NIST definition of cloud computing”, SP 800-145, pp. 2-3, 2011.

[2] J. Rosenberg, A. Mateos, “The cloud at your service”, Manning Publications, 1st Ed., United States of America, pp. 10-12, 2011.

[3] S. Kumar, R.H. Goudar, “Cloud computing – research issues, challenges, architecture, platforms and applications: a survey”, *Int. J. of Future Computer and Communication*, Vol. 1, No. 4, pp.1-2, 2012.

[4] Search Unified Communications, http://searchunifiedcommunications.techtarget.com/definition/UCaaS-Unified-Communications-as-a-Service , last visited on August 2014.

[5] D. Reilly, C. Wren, T. Berry, “Cloud computing: pros and cons for computer forensic investigations”, *Int. J. Multimedia and Image Processing* (IJMIP), Vol. 1, Issue 1, pp. 32-33 2011.

[6] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Abkin, I. Stoica, M. Zaharia, “A view of cloud computing”, *Comm. of ACM*, Vol. 53, No. 4, pp. 53-58, 2010.

[7] S. Nanda, T. Chiueh, “A survey on virtualization technologies”, State University of New York, Stony Brook, Tech Report, pp. 1-10, 2005.

[8] D.G. Cattrysse, L. N. V. Wassenhove, “A survey of algorithms for the generalized assignment problem”, *European J. of Operational Research*, Vol. 60, Issue 3, pp. 1-7, 1992.

[9] D. W. Pentico, “Assignment problems: a golden anniversary survey”, *European J. of Operational Research*, Vol. 176, Issue 2, pp.775-776, 784-787, 2007.

[10] C. R. Reeves, *Modern Heuristic Techniques for Combinatorial Problems*, pp. 1-14, 243-282, 1993.

[11] D. P. Bertsekas, “Auction Algorithms”, *Encyclopedia of Optimization*, Kluwer, pp. 2-5, 2001.

[12] S. Janacek, "Modeling and approaching a cost transparent specific data center power consumption", *Proc. of the Int. Conf. on Energy Aware Computing*, pp. 2-4, 2012.

[13] T.R.G Nair, M. Vaidehi, "Efficient resource arbitration and allocation strategies in cloud computing through virtualization", in *Proc. of the IEEE Int. Conf. on Cloud Computing and Intelligence Systems* (CCIS 2011), pp. 398-400, 2011.

[14] W. Tian, Y. Zhao, Y. Zhong, M. Xu, C. Jing, "A dynamic and integrated load-balancing scheduling algorithm for cloud data centers", in *Proc. of the IEEE Int. Conf. on Cloud Computing and Intelligence Systems* (CCIS 2011), pp. 312-315, 2011.

[15] T. Wood, P. Shenoy, A. Venkataramani, "Black-box and gray-box strategies for virtual machine migration", in *Proc. of the 4th USENIX Conf. on Networked Systems Design & Implementation* (NSDI 2007), pp. 7-10, 2007.

[16] J. Baliga, R. W. A. Ayre, K. Hinton, R. s. Tucker, "Green cloud computing: balancing energy in processing, storage and transport", *Proc. of the IEEE,* Vol. 99 , Issue 1, pp. 158-160, 2011.

[17] P. Kumar, A. Verma, "Independent task scheduling in cloud computing by improved genetic algorithm", *Int. J. of Advanced Research in Computer Science and Software Engineering,* Vol. 2, Issue 5, pp. 112-113, 2012.

[18] I. Iyoob, "Cloud computing Operations Research", *Service Science,* Vol. 5, Issue 2, pp. 9-15, 2013.

[19] MSI Global, http://www.msi.com/power-supply-calculator/, last visited on August 2014.

[20] Free calculators and Converters, http://easycalculation.com/physics/electromagnetism/computer-power-supply.php, last visited on August 2014.

[21] Global broadband and mobile performance data compiled by Ookla, http://www.netindex.com/download/allcountries/#, last visited on August 2014.

[22] Kioskea, Online Community, http://en.kioskea.net/faq/31342-the-internet-bandwidth-in-the-countries-of-the-world, last visited on August 2014.

[23] A. Kansal, F. Zhao, N. Kothari, A. A. Bhattacharya, “Virtual machine power metering and provisioning”, in *Proc. of the 1st ACM Symp. on Cloud Computing* (SoCC '10), pp. 1-5, 2010.

[24] Oracle VM VirtualBox, https://www.virtualbox.org/, last visited on August 2014.

[25] K. Hwang, G.C. Fox, J.J. Dongarra, "Distributed and Cloud Computing", Morgan Kaufmann is an imprint of Elsevier, USA, pp. 130-140, 2014.

[26] T. Dillon, Chen Wu, E. Chang, “Cloud Computing: Issues and Challenges”, Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on, pp. 27-33, 2010.

[27] B.Sosinsky, “Defining Cloud Computing”, Cloud Computing Bible, Wiley Publishing Inc., Indianapolis, Indiana, 2010.

[28] T. Dokeroglu, A. Sert,, S. Cinar, , "Evolutionary Multiobjective Query Workload Optimization of Cloud Data Warehouses," The Scientific World Journal, vol. 2014, 2014

[29] T. Dokeroglu, S.A. Sert , M.S. Cinar and A. Cosar (2014), “Designing Cloud Data Warehouses using Multiobjective Evolutionary Algorithms”, International Conference on Agents and Artificial Intelligence (ICAART) Eseo, Angers, Loire Valley, France, 2014.

APPENDIX A

SAMPLE TEST RESULTS FOR DIFFERENT SCALING FACTORS

Table A-1 – Power Consumption Comparison of Literature Models For α = 0.2

5x30	Joint Model TPC (Watt)	CM1 TPC (Watt)	CM2 TPC(Watt)	CM3 TPC (Watt)
Pool 1	2249.28	3199.02	2994.62	3031.95
Pool 2	2264.15	3246.36	2864.47	2864.47
Pool 3	2228.74	3400.13	2864.47	3018.74
Pool 4	2186.51	3155.69	3002.66	2918.34
Pool 5	2253.03	3193.07	3014.49	2994.32
Pool 6	2243.90	3104.35	3039.35	2966.53
Pool 7	2255.78	3151.68	2909.20	2909.20
Pool 8	2217.50	3383.58	3087.77	3029.78
Pool 9	2191.94	3021.95	3070.36	2998.54
Pool 10	2243.85	3098.40	3090.88	3067.28
Pool 11	2172.04	3108.35	3231.08	3010.96
Pool 12	2139.00	3392.51	3418.45	3715.72
Pool 13	2163.27	3150.01	2862.29	2862.29
Pool 14	2172.46	3147.41	3291.38	3694.77
Pool 15	2139.00	3027.90	3179.75	2969.57
Pool 16	2148.45	3211.30	3312.53	3045.62
Pool 17	2142.75	3400.78	3505.27	3740.55
Pool 18	2176.19	3289.42	3303.90	3719.61
Pool 19	2172.45	3163.96	3361.34	3700.85
Pool 20	2124.19	3013.67	3246.92	3004.23
Pool 21	2098.24	2847.34	2569.34	2569.34
Pool 22	2245.19	2932.48	2563.76	2563.76
Pool 23	2310.01	3035.30	2574.01	2574.01
Pool 24	2302.52	3210.37	2572.46	2572.46
Pool 25	2367.54	3043.29	2570.90	2570.90

Table A-1 (continued)

Pool 26	2376.22	5800.28	2789.97	2789.97
Pool 27	2281.94	3090.91	2582.71	2582.71
Pool 28	2283.63	3352.80	2598.55	2598.55
Pool 29	2316.96	3274.26	2581.15	2581.15
Pool 30	2329.00	3089.45	2575.25	2575.25
Pool 31	2041.61	3192.56	2660.08	2660.08
Pool 32	2121.82	3480.86	2866.47	3644.53
Pool 33	2091.82	3619.66	2909.31	3830.16
Pool 34	2091.82	3534.80	2783.68	2717.36
Pool 35	2085.77	3651.98	2907.92	2782.97
Pool 36	2046.04	3767.48	2873.77	2740.13
Pool 37	2023.78	3308.06	2709.15	2709.15
Pool 38	2072.31	3774.22	2967.08	3875.21
Pool 39	2072.31	4001.84	2850.15	2797.03
Pool 40	2105.75	4001.84	2972.82	2850.13
Pool 41	2048.40	3318.38	2660.08	2660.08
Pool 42	2080.72	3569.30	2763.80	2763.80
Pool 43	2144.70	3778.22	2740.83	2740.83
Pool 44	2220.82	3375.85	2714.73	2714.73
Pool 45	2194.40	3431.47	2763.80	2763.80
Pool 46	2048.53	3694.75	2700.45	2700.45
Pool 47	2141.68	4129.76	2753.87	2753.87
Pool 48	2199.80	3635.05	2730.57	2730.57
Pool 49	2195.98	3517.87	2746.40	2746.41
Pool 50	2189.16	3626.77	2732.12	2732.13

Same tests are also done for $\alpha = 0.5$ and $\alpha = 0.8$ but only the above values are presented for illustration purposes.

Table A-2- Bandwidth Comparison of Literature Models For α = 0.5

15x30	Joint Model TBW(Mbps)	CM1 TBW (Mbps)	CM2 TBW(Mbps)	CM3 TBW(Mbps)
Pool 1	106.88	56.43	107.67	85.12
Pool 2	111.10	59.96	107.67	103.34
Pool 3	98.21	59.96	107.67	79.78
Pool 4	103.65	59.96	107.67	86.12
Pool 5	110.10	59.96	107.67	84.12
Pool 6	106.88	59.96	107.67	79.78
Pool 7	111.10	59.96	107.67	103.34
Pool 8	106.76	59.96	107.67	86.23
Pool 9	103.54	59.96	107.67	86.12
Pool 10	106.87	59.96	107.67	84.12
Pool 11	103.68	57.85	112.01	75.56
Pool 12	106.79	57.85	112.01	62.73
Pool 13	134.79	57.85	112.01	112.01
Pool 14	84.24	57.85	112.01	62.73
Pool 15	105.85	57.85	112.01	79.89
Pool 16	103.68	57.85	112.01	75.56
Pool 17	102.63	57.85	112.01	62.73
Pool 18	106.80	57.85	112.01	66.38
Pool 19	90.68	57.85	112.01	62.73
Pool 20	106.96	57.85	112.01	75.56
Pool 21	110.18	75.48	103.34	103.34
Pool 22	112.26	72.80	103.34	99.59
Pool 23	109.05	72.80	103.34	103.38
Pool 24	109.63	72.80	103.34	95.83

Table A-2 (continued)

Pool 25	111.76	72.80	103.34	95.83
Pool 26	110.25	72.80	103.34	94.67
Pool 27	112.85	72.80	103.34	103.38
Pool 28	112.88	72.80	103.34	95.83
Pool 29	112.88	72.80	103.34	95.83
Pool 30	109.65	72.80	103.34	95.83
Pool 31	97.29	73.13	135.84	135.84
Pool 32	90.32	73.13	135.84	106.46
Pool 33	90.32	73.13	129.34	114.72
Pool 34	98.83	73.13	135.84	118.44
Pool 35	98.83	73.13	135.84	118.44
Pool 36	113.29	73.13	135.84	118.44
Pool 37	97.29	73.13	135.84	135.84
Pool 38	95.11	73.13	135.84	114.72
Pool 39	98.83	73.13	135.84	118.44
Pool 40	108.50	73.13	135.84	118.44
Pool 41	112.18	79.35	112.01	112.01
Pool 42	118.75	79.35	112.01	112.01
Pool 43	118.68	79.41	112.01	116.38
Pool 44	113.25	79.35	112.01	116.38
Pool 45	115.47	79.35	103.34	116.38
Pool 46	117.75	85.85	112.01	112.01
Pool 47	120.85	79.41	112.01	116.38
Pool 48	115.47	79.35	112.01	116.38
Pool 49	115.47	79.35	112.01	116.38
Pool 50	112.25	79.35	112.01	116.38

Same tests are also done for α = 0.2 and α = 0.8 but only the above values are presented for illustration purposes.

Table A-3- Priority Based Load Imbalance Rate Comparison of Literature Models For α = 0.8

20x30	Joint Model LIBR	CM1 LIBR	CM2 LIBR	CM3 LIBR
Pool 1	1.99	1.52	2.71	1.16
Pool 2	3.44	1.44	2.44	1.57
Pool 3	3.62	2.01	3.20	0.78
Pool 4	3.14	1.52	2.65	1.95
Pool 5	1.78	1.59	1.67	1.76
Pool 6	2.77	1.48	3.25	1.29
Pool 7	2.85	0.82	2.26	1.33
Pool 8	2.95	3.13	4.02	0.82
Pool 9	4.04	1.90	2.22	1.31
Pool 10	1.82	1.44	1.81	1.38
Pool 11	1.22	1.70	6.00	4.52
Pool 12	1.47	1.08	5.12	1.00
Pool 13	3.74	1.53	1.58	1.41
Pool 14	2.56	2.38	2.27	1.59
Pool 15	4.31	0.80	2.62	3.29
Pool 16	1.21	2.24	6.98	3.16
Pool 17	1.99	1.03	6.73	0.49
Pool 18	2.94	4.16	2.60	1.13
Pool 19	2.37	2.17	4.48	2.16
Pool 20	2.25	1.73	4.73	6.70
Pool 21	1.50	1.02	1.50	1.50
Pool 22	2.67	2.60	3.06	1.23
Pool 23	2.81	0.60	0.90	2.48
Pool 24	1.93	1.35	1.08	1.69
Pool 25	7.99	1.90	1.27	1.22

Table A-3 (continued)

Pool 26	2.44	0.83	1.85	1.27
Pool 27	2.32	0.61	1.28	2.42
Pool 28	3.71	1.00	1.19	1.21
Pool 29	3.65	1.10	1.13	1.03
Pool 30	2.91	1.10	2.10	1.08
Pool 31	0.00	0.00	0.00	0.00
Pool 32	4.16	0.83	0.54	0.55
Pool 33	4.97	1.58	1.79	0.37
Pool 34	4.74	1.19	1.07	1.56
Pool 35	4.03	1.00	1.20	1.30
Pool 36	2.51	1.47	1.31	1.20
Pool 37	5.31	1.07	0.50	0.50
Pool 38	3.45	1.33	1.04	0.60
Pool 39	3.39	1.29	0.99	1.53
Pool 40	2.06	1.00	1.54	0.89
Pool 41	4.00	1.14	0.44	0.44
Pool 42	8.56	1.44	0.95	0.95
Pool 43	4.10	2.45	1.20	1.90
Pool 44	5.86	5.54	1.65	4.58
Pool 45	1.72	3.75	3.24	4.40
Pool 46	2.89	1.57	4.15	4.15
Pool 47	3.14	3.13	1.80	1.06
Pool 48	5.11	5.00	1.87	2.22
Pool 49	3.92	1.29	3.62	3.69
Pool 50	4.20	3.74	3.79	2.77

Same tests are also done for α = 0.2 and α = 0.5 but only the above values are presented for illustration purposes.

In addition to these, priority based bandwidth value tests are also performed for each virtual machine pool and each different scaling factor.