Task Consolidation Algorithm for Heterogeneous Cloud Computing

Reenu Deswal

(Roll No: 213CS1135)

Department of Computer Science and Engineering
National Institute of Technology, Rourkela
Rourkela-769 008, Odisha, India

May, 2015.

Task Consolidation Algorithm for Heterogeneous Cloud Computing

Thesis submitted in partial fulfillment

of the requirements for the degree of

Master of Technology

in

Computer Science and Engineering

by

Reenu Deswal

(Roll No: 213CS1135)

under the guidance of

Prof. Bibhudutta Sahoo

Department of Computer Science and Engineering
National Institute of Technology, Rourkela
Rourkela-769 008, Odisha, India

May, 2015.

Dedicated to my Parents and Siblings

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela - 769008, Odisha, India

CERTIFICATE

This is to certify that the work in the thesis entitled ***Task Consolidation Algorithm for Heterogeneous Cloud Computing*** by ***Reenu Deswal***, having roll number **213CS1135**, is a record of an original research work carried out by her under my supervision and guidance in partial fulfillment of the requirements for the award of the degree of ***Master of Technology*** in ***Computer Science and Engineering Department***. Neither this thesis nor any part of it has been submitted for any degree or academic award elsewhere.

Dr. Bibhudatta Sahoo
Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela-769008, Odisha, INDIA

Place: NIT Rourkela
Date:

Acknowledgment

Preeminent, I might want to express my true appreciation towards my supervisor Prof. Bibhudatta Sahoo, who has been the managing compel behind this work. I am most obligated to him for acquainting me with the field of Cloud Computing and giving me the chance to work under him. His undivided faith in this topic and ability to bring out the best of analytical and practical skills in people has been invaluable in tough periods. Without his priceless guidance and support it would not have been workable for me to complete this theory. His invaluable suggestions, constant encouragement and assistance were always encouraging for me in every aspect of my academic life. I am enormously obligated to him for his consistent consolation and priceless counsel in every part of my scholastic life. I think of it as my favorable luck to have got a chance to work with such a magnificent individual.

I wish to thank all faculty members and secretarial staff of the CSE Department for their sympathetic cooperation.

During my studies at N.I.T. Rourkela, I made many friends. I would like to thank them all, for all the great moments I had with them.

When I look back at my accomplishments in life, I can see a clear trace of my family's concerns and devotion everywhere. My dearest mother, whom I owe everything I have achieved and whatever I have become; my beloved father, for always believing in me and inspiring me to dream big even at the toughest moments of my life; and my brother and sister; who were always my silent support during all the hardships of this endeavor and beyond.

Reenu Deswal

May 25, 2015

Abstract

As the recent advancements are going in the field of computer technologies like network devices, hardware capacities and software applications, cloud computing has emerged as an important paradigm that provides scalable and dynamic virtual resources to the users through the internet. Energy consumed by modern computer systems, particularly by servers in a cloud has almost reached at an unacceptable level. Also the energy consumed due to under utilization of resources accounts almost 60% of the energy consumed at peak load. It has resulted into reduced system reliability, extremely large electricity bills and environmental concerns because of resulting carbon emission. Therefore there is a great need to optimize energy consumption. Methods like memory compression, request discrimination, task consolidation among virtual machines are developed to enhance resource utilization. Task consolidation problem has been addressed as an optimization problem in heterogeneous cloud computing environment. Task consolidation maps user service requests to appropriate resources in cloud computing environment. The resource allocation problem in cloud computing environment is NP- complete. This thesis presents resource allocation problem as LPP to optimize energy consumed by the computing resources in cloud computing environment.

We have used greedy algorithms to obtain sub-optimal solution for task consolidation problem. The performance of the task consolidation algorithm has been simulated with in-house simulator developed by us using Matlab. The simulation is carried out with three different arrival patterns and the result shows in favor of our proposed EATC(Energy Aware Task Consolidation) algorithm.

Keywords : Cloud computing, energy consumption, EATC, resource utilization, task heterogeneity, machine heterogeneity.

Contents

List of Figures

List of Tables

List of Acronyms

Acronym	Description
IDC	Internet Data Center
CSP	Cloud Service Provider
CSU	Cloud Service User
SaaS	Software as a Service
IaaS	Infrastructure as a Service
PaaS	Platform as a Service
VM	Virtual Machine
DC	Data Center
LPP	Linear Programming Problem
EATC	Energy Aware Task Consolidation
ICT	Information and Communication Technology
DCD	Dynamic Component Deactivation
DPS	Dynamic Performance Scaling
DVFS	Dynamic Voltage Frequency Scaling
HPC	High Performance Computing
ETC	Expected Time to Compute
CPU	Central Processing Unit
DRAM	Dynamic Random Access Memory
SLA	Service Level Agreement
CMOS	Complementary Metal-Oxide Semiconductor
API	Application Programmable Interface
SOAP	Service Oriented Access Protocol
GBEAS	Genetic Based Energy Aware Scheduler
EAA	Energy Aware Algorithm
PBSA	Priority Based Scheduling Algorithm
MESFA	Most Efficient Server First Algorithm
VLAN	Virtual Local Area Network
IT	Information Technology
QoS	Quality of Service
EWRR	Enhanced Weighted Round Robin
RT-VM	Real Time- Virtual Machine
SPF	Single Point of Failure

Chapter 1

Introduction

1.1 Introduction

As the recent advancements are going in the field of computer technologies like network devices, hardware capacities and software applications, Cloud computing has emerged as an important paradigm that provides scalable and dynamic virtual resources to the users through the internet. Cloud environment is a delivery model that delivered the on-demand computational resources to the applications running in data centers over internet according to pay-for-use basis. Some of the definitions given by well known people and organization in this area include:

1) According to Buyya et al. (8), (7)- " *A Cloud is a type of parallel and distributed system consisting of a collection of interconnected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements established through negotiation between the service provider and consumers"*.

2) According to the various researchers in (12),(9),(17)- " *The Cloud is a model for enabling service users to have convenient, ubiquitous and on-demand network access to a shared pool of configurable computing resources (e.g., servers, networks, services and applications), that can be rapidly provisioned and released with minimal management effort or service-provider interaction*.

The cloud computing exhibits several benefits such as reduced cost, security, reliability, scalability etc. The cloud is composed of several essential characteristics that include:

1) *On-Demand Self-Service*: A cloud service user can anytime access cloud computing capabilities, such as network storage, server time, collaboration and communication services whenever needed automatically without any human interaction with each service's CSP.

2) *Broad Network Access:* Broad network capabilities are available and accessed through standard mechanisms which is promoted by use of heterogeneous thick or thin client platforms (e.g., laptops and mobile phones).

3) *Resource pooling:* The computing resources of cloud service provider's are pooled to provide services to multiple users using a multi-tenant model, with different virtual physical and resources that are assigned dynamically and reassigned according to user demand. The resource examples include memory typically DRAM, storage typically on optical or hard disc drives, processing, virtual machines and network bandwidth.

4) *Rapid Elasticity:* Computing capabilities can be elastically and rapidly provisioned automatically in some cases for quick scaling out, and released rapidly for quickly scale in. The capabilities available to the cloud service user for provisioning are unlimited and can be purchased in required quantity at any point of time.

5) *Measured Service:* The Cloud system is made to automatically optimize and control the resource usage (e.g., processing, storage and bandwidth) by providing some abstraction level according to the service type (e.g., active user account number). Resource usage is controlled, monitored, and reported, providing transparency to both the cloud service provider and cloud service user of the utilized service.

1.1.1 Cloud Services

Cloud computing delivers various services to the consumers under the pay-as-you-go model.These services are delivered and consumed according to the demand at any time and are accessed thorough some network devices. The services include Software as a Service (SaaS), Infrastructure as a Service (IaaS) and Platform as a Service (PaaS). Figure 1.1 describes each of the service provided by cloud computing. Each of these services are defined as follows:

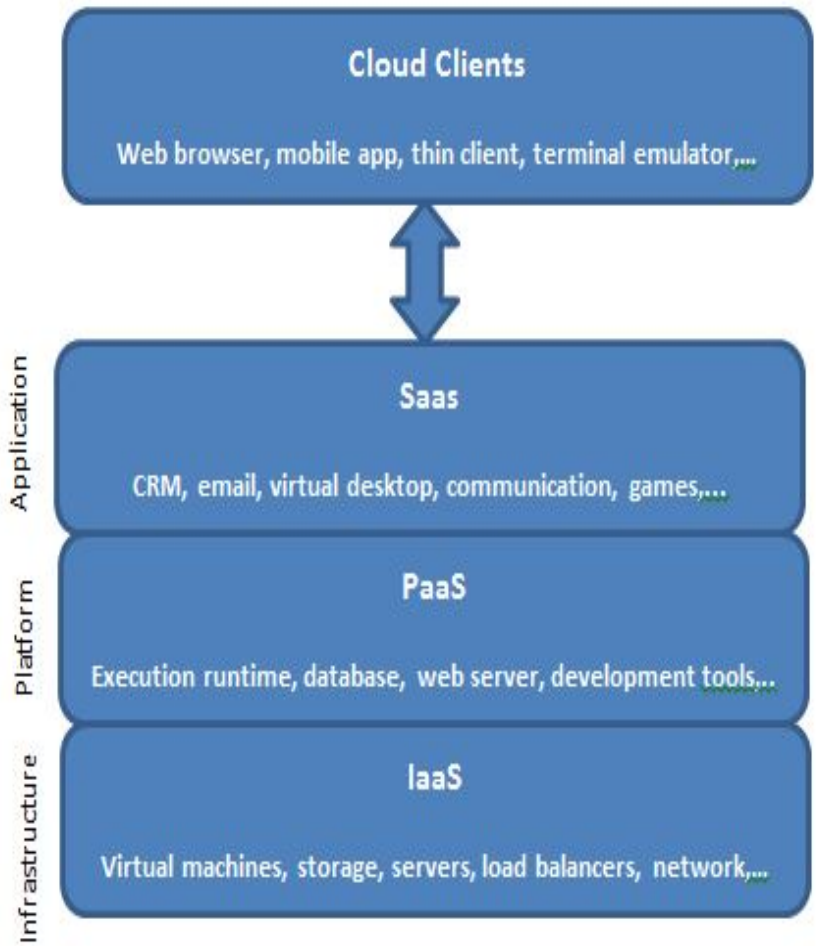

Figure 1.1: Cloud Service Models

1) *Software as a service(SaaS):* A cloud service category where the cloud service user is provided with the capability to make use of cloud service provider's applications which are running on a cloud infrastructure. All applications are similar in terms of their characterisitics to be of non-real-time and may be of different types, including business and IT applications, and also they may be accessed by different user devices.

2) *Infrastructure as a service (IaaS):* A cloud service category where the cloud service user is provided with the capability to make use of services like intra-cloud network connectivity, storage, processing (e.g. application acceleration, load balancer, firewall and VLAN), and other basic computing resources in the cloud infrastructure where the cloud service users can deploy and run their arbitrary applications.

3) *Platform as a service (PaaS):* A cloud service category where the cloud service user is provided with the capability deploy user-created or acquired applications onto the cloud infrastructure using platform tools supported by the cloud service provider. Platform tools may include tools for application development and programming languages, testing, storage, database development and interface development, .

Resources in cloud are widely distributed and aims to provide reliable Qos while meeting SLA.

SLA: It is an agreement between the consumer and the service provider regarding the technical performance promises that are made to the consumers. SLA includes the remedies for performance failures. The SLA is usually composed of three parts :

1) A collection of promises made to subscribers.

2) A collection of promises explicitly not made to subscribers, i.e., limitations.

3) A set of obligations that subscribers must accept.

Cloud Services may have different type of SLA. The main advantage of cloud environment is that it reduces the hardware cost and users can access high quality of services at a low cost.

1.1.2 Cloud Deployment Models

The cloud computing deployment models are basically classified into four categories i.e., public, private, hybrid and community. The figure 1.2 shows the various deployment models.

All the deployment models are basically the same except for the class of users they are designed.

Public Cloud Infrastructure is designed for the large organizations groups and it is also open for public use. Public and private cloud are almost the same with a substantial difference in the security concerns. This type of cloud is mainly owned by cloud service selling organizations.

Private Cloud Infrastructure is designed solely for an organization. It is managed either by a third party or by that organization itself and can be hosted either internally or externally.

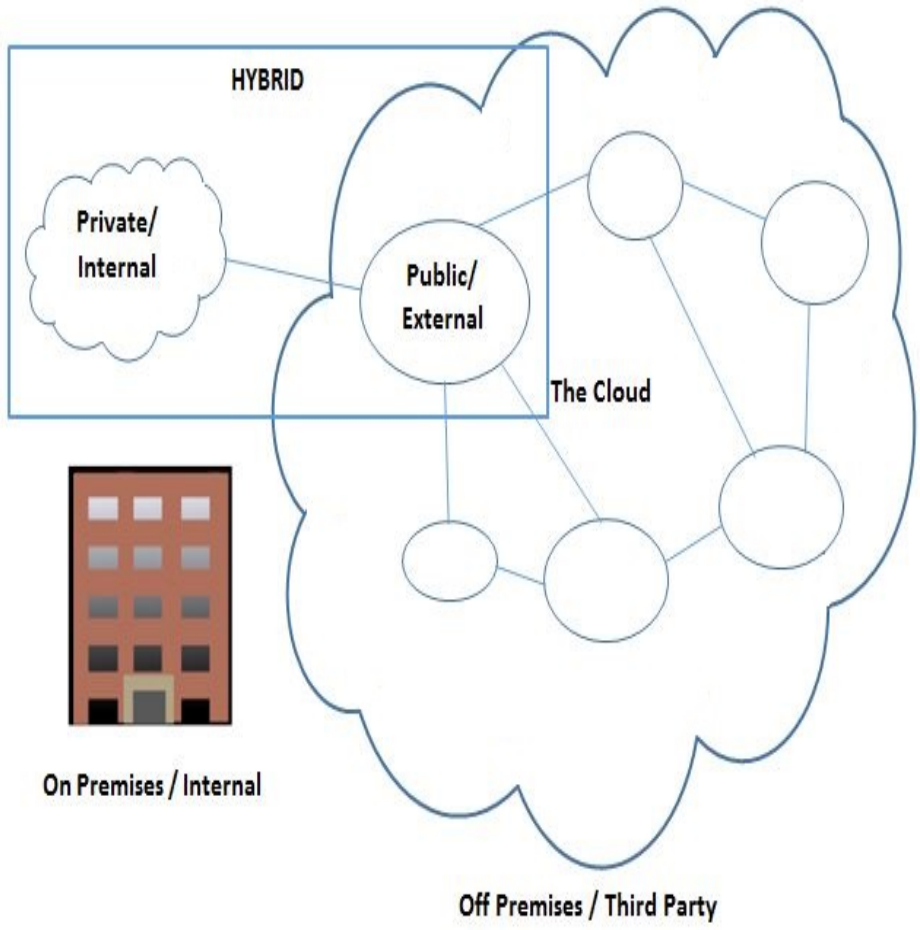

Figure 1.2: Cloud Deployment Models

Hybrid Cloud Infrastructure is designed by combining two or more clouds (public, community and private) that are bound together to offer the advantages of different deployment models.

Community Cloud Infrastructure is designed for sharing by several organizations from s specific community that are having a common concern. It is managed either by a third party or by that organization itself and can be hosted either internally or externally.

1.2 Energy Consumption in Data Centers

Nowadays power consumed by servers in a cloud has almost reached at an unacceptable level resulting onto a financial burden to the operating organizations, an environmental burden on society ans infrastructure burden on power utilities(18). According to (10),(43), around 2% of the global carbon is emitted by ICT itself. The increased use of Cloud com-

puting, representing the need to reduce carbon emissions and increasing energy costs calls for the energy-efficient technologies that can sustain Cloud data centers. Large Internet companies like Microsoft and Google have worked to significantly improve the energy savings of their multimegawatt DC's, focusing mostly on hardware aspects. According to(3),(43), the energy consumed by computers was around 2% of the total electricity consumption in US.

Among the major reasons of energy inefficiency, one is the idle power wastage. Even at very low utilization(10%) the energy consumed is 50-60% of the peak energy (23),(36),(31), (39),(1),(16). This has resulted into reduced system reliability, extremely large electricity bills and environmental issues generating due to emission of carbon in large quantity. Microsoft Dublin DC consumes 5.4 megawatts of electricity and may be expanded to 22.2 MW in the near future . The Tianhe-1, a cluster computer in Tianjin, China, consumes 128KW electricity per hour. This is equivalent to the electricity consumption of 2 million ordinary families(25). The key areas where energy consumption is maximum inside a DC involves various critical computational server that provides storage and CPU functionalities, power conversion units and cooling systems(25).

Power-aware technologies either make use of low power energy-efficient hardware equipments like power supplies and CPU for reducing the energy consumption and peak power consumption, or they try to reduce energy consumption by looking at the current application workloads and resource utilization. Since power density is in close relation with the temperature, the power factor is involved in the process of calculation of dynamic criticalities in power-aware allocation and scheduling. Power-aware scheduling process works at operating system, architectural, circuit, device, compiler and networking layers(37). The most direct and efficient method is to make use of more power efficient components during the hardware designing phase. Other alternatives have also been developed that include algorithms to scale down power or even shut down a system when not in use. In this context, the authors in(13),(5) have proposed a high-level taxonomy for the power management which is shown in figure 1.3.

According to recent research in (5),(35),(26),(27) , the major part of power consumption by a server is shared by the CPU then followed by the memory and then by losses occuring from power supply inefficiency. Also in (30) a data provided by Intel Labs, most of the power in a server is consumed by the CPU and then by the memory. The data in figure 1.4 (35) shows the power consumed by various components in a server. But as a result of continuous improvement in the application of various power saving techniques

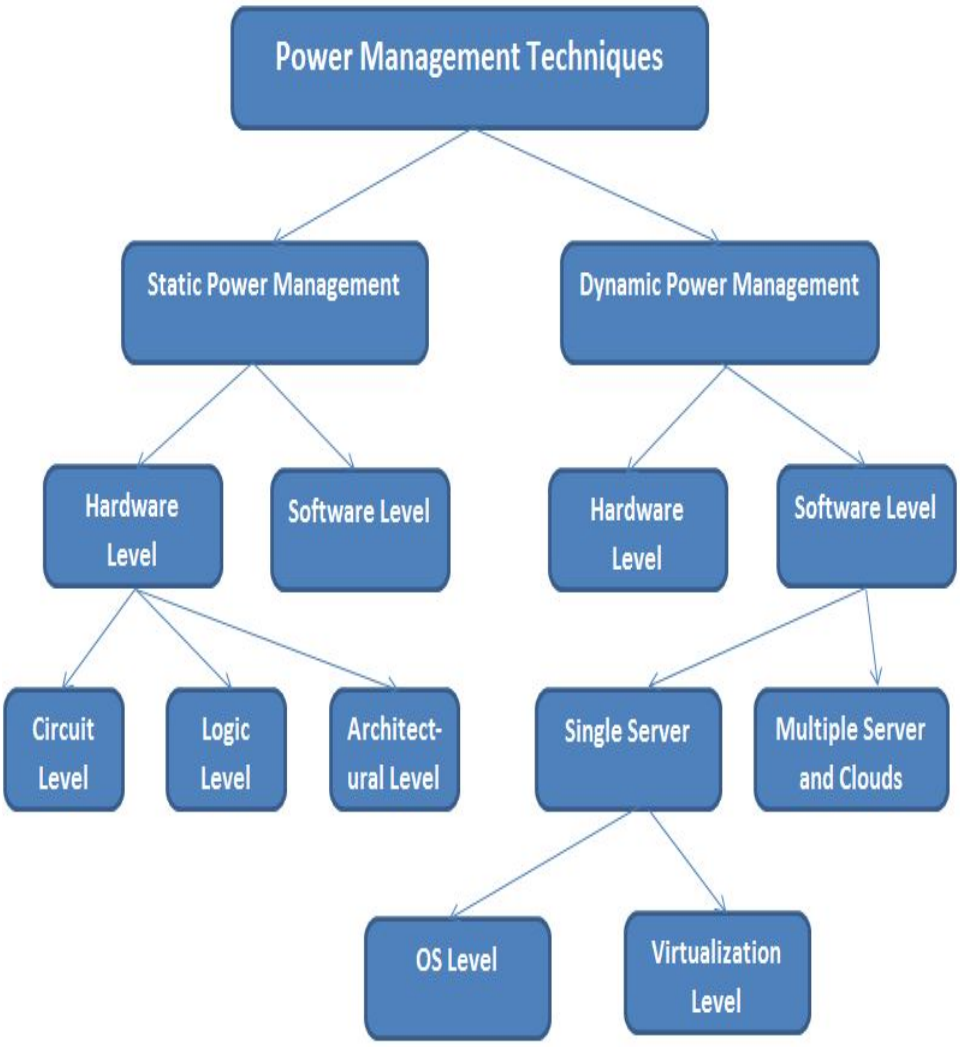

Figure 1.3: Taxonomy of Power Management Techniques

like DVFS that are capable of enabling active low-power modes and rapidly increasing CPU power efficiency, CPU is no longer dominating in the consumption of power by a server. Dynamic power range for all other server's components are much narrow i.e., less than 25% for disk drives, 50% for DRAM and 15% for network switches, and for all other components is negligible (6). The reason behind is that various active low-power modes are supported only by CPU, whereas all the other components can either be partially or completely switched off. Most of the techniques for power management mainly focuses only on the CPU; however, the constant increase in capacity and frequency of memory chips raises the cooling requirements along with the issue of high energy consumption. These are the reasons that makes memory one of the most important components of server that should be managed efficiently. New techniques and approaches for the the reduction of the memory power consumption have to be developed. The problem of low average resource utilization is equally applicable to the disk storage devices in any data center,

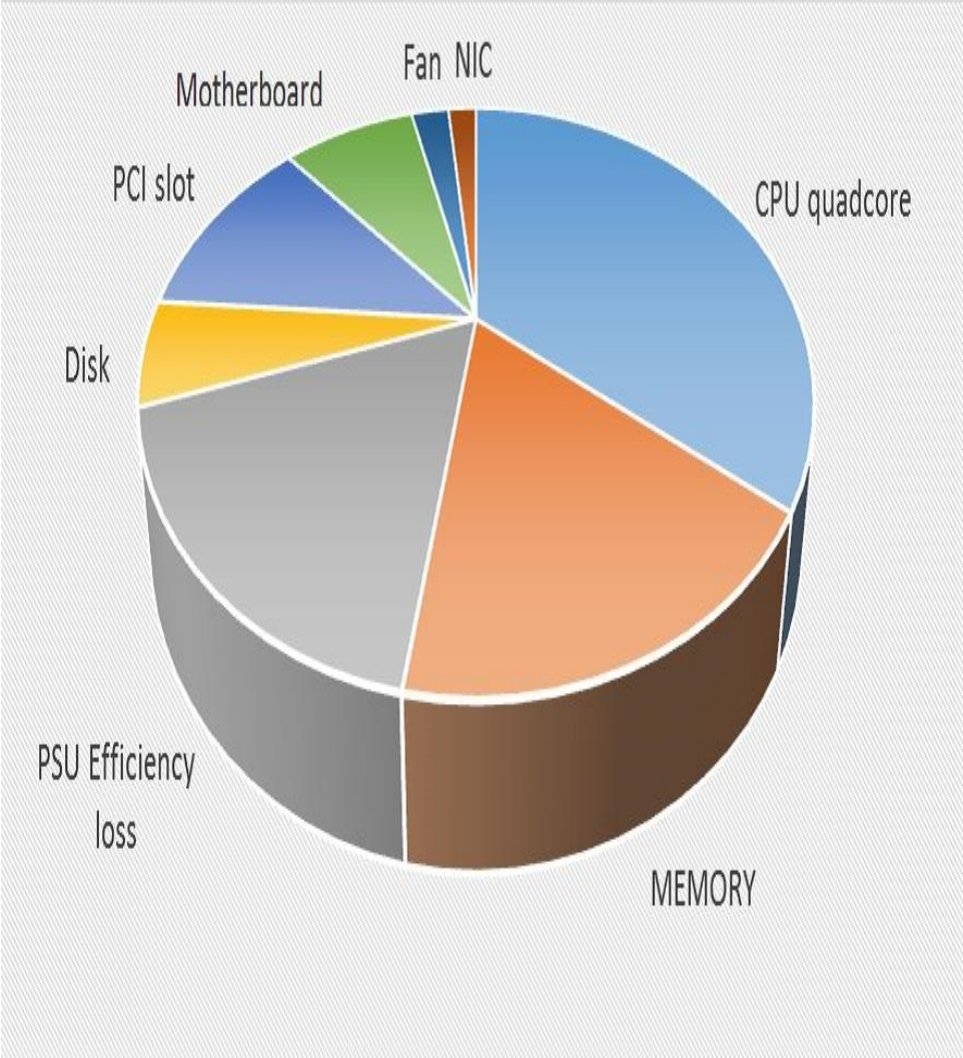

Figure 1.4: Power Consumed by various Server Components

especially when the disks are attached to servers.

However, this problem can be somewhat resolved by moving the disks to an external centralized storage array. To meet this problem effectively, policies should be used that will efficiently manage a storage system containing thousands of disks.

The problem of energy aware allocation of different virtualized resources(processor, database servers, ram, network etc) is complex because of the heterogeneous nature of workload application having different resource requirements. Different researchers have tried to address this problem with some degree of success. The service requests submitted by users at application layer of cloud framework are realized as tasks in the cloud environment. One of the major challenges for heterogeneous cloud is how to meet a huge number of heterogeneous tasks while providing the QoS guarantee. This creates another opportunity for the researchers to work aiming at power and energy usage and optimization of resource of servers hosted in the data centers.

1.3 Resource Allocation

Resource allocation problem is one of the major challenges in cloud computing because the consumers have unlimited access to the resources over internet anywhere and at any point of time. The cloud resources are not requested directly but with the help of SOAP/Restful web APIs that help in mapping the storage and computational request onto the virtualized resources. In a cloud, resources are managed using a resource manager which can be both centralized and distributed depending upon the size of cloud (i.e., number of physical servers). Since cloud computing model offers almost infinitely available resources, it is capable of supporting on-demand and elastic allocation of resources. But sometimes this may also lead to non-optimal allocation of resources.

In the cloud environment anything such as memory, CPU, storage,application and bandwidth can be termed as an ICT resource. For an energy efficient data center, it is very important to properly utilize the resources. The problem of resource allocation in cloud is a NP-complete problem (38),(22), hence no optimal solution can be found. The solution space is exponential and different heuristic algorithms have been developed to search the solution space and get a sub-optimal solution in acceptable amount of time. The problem of resource allocation is very complex and its complexity further increases as the cloud infrastructure size increases. Thus it also require certain assumption including set of tasks, set of operational servers, set of virtual machines , reduction in power and energy consumption.

To efficiently utilize the resources various efficient methods have been developed. Methods like memory compression, request discrimination, task consolidation among virtual machines are developed to enhance resource utilization (15). In response to the poor utilization of resources, *Task Consolidation* plays the role of an effective technique for maximizing utilization of resources. Maximizing resource utilization improves various benefits like IT service customization, rationalization of maintenance and reliable and QoS services.

The *Task consolidation problem sometimes also called as workload or server consolidation can be defined as the process of assigning a set N of n tasks to a set R of r cloud resources without violating SLA and aiming to minimize energy consumption.*

It allows running of multiple virtual servers inside a single physical server at the same time and is a strong approach for achieving energy efficient utilization of resources in any data center.

Recent studies demonstrates that vitality utilization of physical servers shifts directly with the resource usage. Task consolidation can also help in freeing up the resources sitting idle yet consuming huge power. This technique is faciliated by another technology called *virtualization* which provides necessary abstraction to the underlying hardware and allow the running of several tasks concurrently on a single physical server. In today's world, most of the IDC uses virtualization technique to create multiple instances of virtual machine on a physical server. This unique property of cloud plays an important role in task consolidation. It is the ability to create multiple instances of virtual machines dynamically on demand and has proved to be a popular solution to manage the resources of a physical server. Virtualization allows the running of multiple virtual machines on a single physical host thereby improving utilization of resources and also the running independency of user's applications is ensured. It facilitates the execution of several tasks concurrently on a single physical resource. Thus virtualization is a critical aspect of cloud computing and is equally important for both the providers and consumers. According to (29), virtualization plays the important role of:

1) Performance and Reliability by allowing applications to migrate from one platform to another.

2) Consolidation.

3) System security as it allows isolation of applications from each other running on the same hardware.

4) Performance isolation.

5) Ease of testing.

Figure 1.5 shows the concept of guest OS achieved using virtualization. It is clear from the figure that after virtualization, different user applications managed by their own operating system (guest OS) can run on the same hardware, independent of the host OS. Virtualization isolates the software from hardware and provide rapid software development and requires no or minimum physical hardware provisioning and thereby significantly reducing the time required for an application to run. For any incoming requests in the cloud, resources are allocated in 2 steps namely *VM Provisioning* and *Resource Provisioning* (34). First step is to create multiple instances of VM's on a physical server to host the incoming application requests. VM instances are created by matching to the specific

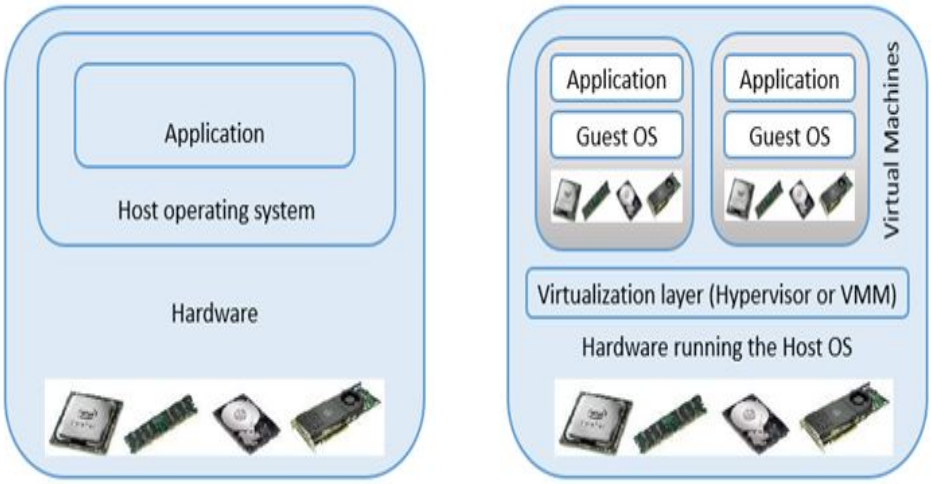

Figure 1.5: Computer Architecture without and with Virtualization

requirements of the request called as *VM provisioning*. The next step also called as *Resource provisioning* maps these incoming requests onto the distributed physical servers. In this work, we have focused on the *Task Consolidation Problem in Heterogeneous Cloud* using virtualization technique to reduce the total energy consumption. A considerable amount of research work has been done using various resource allocation and software approaches.

1.4 Related Work

Energy consumption is an important issue in heterogeneous cloud and has received more attention because of green computing in trend. The cloud service providers want their product to use less power to increase financial savings. Research results shows that CPU utilization greatly affects the energy consumption. Many methods have been developed to enhance the utilization of resources in cloud that include DVFS, request discrimination, memory compression defining a usage threshold value for resources, task scheduling among virtual machines. One of the key techniques for energy efficient resource allocation is task consolidation. This section describes various task consolidation algorithms developed by researchers. These algorithms vary in their resource allocation strategy, objective function, simulator used, resource used and the system model undertaken.

In (23) , the authors have presented two energy aware task consolidation algorithms (ECTC and MaxUtil) which aims to maximize utilization of resources and considers both

idle and active energy consumption into account. The author has considered homogeneous resources having similar capacity and computing capabilities. The algorithms tries to assign tasks on to the resources for which energy consumption is minimized without any degradation in performance. Energy model is built based on the utilization of resources, CPU being the only resource they have considered. Only CPU is considered as the resource because of the fact that energy consumption is directly proportional to resource utilization. Thus only processor utilization and processing time information about tasks is sufficient for measuring the energy consumed by that task. Task processing times are considered as hard deadlines and as the turn OFF/ON of a machine takes non negligible amount of time so idle resources are not considered in their study.
Both the algorithms almost follow similar steps except for difference in their cost functions. The results showed that regardless of migration policy, ECTC and MaxUtil outperformed random algorithm by 18% and 13% respectively.

The authors in (3) have designed an Enhanced First-fit Decreasing Algorithm integrated with virtual machine reuse strategy, DVFS technology and live migration to reduce energy consumption within a data center without violating SLA in terms of task execution deadline. The algorithm tries to control the best frequency depending on the CPU load. As the load increases, the frequency increase and so is the energy consumption. Thus depending on the task deadline, frequency is controlled and energy consumption in reduced. For every virtual machine falling below the minimum utilization, the virtual machine with least load that can handle this virtual machine is searched. All the running tasks are migrated onto that machine and the other virtual machine is shut down. CloudReport was used to simulate real cloud environment and the performance was compared with greedy and round robin algorithm. The results showed that proposed EWRR algorithm makes better utilization of resources by consolidating tasks onto a fewer nodes.

The authors in (20) has given the task communicational demands equal importance as that of computational demands. The author has developed an energy-efficient scheduler for cloud computing services with load balancing of traffic(e-STAB). The algorithm considers the traffic requirements of cloud applications and along with energy efficient job scheduling, it also provides load balancing for the incoming traffic in data center networks. e-STAB aims to balance the communication flows created among tasks also consolidate the workload on the minimum number of computing servers. The scheduler

is implemented in GreenCloud simulator and compared against green scheduler. The performance is measured in terms of time of job execution time, produced network load and impact on the energy consumed by the system. Both the schedulers almost shows same performance except for distribution of traffic load. e-STAB scheduler provides better traffic load balancing without compromising in energy consumption.

The work in (11) focuses on a batch mode algorithm with the objective of minimizing energy consumption in HCS. The system model consists of variably capable machines that are built with an effective mechanism for saving energy during the idle time slots. The energy consumption model has been derived from the power consumption model used in digital CMOS circuitry. The tasks are considered to be independent and indivisible workload and the computational model is taken as ETC model(2). The simulation work is carried using a set of randomly generated ETC matrices and the algorithm is compared with an existing algorithm min-min. Performance parameters considered for comparison are makespan, flow time and energy consumption and the results showed that the algorithm bahaves similar to min-min but with lower complexity.

The work (24) has presented an optimized task scheduling model for minimizing the task processing times and consumption of energy in the data enters for cloud computing. To minimize the energy expenditure of homogeneous tasks, the author has proposed the most-efficient-server first greedy task scheduling algorithm. The algorithm also provide a bound on the average waiting time of tasks and also minimizing number of active servers. The proposed algorithm is simulated in Matlab and performance is measured based on total energy consumed and average waiting time of tasks inside DC versus total number of active servers.

The author in (15) has presented an Energy Conscious Task Consolidation technique by restricting the CPU usage below a specified peak threshold. The cloud model undertaken is made up of several virtual clusters having the virtual machines limited in number. energy consumption is separated into two states: idle and running. Energy consumed by any virtual machine at any instant of time is computed based on its CPU utilization. The task consolidation strategy makes use of the best-fit technique for optimizing resources and has defined a 70% upper threshold for CPU utilization for allocating any virtual machine. Simulation results showed significant power saving of developed algorithm over recently developed greedy algorithm MaxUtil by 17%.

In a heterogeneous computing(HC) environment, diversely capable machines are harnessed together for executing a variety of tasks varying in their resource requirements. The degree to which the task execution time varies for a given physical machine is referred as task heterogeneity and the degree to which the machine execution time varies for a given task is referred as machine heterogeneity. From the work studied above, we observe that most of the research work is done assuming homogeneous systems. But in real applications systems greatly vary in terms of their resource capabilities. Also the service requests submitted by users vary greatly in terms of their computational and communicational complexities. To the best of our knowledge only a very few researchers have modeled both task heterogeneity and machine heterogeneity in their research. In a work (2) the author has described an ETC model to introduce heterogeneity in distributed Heterogeneous computing systems. Based on this, four categories of ETC matrix were proposed:

1) Low Machine Heterogeneity and Low Task Heterogeneity.

2) High Machine Heterogeneity and Low Task Heterogeneity.

3) Low Machine Heterogeneity and High Task Heterogeneity.

4) High Machine Heterogeneity and High Task Heterogeneity.

A coefficient-of-variation based method and a range based method to generate ETC matrix are discussed. In our work, we have used range based method to generate the ETC matrix. On the basis of above work done, the following observations can be made:

1) Most of the researchers have considered physical server considered as homogeneous in terms of their resource capabilities.

2) Most of the researchers have only considered CPU as the computing resource for calculating the total energy consumption by the cloud.

3) Computational tasks are given more importance neglecting the communicational(traffic) requirements of tasks.

4) SLA agreement is violated and the system performance is degraded in terms of waiting time, network delays, response time, makespan, throughput etc.

1.5 Research Motivation

There are various research issues in a cloud computing environment such as Virtual Machine Migration, Server Consolidation, Data Security, Energy Consumption etc. The rapid growth of cloud computing environment with the virtualized DC's has made serious issues including energy consumption, cooling infrastructures and air conditioning concerns in terms of increasing operational costs (34). Also the cooling and power rates are increasing eight times every year. A typical DC contains thousands of densely packed blade servers to better manage the efficiency and maximize the space utilization (42). So this rapid growth in server quantities,the energy consumption, which varies in direct proportion to the number of physical machines and their workload is coming as a great challenge. The energy consumption issue is gaining much importance because the energy consumption in data centers has reached at an unacceptable level creating financial as well as environmental burdens on the organizations and the society respectively. It has been found that average utilization if resources in a data center can be low as 20% (16). Even at very low load resources consume 50-60% of the peak power. Also the Data Centers emit substantial amount of CO_2 which contributes to greenhouse effect (4),(43). Typically a data center include hundreds or thousand of servers and other resources and with the rapid increase in Cloud computing technology, this size is getting huge expansion. Figure 1.6 shows the various consequences of rapid increase in energy consumption in DC's. This increase in energy results into high emission of carbon gases and high energy cost which further results into low profit for the CSP. High emission of carbon is not good for environment and low profit is not good for CSP, there is a great need of reducing the power consumption and proceed towards green computing. Hence from both perspectives i.e., environment and cloud provider, it calls for developing an energy efficient solution. Various methods are developed by researchers to meet this goal, task consolidation, VM consolidation and live migration of VM's are the some of the most crucial methods for achieving energy savings and load balancing (42). In this work, we have worked on the task consolidation problem in heterogeneous cloud for reducing the energy consumption in DC's. Some of the current researches in energy efficient resource allocation have identified the following key areas for optimizing energy consumption in a cloud infrastructure:

- *DCD* : When a computer component not supporting performance scaling is in idle state can also be deactivated (5). But such transition may load to delays, performance degradation but also extra power draw. Hence for better efficiency transition

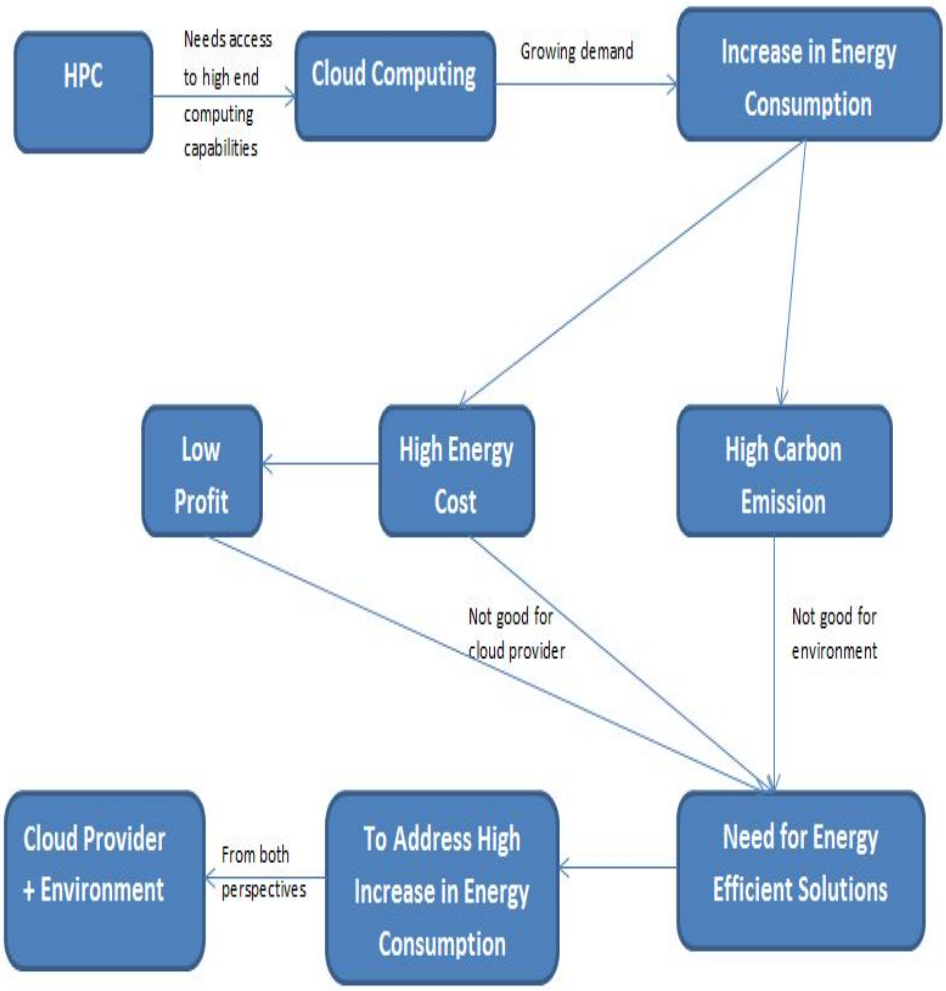

Figure 1.6: Green Cloud Computing

should be made only when idle period is sufficiently large to cover transition overhead.

- *DPS* : It includes various techniques that are applied to components of computer for supporting the dynamic adjustment in the performance level according to the consumption of power. For eg., CPU when not in complete use allows gradual changes in the clock frequency by adjusting voltage supply. One of the widely adopted technology called DVFS uses this idea (5).

- *Server Consolidation* : It allows running of multiple servers in a single host simultaneously to achieve energy savings in a DC.

- *Resource Scaling* : A task is assigned in a way that it uses the minimum number of resources at any point of time and also achieving the requirements mentioned in SLA.

1.6 Problem Statement

The energy consumed by the under-utilized resources inside the cloud environment has reached at an unacceptable level. The task consolidation problem in cloud can be presented as a minimization problem with the primary objective of minimizing the total energy consumed in executing a set of incoming tasks. A resource allocation strategy that maximizes the utilization of resources and in turn reduces the power consumption is required. The task Consolidation problem also known is the process of allocating a set of 'n' tasks to a set of 'r' resources without violating the QoS constraints . It aims to allocate resources to the task that explicitly or implicitly minimizes the energy consumption and also meeting the constraint specified in SLA. The problem is NP complete as there exist a large solution space which can be minimized by putting various performance constraints but still very large. In this thesis we have assumed a centralized cloud hosted in a DC that comprises of large number of diversely capable physical servers. Also the incoming tasks can vary greatly in terms of their computational and communicational requirements. The cloud infrastructure is modeled as set $H = \{h_1, h_2, ..., h_m\}$ where H represents the set of physical hosts. For each host h_i, Vm_i is the set of finite virtual machine $Vm_i = \{v_{1i}, v_{2i}, ..., v_{li}\}$. T =$\{\ t_1, t_2, ..., t_n\ \}$ describes the set of incoming tasks. Detailed description of the modeling is provided in chapter 3. In this thesis, the problem is addressed with the allocation of of tasks to the set of VM's so that total energy consumption is minimized.

1.7 Research Objective

Various resources(eg., CPU, RAM, disk, network bandwidth etc.) in cloud consume enormous amount of energy. Studies also shows that average utilization of resources in cloud is very low i.e., around 20% (16). In this thesis, an energy efficient approach has been proposed that makes use of task consolidation and virtualization technique to minimize the total energy consumption in the cloud. It also tries to allocate the tasks on the machines that will take minimum time for executing that task.

1.8 Research Contributions

The research contribution of *Task Consolidation Algorithm for Heterogeneous Cloud Computing* are summarized as follows:

1) Mathematical formulation for task consolidation problem and proposal of system and a workload model for the above defined problem.

2) Designing and analysis of energy efficient task consolidation algorithm using greedy approach.

1.9 Organization of Thesis

In this thesis, task consolidation problem in heterogeneous cloud has been addressed as an optimization problem with the main objective of minimizing the energy consumption. The thesis is being organized into five chapters. In this chapter, a brief introduction of cloud (services and deployment models), energy consumption in DC's, related work done, research motivation along with objective etc. are discussed. The rest part of this thesis is organized into the following chapters :

In **Chapter 2**, cloud computing architecture is discussed. The problem is defined and energy consumption model and system heterogeneity model are discussed.

In **Chapter 3**, workload model proposed by different researchers are discussed. Also our proposed system and workload model is defined. The problem is mathematically formulated and scheduling architecture is discussed.

In **Chapter 4**,we have proposed an energy aware task consolidation algorithm based on greedy approach. The simulation is carried out and performance is compared against two recently developed heuristics.

In **Chapter 5**, the conclusions are drawn from thesis and scope for future work is given.

Chapter 2

Heterogeneous Cloud Computing Architecture

2.1 Introduction

Cloud computing environment is designed to offer the on demand scalable services to the users over the internet through web browser or other devices.One of the vital features of cloud computing is to provide a desired level of QoS. QoS also called as Quality of Service can be defined using the term SLA that describes various characteristics like minimizing response time or latency, maximal throughput, makespan minimization etc. by the deployed system. To meet the growing demand for large volumes of data, DC's host high performance storage devices and computing servers. These servers consume the major part of energy in data centers. As a result, CSP's have to deal with energy performance trade-off of minimizing energy consumption while meeting QoS requirements. Energy usage in large scale computer systems like cloud may yield many other serious issues like carbon emission and system reliability. The recent advancement of the term green or sustainable computing is not limited to the main computing components (processors, storage device etc.), but it can be extended to a much larger range of resources associated with computing facilities including auxiliary equipments like water used for cooling and even physical floor space used by these resources. This calls for the development of various software energy saving techniques including scheduling and virtualization. In response to poor utilization of resources in a DC, task consolidation is an effective technique to increase resource utilization. This technique is enabled by virtualization that facilitates the running of several tasks on a single physical resource concurrently.

2.2 Cloud Computing Architecture

In this section, we have demonstrated the cloud computing architecture with the help of figure 2.1. The cloud can be distributed among various geographical locations but in our work we have taken cloud confined at a single location. The Cloud computing model consists of fully interconnected set of resources. These resources can be physical machines, database servers, network devices etc. The physical machine or host represents a physical computing node in the cloud with pre-configured resources like CPU, memory, storage, network latency, etc. In our system model, we have taken heterogeneous physical servers that vary greatly in their computing capabilities. As shown in the figure , the top layer represents the consumers. The consumers can be either service brokers or the users that submit their service requests at the application layer. The requests submitted are treated as tasks in the cloud during scheduling. So a task is defined as an independent service request made by the user with certain resource requirements and other QoS parameters depending upon the type of service desired. In our model, we have considered that tasks are arriving dynamically into the system. The tasks then waits into a global queue before they are allocated resources. After all tasks are arrived, next work is performed by Service Scheduler. It is also a physical node and it assigns service requests to virtual machines and determines resource entitlements for allocated virtual machines. Virtualization technology creates multiple virtual machines on the physical machine. It also allow the task to be assigned to any virtual machine meeting its resource requirements. Even when no task has arrived, i.e., the machines are sitting idle, the machines still consumes the energy. The decision of adding or removing virtual machines according to demand is also taken by scheduler. The scheduler can be both centralized or distributed depending upon the size of cloud. Here we have taken a centralized service scheduler. Finally if a task is meeting all its requirements it is assigned else it is rejected. Virtual machine is the basic unit to execute a task. Virtual Machine(VM) Manager Keeps track of the availability of virtual machines and their resource entitlements. It also handles the migration of virtual machines across physical machines.

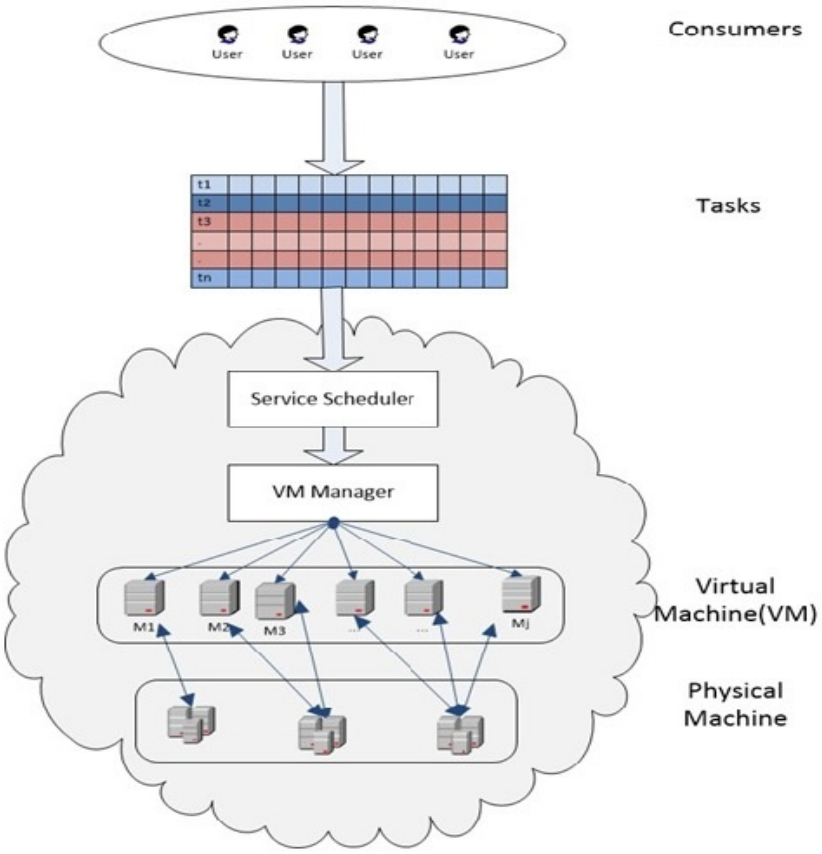

Figure 2.1: Cloud Framework

2.3 Energy Consumption Model in Cloud

According to (14), (23),(33) the total energy consumption of a virtual machine varies with the CPU utilization. These authors have considered only utilization of CPU to compute the total energy consumed in a cloud. But in actual scenario, other resources like Ram ,disk etc. also have an impact on the energy consumption and cannot be neglected. The author in (14) said that higher CPU utilization does not mean utilizing energy efficiently, and hence concluded in order to save energy, CPU should not be exhausted above a peak threshold level. Based on the various research conducted in this field, a 70% utilization is considered as the appropriate threshold value. Also in (23), the author have devised the energy consumption of a particular task based only on processor utilization and its processing time. Since the overheads of turning off and on cannot be neglected, they have considered the idle resources in their study. Different researchers have adopted different energy models based on their requirements. The energy model that we have adopted in our work is derived from (15). According to this model the energy consumption of any

virtual machine is separated in two states: Idle state and Running state. The running state is further divided into six different levels based on the mean utilization of resources. To compute energy the author has only considered CPU, but in our work we are dealing with other resources also(RAM, disk). So we are computing energy based on the mean utilization of all the resources. The different levels of energy consumption are described as below:

1) α if idle

2) $\beta + \alpha$ if $0\%<$ utilization $\leq 20\%$

3) $3\beta + \alpha$ if $20\%<$ utilization $\leq 50\%$

4) $5\beta + \alpha$ if $50\%<$ utilization $\leq 70\%$

5) $8\beta + \alpha$ if $70\%<$ utilization $\leq 80\%$

6) $11\beta + \alpha$ if $80\%<$ utilization $\leq 90\%$

7) $12\beta + \alpha$ if $90\%<$ utilization $\leq 100\%$

The energy is computed in unit of Joules. We have taken an important assumption regarding the α and β value for different servers. As we have considered that data center uses heterogeneous servers so these value may be different them. The value for α and β mainly depends on the hardware architecture of physical servers and may vary on different cloud systems. Also to achieve better load balancing in our system, we have assumed that each server receives the traffic with almost same rate.

2.4 Task Consolidation Problem

The task Consolidation problem also known is the process of allocating a set N of 'n' tasks to a set R of 'r' resources without violating the QoS constraints. It aims to allocate resources to the task that maximize utilization of resources, explicitly or implicitly minimizes the energy consumption and also meeting the constraint specified in SLA. Let T be a set of n tasks and H be a set of m physical hosts. Assign the n tasks to the m resources with the aim of maximizing resource utilization and minimize energy consumption. The problem can be modeled as a multi-dimensional bin-packing problem where physical servers are bins with each resource (CPU, disk, network, etc) being one dimension of the

bin. Each of the incoming task with known resource utilization can be treated as an object with given size in each dimension. Minimizing the number of bins should minimize the idle power wastage. However, that is not true in general, causing the energy aware task consolidation problem to differ from traditional vector bin packing. The task consolidation problem is a NP-Complete problem. The problem is a multi-objective problem and the objectives are as follows:

1) Minimize energy consumption

2) Maximize resource utilization

3) Makespan minimization

4) Load balancing

5) Guarantee QoS

6) Enhance throughput

8 Robustness

9 Scalability

In my research work, I have taken minimization of energy consumption as the primary objective. The designed heuristic also tries to minimize the makespan i.e., total execution time required by all the tasks. Time and resource requirements should also be met by the task consolidation strategy; i.e., the resources assigned to a task should be enough to meet the resource requirements of that particular task.

2.5 Current State of Work

The researchers have developed various task consolidation algorithms that vary greatly in different parameters.These parameters can be the approach used for developing eg., greedy or genetic or some other approach,in terms of the resources they have considered, objective functions, resources considered, simulator used and SLA parameter used for performance evaluation. I have surveyed some of the recently developed task consolidation algorithms. The comparison is shown in table 2.1.the blank field indicates that the required information is not discussed in that paper.

Table 2.1: Task Consolidation Approaches

Approach	Resource Utilization	Energy Minimization	SLA	Resources	Simulator
ECTC (23)	No	Yes	No	CPU	
MaxUtil (23)	Yes	Yes	No	CPU	
EWRR (3)	Yes	Yes	Execution Deadline	CPU	Cloud Report
EAA (36)	Yes	Yes	No	CPU, Disk	
e-STAB (20)	No	Yes	Load Balancing	CPU, Network Bandwidth	Green Cloud
GBEAS (21)	No	Yes	Makespan	CPU	HyperSim-G
GBEAS (11)	No	Yes	Makespan	CPU	
PBSA (32)	Yes	Yes	Makespan	CPU, Memory	
MESFA (24)	No	Yes	Makespan, average waiting time	CPU	Matlab

2.6 Modeling Heterogeneity

In (23),(24), the researchers have considered homogeneous computing servers for task consolidation problem. By using the homogeneous systems, it becomes a bit easier to allocate the tasks and compute the energy consumed. But in real time scenario, cloud mainly consist of heterogeneous communication and computing resources that may include heterogeneous communication interconnections, heterogeneous memories and heterogeneous processors. In any heterogeneous cloud, machines vary greatly in terms of their computational and communicational capabilities. Moreover the service requests (or tasks) submitted by user are also not similar in terms of their size, complexity and other requirements. As the number of heterogeneous processors increases in cloud, the same task can be performed by different processor with consuming different amount of energy. So the task consolidation problem become more complex and efficient heuristics are needed that map these requests to the machines. For this purpose ETC(Expected Time to Compute) model has been followed by (2), to express the heterogeneous nature of the incoming tasks in terms of their running times and also among the machines in the cloud. In ETC matrix, the entry (i,j) indicates the expected execution time of task *i* on the

machine *j*. For all the tasks that are expected to arrive during a given period of time, ETC matrix contains their expected execution time on every machine. Each column of ETC matrix gives the expected execution time of different tasks on a single machine which is also called as task heterogeneity. Similarly each column shows the expected execution time of a given task on different machines which is also called as machine heterogeneity. The table 2.2 represents the suggested values for T_Hetro and m_Hetro (2) where T_Hetro represents the task heterogeneity and M_Hetro represent the machine heterogeneity. Figure

Table 2.2: Suggested values for T_Hetro and M_Hetro

	Low	High
T_Hetro	10	10^5
M_Hetro	10	10^2

2.2 shows an ETC matrix generated for 12 tasks and 7 machines using Matlab. The values represent the expected execution time in milliseconds.

2.7 Conclusion

In this chapter, we have studied the cloud computing architecture. We have also formulated the task consolidation as a multi-objective problem and heuristics are required to solve the problem. The energy consumption model and heterogeneity model are also discussed in detail here. In next chapters, we have proposed a system and workload model to deal with the task consolidation problem. Also the solution to task consolidation problem is developed using greedy approach.

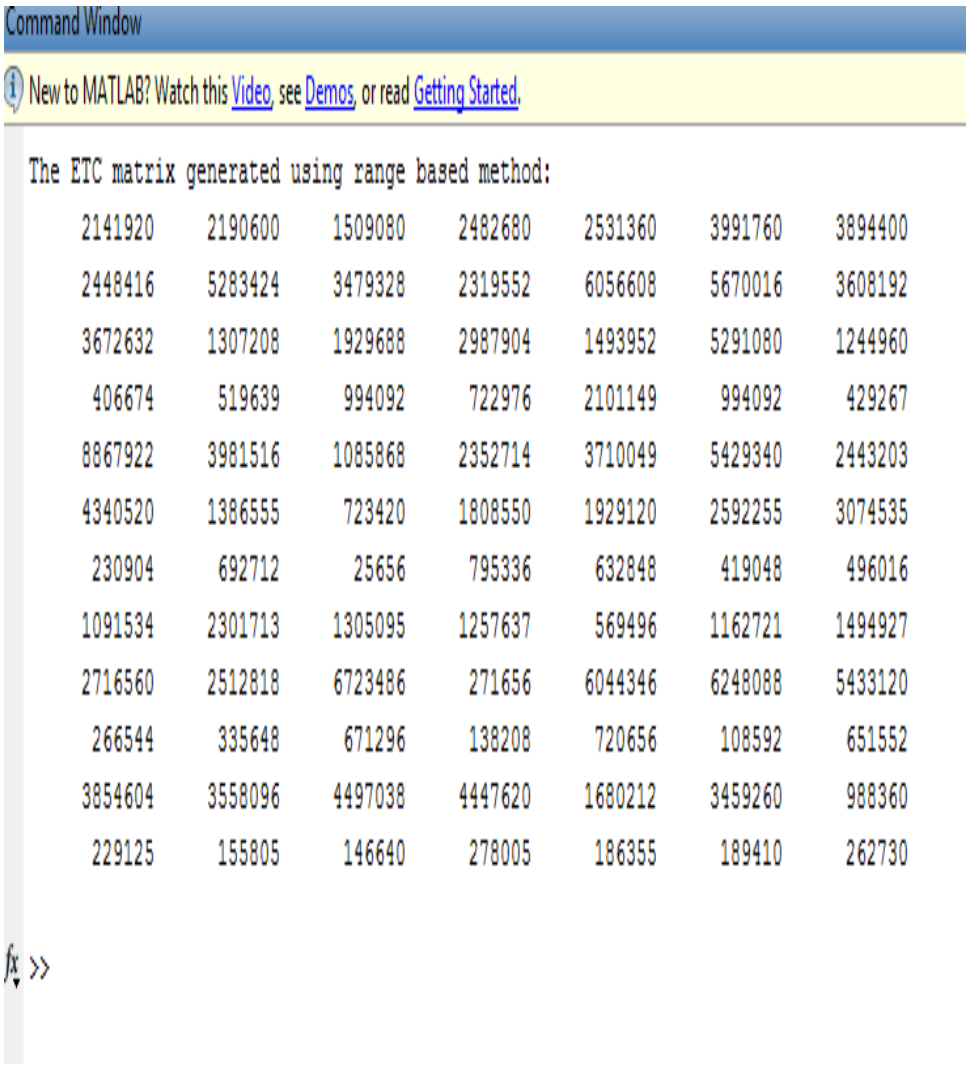

Figure 2.2: ETC Matrix [12*7]

Chapter 3

Task Consolidation Problem in Heterogeneous Cloud Computing

3.1 Introduction

As the workload submitted by a user may vary greatly in terms of their complexity, resource requirements and other parameters. Thus the workload model should be flexible enough in defining the task requirements. The user may submit the workload in form of multiple independent jobs or time and precedence constrained jobs. Each job is further divided into a number of tasks that can be dependent on each other. Independent tasks can be executed concurrently leading to faster execution of the submitted job while as for precedence-constrained task, execution takes place on the basis of DAG. The tasks can be preemptive as well as non-preemptive . A single task undergo various phases from the time of its arrival and until it gets completely executed.

A number of researchers have proposed a number of workload models to deal with the huge number of application requests. The author in (41) has described heterogeneous multiprocessor system model and task scheduling system model for their proposed algorithm. The architectural model describes a set of connected heterogeneous processors. Each processor is tightly coupled with its local memory which are different from one other in terms of their capacity, energy consumption, access time, access concurrency etc. To model the incoming task requests, memory-access data flow graph(MDFG) is used. Tasks are represented using DAG, where a DAG is a node-weighted graph represented by G = (*V*, *E*, *D*, *in*, *out*, *ET*, *EE*).

$V = v_1, v_2, ..., v_n$ is a set of task nodes, $E \subseteq V \times V$describes the set of edges for prece-

dence constraints among tasks in V. D denotes the set of data, *in*(v_i) $\subseteq$ *D* is set of input data for task v_i and *out*(v_i) $\subseteq$ *D* is set of output data for task v_i. $ET(v_i)$ represents the execution time of task on different processors while $EE(v_i)$ is used to represent energy consumption by that task on different processors.

Another model is followed by (19), where each real-time service is analyzed as group of multiple tasks. The author has defined the real-time service by $\tau_i(f_i, p_i, d_i, c_i, r_i) \mid i = 1, ..., n$.

Each task is described by following parameters: f_i as the finish time, p_i specifies the periodicity of task, d_i defining the relative deadline, c_i is the execution time in worst case and r_i is the time at which task is released. These real- time tasks are realized using virtualization technology. RTVM is used to describe requirements of a virtual machine. Every VM V_i is represented using d_i describing its lifetime, m_i describing the MIPS rate of based virtual machine and u_i, describing utilization of real-time applications respectively.

The author in (15) has taken a cloud system containing several VC's. Every cluster contain some limited number of VM's considering CPU utilization as the only resource for a VM availability and allocation. Each cluster maintains its own queue for task submission and the queue contain all the required information of the tasks. This information includes CPU utilization, processing time, arrival time, task ID and data size required by that task. When a VC is not able to fulfill a task requirement, task is migrated to other clusters for execution consuming some amount of network bandwidth and some other overheads.

In (23), the author have assumed that various resources in cloud are homogeneous in terms of their capacity and computing capabilities. A fully interconnected network is taken for direct communication between resources. For any incoming task the information about its processor utilization and processing time is considered sufficient for computing the energy consumption.

3.2 System Model

The research work lacked in terms of a proper defined system model. The system model gives an idea about the nature of physical hosts, their resource capabilities, inter-connection among them. So here we defines a proper system model which includes host model and virtual machine model. It is a generalized model and can be used in different scenarios depending upon the application requirements. The host contains all the physical resources required for task implementation in including storage resources, computational resources,

network resources and some other hardware devices. The model is described as below:

3.2.1 Host Model

The set $H = \{h_1, h_2, ..., h_m\}$ is defined as the set of physical hosts such that $\mid H \mid = m$. In this set, each $h_j, j \in [1, m]$ indicates host j, such that $h_j = \{hId, hTRes, hFRes , hTask\_set, h\_Vm\}$. Each attribute of set is defined as follows:

1) hId is the host identification.

2) $hTRes$ is total resource capability of a host, $hTRes = \{hTR_1, hTR_2, ..., hTR_k\}$ such that $hTR_j, j \in [1, k]$ is the total resource capability of resource R_j .

3) $hFRes = \{hFR_1, hFR_2, ..., hFR_k\}$ such that $hFRes_j, j \in [1, k]$ is the free resource capability of resource R_j

4) $hTask\_set$ describes the set of tasks that are allocated to a host.

5) $hComp\_time$ describes the time at which the host completes the execution of all tasks allocated to it.

5) h_Vm_i is set of virtual machines that are running on i^{th} host.

3.2.2 Virtual Machine Model

For each host h_i, Vm_i is the set of finite virtual machine $Vm_i = \{v_{1i}, v_{2i}, ..., v_{li}\}$ such that $\mid Vm_i \mid = l$.
Each $v_{ji} (j \in [1, l])$ and $(i \in [1, m])$ indicates virtual machine v_j running on host h_i. Each v_{ji} is represented by $\{vId, vTRes, vFRes, vFree, vPower\}$.Each attribute of the set is defined as follows:

1) vId virtual machine identification.

2) $vTRes$ is the total resource capacity of a virtual machine, $vTRes = \{vTR_1, vTR_2, ..., vTR_k\}$ such that $vTR_j (j \in [1, k])$ is the total resource capacity of resource R_j.

3) $vFRes = \{vFR_1, vFR_2, ..., vFR_k\}$ such that $vFRes_j, j \in [1, k]$ is the free resource capability of the resource R_j.

4) $vFree$ is a boolean variable signifying whether a virtual machine is free or not.

5) $vPower$ describes the total power consumed by a virtual machine.

3.2.3 Task Model

In general,service requests are heterogeneous in terms of their resource requirements. Some requests are I/O-Intensive and some are CPU-Intensive. These service requests are realized as tasks in cloud and are not homogeneous and varies greatly in their computational requirements. From application to application task vary in terms of their resource requirements, performance metric. Hence there is a need of defining a task model that can be easily mapped onto system model. The figure 3.1 shows the different levels that a task undergoes during its execution. All the service requests or jobs submitted by user

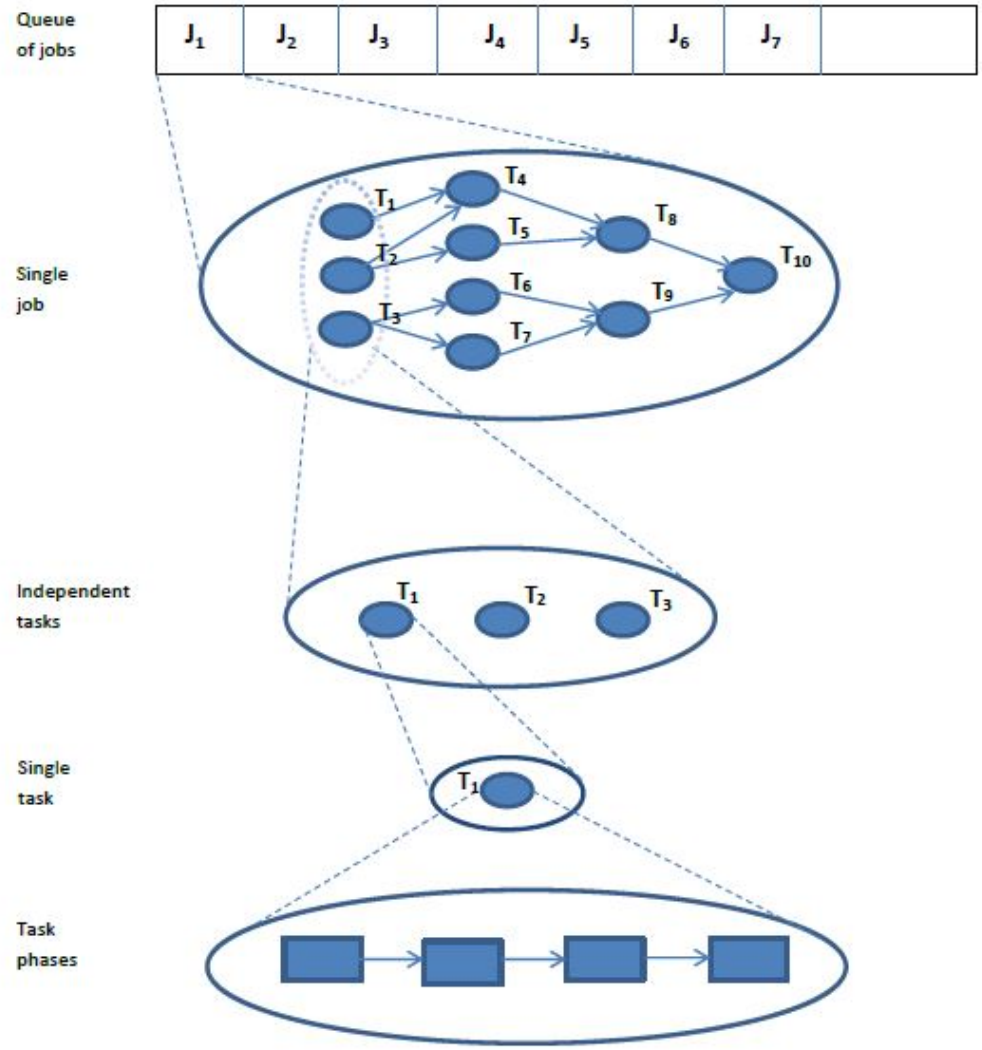

Figure 3.1: Workload Model

are stored in a queue. Each job can be a set of different tasks that can be independent or may depend on other tasks. Task is the smallest unit that executes using cloud resources. A task may have different parameters and it may undergo different task phases. To deal with the heterogeneous nature of workload, we have defined task model. The task model is as follows:
Set of finite tasks is T =$\{ t_1, t_2, ..., t_n \}$ such that $\mid T \mid = n$.Each $t_i, i \in [1, n]$ indicates task $i, t_i = \{tId, tarrival, tRes, tETC, tVm\_type, tAssign\}$. Each attribute of set is defined as follows:

1) tId is the task identification.

2) $tarrival$ is the task arrival time.

3) Required resource of a task is defined by $tRes = \{tR_1, tR_2, \ldots\ldots, tR_k\}$ such that $tR_j, j \in [1, k]$ is the requirement of resource R_j by the task.

4) $tETC$ describes the ETC(Expected Time to Compute) matrix for a task.For each task it is a 1$*$n matrix.

5) tVm_type describes the virtual machine type required by a task.

6) $tAssign$ = boolean variable representing whether task is scheduled or not.

3.3 Scheduling Architecture

For the above described host model, virtual machine model and task model, figure 3.2 describes the scheduling architecture. The architecture consists of a service scheduler and a VM controller. Service scheduler can be both central and distributed depending upon the requirements. In our work we have taken a centralized scheduler. The job of scheduler is to assign tasks to VM's. It also decided when VM's are to be added or removed to meet the demands. VM controller keep track of the availability of VM's and their available resources. It is also the in charge of migrating VM's across physical machines. When a task arrives, the scheduling process follows the following steps:

1) The scheduler checks the system status information about running task remaining execution time, active hosts, currently allocated VM's.

2) The tasks are sorted according to their arrival time.

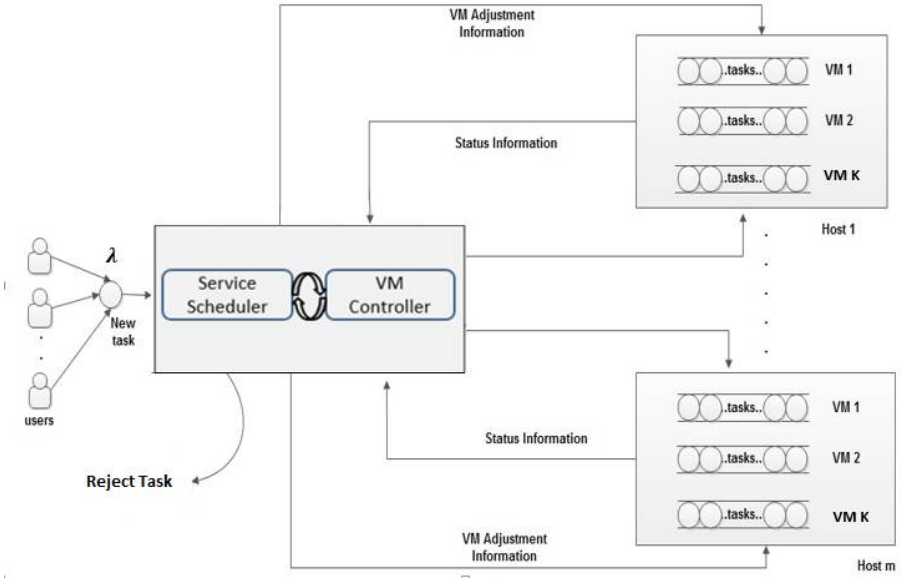

Figure 3.2: Simulation Architecture

3) The scheduler checks if the tasks can be allocated or not. If not, scheduler informs VM controller about it. To meet the task requirements, controller adds virtual machines. If schedule is found, task is allocated else task is rejected.

4) Allocated VM's, active hosts,available resources, task completion time etc. are updated.

3.4 LPP Formulation of Task consolidation Problem

Task Consolidation problem can be formulated as a linear programming problem. The LPP formulation of the problem is given below:

Minimize $E(0,t) = \sum_{j=1}^{m} \sum_{i=1}^{l} e_{ij}(0,t)$

Subject to:

1. $\sum_{i=1}^{l} e_{ij} \leq e_j, \qquad \forall j \in [1,m].$
2. $\sum_{i=1}^{l} vTRes_{ij} \leq hTRes_j, \qquad \forall j \in [1,m].$
3. $\sum_{i=1}^{l} vFRes_{ij} \leq hFRes_j, \qquad \forall j \in [1,m].$

where $E(0,t)$ describes the total energy consumed by the cloud in the time interval $[0,t]$. $e_{ij}(0,t)$ represents the energy consumed by virtual machine i running on host j in time interval $[0,t]$. The first condition restricts the total energy consumed by all the virtual machine inside a host to be less than the energy consumed by that host. Second constraint

says that total resources of all the virtual machines running on a host should always be less than the total resources of that host. Similarly third constraint states that total free resources of all the virtual machines running on a host should always be less than the total available resources of that host. All these conditions must hold true at every instant of time.

3.5 Conclusion

In this chapter, we discussed the workload models already existing in literature. We also proposed a system and workload model for the task consolidation problem. Also the scheduling architecture is discussed here. In next chapters, we have developed a greedy algorithm to the task consolidation problem. The algorithm is based on the models defined in this chapter.

Chapter 4

Greedy Algorithms for Task Consolidation Problem

4.1 Introduction

The rapid growth of cloud computing environment with the virtuallized DC's has made serious issues including energy consumption, cooling infrastructures and air conditioning concerns in terms of increasing operational costs (34). The increasing size of cloud infrastructure along with poor resource utilization is coming as a great challenge. The energy consumption varies with the number of blade servers and the incoming workload and has emerged as one of the biggest challenge for cloud computing.

Among the major reasons of energy inefficiency, one is the idle power wastage. Even at very low utilization(10%) the energy consumed is 50-60% of the peak energy (23),(36),(31), (39),(1),(16). This has resulted into reduced system reliability, extremely large electricity bills and environmental issues generating due to emission of carbon in large quantity. Thus an energy efficient task consolidation strategy that maximizes the utilization of resources and in turn reduces the energy consumption is required. The task consolidation problem is a NP-Complete problem and requires the heuristics technique to solve. In a homogeneous cloud the problem is a bit easy to solve because of the similar resource capabilities and capacities of servers. But in heterogeneous cloud, the problem becomes more complex as all the servers vary in their processor capabilities and capacities.

And as the cloud infrastructure size increase, its complexity increases leading to exponential solution space. Brute force technique will require huge amount of time to search the entire solution space and hence a heuristic that given a sub-optimal solution in an ac-

ceptable amount of time is required. The developed algorithm have to meet the following requirements:

- Decentralization and parallelism to eliminate SPF.
- Provide Scalability.
- High Performance.
- Guaranteed QoS.
- Independence of workload type.

In this chapter, greedy algorithm for task consolidation problem is proposed that tries to minimize the total energy consumption of cloud. The algorithm also tries to allocate the task on the server that takes minimum execution time for that task. The algorithm works on the system and workload model defined in chapter 3. Also some other heuristics like MaxUtil (14), random have been selected from literature. These approaches are analyzed and implemented and compared with our developed approach. All these approaches are implemented under a common set of assumptions. To generate the heterogeneity among machines and tasks, ETC model (2) is used.

4.2 Task Consolidation Algorithms

As the task consolidation problem is NP-Complete, no optimal solution can be found. Heuristics algorithms are required to solve the task consolidation problem obtaining a sub-optimal solution in acceptable amount of time. Using the greedy approach, I have developed an *Energy Aware Task Consolidation (EATC)* algorithm. Whenever a consumer submits a service requests, it is first handled by a front end web portal server, then the algorithm allocates the required resources if available and finally the request is forwarded to one server at a specific location. The algorithm runs on a special server called as scheduler specifically meant for scheduling the incoming tasks. The schedulers can be centralized as well as distributed depending upon the cloud infrastructure size. Rather than executing on individual task, the algorithm works in batch mode. The algorithm is dynamic is nature in the sense that tasks are coming dynamically.

4.2.1 EATC Algorithm

The developed approach for task consolidation problem is presented in algorithm 1. The working flow of EATC algorithm is shown in figure 4.1. In the first step, hosts and virtual machines resource details are input to the system. Then task details are input from the job queue maintained for storing the resource requirements for tasks. Then the algorithm calls *Resource_Generation* subroutine to generate the resource requirements for the task. After the resources are generated, another subroutine *ETC_Generation* is called to generate the ETC matrix. According to the VM requirement for the task, a specific virtual machine is initialized. The scheduling criteria for EATC is ETC matrix. The ETC matrix is stored in form of a min-heap so that host that takes minimum amount of time to compute the task can be searched in a constant amount of time. The root of min-heap gives the minimum time of execution for a specific task. The algorithm tries to allocate the task on to the resource taking minimum time for it. Deletion of root node takes place repetitively until a resource with required resources is found. After the host is found, next step is to check for the availability of required VM in that particular host. The searching for resources goes like that until we find the required resources. If the task requirements cannot be fulfilled, the task is added to reject queue. For the task with successful allocation, the available resources are updated. The task is added to $hTask\_set$ of that particular host and $hComp\_time$ of the host is updated. Based on the current utilization of resources of VM, the energy consumption is calculated following the energy consumption model defined in chapter 3. Same procedure is followed for all the incoming tasks. Tasks are selected in batch-mode, based on the arrival time i.e., FCFS mode. After the algorithm finishes all the task allocation work, it returns the scheduled details along with the total energy consumed.

The EATC algorithm works upon the system and workload model defined in chapter 3. For the defined algorithm, we have considered a number of assumptions which we used for simulation purpose. The assumptions are discussed below:

- Time required to execute a task includes both the computational time as well as communicational time. It means that $tETC$ includes both the computational time as well as communicational time.

- All the tasks are independent and heterogeneous in nature. This specific assumption model heterogeneity among tasks because in real time scenario tasks vary greatly in their computational complexities and other resource requirements.

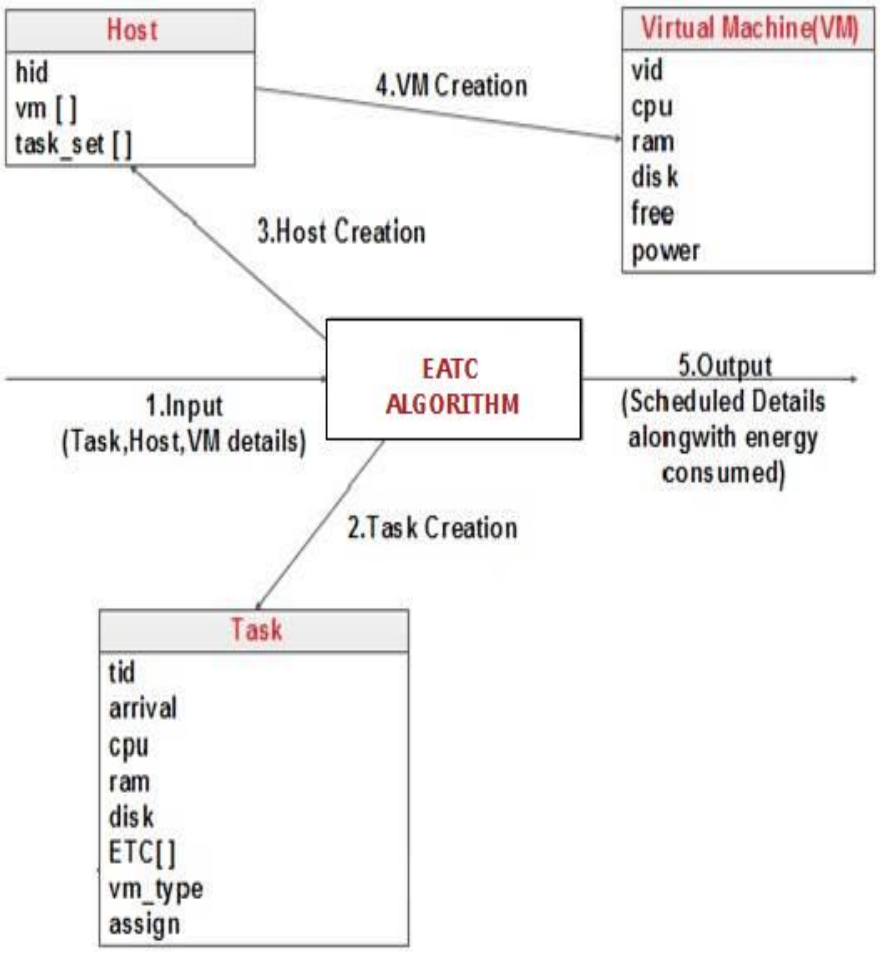

Figure 4.1: Working of EATC

- All the tasks are considered as non-preemptive in nature.
- Arrival time is considered to be Poisson distribution (λ).
- All the systems are heterogeneous in terms of their resource capabilities. It models system heterogeneity because in actual, systems vary greatly in terms of their processor speed, RAM size and other resource capabilities
- A task is allowed to execute only on a single machine.
- The task resource requirements should be positive.
- All the other overheads like start and shutdown time of virtual machines are considered to be constant.
- All virtual machines are installed on all physical hosts and initialized at beginning.

- Resources are updated after every allocation.

Algorithm 1: EATC

Input: $n\_host, T\_hetro, M\_hetro, st, \lambda, n\_vms$
Output: hTask_set, E

```
1  begin
2      n_tasks = 0.
3      for i in range from 1 to st do
4          a ~ Poisson(λ)
5          for j in range from 1 to a do
6              n_tasks = n_tasks + 1.
7              Required resources=Resource_Generation(n_vms)
8              ETC[n_tasks] = ETC_Generation(n_host, T_hetro, M_hetro)
9              sorted_host = Sort(ETC[n_tasks]).
10     for k in range from 1 to n_tasks do
11         for l in range from 1 to n_host do
12             for m in range from 1 to n_host do
13                 if sorted_host(1,l) == ETC(1,m) then
14                     if the required VM of m^th host can fulfill the task requirements
                       then
15                         h_l Task_set = h_l Task_set ∪ tId_k.
16                         tAssign =TRUE Update the vFRes of alloted VM.
17                         Update E. Update h_l Comp_time.
18                         break.
19             if tAssign == TRUE then
20                 break.
21         if tAssign != TRUE then
22             Display error message sufficient resources not available.
23             Reject the task.
24     return E, hTask_Set.
```

The algorithm returns the total energy consumed for executing all the arrived tasks. It also returns the hComp_time and hTask_Set for all the hosts. The algorithm reduces the overall makespan of servers but as we have focused only on energy so we have not shown makespan in our results. The algorithm *EATC* further calls two subroutines named *ETC_Generation* and *Resource_Generation*. The working of the subroutines is describes in the next sections.

4.2.2 ETC_Generation

As we have considered the cloud environment heterogeneous in nature, both the physical machines and tasks vary greatly in terms of their resource requirements. To introduce this heterogeneity among tasks and machines, we have followed the ETC matrix model (2). This matrix gives the expected time to execute a task on every machine. In ETC matrix, the entry (i,j) indicates the expected execution time of task *i* on the machine *j*. Two methods called *Range-Based* and *Cofficient-of-Variation Based* (2) have been developed to generate this ETC matrix. We have presented the algorithm 2 that uses the range-based method for generating ETC matrix. For every incoming task, the subroutine takes three parameters n_host, representing number of physical hosts, T_hetro, representing task heterogeneity and M_hetro representing machine heterogeneity. The algorithm uses the *Uniform Distribution* and returns the ETC matrix. As we have called this subroutine for every incoming task, size of ETC matrix is of order $1 \times$n_host.

Algorithm 2: ETC_Generation

Input: n_host, T_hetro, M_hetro
Output: An ETC matrix of order $[1 * n\_host]$

```
1 begin
2     Compute a = ∪(1, T_hetro)
3     for i in range from 0 to (n_host − 1) do
4         b=∪(1, M_hetro)
5         ETC[1, i] = a ∗ b
6     return ETC
```

4.2.3 Resource_Generation

Every incoming task in cloud will have certain resource requirements, that can be CPU, RAM, disk, virtual machine required etc. These resource requirements are generated using algorithm 3. The algorithm is called for every incoming task. It takes n_vms i.e., number of virtual machine as the input parameter and returns the task with its required resources i.e., $CPU, RAM, Disk, vm\_type$. The resources are generated using *Uniform Distribution*.

Algorithm 3: Resource_Generation

Input: n_vms
Output: Task along with its resource requirements i.e. $CPU, RAM, Disk, vm\_type$

1 **begin**
2 **Compute** CPU = $\cup(x, y)$
3 **Compute** RAM = $\cup(x, y)$
4 **Compute** Disk = $\cup(x, y)$
5 **Compute** vm_type = $\cup(1, n\_vms)$
6 return $Allresources : CPU, RAM, Disk, vm\_type$

4.3 Experimental Evaluation and Simulation Result

The experimental evaluation has been carried out using the in-house simulator using Matlab2012. All experiments were run on systems with Windows 8 (32 bit) operating system on Intel Core i3 processor. We have conducted various experiments for ten times and the average result obtained is shown here. We have used three heuristic algorithms on different task arrival patterns to observe the energy consumption. A total of three type of arrival patterns namely low traffic arrival, moderate traffic arrival, high traffic arrival using Poisson distribution were generated. After implementing the algorithms, the performance is compared between our proposed algorithm EATC, another greedy algorithm MaxUtil (15) and a random algorithm. The algorithm differ in their way of resource selection, resource usage for the incoming workload. In our simulation, we have taken three resources namely CPU, RAM and disk. All the required resources for the incoming task are generated using uniform distribution. The performance parameter is taken as total energy consumed in allocating all the tasks that are arriving in a given interval of time. Graphs are plotted for total number of tasks arrived versus total energy consumed in executing those tasks. The results are obtained for different task arrival patterns. The energy measurement unit was taken to be Joule. After simulating the algorithm, the comparative results are shown in figure 4.2 to figure 4.13. All the related values for simulation environment were taken using table 4.1.

Figure 4.2 shows the result for low arrival rate. A total of 100 tasks are arriving in the given time and for the first 50 tasks the energy consumption by three different algorithms do not vary greatly. But for next 50 tasks, EATC shows significant improvement in total energy consumed over MaxUtil and Random algorithm. Random algorithm consumes maximum amount of energy among all three algorithms. In figure 4.3, a total of more than 200 tasks are arriving. After the first 50 tasks are executed the energy consumption by EATC is significantly less than those of other two algorithms. Similarly in figure 4.4, i.e., high arrival pattern a total of more than 250 tasks are arriving. MaxUtil and Random are almost showing the same behaviour while EATC has consumed almost half of the energy consumed by other two algorithms.

In figure 4.5 and 4.6, EATC and Random do not vary greatly in terms of energy consumption, but EATC showed a significant energy saving over MaxUtil. The result obtained in figure 4.7 are different from those obtained in figure 4.5 and 4.6. Here EATC and MaxUtil almost showed the same with slighter improvement in EATC while Random shows the

Table 4.1: Simulation Values Taken

Variable	Value
simulation time	20 sec
n_host	30
λ (low arrival)	5
λ (moderate arrival)	10
λ (high arrival)	15
T_hetro=10, M_hetro=10	ETC matrix for Low Task and Low Machine Heterogeneity
T_hetro=10, M_hetro=10^2	ETC matrix for Low Task and High Machine Heterogeneity
T_hetro=10^5,M_hetro= 10	ETC matrix for High Task and Low Machine Heterogeneity
T_hetro=10^5,M_hetro= 10^2	ETC matrix for High Task and High Machine Heterogeneity
n_vms	3
α	5 W
β	10 W

worst behaviour consuming maximum amount of energy.

In figure 4.8, a total of more than 100 tasks are arriving. For the first 60 tasks, MaxUtil and Random are showing almost the same energy consumption while later on EATC showing significant energy savings over these two approaches. In figure 4.9, Random and MaxUtil are showing almost the same energy consumption for first 130 tasks while for remaining tasks MaxUtil consumes less amount of energy than the Random algorithm. EATC shows the best result for the whole set of tasks. Similarly in figure 4.10, MaxUtil and Random shows almost the same behaviour with Random consuming a bit lesser energy than MaxUtil. EATC shows more than 100% energy savings over the other two approaches.

For figure 4.11, 4.12 and 4.13 all the three algorithms are showing same type of behaviour i.e., MaxUtil consuming maximum energy, EATC consuming minimum energy and Random consuming energy in between these two algorithms. In all three arrival patterns. EATC is consuming minimum amount of energy.

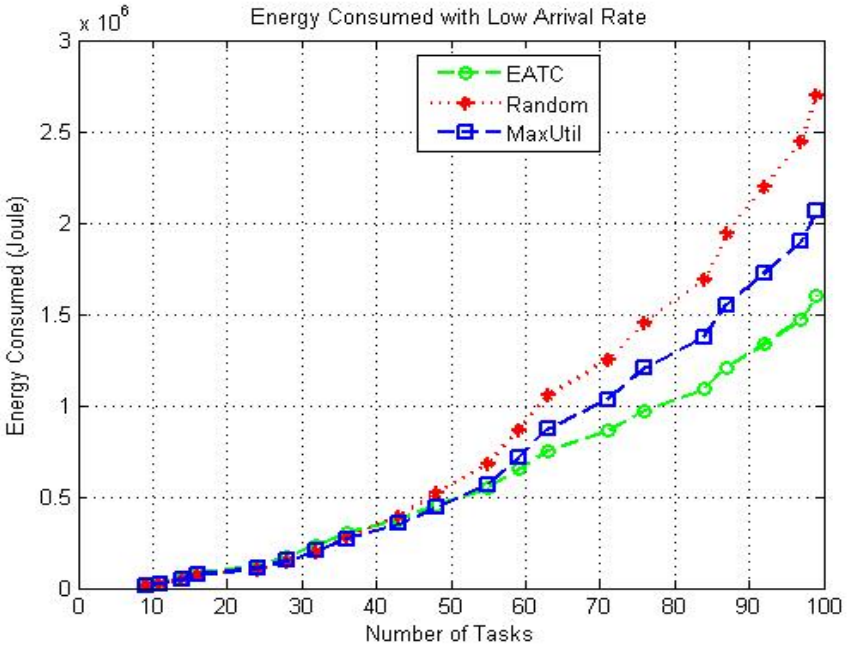

Figure 4.2: ETC matrix for Low Task and Low Machine Heterogeneity

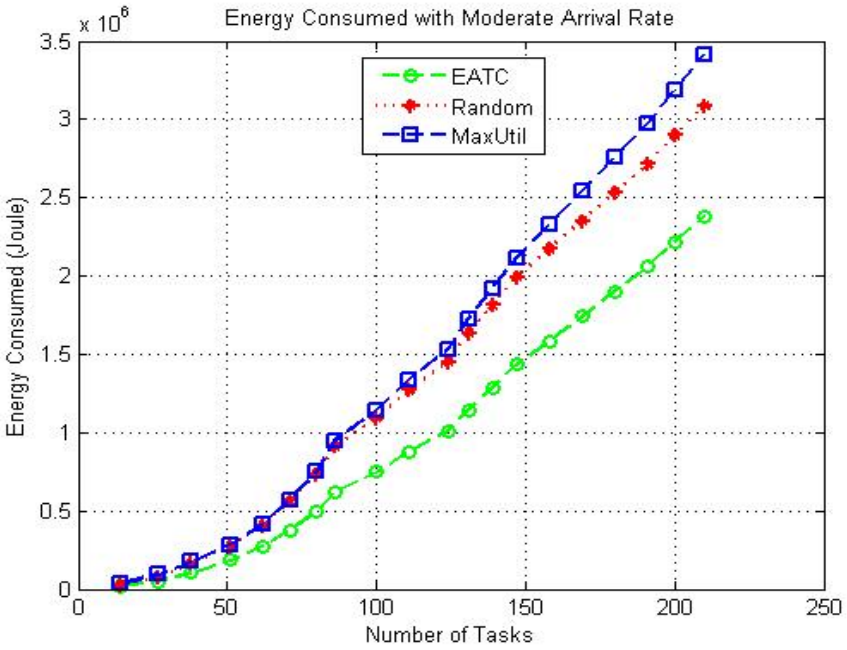

Figure 4.3: ETC matrix for Low Task and Low Machine Heterogeneity

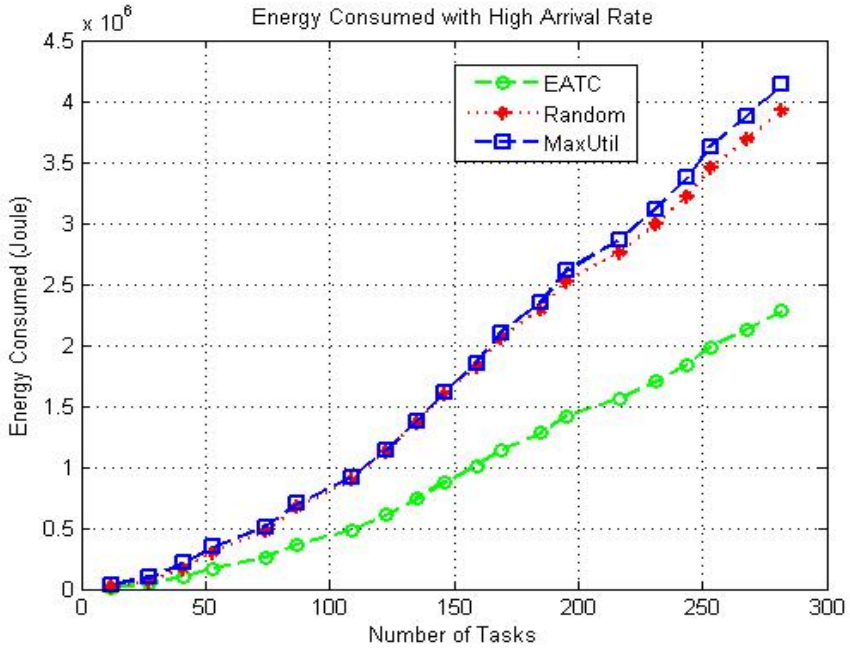

Figure 4.4: ETC matrix for Low Task and Low Machine Heterogeneity

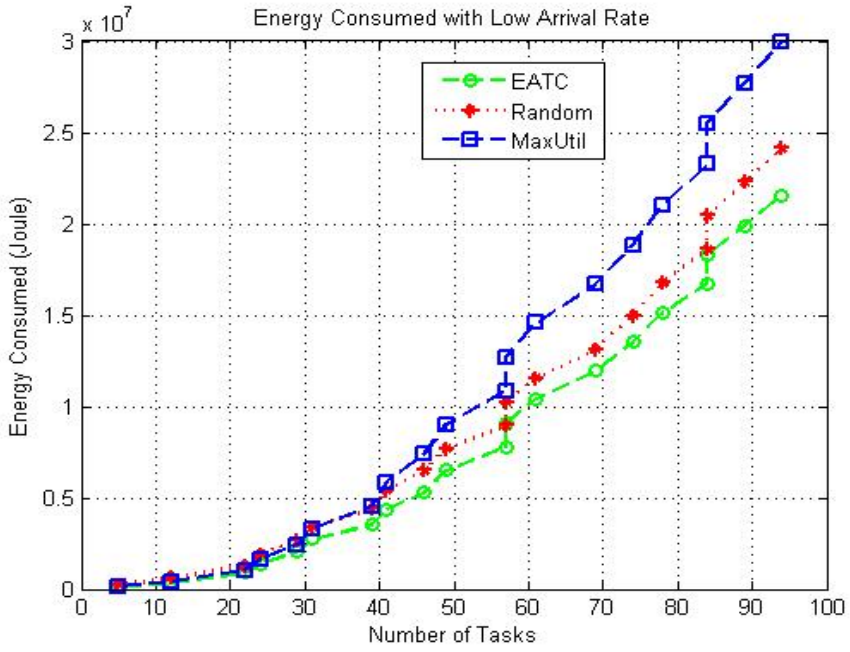

Figure 4.5: ETC matrix for Low Task and High Machine Heterogeneity

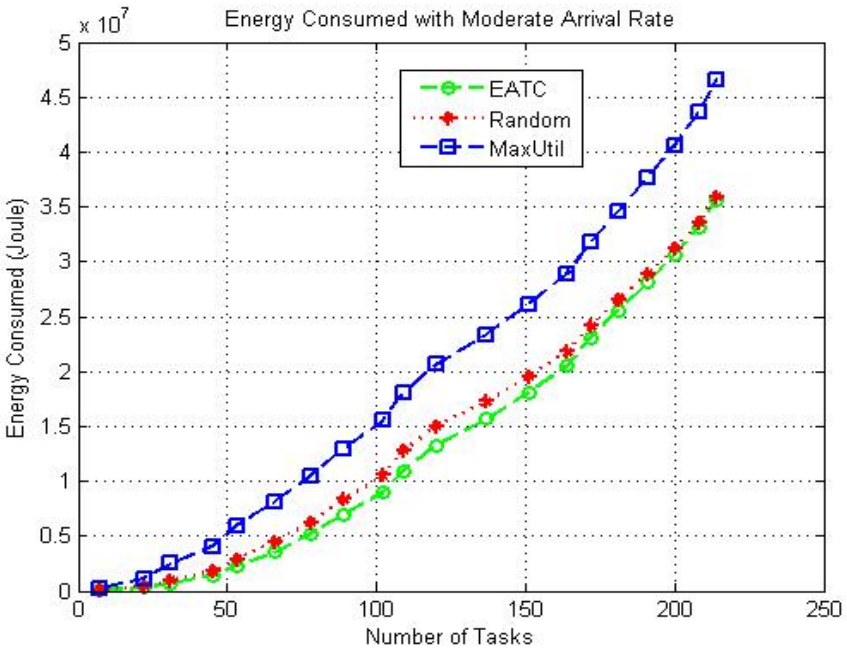

Figure 4.6: ETC matrix for Low Task and High Machine Heterogeneity

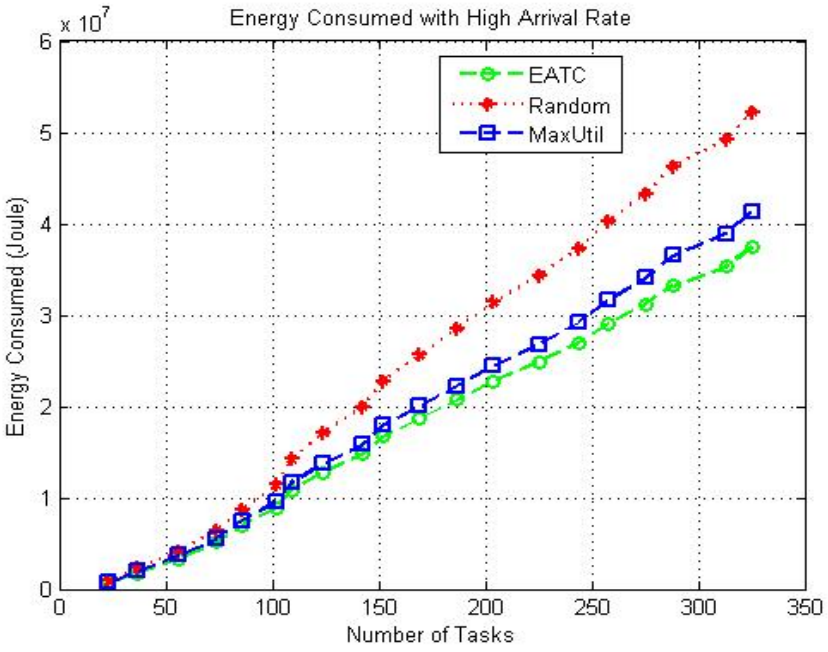

Figure 4.7: ETC matrix for Low Task and High Machine Heterogeneity

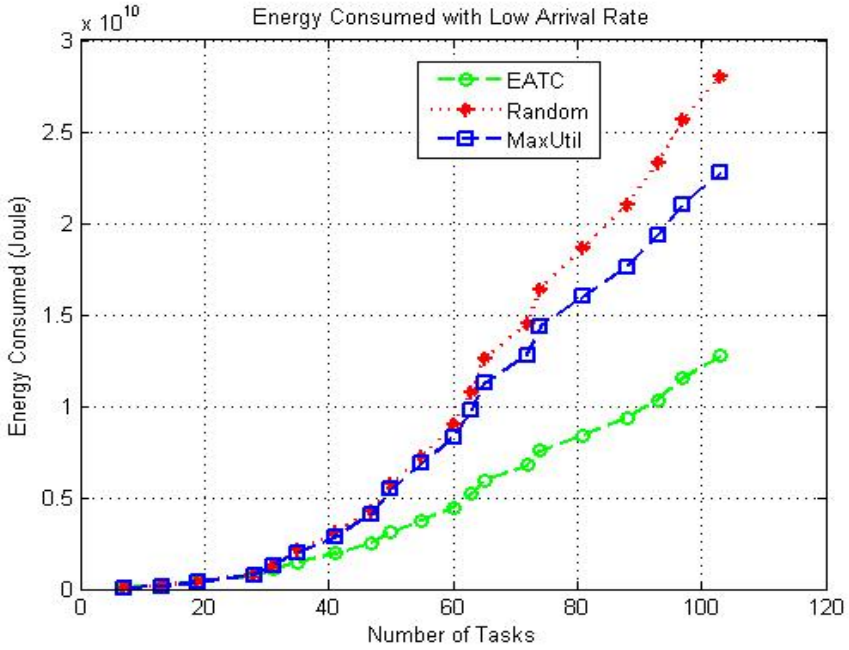

Figure 4.8: ETC matrix for High Task and Low Machine Heterogeneity

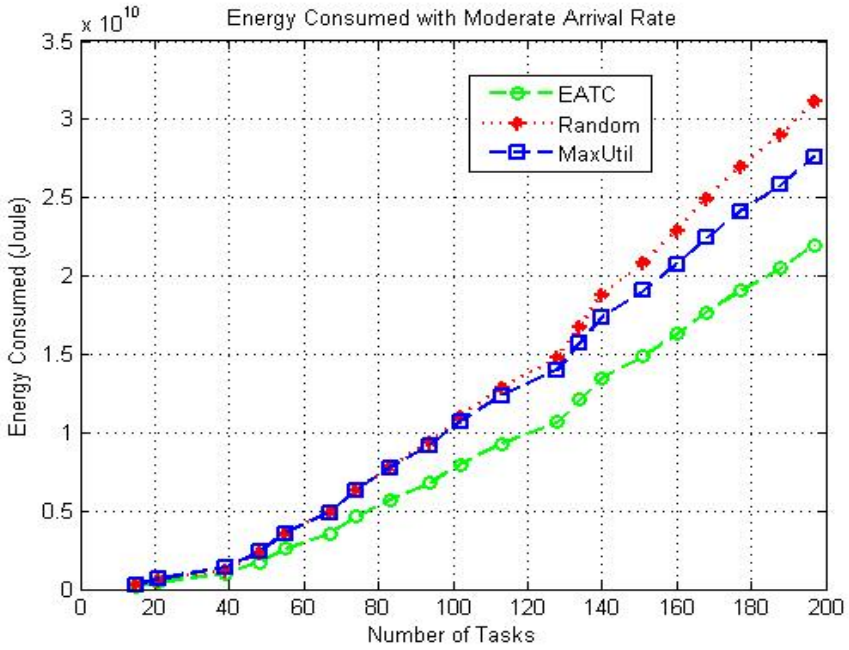

Figure 4.9: ETC matrix for High Task and Low Machine Heterogeneity

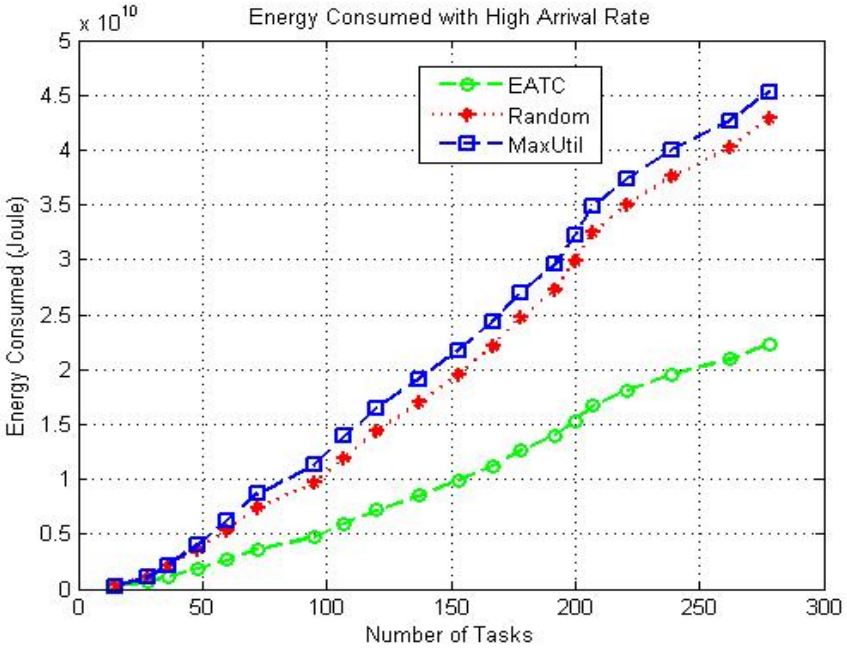

Figure 4.10: ETC matrix for High Task and Low Machine Heterogeneity

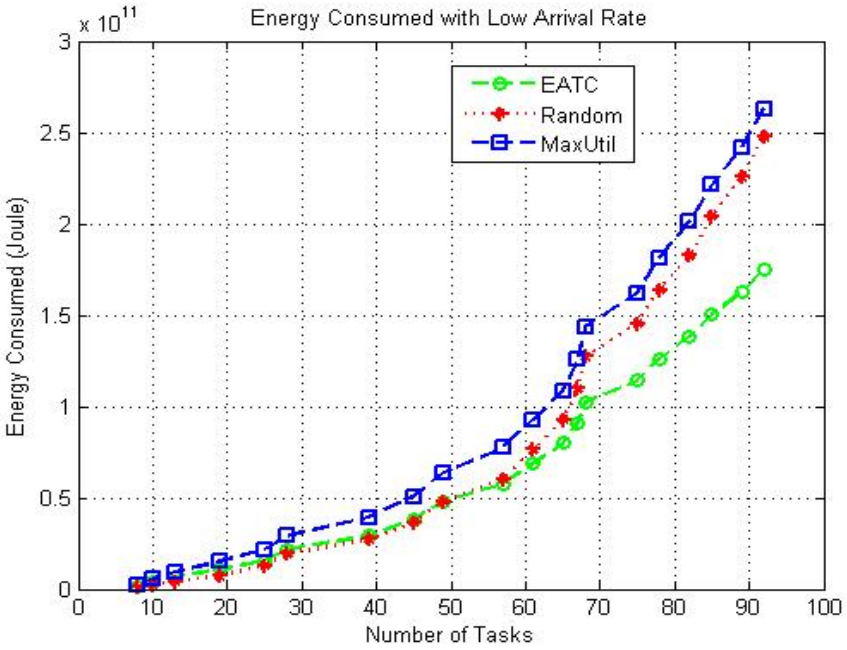

Figure 4.11: ETC matrix for High Task and High Machine Heterogeneity

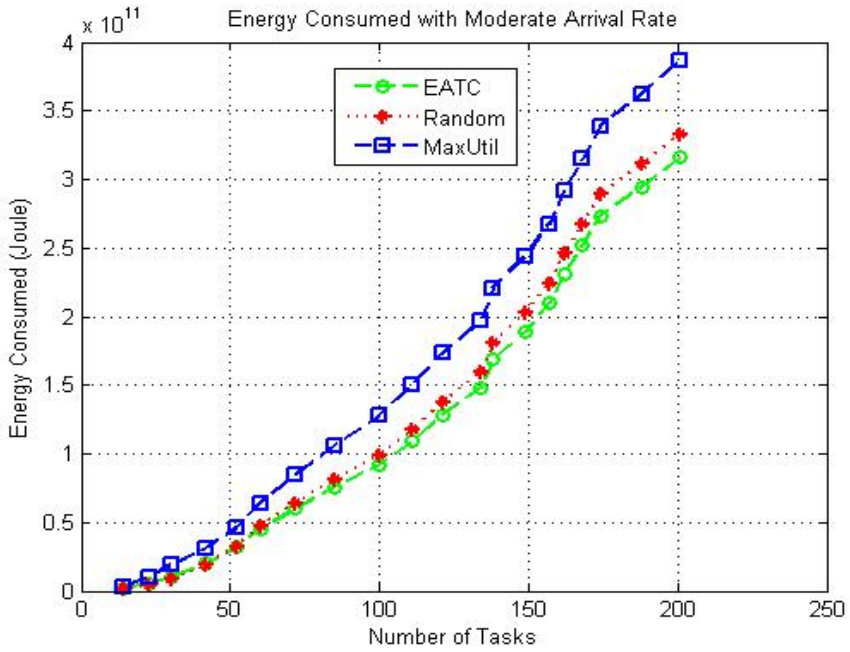

Figure 4.12: ETC matrix for High Task and High Machine Heterogeneity

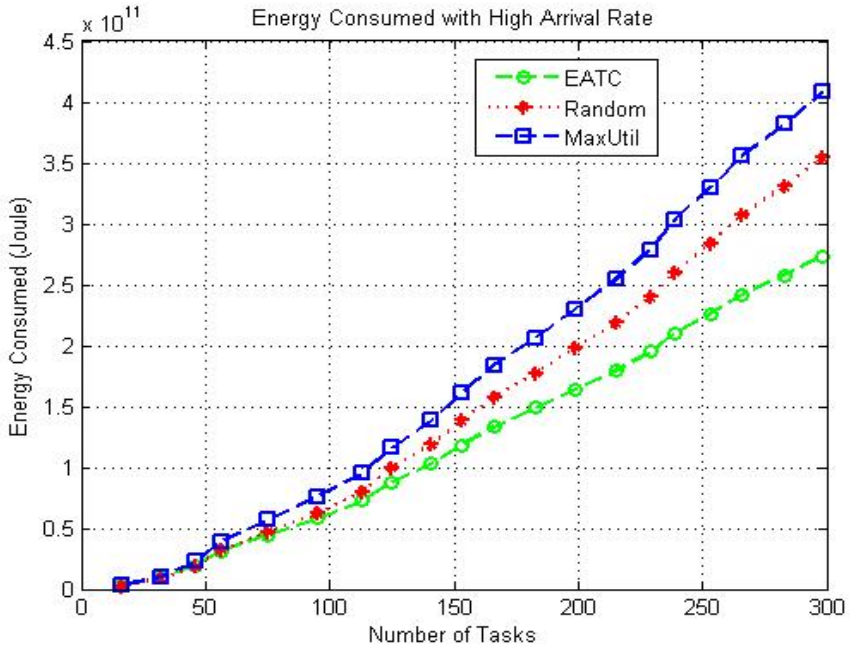

Figure 4.13: ETC matrix for High Task and High Machine Heterogeneity

4.4 Conclusion

In this chapter, we have developed an Energy Aware Task Consolidation(EATC) Algorithm for heterogeneous cloud computing using greedy approach. The simulation was carried out for all four types of ETC matrix. For each ETC matrix, the results are obtained for three traffic arrival rates namely(low, moderate, high). Energy consumption is taken as the performance parameter and performance of EATC was compared with a recently developed algorithm MaxUtil (15) and Random algorithm. The results obtained shows the change in energy consumption with varying the workload. The propsed algorithm i.e., EATC shows significant improvement in energy saving over the MaxUtil and Random algorithm.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

Task consolidation problem in cloud has been addressed as an optimization problem. Also due to heterogeneous nature of physical servers and incoming tasks, this problem becomes more complex. In this thesis we have discussed about different task consolidation strategies proposed by various researchers. Most of the existing work is focused for homogeneous cloud environment and only a little work is done for addressing the task and machine heterogeneity. As it is a NP-complete problem, heuristics techniques are preferred by the researchers to address the problem. To model the heterogeneity, we have used the ETC model (2). We have also developed a generalized system and workload model to handle a variety of tasks. For the proposed model, we have used the greedy algorithms for task consolidation problem. The developed algorithm tries to make optimized use of cloud resources in order to reduce energy consumption. Simulation experiments were conducted to examine the performance of developed EATC algorithm to optimize the energy consumption in cloud computing system. The performance was compared against two other algorithms named MaxUtil (15) and a randomized algorithm. The results showed a significant improvement in energy savings of EATC over the other two heuristics.

5.2 Future Work

The cloud environment is dynamic in nature as the workload may vary from time to time. At high peak load, the system performance may degrade in terms of average waiting time,

response time, throughput etc. The work in (40), (28) have taken into account the scalability to meet the deadline during high workload. Thus in order to prevent performance degradation, we can extend our algorithm to be more scalable in terms of adding or removing virtual machines if required. We can also introduce more resources(e.g., storage servers) to study their impact on energy consumption. The work can be done in order to prevent the idle power wastage i.e., to minimize the number of active servers.

Bibliography

[1] Abdulrahman Alahmadi, Abdulaziz Alnowiser, Michelle M Zhu, Dunren Che, and Parisa Ghodous. Enhanced first-fit decreasing algorithm for energy-aware job scheduling in cloud. In *Computational Science and Computational Intelligence (CSCI), 2014 International Conference on*, volume 2, pages 69–74. IEEE, 2014.

[2] Shoukat Ali, Howard Jay Siegel, Muthucumaru Maheswaran, and Debra Hensgen. Task execution time modeling for heterogeneous computing systems. In *Heterogeneous Computing Workshop, 2000.(HCW 2000) Proceedings. 9th*, pages 185–199. IEEE, 2000.

[3] Abdulaziz Alnowiser, Eman Aldhahri, Abdulrahman Alahmadi, and Michelle M Zhu. Enhanced weighted round robin (ewrr) with dvfs technology in cloud energy-aware. In *Computational Science and Computational Intelligence (CSCI), 2014 International Conference on*, volume 1, pages 320–326. IEEE, 2014.

[4] Anton Beloglazov and Rajkumar Buyya. Energy efficient resource management in virtualized cloud data centers. In *Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pages 826–831. IEEE Computer Society, 2010.

[5] Anton Beloglazov, Rajkumar Buyya, Young Choon Lee, and Albert Zomaya. A taxonomy and survey of energy-efficient data centers and cloud computing systems. *arXiv preprint arXiv:1007.0066*, 2010.

[6] Anton Beloglazov, Rajkumar Buyya, Young Choon Lee, Albert Zomaya, et al. A taxonomy and survey of energy-efficient data centers and cloud computing systems. *Advances in Computers*, 82(2):47–111, 2011.

[7] Rajkumar Buyya, Anton Beloglazov, and Jemal Abawajy. Energy-efficient manage-

ment of data center resources for cloud computing: A vision, architectural elements, and open challenges. *arXiv preprint arXiv:1006.0308*, 2010.

[8] Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal, James Broberg, and Ivona Brandic. Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation computer systems*, 25(6):599–616, 2009.

[9] Huangke Chen, Xiaomin Zhu, Jianghan Zhu, and Jianjiang Wang. Eres: An energy-aware real-time elastic scheduling algorithm in clouds. In *High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing (HPCC_EUC), 2013 IEEE 10th International Conference on*, pages 777–784. IEEE, 2013.

[10] Mehiar Dabbagh, Bechir Hamdaoui, Mohsen Guizani, and Ammar Rayes. Toward energy-efficient cloud computing: Prediction, consolidation, and overcommitment. *Network, IEEE*, 29(2):56–61, 2015.

[11] Cesar O Diaz, Mateusz Guzek, Johnatan E Pecero, Gregoire Danoy, Pascal Bouvry, and Samee Ullah Khan. Energy-aware fast scheduling heuristics in heterogeneous computing systems. In *High Performance Computing and Simulation (HPCS), 2011 International Conference on*, pages 478–484. IEEE, 2011.

[12] Pragya Gupta and Sudha Gupta. Mobile cloud computing: the future of cloud. *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, 1(3):134–145, 2012.

[13] Abdul Hameed, Alireza Khoshkbarforoushha, Rajiv Ranjan, Prem Prakash Jayaraman, Joanna Kolodziej, Pavan Balaji, Sherali Zeadally, Qutaibah Marwan Malluhi, Nikos Tziritas, Abhinav Vishnu, et al. A survey and taxonomy on energy efficient resource allocation techniques for cloud computing systems. *Computing*, pages 1–24, 2014.

[14] Ching-Hsien Hsu, Kenn D Slagter, Shih-Chang Chen, and Yeh-Ching Chung. Optimizing energy consumption with task consolidation in clouds. *Information Sciences*, 258:452–462.

[15] Ching-Hsien Hsu, Kenn D Slagter, Shih-Chang Chen, and Yeh-Ching Chung. Optimizing energy consumption with task consolidation in clouds. *Information Sciences*, 258:452–462, 2014.

[16] Liting Hu, Hai Jin, Xiaofei Liao, Xianjie Xiong, and Haikun Liu. Magnet: A novel scheduling policy for power reduction in cluster with virtual machines. In *Cluster Computing, 2008 IEEE International Conference on*, pages 13–22. IEEE, 2008.

[17] Jing Jiang. Optimised auto-scaling for cloud-based web service. 2015.

[18] Gregory Katsaros, Josep Subirats, J Oriol Fitó, Jordi Guitart, Pierre Gilet, and Daniel Espling. A service framework for energy-aware monitoring and vm management in clouds. *Future Generation Computer Systems*, 29(8):2077–2091, 2013.

[19] Kyong Hoon Kim, Anton Beloglazov, and Rajkumar Buyya. Power-aware provisioning of cloud resources for real-time services. In *Proceedings of the 7th International Workshop on Middleware for Grids, Clouds and e-Science*, page 1. ACM, 2009.

[20] Dzmitry Kliazovich, Sisay T Arzo, Fabrizio Granelli, Pascal Bouvry, and Samee Ullah Khan. e-stab: Energy-efficient scheduling for cloud computing applications with traffic load balancing. In *Green Computing and Communications (GreenCom), 2013 IEEE and Internet of Things (iThings/CPSCom), IEEE International Conference on and IEEE Cyber, Physical and Social Computing*, pages 7–13. IEEE, 2013.

[21] Joanna Kolodziej, Samee Ullah Khan, and Fatos Xhafa. Genetic algorithms for energy-aware scheduling in computational grids. In *P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2011 International Conference on*, pages 17–24. IEEE, 2011.

[22] Dilip Kumar and Bibhudatta Sahoo. Energy efficient heuristic resource allocation for cloud computing. *International Journal*, 2014.

[23] Young Choon Lee and Albert Y Zomaya. Energy efficient utilization of resources in cloud computing systems. *The Journal of Supercomputing*, 60(2):268–280, 2012.

[24] Ning Liu, Ziqian Dong, and Roberto Rojas-Cessa. Task and server assignment for reduction of energy consumption in datacenters. In *Network Computing and Appli-*

cations (NCA), 2012 11th IEEE International Symposium on, pages 171–174. IEEE, 2012.

[25] Liang Luo, Wenjun Wu, Dichen Di, Fei Zhang, Yizhou Yan, and Yaokuan Mao. A resource scheduling algorithm of cloud computing based on energy efficient optimization methods. In *Green Computing Conference (IGCC), 2012 International*, pages 1–6. IEEE, 2012.

[26] Liang Luo, Wenjun Wu, WT Tsai, Dichen Di, and Fei Zhang. Simulation of power consumption of cloud data centers. *Simulation Modelling Practice and Theory*, 39:152–171, 2013.

[27] Zoltán Adám Mann. Allocation of virtual machines in cloud data centers–a survey of problem models and optimization algorithms. 2014.

[28] Ming Mao and Marty Humphrey. Auto-scaling to minimize cost and meet application deadlines in cloud workflows. In *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, page 49. ACM, 2011.

[29] Dan C Marinescu. *Cloud computing: Theory and practice*. Newnes, 2013.

[30] Lauri Minas and Brad Ellison. *Energy efficiency for information technology: How to reduce power consumption in servers and data centers*. Intel Press, 2009.

[31] Elizabeth Sylvester Mkoba and Mokhtar Abdullah Abdo Saif. A survey on energy efficient with task consolidation in the virtualized cloud computing environment.

[32] Chandrashekhar S Pawar and Rajnikant B Wagh. Priority based dynamic resource allocation in cloud computing. In *Cloud and Services Computing (ISCOS), 2012 International Symposium on*, pages 1–6. IEEE, 2012.

[33] Lei Rao, Xue Liu, Le Xie, and Wenyu Liu. Minimizing electricity cost: optimization of distributed internet data centers in a multi-electricity-market environment. In *INFOCOM, 2010 Proceedings IEEE*, pages 1–9. IEEE, 2010.

[34] Ivan Rodero, Juan Jaramillo, Andres Quiroz, Manish Parashar, Francesc Guim, and Stephen Poole. Energy-efficient application-aware online provisioning for virtualized clouds and data centers. In *Green Computing Conference, 2010 International*, pages 31–45. IEEE, 2010.

[35] Pankaj Singh, Binay Kumar Pandey, and HL Mandoria. Review of energy aware policies for cloud computing environment.

[36] Shekhar Srikantaiah, Aman Kansal, and Feng Zhao. Energy aware consolidation for cloud computing. In *Proceedings of the 2008 conference on Power aware computing and systems*, volume 10. San Diego, California, 2008.

[37] Qinghui Tang, Sandeep KS Gupta, and Georgios Varsamopoulos. Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers: A cyber-physical approach. *Parallel and Distributed Systems, IEEE Transactions on*, 19(11):1458–1472, 2008.

[38] Nikos Tziritas, Samee Ullah Khan, Cheng-Zhong Xu, Thanasis Loukopoulos, and Spyros Lalis. On minimizing the resource consumption of cloud applications using process migrations. *Journal of Parallel and Distributed Computing*, 73(12):1690–1704, 2013.

[39] Giorgio L Valentini, Samee U Khan, and Pascal Bouvry. Energy-efficient resource utilization in cloud computing. *Large Scale Network-centric Computing Systems, John Wiley & Sons, Hoboken, NJ, USA*, 2013.

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

[41] Yan Wang, Kenli Li, Hao Chen, Ligang He, and Keqin Li. Energy-aware data allocation and task scheduling on heterogeneous multiprocessor systems with time constraints. *Emerging Topics in Computing, IEEE Transactions on*, 2(2):134–148, 2014.

[42] Kejiang Ye, Dawei Huang, Xiaohong Jiang, Huajun Chen, and Shuang Wu. Virtual machine based energy-efficient data center architecture for cloud computing: a performance perspective. In *Proceedings of the 2010 IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing*, pages 171–178. IEEE Computer Society, 2010.

[43] Luna Mingyi Zhang, Keqin Li, Dan Chia-Tien Lo, and Yanqing Zhang. Energy-efficient task scheduling algorithms on heterogeneous computers with continuous

and discrete speeds. *Sustainable Computing: Informatics and Systems*, 3(2):109–118, 2013.

Dissemination

Journal

1. Reenu Deswal, Sambit Kumar Mishra, Bibhudutta Sahoo, “Optimizing Power Consumption in Cloud Using Task Consolidation”, CiiT International Journal of Networking and Communication Engineering, vol. 7, no. 4, pp.155-162, 2015