Task-Aware Priority Based Scheduling in Cloud Computing

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering
in
Computer Science and Engineering

Submitted By
Atul Kumar Ramotra
(Roll No. 801132003)

Under the supervision of:
Ms. Anju Bala
Assistant Professor

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004

July 2013

Certificate

I hereby certify that the work which is being presented in the thesis entitled, **"Task-Aware Priority Based Scheduling in Cloud Computing"**, in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Computer Science and Engineering* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of *Ms. Anju Bala* and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

(Atul Kumar Ramotra)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

(Ms. Anju Bala)
Assistant Professor
Computer Science and Engineering Department
Thapar University
Patiala

Countersigned by:

(Dr. Maninder Singh)
Associate Professor and Head
Computer Science and Engineering Department
Thapar University
Patiala

(Dr. S. K. Mohapatra)
Dean (Academic Affairs)
Thapar University
Patiala

Acknowledgement

I would like to express my deep sense of gratitude to my supervisor, **Ms. Anju Bala**, Assistant Professor, Computer Science and Engineering Department, Thapar University, Patiala, for her invaluable help and guidance during the course of thesis. I am highly indebt to her for constantly encouraging me by giving critics on my work. I am grateful to her for giving me the support and confidence that helped me a lot in carrying out the research work in the present form. And for me, it's an honor to work under her.

I also take the opportunity to thank **Dr. Maninder Singh**, Associate Professor and Head, Computer Science and Engineering Department, Thapar University, Patiala, for providing us with the adequate infrastructure in carrying out the research work.

I would also like to thank my parents and friends for their inspiration and ever encouraging moral support, which went a long way in successful completion of my thesis.

Above all, I would like to thank the almighty God for his blessings and for driving me with faith, hope and courage in the thinnest of the times.

Atul Kumar Ramotra

(801132003*)*

Abstract

Cloud computing is an internet-based computing where resources, software and information are provided to computers on-demand, like a public utility. It is emerging as a platform for sharing resources like infrastructure, software and various applications. The majority of cloud computing infrastructure consists of reliable services delivered through data centers. Modern data centers, operating under the cloud computing model are hosting a variety of applications ranging from those that run for a few seconds to those that run for longer periods of time. When a job is submitted to the clouds, it is partitioned into several tasks. Scheduling a set of tasks in is one of the traditional challenges in parallel and distributed computing. It is very important to decide the order in which these tasks should be executed in order to increase the overall efficiency.

Task scheduling is a key process for assigning the requests to resources in an efficient way considering cloud characteristics. Proper scheduling can have significant impact on the performance of the system. In this thesis work a task level scheduling approach has been proposed which prioritize the tasks to reduce the execution time, waiting time and response time. The simulator CloudSim has been used to validate the proposed algorithm.

Table of Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1
Introduction

This chapter includes a brief introduction of cloud computing, its evolution, deployment models, service models, key characteristics, issues and advantages.

1.1 Cloud Computing

Cloud Computing is gaining in popularity, both in the academic world and in software industry. Cloud Computing include on-demand self service and dynamic scalability. As Cloud Computing utilizes pay-as-you-go payment solution, customers are offered fine grained costs for rented services [1]. These advantages can be utilized to prevent web-applications from breaking during peak loads supporting end users. According to Amazon, provider of Amazon Web Service (AWS), a major Cloud Computing host: "much like plugging in a microwave in order to power it doesn't require any knowledge of electricity, one should be able to plug in an application to the cloud in order to receive the power it needs to run, just like a utility" [2]. Cloud Computing refers to both the applications delivered as services over the internet and systems software in the datacenters that provide those services. These datacenter hardware and software called as a Cloud [1].

Cloud computing is an IT deployment model established on virtualization, where resources in terms of applications, infrastructure and data are deployed via the internet as a distributed service by one or several service providers. These services are scalable on demand which can be priced on a pay-per-use basis [3]. Cloud Computing differs from traditional computing paradigms as it is scalable and can be encapsulated as an abstract entity providing different level of services to the clients which are driven by economies of scale and the services are dynamically configurable [4]. There are many benefits of Cloud Computed stated by different researchers which make it more preferable to be adopted by enterprises. Cloud Computing infrastructure allows enterprises to achieve more efficient use of their IT software and hardware investments being achieved by breaking down the physical barrier inherent in isolated systems and automating the management of the group of the systems as a single entity. Cloud Computing can also be defined as ultimately virtualized system and a natural evolution for data centers which offer automated systems management [4]. Cloud Computing Services are an emerging network architecture where applications

reside on third party servers which are managed by private firms that provide remote access with the help of web based devices. Customers mostly do not own the infrastructure. This model of service delivery is different from the architecture in which data and applications typically reside on servers or computers within the control of the end user. Such type of model, which aims to offer distributed, virtualized, and elastic resources as utilities to end users, having the potential to support full realization of "computing as a utility" in the near future [5].

Cloud computing is not a total new concept; it is originated from the earlier large-scale distributed computing technology. Along with the advancements of the Cloud technology there are new possibilities for Internet-based applications development which are also emerging. These new application models can be grouped into two forms. On one side there are the cloud service providers that are willing to provide large-scale computing infrastructure at a price based primarily on usage patterns which eliminates the initial high-cost for application developers of environment set up an application deployment and on the other side there are large-scale software systems providers. They develop applications like e-commerce and social networking sites which are gaining popularity on the Internet. These applications can benefit Cloud infrastructure services to minimize costs and improve service quality to end users. With the help of the Cloud Computing, deployment and hosting became cheaper and easier with the use of pay-per-use and flexible infrastructure services provided by Cloud providers [6]. Figure1.1 shows the hardware and software infrastructure for cloud Computing.

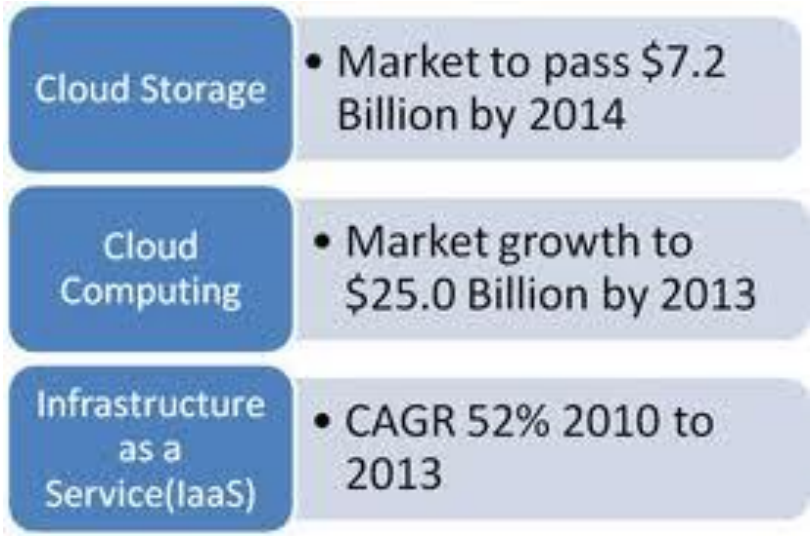

Figure 1.1 Hardware and Software Infrastructure [7]

1.2 Evolution of Cloud Computing

There has always been a debate about the evolution of Cloud Computing and the most important point in that is Grid Computing. Some call Cloud Computing and Grid Computing the same phenomena while others call Cloud Computing an extension of Grid computing. To find the truth we need to know about the Grid computing. Grid Computing is a complex phenomenon which has evolved through earlier developments in parallel, distributed and HPC (High Performance Computing) [8]. Some of the organizations have also defined the Grid computing with respect to the features. According to IBM "Grid computing allows you to unite pools of servers, storage systems, and networks into a single large system so you can deliver the power of multiple-systems resources to a single user point for a specific purpose. To a user, data file, or an application, the system appears to be a single enormous virtual computing system." [9]. So the definition of Cloud Computing earlier and of Grid computing here shows that Cloud Computing and grid computing have many similarities. So it is important to discuss about the differences in these two technologies. The technical differences among Grid computing and Cloud Computing [8] are shown below in the Table 1.1.

Table: 1.1 Differences between Grid Computing and Cloud Computing [8]**.**

	Grid Computing	**Cloud Computing**
Means of Utilization	Allocation of multiple servers onto a single task or job.	Virtualization of servers; one server to compute several tasks concurrently.
Typical usage pattern	Typically used for job execution, i.e. the execution of a program for a limited time	More frequently used to support long-running services.
Level of Abstraction	Provide high level of detail	Provide higher-level Abstractions
User Management	Decentralized and also Virtual Organization based	Centralized or can be delegated to third party

As presented in the table what makes Cloud Computing different from Grid computing is "virtualization". Cloud Computing supplements virtualization to maximize the computing power. Virtualization resolves some of the challenges faced by grid computing. Grid computing achieves high utilization by the allocation of multiple servers onto a single task or job. In Cloud Computing the virtualization of servers achieves high utilization by allowing one server to compute several tasks concurrently [10]. Along with the differences in technology among Grid computing and Cloud Computing, the usage patterns are also different between them. Grid is commonly used for job execution while clouds are more frequently used to support long-running services. Thus we can summarize that Grid Computing is the starting point and basis for Cloud Computing. Cloud [11] Computing necessarily represents the increasing trend towards the external deployment of IT resources, which includes computational power, storage, business applications and obtaining them as services.

1.3 Cloud Computing Service Models

- **Software-as-a-Service (SaaS).** The SaaS service model offers the services as applications to the consumer, using standardized interfaces. The cloud provider is responsible for the management the application, operating systems and underlying infrastructure. The services run on top of a cloud infrastructure and are invisible for the consumer. The consumer can only control some of the user-specific application configuration settings. Example: Rackspace Mosso, Web Fusion [12] .
- **Platform-as-a-Service** (PaaS). The PaaS service model offers the services as operation and development platforms to the consumer. The consumer does not manage the underlying cloud infrastructure such as network, operating systems, servers, or storage, but only has control over the deployed applications and possibly application hosting environment configurations [3]. The consumer can use the platform to develop and run his own applications, supported by a cloud-based infrastructure Examples: Microsoft Windows Azure, Google App Engine,
- **Infrastructure-as-a-Service** (IaaS). The IaaS service model is the lowest service model in the technology stack, offering infrastructure resources as a service which includes raw data storage, processing power and network capacity [6]. The consumer can the use IaaS based services to deploy his own operating systems and applications hence offer a wider variety of deployment possibilities for a consumer than the PaaS and SaaS models. The consumer does not manage or

control the underlying cloud infrastructure but has control over operating systems; storage, deployed applications, and has limited control of select networking components (e.g., host firewalls). Example: Amazon EC2.

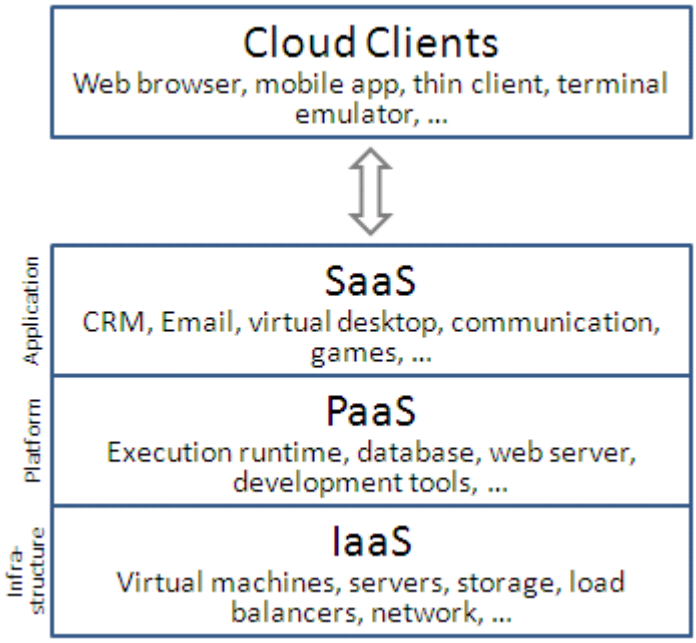

Figure 1.2 Cloud Computing Service Models [6]

1.4 Cloud Computing Deployment Models

Cloud offerings can be deployed in four capital ways, each with their own characteristics. These characteristics to describe the deployment models are (i) Who owns the infrastructure; (ii) Who manages the infrastructure (iii) Where *is* the infrastructure located; (iv) Who accesses the cloud services.

- **Public clouds:** Public cloud computing is based on massive scale offerings to the general public. The infrastructure is located on the premises of the provider, who owns and manages the cloud infrastructure. Public cloud users are considered to be untrusted, which implies that they are not tied to the organization as employees and that the user has no contractual agreements with the provider [13].
- **Private clouds**: Private clouds run in service of a single organization and resources are not shared by other entities. The infrastructure may be owned by and/or physically located in the organization's datacenters (on-premise) or that of a designated service provider (off-premise) with an extension of management and security control planes controlled by the organization or designated service

provider respectively [14]. Private cloud users are considered as trusted by the organization; as they are have contractual agreements with the organization.

- **Community clouds**: In the community deployment model, the infrastructure is shared by several organizations with the same policy and compliance considerations. Community clouds run in service of a community of organizations, those having the same deployment characteristics as private clouds. Community cloud users are also considered as trusted by the organizations that are part of the community.
- **Hybrid clouds.** Hybrid clouds are a combination of all the cloud models, public, private, and community clouds. Hybrid clouds leverage the capabilities of each cloud deployment model. Every part of a hybrid cloud is connected to the other by a gateway, controlling the applications and data that flow from each part to the other. Where private and community clouds are managed and located on either organization or third party provider side per characteristic, hybrid clouds is having these characteristics on both organization and third party provider side [15]. In hybrid clouds untrusted users are prevented to access the resources of the private and community parts of the hybrid cloud. A hybrid cloud is typically offered in one of two ways: a vendor has a private cloud and forms a partnership with a public cloud provider, or a public cloud provider forms a partnership with a vendor that provides private cloud platforms.

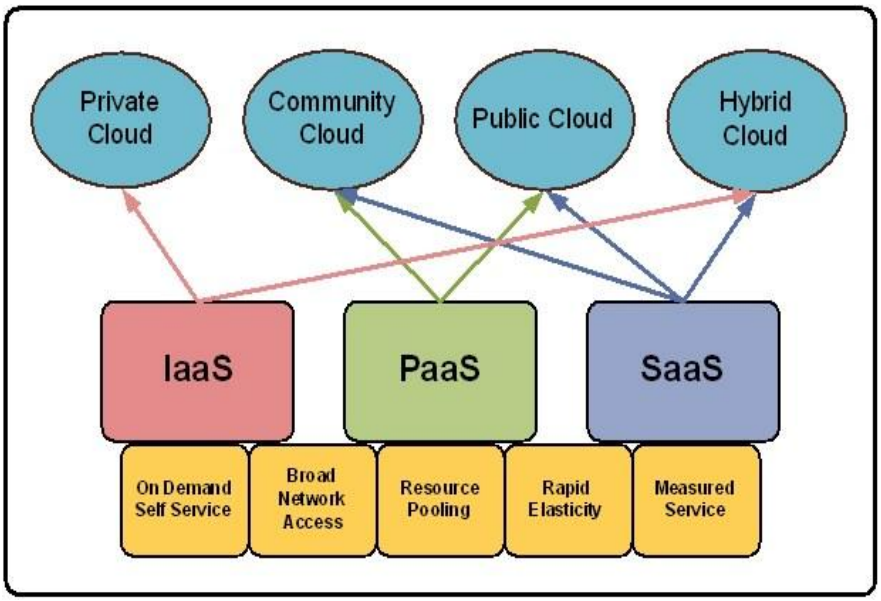

Figure 1.3 Cloud Computing Deployment Models [15]**.**

1.5 Cloud Computing Key Characteristics

- **On-demand self-service.** Cloud computing resources can be procured and disposed of by the consumer without human interaction with the cloud service provider. This automated process reduces the personnel overhead of the cloud provider, cutting costs and lowering the price at which the services can be offered.
- **Resource pooling.** By using a technique called virtualization, the cloud provider pools his computing resources. This resource pool enables the sharing of virtual and physical resources by multiple consumers, dynamically assigning and releasing resources according to consumer demand [16]. The consumer has no explicit knowledge of the physical location of the resources being used, except when the consumer requests to limit the physical location of his data to meet legal requirements.
- **Broad network access.** Cloud services are accessible over the network via standardized interfaces, enabling access to the service not only by complex devices such as personal computers, but also by light weight devices such as smart phones.
- **Rapid elasticity.** The available cloud computing resources are rapidly matched to the actual demand, quickly increasing the cloud capabilities for a service if the demand rises, and quickly releasing the capabilities when the need for drops. This automated process decreases the procurement time for new computing capabilities when the need is there, while preventing an abundance of unused computing power when the need has subsided.
- **Measured service.** Cloud computing enables the measuring of used resources, as is the case in utility computing. The measurements can be used to provide resource efficiency information to the cloud provider, and can be used to provide the consumer a payment model based on "pay-per-use." For example, the consumer may be billed for the data transfer volumes, the number of hours a service is running, or the volume of the data stored per month.
- **Ultra large-scale:** The scale of cloud is large. The cloud of Google has owned more than one million servers. Even in Amazon, IBM, Microsoft, Yahoo, they have more than hundreds of thousand servers. There are hundreds of servers in an enterprise. Cloud enlarges the user's computing power.

- **Versatility:** Cloud computing doesn't aim at certain special application. It can produce various applications supported by cloud, and one cloud can support different applications running it at the same time.
- **Consumption-based billing:** Consumers pay for only what resources they use and therefore are charged or billed on a consumption-based model.

1.6 Cloud Computing Issues

In the last few years, cloud computing has grown from being a promising business concept to one of the fastest growing segments of the IT industry. Presently recession-hit companies are increasingly realizing that simply by tapping into the cloud they can gain fast access to best-of-breed business applications or drastically boost their infrastructure resources, all at minimal cost [17]. But as more and more information on individuals and companies is placed in the cloud more concerns are beginning to grow about the fact that how safe an environment it is. Main issues related to Cloud Computing are:

- **Privileged User Access:** Data stored and processed outside the enterprises direct control, brings with an inherent level of risk, because outsourced services bypass the physical, logical and personnel controls IT shops exert over in-house programs. So it is more important to get as much information as possible about the people who manage your data and the controls they implement.
- **Regulatory Compliance:** Data owners are responsible for the integrity and confidentiality of their data, even when the data is outside their direct control, which is the case with external service providers such as cloud providers. Where traditional service providers are forced to comply with external audits and obtain security certifications so should cloud computing providers [17]. The leading cloud providers do not support on-site external audits on customer's request. As a result, some compliance cannot be achieved because on-site auditing is a requirement that cannot be satisfied.
- **Data Location:** The exact location of data in the cloud is often unknown. Data may be located in systems in other countries, which may be in conflict with regulations prohibiting data to leave a country or union.
- **Data Segregation:** The shared, massive scale characteristics of cloud computing makes it likely that one's data is stored alongside data of others consumers. Encryption is often used to segregate data-at-rest, but it is not a cure-all. It is

important to do a thorough evaluation of the encryption systems used by the cloud provider [18]. A proper built, but poorly managed encryption scheme may be just as lethal as no encryption at all, because as the confidentiality of data may be preserved, availability of data may be at risk when data availability is not guaranteed.

- **Recovery:** Cloud providers should have recovery mechanisms in place in case of a disaster. Any provider which does not replicate the data and application infrastructure across multiple sites is at a risk of total failure [19]. Cloud providers should provide its guidelines concerning business continuity planning, detailing how long it will take for services to be fully restored.
- **Investigative Support:** Investigating inappropriate or illegal activity may be impossible in cloud computing, because logging and data may be replicated and spread across ever-changing sets of hosts and data centers.
- **Data Lock-in:** Availability of customer's data may be at risk if a cloud provider shuts down or is acquired by another organization. Providers should provide procedures how customers can retrieve their data when the needed, and in which format the data is presented to the customer. If the data is presented in a format patented to the cloud provider, it may be unusable by any other provider [18]. The use of open standards by providers to prevent data lock-in is recommended, but not always supported.
- **Legal Issues:** Regardless of efforts to bring into line the lawful situation, as of 2009, supplier such as Amazon Web Services provide to major markets by developing restricted road and rail network and letting users to choose availability zones [20]. On the other hand, worries stick with safety measures and confidentiality from individual all the way through legislative levels.

1.7 Advantages of Cloud Computing

If used properly working with data in the cloud can be very beneficial Mentioned below are some of the advantages of this technology:

- **Cost Efficient:** Cloud computing is probably the most cost efficient method to use and maintain. Traditional desktop software costs companies a lot in terms of cost. Adding up the licensing fees for multiple users can prove to be very expensive for the establishment concerned. On the other hand cloud is available at much cheaper rates and hence, can significantly decrease the company's IT expenses [21].

- **Backup and Recovery:** As all of data is stored in the cloud, backing and restoring the same is relatively much easier than storing the same on a physical device. Not only that, most cloud service providers are usually competent enough to handle recovery of information. So, this makes the entire process of backup and recovery much simpler than other traditional methods of data storage [21].
- **Minimizing Startup Costs:** For companies that are just starting out, organizations in emerging markets, or "Skunk Works" groups in larger organizations, cloud computing greatly reduces startup costs. The new organization starts with an infrastructure already in place, so the time and other resources that would be spent on building a data center are borne by the cloud provider, whether the cloud is private or public [18].
- **Easy Access to Information:** Once user gets registered in the cloud, he can access the information from anywhere with an Internet connection. This convenient feature lets users to move beyond time zone and geographic location issues.
- **Streamlining the Data Center**: An organization of any size will have a substantial investment in its data center. That includes buying and maintaining the hardware and software, providing the facilities in which the hardware is housed and hiring the personnel who keep the data center running. An organization can streamline its data center by taking advantage of cloud technologies internally or by offloading workload into the public [21].
- **Improving Business Processes:** The cloud provides an infrastructure for improving business processes. An organization and its suppliers and partners can share data and applications in the cloud, allowing everyone involved to focus on the business process instead of the infrastructure that hosts it.
- **Quick Deployment:** Cloud computing gives the advantage of quick deployment. Once user opts for this method of functioning, the entire system can be fully functional in a matter of a few minutes. The amount of time taken here will depend on the exact kind of technology those user needs [22].
- **Lower IT operating costs**: Organizations can rent added server space for a few hours at a time rather than maintain proprietary servers without worrying about upgrading their resources. They also have the flexibility to host their virtual IT infrastructure in locations offering the lowest cost.

1.8 Organization of Thesis

Chapter 2 - This chapter discusses the concept of virtualization, its benefits, VM migration, its advantages and difficulties that occur during vm migration, migration policies, scheduling process, scheduling algorithms, workflow, workflow management system and work flow fault tolerance techniques.

Chapter 3 - This chapter describes the simulation and simulation tools used in experimental setup which includes cloudsim, its architecture, design and implementation, gridsim, its features and its architecture.

Chapter 4 - This chapter discusses about the gap analysis and need and significance of the research work.

Chapter 5 - This Chapter discusses about the solution to the problem and the proposed algorithm.

Chapter 6 - This chapter focuses on the performance and result evaluation of our results.

Chapter 7 - This chapter discusses the conclusions of work presented in this thesis. The chapter ends with a discussion of the future direction which thesis work can take.

Chapter 2

Literature Survey

This chapter discusses the concept of virtualization, its benefits, VM migration, its advantages and difficulties that occur during vm migration, migration policies, scheduling process, scheduling algorithms, workflow, workflow management system and work flow fault tolerance techniques.

2.1 Virtualization

Virtual machine (VM) technology has recently emerged as an essential building-block for data centers and cluster systems, largely due to its capabilities of migrating and consolidating workload [23]. Altogether, these features allow a data center to serve multiple users in a secure and efficient way. These virtualized infrastructures are considering a key component to drive the emerging Cloud Computing paradigm. Migration of virtual machines seeks to improve efficiency and fault tolerance of systems. More specially, the reasons that justify VM migration in a production system which includes: the need to balance system load, done by migrating VMs out of overloaded/overheated servers; and the need of selectively bringing servers down for maintenance after migrating their workload to other servers. The facility to migrate an entire operating system overcomes most difficulties that traditionally have made process-level migration a complex operation [24]. The applications themselves and their corresponding processes do not need to be aware that a migration is occurring. Virtualization the Cloud Computing paradigm allows workloads to be deployed and scaled-out quickly through the rapid provisioning of virtual machines or physical machines. Any request of resources will be delivered by Cloud in terms of Virtual Machine. So placement of Virtual machine is most important part in Cloud Computing [20].

2.1.1 Virtualization Benefits

Virtualization is the single most effective way to reduce IT expenses while boosting efficiency and agility. The benefits include:

- On a single computer, running multiple applications and operating systems.
- Consolidating hardware to get vastly higher productivity from fewer servers.
- Save 50% or more on overall IT costs.

- Speed and simplify IT maintenance, deployment and management of new applications [25].
- For software testing a testbed can be created with virtual machines.
- For software distribution, software can be installed on one virtual machine, which can be distributed as virtual appliance, requiring few configuration changes [25]

2.2 VM Migration

Cloud computing infrastructure consists of resource virtualization and resource scheduling based on service level agreement. Resource virtualization forms resource set by collecting resources that can be virtualized. This resource set is in turn converted to dynamic resource set that is utilized dynamically as demand arises. Scheduling of dynamic resources assigns resources to demands according to the service level agreement [20]. The underlying technologies and issues for cloud computing, therefore, include resource virtualization, scalable resource management, and load balancing of resources across time and location, and quality of service. Efficient assignment and scheduling of resources is yet to be developed to dynamically match workload demands in the absence of a clear picture of the future usage. Human intervention, therefore, is indispensable and infrastructure administrators manually schedule and/or move virtualized resources to sustain the fluctuating demands, resulting in added burden on the already complex operation [26]. Solutions to these pressing needs for cloud computing infrastructure hinge on virtual machine migration. VM migration is designed to move VMs from an overloaded physical machine to a lightly loaded machine. Moving VMs will lessen the burden on the overloaded machine while utilizing the idling physical machine [20]. Numerous critical issues need to be addressed to make VM migration approach successful. Among the issues is an important parameter threshold that dictates what constitutes a machine underutilized or overloaded. If the overall resource utilization of a physical machine is over a certain fixed threshold, the machine is deemed overloaded and one or more of the VMs may be selected for migration. The survey of Milojicic [27] identifies the following categories of reasons for using migration:

- Accessing more processing power (in terms of load balancing).
- Exploitation of resource locality (for performance).
- Commercial relation exists between clouds.

- Resource sharing (meaning sharing of expensive and rare resources or large amounts of free memory by processes over a network).
- Fault resilience.
- Transmission of mass data.
- Load Balancing.
- Reconfiguration.

2.2.1 Advantages of VM Migration

Migration can be considered to be inherently related up to a point to resource virtualization. For instance, when considering migration for improving locality, for example the case of moving a server closer to a database that it makes use of, obviously the best solution we could think of, if feasible, would be to actually "teleport" (in as little time as possible) the actual physical resource/machine on which the server runs (also including all its outer network connections - wired and/or wireless, if possible) to the new location, without disrupting the service at all, if we can. Then obviously problems like transparent process checkpointing or transparent connection redirection would basically disappear, while the goal of improving performance by bringing the server closer to the data has been achieved. Since physical teleportation is not yet available as an "empowering technology", the next best thing we can do is to actually "teleport" a virtualized version of the physical machine (ideally also including its network connections), i.e. perform a virtual machine migration [20]. It is therefore clear that migrating entire VMs brings an overwhelming advantage in terms of implementation effort and simplicity over migrating individual processes, since the running state of the VM is readily available and does not have to be extracted.

2.2.2 Difficulties Related to VM Migration

To perform a correct migration, besides the checkpointed state of the VM, the memory image of that VM also has to be migrated, for the state to be correctly preserved. All this should be done while programs in the VM are still running; therefore memory pages are still getting dirtied. Therefore, we see that increased simplicity comes at a certain price, as the VM’s memory image and state is undoubtedly much larger than the process checkpointed state in the case of process migration [27]. This cost comes in the form of a quite large performance penalty, but, as will be seen in the next section, this can be ameliorated by techniques previously

employed in the process migration field. Additionally, as with all migrations, resources used by processes running within the migrated VM should still be available after the migration attached. Since these resources might be hard to migrate (because of large sizes or consistency constraints for instance), this brings back the problem of residual dependencies. For instance, the problem of migrating the file system present on the virtual disk of a Virtual Machine [25]. In the context of Virtual Machine migration, "residual dependencies" are especially important, considering that the size of a virtual machine can be much larger than that of a process. While migrating the entire VM has the advantage that support for checkpointing is readily provided, unlike in the case of a regular process, the VM's address space, and especially it's virtual disk are of considerable size, therefore leaving residual dependencies may be unavoidable to ensure reasonable migration times (at least with current typical network resources). As with all migration systems, transparency remains an issue also for VM migration [20].

2.3 Existing Work on VM migration Policies

- Elmroth et al. [28] have formulated technology, neutral interfaces and architectural additions for handling migration, and monitoring of VMs in cloud environments. The interfaces presented follow the practices of general requirements of scalability and security in addition to specific requirements related to the particular issues of exchanging and using information (interoperability) and business relationships between competing cloud computing infrastructure providers. They may be used locally and remotely which create a layer of abstraction that simplifies management of virtualized service components.
- Beloglazov et al. [29] have proposed a method for dynamic consolidation of VMs based on adaptive utilization thresholds, ensuring a high level of reaching the SLA (Service Level Agreements).They also validate the high efficiency of the proposed technique across different kinds of workloads using workload traces from more than a thousand Planet Lab servers. Dynamic consolidation of virtual machines (VMs) and switching idle nodes off allow Cloud providers to optimize resource usage and reduce energy consumption.
- Beloglazov et al. [30] have invented an efficient resource management policy for virtualized Cloud data centers. The main objective is to continuously consolidate VMs live migration and switch off idle nodes to minimize power consumption,

providing required Quality of Service. Modern Cloud computing environments have to provide high Quality of Service (QoS) for their customers resulting in the necessity to deal with power performance trade-off.

- Voorsluys et al. [31] have formulated a performance evaluation on the effects of live migration of virtual machines on the performance of applications that run inside Xen VMs. The results shows that in most of the cases, migration overhead is acceptable but they cannot be disregarded, mainly in systems where service availability and responsiveness are governed by strict Service Level Agreements (SLAs). Despite that, there is a high potential for live migration applicability in data centers serving enterprise-class internet applications. The capability of virtual machine (VM) migration brings multiple benefits such as higher performance, manageability and fault tolerance. Live migration of VMs often allows work-load movement with a short service downtime.
- Sonnek et al. [32] have formulated a decentralized affinity-aware migration technique that incorporates heterogeneity and dynamism in network topology and job communication patterns to allocate virtual machines on the available physical resources. This technique monitors network affinity between pairs of VMs and uses a distributed bartering algorithm which is coupled with migration, to dynamically adjust VM placement so that communication overhead is minimized. Besides, their technique is able to adjust to dynamic variations in communication patterns and provides both good performance and low network contention with minimal overhead.

2.4 Fault Tolerance Based VM Migration Policy & Consolidation of VMs

Several heuristics for dynamic consolidation of VMs based on an analysis of historical data of the resource usage by VMs are proposed.

2.4.1 Dynamic Consolidation Issue:

The following are the main four parts of dynamic VM consolidation:

- To determine when a host is considered as being overloaded requiring migration of one or more VMs from this host.
- To determine when a host is considered as being under loaded leading to a decision to migrate all VMs from this host and switch the host to the sleep mode.
- Selecting VMs that should be migrated from an overloaded host.

- To find a new placement of the VMs selected for migration from the overloaded and under loaded hosts [33].

2.4.2 Analysis of Policies for VM migration.

It mainly policies for VM migration are:

- **The Minimum Migration Time Policy**: The Minimum Migration Time (MMT) policy migrates a VM v that requires the minimum time to complete a migration relatively to the other VMs allocated to the host. The migration time is estimated as the amount of RAM utilized by the VM divided by the spare network bandwidth available for the host j.
- **The Random Choice Policy**: The Random Choice (RC) policy selects a VM to be migrated according to a uniformly distributed discrete random variable whose values index a set of VMs allocated to a host.
- **The Maximum Correlation Policy**: The Maximum Correlation (MC) policy is based on the idea that the higher the correlation between the resource usage by applications running on an oversubscribed server, the higher the probability of the server overloading. According to this idea, we select those VMs to be migrated that have the highest correlation of the CPU utilization with other VMs [33]. To estimate the correlation between CPU utilizations by VMs, we apply the multiple correlation coefficients. It is used in multiple regression analysis to assess the quality of the prediction of the dependent variable. The multiple correlation coefficients correspond to the squared correlation between the predicted and the actual values of the dependent variable. This can also be interpreted as the proportion of the variance of the dependent variable explained by the independent variables [30]. So the main idea is that the higher the correlation between the resource usage by applications running on an oversubscribed server, the higher the probability of the server overloading.

2.5 Fault Tolerance

Fault-tolerant computing is the art and science of building computing systems that continue to operate satisfactorily in the presence of faults the ability for a system to remain in operation even if some of the components used to build the system fail [30]. Fault tolerance is a main concern for the assurance of availability and reliability of critical services as well as application execution. To minimize failure impact on the system and application execution, failures must be predicted and proactively handled.

In some cases, if fault occurs, a reactive measure needs to be implemented to protect the system from its consequences Fault tolerance techniques are used to predict these failures and take an appropriate action before failures actually occur. Fault tolerance makes a system capable to operate correctly in a faulty condition, protect against accidental or malicious destruction of information, protect against generating erroneous output and guarantees that confidential information cannot be divulged. Fault Tolerance is one of the key issues of cloud computing. Fault tolerance is concerned with all the techniques necessary to enable a system to tolerate software faults remaining in the system after it has been developed.

These faults may or may not manifest themselves during systems operations, but when they do, and then the fault tolerant techniques should provide the necessary mechanisms of the system to prevent system failure occurrences [30]. With increasing heterogeneity and complexity of computational clouds, and due to the inherent unreliable nature of large-scale cloud infrastructure, fault tolerance techniques have become a major concern. Fault tolerance techniques are designed to allow a system to tolerate software faults that remain in the system after its development. Fault tolerance techniques are employed during the planning, or development, of the software. If a fault occurs, these techniques provide mechanisms to the software system to prevent system failure from occurring [34]. It should also be incorporated in autonomic cloud computing environment.

2.6 Scheduling Process

Task scheduling is one of the major activities which are performed in all the computing environments. To increase the working of cloud computing environments, task scheduling is one the main tasks performed in order to gain maximum profit [35]. The goal of scheduling algorithms is spreading the load on processors and maximizing their utilization while minimizing the total task execution time.

Scheduling process in cloud can be generalized into three stages as:

- Resource Discovering and Filtering – Datacenter Broker discovers all the resources present in the network system and collects status information related to them.
- Resource Selection – Target resource is selected based on some parameters of task and resource.
- Task Submission -Task is submitted to resource that has been selected.

There are various types of scheduling algorithm for distributed computing system. Most of them are applied in the cloud environment with suitable verifications. The main advantage of job scheduling algorithm is to achieve the best system throughput and a high performance computing. Main categories of scheduling algorithms are:

- **First Come First Serve Algorithm:** Job in the queue which comes first is served. This algorithm is simple and fast.
- **Round Robin Algorithm**: In the round robin scheduling the processes are dispatched in a FIFO manner but they are given a limited amount of CPU time called a time-slice. If a process does not complete before its CPU-time expires then the CPU is given to the next process waiting in a queue.
- **Min–Min algorithm:** This algorithm chooses small tasks to be executed firstly, which in turn large task delays for long time.
- **Max – Min algorithm:** This algorithm chooses large tasks to be executed firstly, which in turn small task delays for long time.
- **Most fit task scheduling algorithm:** In this algorithm task which fit best in queue are executed first. This algorithm has high failure ratio.
- **Priority scheduling algorithm:** The basic idea is straightforward: each process is assigned a priority, and priority is allowed to run. Equal-Priority processes are scheduled in FCFS order. The shortest-Job-First (SJF) algorithm is a special case of general priority scheduling algorithm. An SJF algorithm is simply a priority algorithm where the priority is the inverse of the (predicted) next CPU burst. That is, the longer the CPU burst, the lower the priority and vice versa. Priority can be defined either internally or externally. Internally defined priorities use some measurable quantities or qualities to compute priority of a process [35].

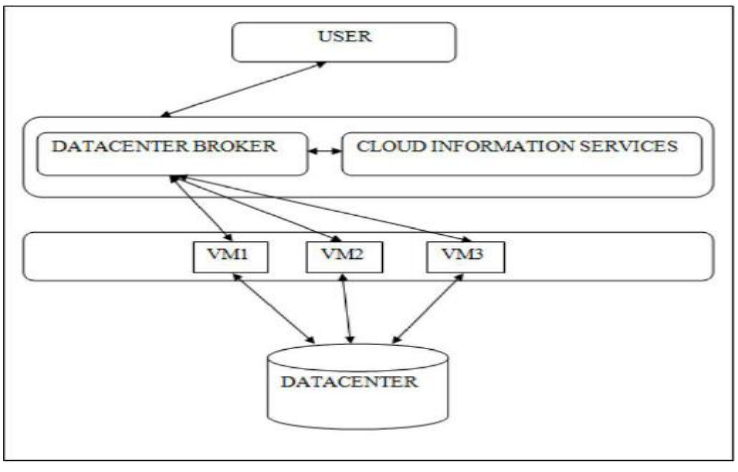

Figure 2.1 Scheduling Process [35]

2.6.1 Existing Scheduling Algorithms

For the scheduling process, the following scheduling algorithms has been discussed:

- Resource-Aware-Scheduling algorithm (RASA): Saeed Parsa et al. [36] proposed a new task scheduling algorithm RASA. It is composed of two traditional scheduling algorithms; Max-min and Min-min. RASA uses the advantages of Max-min and Min-min algorithms and covers their disadvantages. Though the deadline of each task, arriving rate of the tasks, cost of the task execution on each of the resource, cost of the communication are not considered. The experimental results show that RASA is outperforms the existing scheduling algorithms in large scale distributed systems.
- RSDC (Relaible Scheduling Distributed in Cloud Computing): Arash Ghorbannia Delavar et al. [35] proposed a reliable scheduling algorithm in cloud computing environment. In this algorithm major job is divided to sub jobs. In order to balance the jobs the request and acknowledge time are calculated separately. The scheduling of each job is done by calculating the request and acknowledges time in the form of a shared job. So that efficiency of the system is increased.
- An Optimal Model for Priority based Service Scheduling Policy for Cloud Computing Environment: Dr. M. Dakshayini et al. [37] proposed a new scheduling algorithm based on priority and admission control scheme. In this algorithm priority is assigned to each admitted queue. Admission of each queue is decided by calculating tolerable delay and service cost. Advantage of this algorithm is that this policy with the proposed cloud architecture has achieved very high (99%) service completion rate with guaranteed QoS. As this policy provides the highest precedence for highly paid user service-requests, overall servicing cost for the cloud also increases
- A Priority based Job Scheduling Algorithm in Cloud Computing: Shamsollah Ghanbari and Mohamed Othman [38] proposed a new scheduling algorithm based on multi – criteria and multi - decision priority driven scheduling algorithm. This scheduling algorithm consist of three level of scheduling: object level, attribute level and alternate level. In this algorithm priority can be set by job resource ratio. Then priority vector can be compared with each queue. This algorithm has higher throughput and less finish time.

- Extended Max-Min Scheduling Using Petri Net and Load Balancing: El-Sayed T. et al. [39] have proposed a new algorithm based on impact of RASA algorithm. Improved Max-min algorithm is based on the expected execution time instead of complete time as a selection basis. Petri nets are used to model the concurrent behavior of distributed systems. Max-min demonstrates achieving schedules with comparable lower makespan rather than RASA and original Max-min.
- An Optimistic Differentiated Job Scheduling System for Cloud Computing: Shalmali Ambike et al. [40] have proposed a differentiated scheduling algorithm with non-preemptive priority queuing model for activities performed by cloud user in the cloud computing environment. In this approach one web application is created to do some activity like one of the file uploading and downloading then there is need of efficient job scheduling algorithm. The Qos requirements of the cloud computing user
- Improved Cost-Based Algorithm for Task Scheduling: Mrs.S.Selvarani, and Dr.G.Sudha [41] proposed an improved cost-based scheduling algorithm for making efficient mapping of tasks to available resources in cloud. The improvisation of traditional activity based costing is proposed by new task scheduling strategy for cloud environment where there may be no relation between the overhead application base and the way that different tasks cause overhead cost of resources in cloud. This scheduling algorithm divides all user tasks depending on priority of each task into three different lists. This scheduling algorithm measures both resource cost and computation performance, it also Improves the computation/communication ratio.
- Performance and Cost evaluation of Gang Scheduling in a Cloud Computing System with Job Migrations and Starvation Handling: Helen D. Karatza has proposed a gang scheduling algorithm with job migration and starvation handling in which scheduling parallel jobs, already applied in the areas of Grid and Cluster computing. The number of Virtual Machines (VMs) available at any moment is dynamic and scales according to the demands of the jobs being serviced. The aforementioned model is studied through simulation in order to analyze the performance and overall cost of Gang Scheduling with migrations and starvation handling. Results highlight that this scheduling strategy can be effectively

deployed on Clouds, and that cloud platforms can be viable for HPC or high performance enterprise applications.

2.7 Workflow

A workflow expresses an automation of procedures wherein files and data are passed between procedures applications according to a defined set of rules, to achieve an overall goal [42]. It is the automation of a business process, in whole or a part, during which the information, documents or tasks are passed from one participant to another for action, according to a set of procedural rules [43]. A workflow is composed by connecting multiple tasks according to their dependencies. It consists of a sequence of steps that simplifies the complexity of execution and management of applications. The operational aspects of workflow are:

- How tasks are structured.
- Who performs them.
- What their relative order is.
- How they are synchronized.
- How the information flows to support the tasks.
- How tasks are being tracked.

2.7.1 Workflow Management (WFM)

It is a technology supporting the reengineering of business and information processes. It mainly involves:

1. Defining workflows, i.e., describing those aspects of a process that are relevant to controlling and coordinating the execution of its tasks (and possibly the skills of individuals or information systems required to perform each task).

2. Providing for fast (re)design and (re)implementation of the processes as business needs and information systems change [43].

To effectively support WFM, organizations must evolve their existing computing environments to a new distributed environment so that it:

- Is component-oriented, i.e., supports integration and interoperability among loosely coupled components corresponding to heterogeneous, autonomous, and/or distributed (HAD) legacy and new systems.
- Supports different workflow applications corresponding to business or information process implementations accessing multiple HAD systems.

- Ensures the reliability and correctness of applications in the presence of concurrency and failures.
- Supports the evolution, addition and replacement of workflow applications and component systems as processes are reengineered [44].

2.7.2 Workflow Failure Handling Techniques

Workflow execution failure can occur for various reasons: non-availability of required services or software components, the variation in the execution environment configuration, system running out of memory, overloaded resource conditions and faults in computational and network fabric components. Workflow management systems should be able to identify and handle failures and support reliable execution in the presence of concurrency and failures Hwang [45] divided workflow failure handling techniques into two different levels as task-level and workflow-level. The task-level techniques mask the effects of the execution failure of tasks in the workflow and the workflow-level techniques manipulate the workflow structure such as execution flow to deal with erroneous conditions.

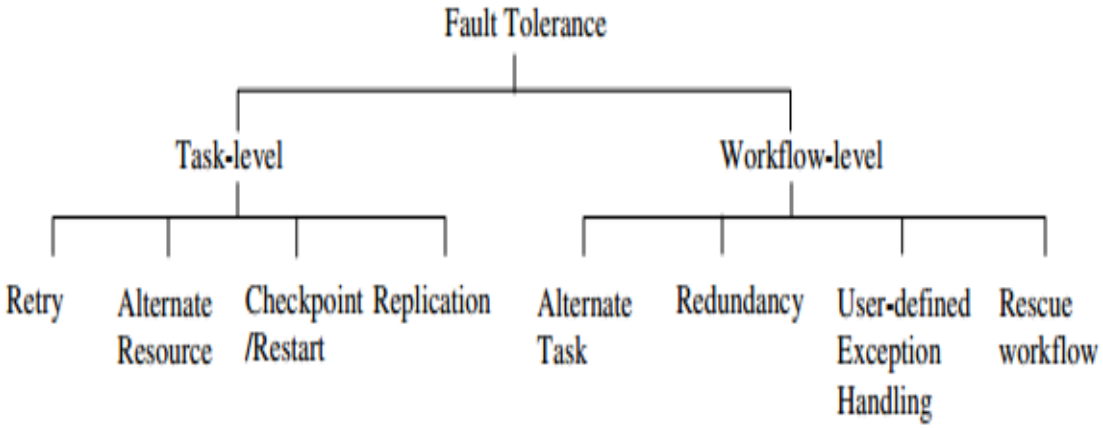

Figure 2.2 Fault Tolerance Taxonomy [46]

Task-level techniques have been widely studied in parallel and distributed systems. They can be cataloged into retry, alternate resource, checkpoint/restart and replication. The retry technique is the simplest failure recovery technique, as it tries to execute the same task on the same resource after failure has occurred. The alternate resource technique simply submits failed task to another resource. The checkpoint/restart technique moves failed tasks transparently to other resources and the task continue its execution from the point of failure. Replication technique [47] runs the same task simultaneously on different resources to ensure task execution provided that at least one of the replicas does not fail.

Workflow-level techniques include alternate task, user-defined exception handling, redundancy and rescue workflow. The first three approaches proposed in [45] assume there is more than one implementation for a certain computation with different execution characteristics. Alternate task technique executes another implementation of a certain task if the previous one failed and the redundancy technique executes multiple alternative tasks simultaneously. User-defined exception handling allows the users to specify a special treatment for a certain failure of a task in a workflow. The rescue workflow technique developed in Condor DAGMan system [48] ignores the failed tasks and continues to execute the remainder of the workflow until no more forward progress can be made. A rescue workflow description called rescue DAG, which indicates failed nodes with statistical information which is generated for later submission.

Chapter 3

Experimental Setup

This chapter describes the simulation and simulation tools used in experimental setup which includes the architecture, design and implementation of cloudsim and gridsim.

3.1Simulation

Simulation is a technique where a program models the behavior of the system (CPU, network etc.) by calculating the interaction between its different entities using mathematical formulas, or actually capturing and playing back observations from a production system. The available Simulation tools in Cloud Computing today are: Simjavas, Gridsim and CloudSim.

3.2 Simulation Tools.

These tools open up the possibility of evaluating the hypothesis in a controlled environment where one can easily reproduce results. Simulation-based approaches offer significant benefits to IT companies (or anyone who wants to offer his application services through clouds) by allowing them to:

- Test their services in repeatable and controllable environment.
- Tune the system bottlenecks before deploying on real clouds.
- Experiment with different workload mix and resource performance scenarios on simulated infrastructures for developing and testing adaptive application provisioning techniques [49].

3.3 CloudSim

CloudSim is a framework developed by the GRIDS laboratory of University of Melbourne which enables seamless modeling, simulation and experimenting on designing Cloud computing infrastructures. CloudSim is a self-contained platform which can be used to model data centers, host, service brokers, scheduling and allocation policies of a large scaled Cloud platform. This CloudSim framework is built on top of GridSim framework which is also developed by the GRIDS laboratory. CloudSim is an extensible simulation toolkit that enables modeling and simulation of Cloud computing systems and application provisioning environments [50]. The CloudSim toolkit supports both system and behavior modeling of Cloud system components such as data centers, virtual machines (VMs) and resource provisioning policies. It implements generic application provisioning techniques that can be

extended with ease and limited effort. Currently, it supports modeling and simulation of Cloud computing environments consisting of both single and inter-networked clouds (federation of clouds). Moreover, it exposes custom interfaces for implementing policies and provisioning techniques for allocation of VMs under inter-networked Cloud computing scenarios. Several researchers from organizations, such as HP Labs in U.S.A., are using CloudSim in their investigation on Cloud resource provisioning and energy-efficient management of data center resources. CloudSim is a new, generalized, and extensible simulation framework that allows seamless modeling, simulation, and experimentation of emerging Cloud computing infrastructures and application services [50]. By using CloudSim, researchers and industry-based developers can test the performance of a newly developed application service in a controlled and easy to set-up environment. The simulation framework has the following novel features:

- Support for modeling and instantiation of large scale Cloud computing infrastructure, including data centers on a single physical computing node and java virtual machine.
- A self-contained platform for modeling datacenters, service brokers, scheduling, and allocations policies.
- Availability of virtualization engine, which aids in creation and management of multiple, independent, and co-hosted virtualized services on a data center node.
- Flexibility to switch between space-shared and time-shared allocation of processing cores to virtualized services [51].

3.3.1 CloudSim Architecture

Figure 3.1 shows the multi-layered design of the CloudSim software framework and its architectural components. Initial releases of CloudSim used SimJava as discrete event simulation engine that supports several core functionalities, such as queuing and processing of events, creation of Cloud system entities (services, host, data center, broker, virtual machines), communication between components, and management of the simulation clock. However in the current release, SimJava layer has been removed in order to allow some advanced operations that are not supported by it.

The CloudSim simulation layer provides support for modeling and simulation of virtualized Cloud-based data center environments including dedicated management interfaces for virtual machines (VMs), memory, storage, and bandwidth [50]. The

fundamental issues such as provisioning of hosts to VMs, managing application execution, and monitoring dynamic system state are handled by this layer. A Cloud provider, who wants to study the efficiency of different policies in allocating its hosts to VMs (VM provisioning), would need to implement their strategies at this layer. Such implementation can be done by programmatically extending the core VM provisioning functionality. There is a clear distinction at this layer related to provisioning of hosts to VMs.

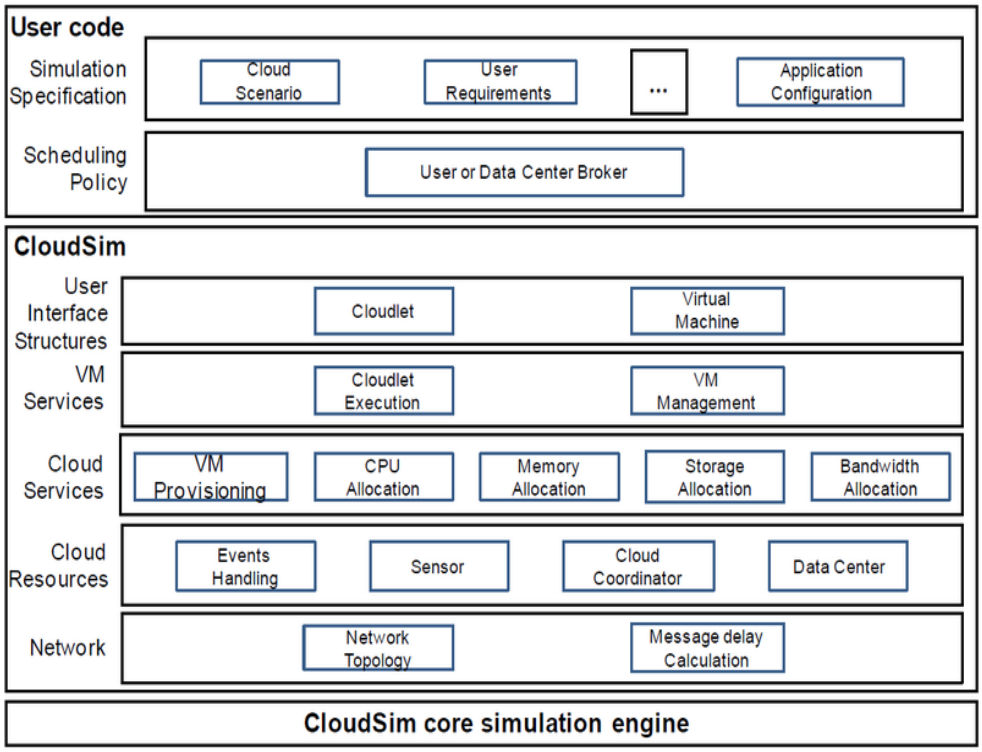

Figure 3.1 - Layered CloudSim Architecture [51]**.**

A Cloud host can be concurrently allocated to a set of VMs that execute applications based on SaaS provider's defined QoS levels. This layer also exposes functionalities that a Cloud application developer can extend to perform complex workload profiling and application performance study. The top-most layer in the CloudSim stack is the User Code that exposes basic entities for hosts (number of machines, their specification and so on), applications (number of tasks and their requirements), VMs, number of users and their application types, and broker scheduling policies. By extending the basic entities given at this layer, a Cloud application developer can perform following activities:

- Generate a mix of workload request distributions, application configurations.

- Model Cloud availability scenarios and perform robust tests based on the custom configurations.
- Implement custom application provisioning techniques for clouds and their federation.

As Cloud computing is still an emerging paradigm for distributed computing, there is a lack of defined standards, tools and methods that can efficiently tackle the infrastructure and application level complexities. Hence, in the near future there would be a number of research efforts both in academia and industry towards defining core algorithms, policies, and application benchmarking based on execution contexts. By extending the basic functionalities already exposed with CloudSim, researchers will be able to perform tests based on specific scenarios and configurations, thereby allowing the development of best practices in all the critical aspects related to Cloud Computing [50].

3.3.2 Design and Implementation of CloudSim

The fundamental classes of CloudSim [50], which are also the building blocks of the simulator, are:

- **BwProvisioner:** This is an abstract class that models the policy for provisioning of bandwidth to VMs. The main role of this component is to undertake the allocation of network bandwidths to a set of VMs that are deployed across the data center. The BwProvisioningSimple allows a VM to reserve as much bandwidth as required, but this is constrained by the total available bandwidth of the host.
- **CloudCoordinator:** This abstract class is liable for monitoring the internal state of data center resources. Based on this dynamic load-shredding decisions are taken. Implementation of this component includes the policy that should be followed during load-shredding. Monitoring of data center resources is performed by the updateDatacenter() method [51].
- **Cloudlet:** This class models the Cloud-based application services such as content delivery and business workflow. Each application service has a pre-assigned instruction length and data transfer overhead which it needs to undertake during its life-cycle. This class can also be extended to support modeling of other performance metrics for applications such as transactions in database-oriented applications.

- **CloudletScheduler:** This abstract class is extended by implementation of different types of policies that determine the share of processing power among Cloudlets in a virtual machine.
- **Datacenter:** This class models the infrastructure level services, offered by Cloud providers. It includes a set of compute hosts that can either be homogeneous or heterogeneous with respect to their hardware configurations such as memory, cores, capacity, and storage). Each Datacenter component instantiates a generalized application provisioning component that implements a set of policies for allocating memory, bandwidth, and storage devices to hosts and VMs.
- **DatacenterBroker:** This class models a broker, which is responsible for mediating negotiations between SaaS and Cloud providers; and such negotiations are driven by QoS requirements. The broker acts on behalf of SaaS providers. It discovers suitable Cloud service providers by querying the Cloud Information Service (CIS) and undertakes on-line negotiations for allocation of resources/services that can meet application's QoS needs.
- **DatacenterCharacteristics:** This class contains configuration information of data center resources.
- **Host:** This class models a physical resource such as a compute or storage server. It encapsulates important information such as the amount of memory and storage, a list and type of processing cores (to represent a multi-core machine), an allocation of policy for sharing the processing power among virtual machines, and policies for provisioning memory and bandwidth to the virtual machines.
- **Vm:** This class models a virtual machine, managed and hosted by a Cloud host component. Every VM component has access to a component that stores some characteristics related to a VM which are: processor, storage size, and the VM's internal provisioning policy, extended from an abstract component called the CloudletScheduler
- **VmScheduler:** This is an abstract class implemented by a Host component which models the policies required by the VMs for allocating processor cores.
- **VmAllocationPolicy:** This abstract class represents a provisioning policy that a VM Monitor utilizes for allocating VMs to Hosts. The main functionality of the VmAllocationPolicy is to select available host in a data center that meets the memory, storage, and availability requirement for a VM deployment.

- **RamProvisioner:** This is an abstract class that represents the provisioning policy for allocating primary memory (RAM) to the VMs.

3.4 GridSim

The GridSim toolkit allows modeling and simulation of entities in parallel and distributed computing (PDC) systems-users, applications, resources, and resource brokers (schedulers) for design and evaluation of scheduling algorithms. It provides a comprehensive facility for creating different classes of heterogeneous resources that can be aggregated using resource brokers for solving compute and data intensive applications [52]. The GridSim simulation framework enables users to:

- Model and simulate parallel and distributed computing systems.
- Determine effective and efficient resource allocation through simulation.
- Explore the significance of local economy and global positioning of resources.

3.4.1 GridSim Features

Main features of GridSim include:

- Incorporating Grid resource failures during runtime.
- Supporting advance reservation of a grid system.
- Reading workload traces from supercomputers to simulate a realistic grid environment.
- Supporting data grid simulations.
- Utilizing a network topology to link together resources and other entities.
- Simulating background network traffic based on a probabilistic distribution to simulate a congested public network.
- Simulating an experiment with multiple virtual organizations.
- New allocation policy can be made and integrated into the GridSim.
- Has the infrastructure or framework to support advance reservation of a grid system.

3.4.2 GridSim Architecture

The GridSim architecture shown below in figure 3.3 is built as a set of layered components. The base foundation is SimJava that is used for communication between the GridSim components [50]. The first layer is contains scalable Java's interface Java Virtual Machine. Forming the second layer are the Core Elements that are used to model Grid resources. Modeling and simulation of core Grid entities such as resources, information services is done in the third layer. The fourth layer is

concerned with the simulation of resource aggregators called grid resource brokers or schedulers. Finally, the fifth and sixth layers consist of components to assist users in implementing schedulers, testing algorithms, defining scenarios, and creating configurations to validate algorithms [53].

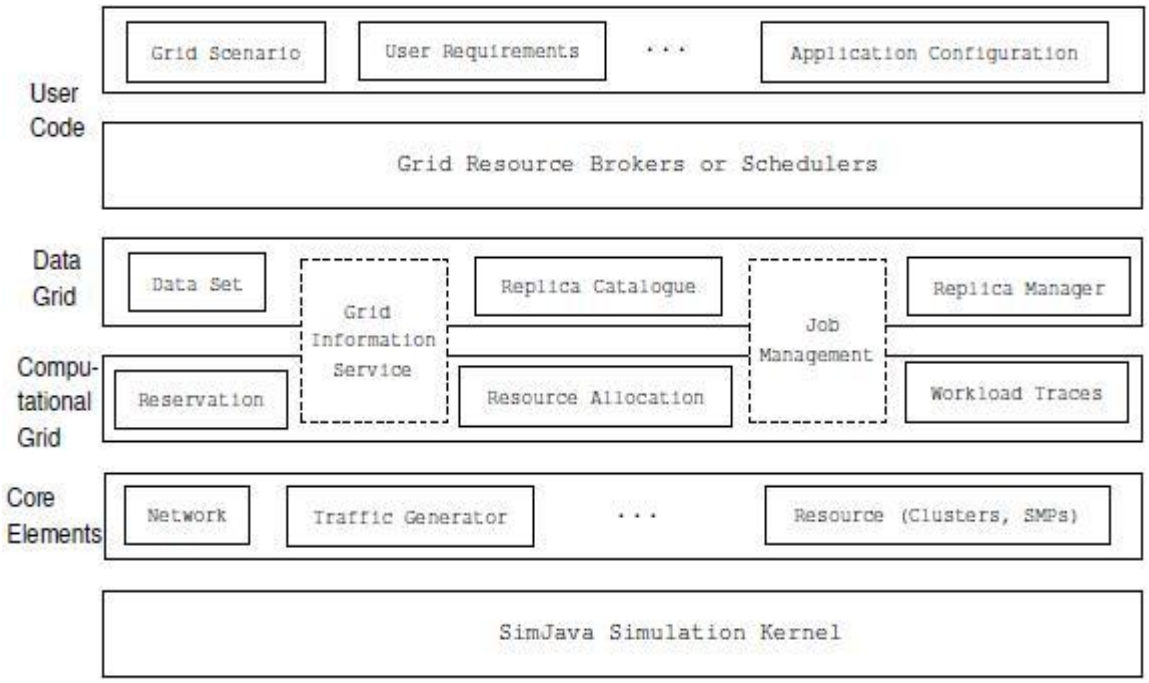

Figure 3.2 GridSim Architecture [52]

Chapter 4

Problem Statement

This chapter discusses about the gap analysis, need and significance of the research work.

4.1 Gap Analysis

Cloud computing is recently a booming area and has been emerging as a commercial reality in the information technology domain. However the technology is still not fully developed. There are still some areas which need improvements. One major area of concern is the Task scheduling. As large scale data processing is increasingly common in cloud computing systems, so it should be able to process the data efficiently in a given frame of time. The scheduling of tasks to the adaptable resources in accordance with adaptable time involves finding out a proper sequence in which tasks can be executed is needed. In such a scenario, tasks should be scheduled efficiently such that the response time, waiting time and execution time can be reduced. For applications such as message passing, parallel applications or multitier web applications, modeling in what order of priority tasks must be executed is very important. So to make effective use of the remarkable abilities of the cloud, efficient scheduling algorithms are mandatory. These scheduling algorithms are generally applied by cloud resource manager to optimally dispatch workloads to the cloud resources. Therefore extension of cloud simulation framework, NetworkCloudsim [54] has been implemented which supports modeling of real cloud datacenter and applications like E-commerce and workflow. The problem of task scheduling is critical not only to achieve high Cloud performance, but also to satisfy various cloud consumers' demands in an equitable fashion thereby enhancing the overall performance of the cloud computing environment.

4.2 Need and Significance of Research

Workflow applications often require very complex execution environments. However, NetworkCloudsim [54] uses only simple four stages in which NetworkCloudlet can be: Send, Receive, Executed and Finished. There is no level of priority used. But for scheduling multitier applications the currently implemented algorithm requires modifications as multitier applications are more event based and synchronized. So when workflow application are introduced in datacenter prioritization of tasks is

needed to be done in such a way that datacenter works smoothly without the bottleneck of load, bandwidth and dependencies. So it is important to design a scheduling algorithm for scheduling of workloads in the Cloud environment so as to simultaneously minimize the execution time, response time and waiting time. Secondly task level scheduling, prioritization of tasks and provision of resources are important issues in both Grid and Cloud Computing, so there is a need to develop a new approach for task level scheduling to increase the overall efficiency.

Chapter 5

Proposed Statement

This Chapter discusses about the solution to the problem and the proposed algorithm.

5.1 Solution to the Problem

Workflows have been used to represent a variety of applications involving high processing. As mentioned in Gap Analysis and Literature Survey there is a need to enhance the tasks of a workload which must incorporate the concept of finding priority in which these tasks must be executed. The Cloud workflow applications often consists of several tasks and each task is implemented by several substitute Cloud services. QoS and resource utilization are necessary and play more important roles in Cloud service workflow applications. In order to efficiently and effectively schedule the tasks and data of applications among cloud services, end user QoS-based scheduling schemas should be implemented, such as those for minimizing total execution time, waiting time, response time and balancing the load of resources. So scheduler needs to take into account the computation stages of applications. Therefore, a new scheduler is formulated to schedule tasks.

Priority scheduling approaches can be broadly classified into three categories:

- Task-level Fixed-Priority (TFP) scheduler assigns a fixed priority to all the jobs of each single task.
- Job-level Fixed-Priority (JFP) scheduler assigns a fixed priority to each single job, and a Job-level
- Dynamic-Priority (JDP) scheduler assigns a priority to each single job that can dynamically change over time.

The main advantage of task level scheduling is high performance computing and the best system throughput. A simulation framework which supports the modeling of essential data center inputs has been presented. The goal of workflow scheduling is spreading the load on processors and maximizing their utilization while minimizing the total task execution time. Its main purpose is to schedule tasks to the adaptable resources in accordance with adaptable time, which involves finding out a proper sequence in which tasks can be executed. The main focuses of this thesis is to understand the fundamentals of task level scheduling and based on that, improved priority assignment algorithm is proposed. A new approach is developed in which

each tasklist is assigned a three level priority and then according to this priority tasklists are allowed to run.

5.2 X-Bar Chart

The X-bar chart is a type of control chart that is used to monitor the arithmetic means of successive samples of constant size, n. This type of control chart is used for characteristics that can be measured on a continuous scale. The X-bar chart displays the centerline, which is calculated using the grand average, and the upper and lower control limits, which are calculated using the average range. Future experimental subsets are plotted compared to these values [55]. This type of control chart is used to monitor a variable's data when samples are collected at regular intervals from process. The limits are calculated as:

- Upper Level = μ +3 ($\sqrt{\sigma}$ / n-1)
- Middle Level = Mean
- Lower Level = μ -3 ($\sqrt{\sigma}$ / n-1)

5.3 Proposed Algorithm

In algorithm 5.1 HUL is historical upper limit, a real integer that stores the upper limit calculated based on historical data using X- bar chart upper limit formula. HML is the historical middle limit and HLL is historical lower limit calculated from the middle level and lower level X-bar chart formulas respectively. Workflow is divided into n number of task list. Each tasklist is divided in m task stages. Available memory is the RAM available for execution of task and DS is the size of data that is being executed.

Algorithm 5.1 Task Aware Priority Based Scheduling Algorithm

Procedure Scheduler (Workflow, Availablememory)

1. Sumratio ← 0 ; HUL ← 0
2. HML← 0
3. HLL ← 0
4. ***for*** each Workflow ***do***
5. n ← |Tasklist|
6. ***for*** i = 1 to n ***do***
7. m ← |Taskstage|

8. ***for*** j = 1 to m ***do***
9. ***if*** i = 1 ***then***
10. Sumratio ← Sumratio + Availablememory / DS_j
11. μ ← Sumratio / n
12. σ ← (Sumratio – μ)^2/ n-1
13. HUL ← μ + 3 ($\sqrt{\sigma}$ / n-1)
14. HML ← μ
15. HLL ← μ - 3 ($\sqrt{\sigma}$ / n-1)
16. ***else***
17. CValue ← Availablememory / DS_j
18. ***if*** CValue ≤ HLL ***then***
19. $Priority_j$ ← HIGH
20. ***if*** CValue ≥ HUL ***then***
21. $Priority_j$ ← LOW
22. ***if*** CValue > HLL and CValue < HUL ***then***
23. $Priority_j$ ← MEDIUM
24. ***end if***
25. Add Taskstage into Priority queue
26. ***end for*** // j
27. ***end for*** // i

Algorithm 5.1 takes workflows and available memory as input and gives priority queue as output. Historical upper level, historical middle level and historical lower level are calculated only for the first time. After calculation of these levels, the new values of memory/data ratios of other task stages are compared with these limits. If the values are less than equal to lower limit, they are assigned high priority as small tasks gets executed fast. If the values are greater than lower limit and lower than higher limit, they are assigned middle level priority and if the values are greater than higher limits, they are assigned low priority as they will take more time to execute, increasing the average waiting time and processor overhead which will decreasing the overall efficiency. After the calculation of priorities all the subtasks are inserted in the priority queue according to their priorities. By prioritizing the task stages the whole task is optimized.

.Chapter 6

Experimental Results

This chapter focuses on the performance and result evaluation of our results. CloudSim is used to create the simulation environment.

6.1 Performance Evaluation

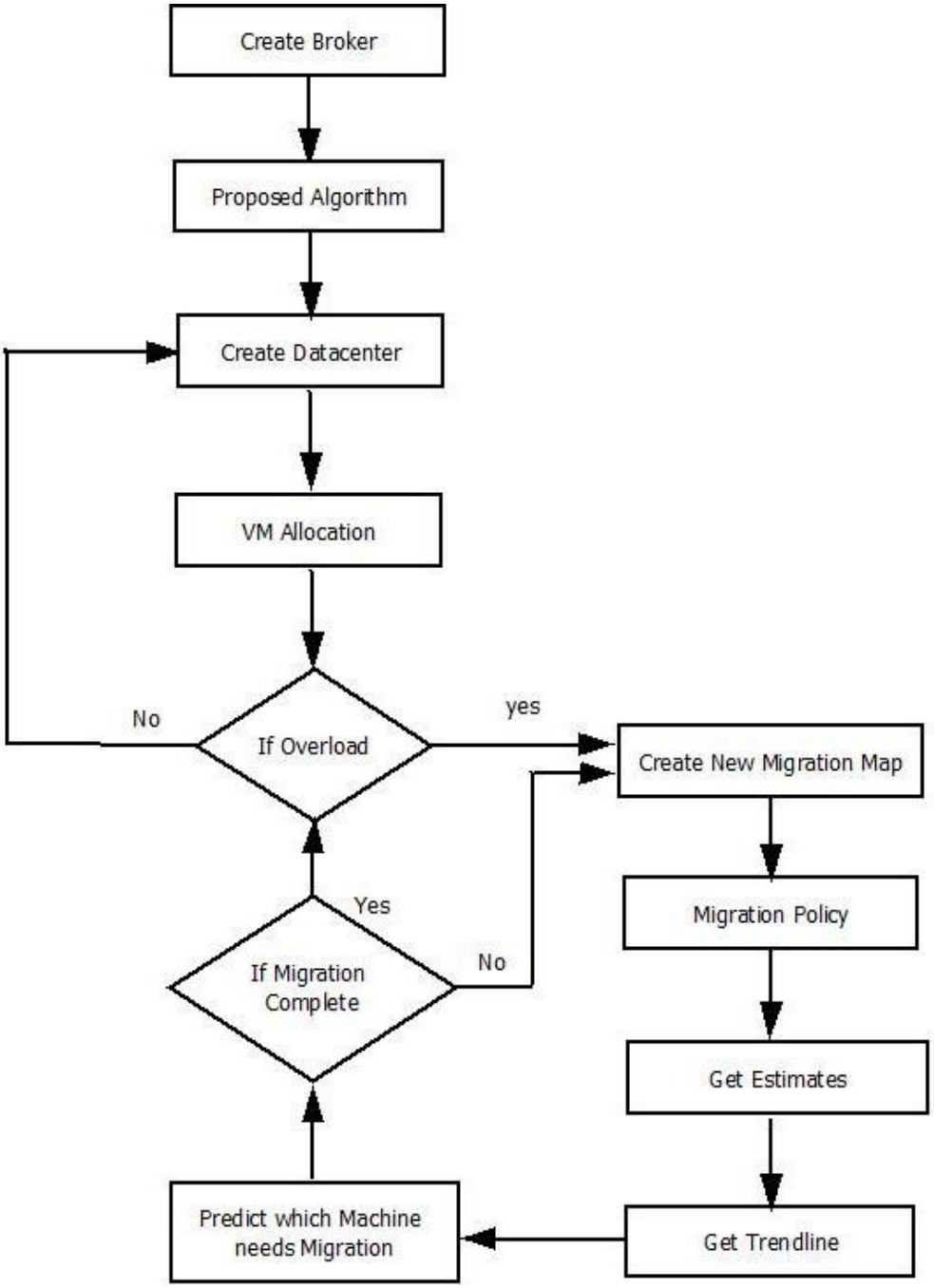

Figure 6.1: Flow Diagram of Experimental Framework

For the best evaluation of our algorithm we have formulated an experimental framework which involves the following steps:

- **Create Broker**: Firstly we create a Broker. It submits tasks to the datacenter. A Broker is responsible for mediating between users and service providers depending on users QoS requirements and it also deploys service tasks across clouds. The broker identifies suitable cloud service providers through the cloud Information Service (CIS) and negotiates with them for an allocation of resources that meets QoS needs of users.
- **Create Datacenter**: A data center, which is home to the computation power and storage, is central to cloud computing and contains thousands of devices. Datacenter models the core infrastructure level services (hardware, software) offered by resource providers in a Cloud computing environment. It encapsulates a set of compute hosts that can be either homogeneous or heterogeneous as regards to their resource configurations such as memory, cores, capacity, and storage.
- **VM Allocation**: VM simply means how the work must be allocated to the VMs. This is implemented by a Host component that models the policies (space-shared, time-shared) required for allocating processing power to VMs.
- **VM Migration**: It refers to the process of moving a running virtual machine or application between different physical machines without disconnecting the client or application.VM migration comes into play if there is a overload. Memory, storage, and network connectivity of the virtual machine are transferred from the original host machine to the destination. Migration of virtual machines improves performance, manageability and fault tolerance of systems.VM migration helps balance system load. In experimental Linear Regression Migration Technique is used.
- **Create Migration Map**: To make a migration Map, the VMs which are best to migrate to should be known. There are different policies to know this. Minimum Migration Time (MMT) VM selection policy is used in the experimental setup. MMT policy is used to know which VM requires the minimum time to complete the migration process compared to other VMs. Time consumed = Memory of VM / Bandwidth of host.

- **VM Migration Policy**: VM Migration comes into play if something goes wrong such as if overloading occurs or power failure etc. The provision of host to the virtual machine in a datacenter needs a careful thought process. It must address how the host must be reserved, and then if those reservations come under over-subscription, over-utilization and performance degradation then VM allocation policies helps to address these issues to that datacenters to take right decisions to re-allocate or migrate the VM machines. Overloading is done the datacenter of the experimental setup by sending extra data packets to the datacenter.
- **Get Estimates and Get Trend Line**: Based on understanding the utilization pattern of hosts, if there is overutilization of datacenters resources, a proactive response is built based on data points (data set of utilization metrics) by using localized subset of dataset, by which it is able to ignore the requirement for the specification of a function to fit a model and only smoothing parameter values. So a trend line is generated to know which VMs are being migrated.

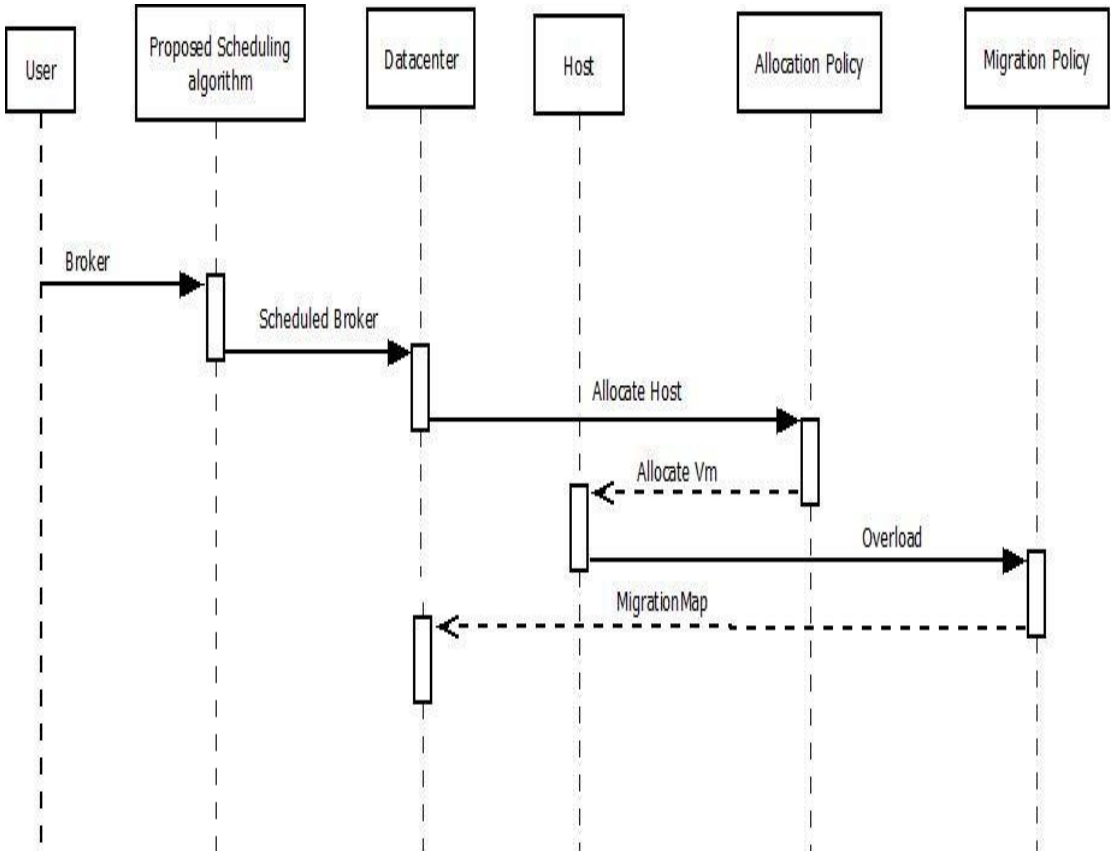

Figure 6.2 Sequence Diagram of Experimental Setup

6.2 Result Evaluation

For evaluation of result, outputs generated from the experimental setup are shown in figure 6.3 and figure 6.4.

Case 1: Figure 6.3 shows the successfully submitted cloudlets for the calculation of three levels of priority, the upper level, middle level and the low level.

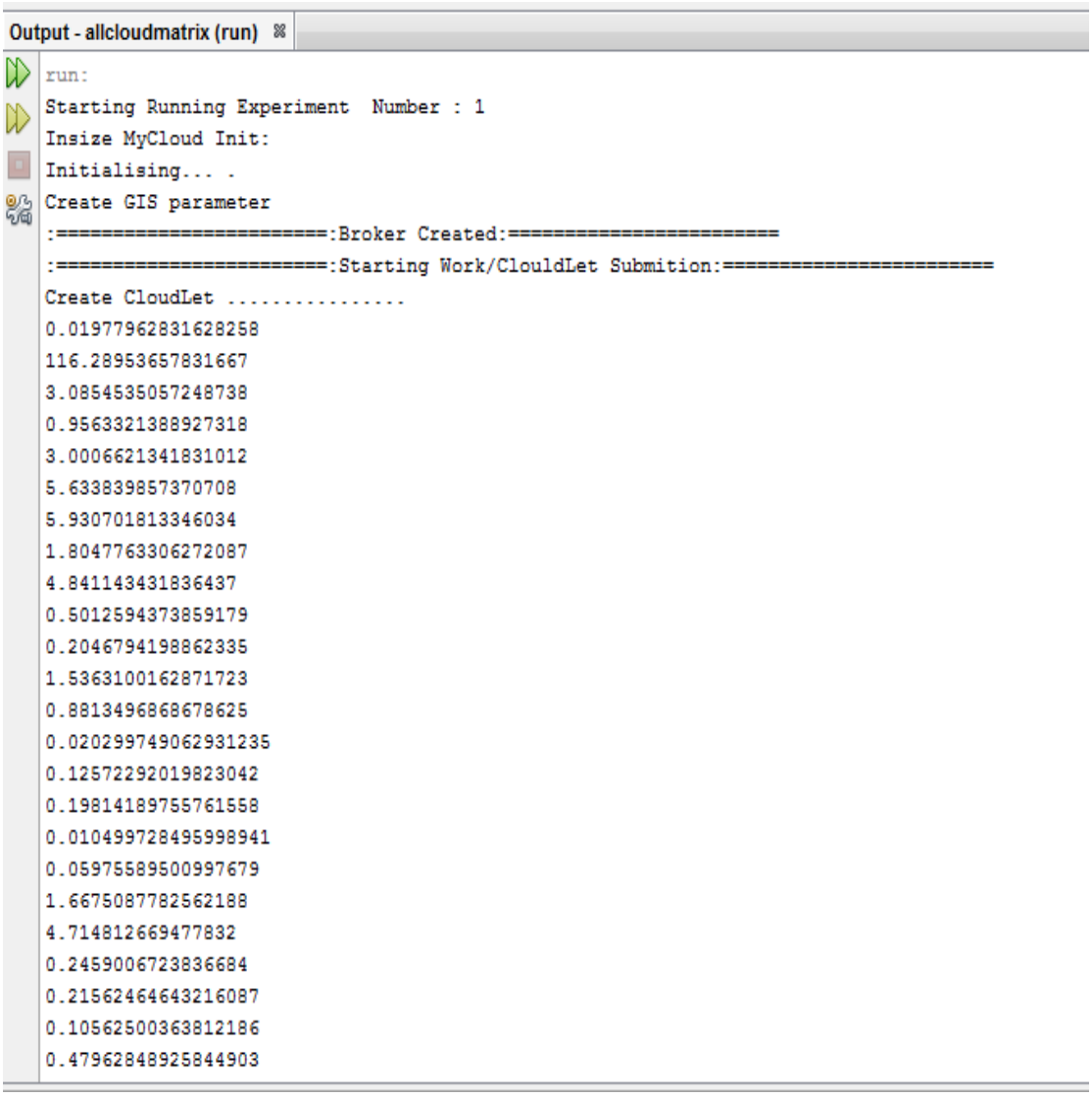

Figure 6.3 Cloudlet Submission

Case 2: Figure 6.4 shows the output generated after the calculation of the three levels of prioritization based on cloudlets which are successfully submitted to the datacenter.

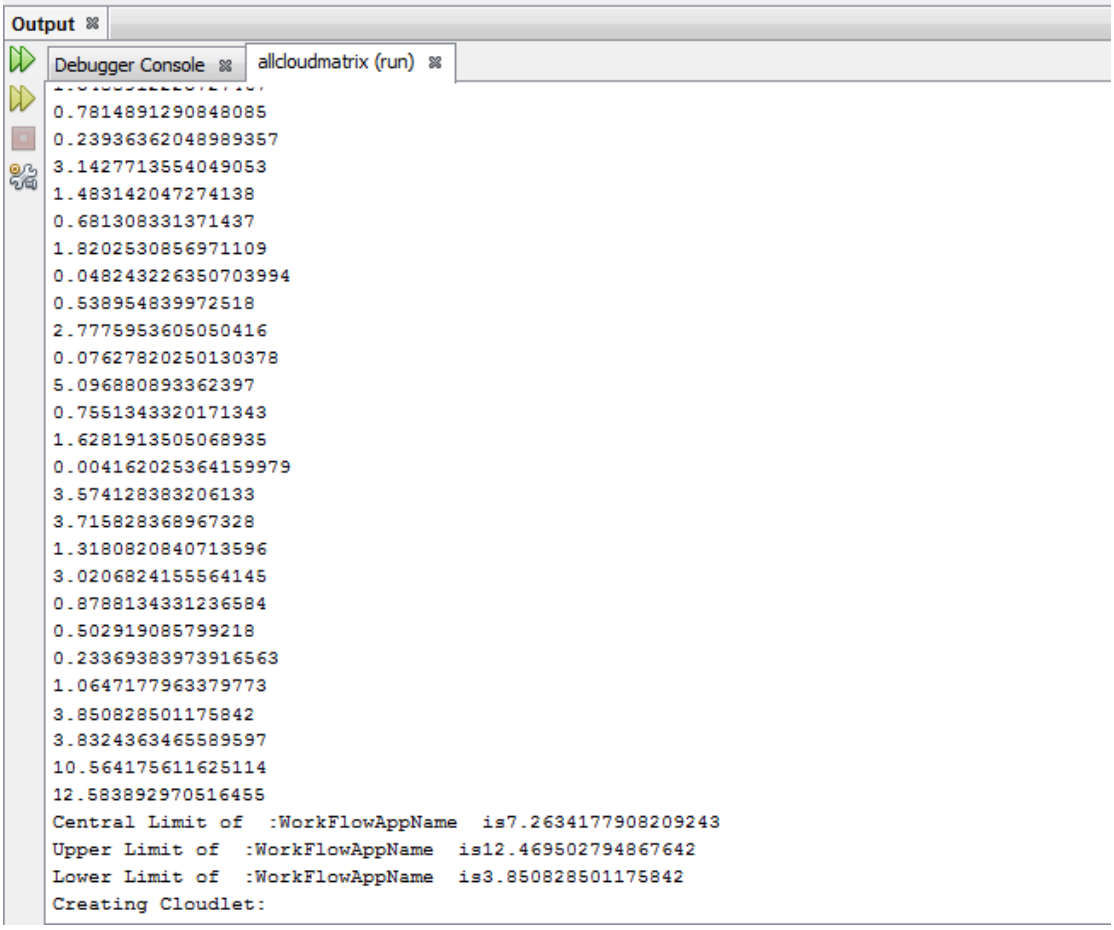

Figure 6.4 Calculation of levels of prioritization

The Execution Time, Response Time and Waiting time is calculated from the outputs generated by running the random overlap algorithm of base paper and the proposed algorithm.

- **Execution Time** = Finish Time – Start Time.
- **Waiting Time** = Arrival time + Execution Time.
- **Average Response Time** = Execution Time + Waiting Time.

The results calculation is done on the basis of 1616 successfully executed cloudlets. A comparative analysis is done for both the algorithms. For best evaluation of results the mean values of execution time, waiting time and response time are calculated. The

finals results are shown in graphical form so that the difference in the values can be clearly seen.

Case 3: Figure 6.5 shows the comparative analysis of mean execution time for both the algorithms.

Mean value of Execution Time for Proposed Algorithm = 16.7706

Mean value of Execution Time for Random Overlap Algorithm = 20.1718

Difference = 20.1718 - 16.7706 = 3.4012

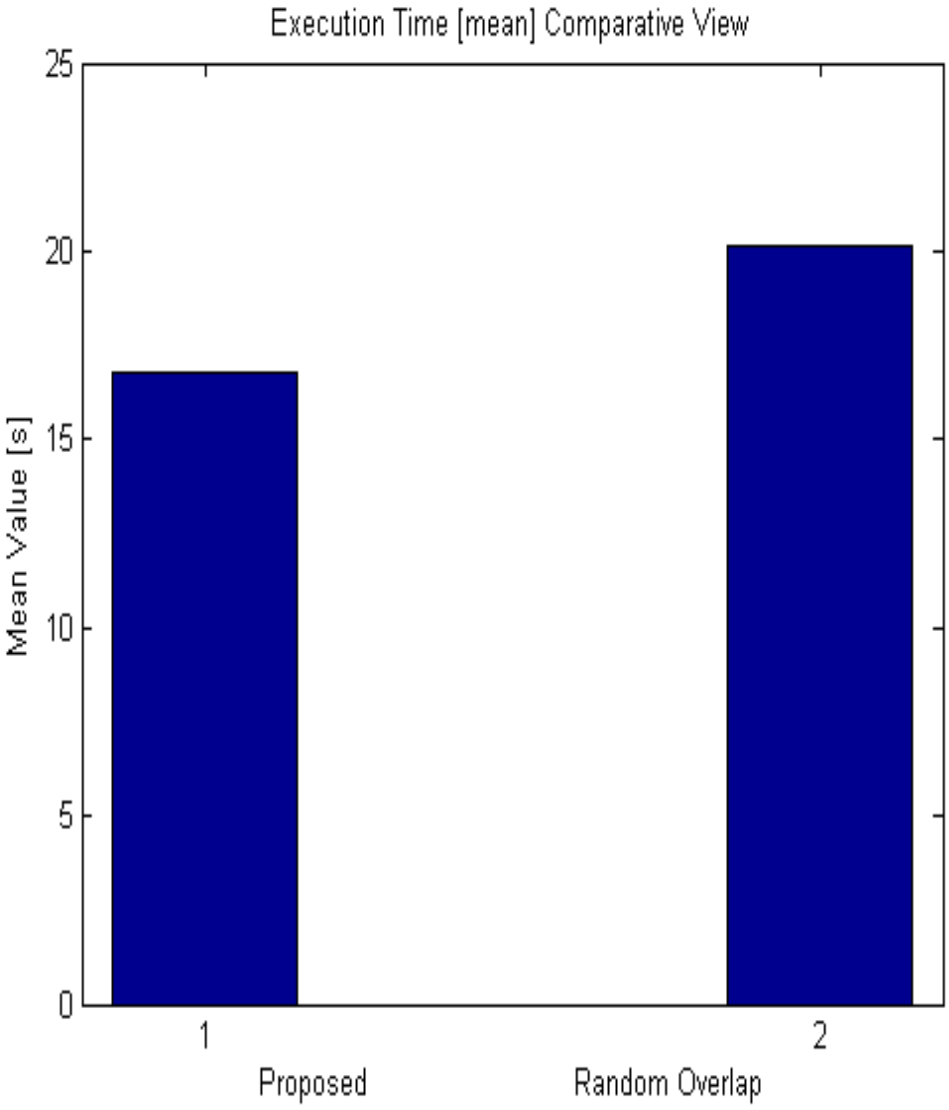

Figure 6.5 Comparative Analysis of Execution Time

Case 4 Figure: 6.6 shows the comparative analysis of mean waiting time for both the algorithms.

Mean value of Waiting Time for Proposed Algorithm = 0.0348

Mean value of Waiting Time for Random Overlap Algorithm = 0.5507

Difference = 0.5507 - 0.0348 = 0.2027

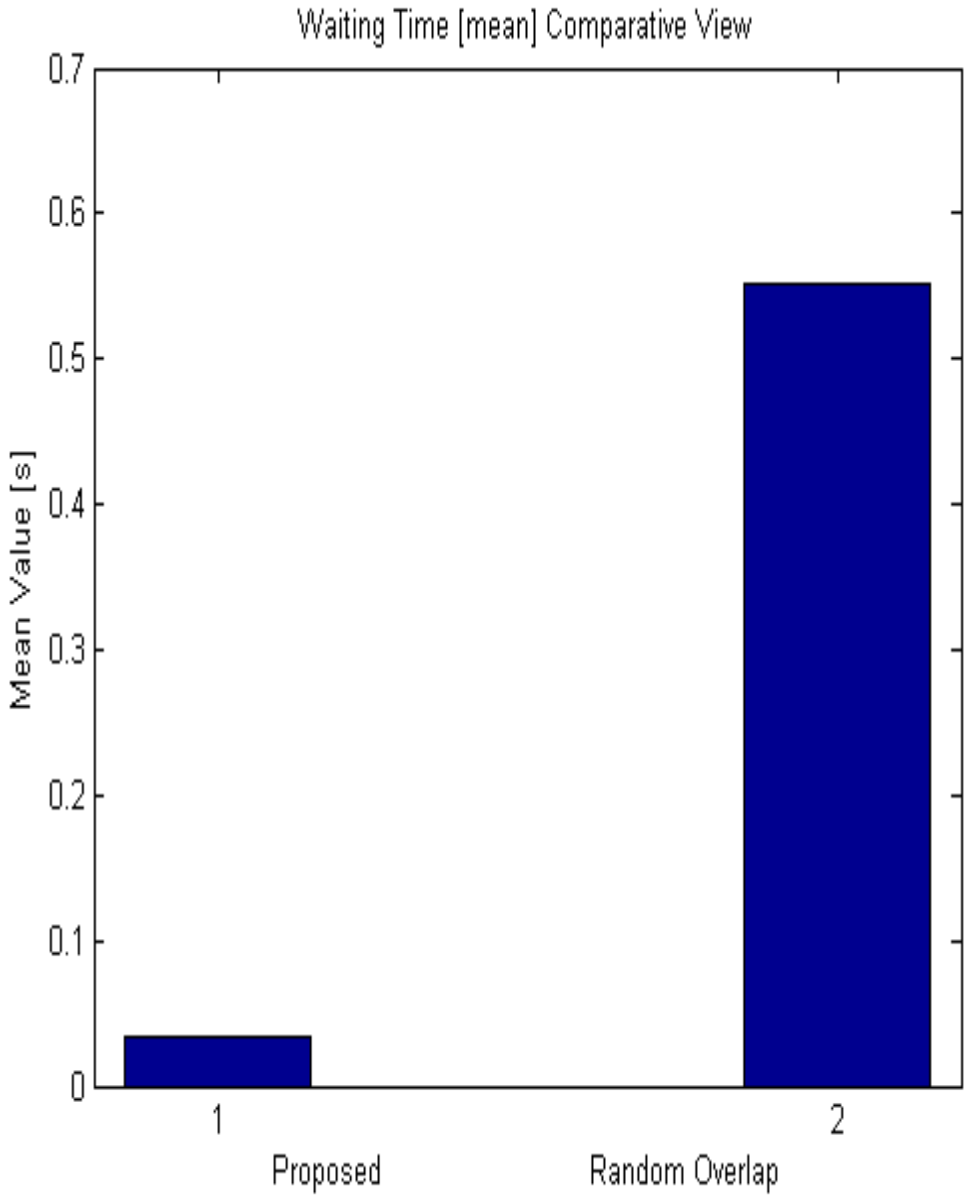

Figure 6.6 Comparative Analysis of Waiting Time

Case 5 Figure: 6.7 shows the comparative analysis of mean response time for both the algorithms.

Mean value of Response Time for Proposed Algorithm = 0.7279
Mean value of Response Time for Random Overlap Algorithm = 1.2135
Difference = 1.2135 - 0.7279 = 0.4856

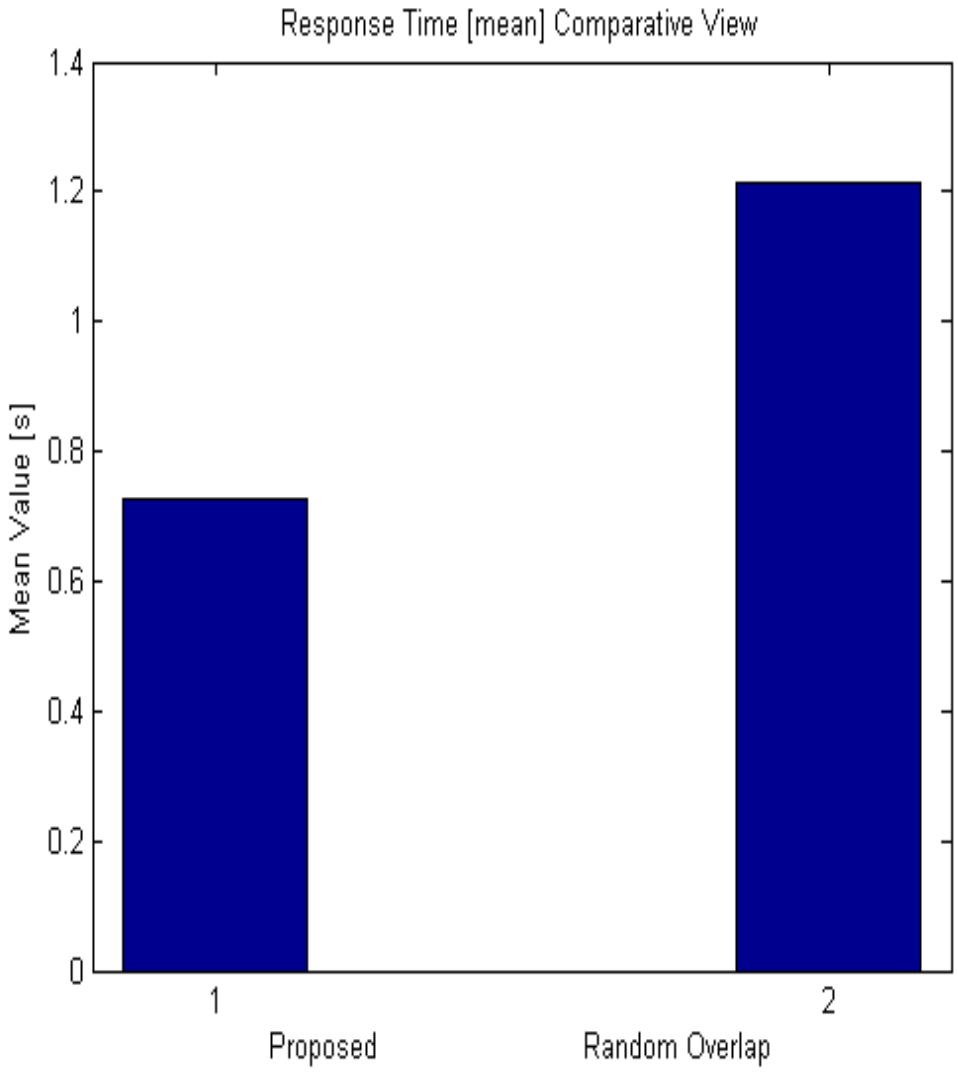

Figure 6.7 Comparative Analysis of Response Time

So it is clear from the above calculations that there is a successful decrease in the Response Time, Execution Time and Waiting Time for the proposed algorithm.

Chapter 7

Conclusion and Future Scope

This chapter discusses the conclusions of work presented in this thesis. The chapter ends with a discussion of the future direction which thesis work can take.

7.1 Conclusion

Task scheduling is a major issue in both large-scale parallel and distributed systems that which affects the system performance. In this thesis a scheduling algorithm for prioritization of tasks has been proposed. By prioritizing the tasks of a workflow the broker can have better control on which task should be processed first. X-bar chart is used to find the order priority for tasks execution. The comparative analysis has been done with previous random overlap algorithm. The comparative analysis depicted that there is a decrease in the execution time, waiting time and response time.

7.2 Thesis Contributions

- X- bar chart is used for prioritization of tasks.
- Setup of cloud simulation environment.
- Task-Aware Priority Based scheduling algorithm is proposed and implemented.

7.3 Future Scope

- Other factors should be taken into consideration such as bandwidth and cpu utilization.
- This concept can be extended for handling fault tolerance.
- This approach can also be extended for optimization of energy efficiency and to reduce power consumption.

.

References

[1] M. Armbrust, A. Fox, R. Griffith, A. d. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson and M. Zaharai, "A view of Cloud Computing," *ACM*, vol. 53, no. 4, pp. 50-58, April 2010.

[2] J. Varia, "Architecting for Cloud Computing:Best Practices," Amazon, January 2010.

[3] X. Wang, B. Wang and J. Wang, "Cloud Computing and its Key Techniques in Science and Automation Engineering (CSAE)," *IEEE*, vol. 2, pp. 404-410, 2010.

[4] R. Clark, "A Break in the Clouds: Towards a Cloud Definition," *ACM*, vol. 39, no. 1, pp. 50-55, January 2009.

[5] G. Heiner, R. Sasse, E. Fuchs and H. Kopetz, "Time-Tiggered Architecture(TTA) Advances in Information Technology," 2011.

[6] B. Hay, K. Nance and M. Bishop, "Strom Cloud rising:Security Challenges for IaaS Cloud Computing," in *System Science*, Hawaii, 2011.

[7] Amazon SLA, 2010. [Online]. Available: aws.amazon.com/ec2-sla/.

[8] H. U, Rehm, Rueter and Wittmann, in *Grid and Cloud Computing*, August 2009, pp. 249-265.

[9] K. E, "Grid Compiting: Past, Present and Future- An Innovation Prespective," IBM white paper, 2006.

[10] L. v. Doorn, "Hardware Virtualization Trends," in *ACM*, Newyork, USA, 2006.

[11] R. Bhuyya, D. Abramson and J. Giddy, "A case for Economy Grid Architecture for Service-Oriented Grid Computing," in *10th IEEE International Heterogeneous Computing Wokshop*, California USA, 2001.

[12] "Advantages of SaaA Based Budgeting, Forecasting & Reporting," Amazon, [Online]. Available: aws.amazon.com. [Accessed 19 1 2013].

[13] A. Goscinski and M. Brock, "Towards Dynamic and Attribute based Publication, Discovery and Selection for Clouc Computing," *Future Generation Computer Systems,* vol. 26, no. 7, pp. 947-970, July 2010.

[14] Bardin, J. Callas, J. Chaput, S. Fusco and P. Gilbert, "Security Guidance for Critical Areas of Focus in Cloud Computing," in *Cloud Security Alliance*, January 2010.

[15] "Amazon Virtual Private Cloud (VPC)," Amazon, [Online]. Available: aws.amazon.com/vcp/. [Accessed 11 January 2013].

[16] S. Crosby and D. Brown, "The Virtualization Reality," *Queue,* vol. 4, no. 10, pp. 34-41, 2007.

[17] J. Brodkin, "Gartner: Seven Cloud-Computing Security," InfoWord, [Online]. Available: www.infoworld.com/article/.

[18] J. Jiang and W. Wen, "Information Security Issues in Cloud Computing Environment," in *Netinfo Security*, 2010.

[19] E. Mills, "Cloud Computing Security Forecast: Clear Skies," January 2009.

[20] C. Clark, K. Fraser, S. Hand, J. G. Hansen, I. Pratt and A. Warfield, "Live Migration of Virtual Machines," in *Proc. of NSDI 05*, Barkley CA, USA, 2005.

[21] Ahmadipour, M. Yadollahi, Sharzid and K. Mahdavi, "Cloud Computing and How to apply it to IT Management," in *2012 International Conference on Digital Object Identifier*, 2012.

[22] P.Pocatilu and C. Boja, "Quality Charactersitics and Matrics related to M-Learning process," in *Amfiteatru Economic*, June 2009.

[23] P.Barham, B.Dragovic, K. Fraser, R. Neugerbauer and I. Wareld, "Xen and the Art of Virtualization," in *SOSP 03: Proceedings of the 19th ACM Symposium on Operating System Principles*, Newyork, NY, USA, 2003.

[24] R.Buyya, C. Yeo, S. Venugopal and S. Broberg, "Cloud Computing and

Emerging IT platforms: Vision, Hype And Reality for Delivering Compuing as the 5th Utility," *Future Generation Computer Systems,* vol. 25, no. 6, pp. 599-612, June 2009.

[25] "Virtualization Basics and its Benefits," [Online]. Available: www.vmware.com/in/virtualization-basics. [Accessed 28 March 2013].

[26] P. Padala, K. G. Shin, X. Zhu, M. Uysal, Z. Wang, S. Singhal, A. Merchant and K. Salem, "Adaptive Control of Virtualized Resources in Utility Computing Environments," in *EuroSys.*, 2007.

[27] Milojicic, D. Douglis, F. Paindaveine, Y. Wheeler and R. Zhou, "Process Migration Survey," in *ACM Computing Survyes 32(3) 241-299*, 2000.

[28] E. Elmroth and L. Larsson, "Interfaces for placement, Migration and Monitoring of Virtual Machines in Federated Clouds," in *8th International Conference on Grid And Cooperative Computing pp 253-260*, 2009.

[29] R.Buyya and Beloglazov, "Adaptive Thrsehold-Based Approach for Energy Efficient Consolidation of Virtual Machines in Cloud Data Centers," in *8th International Workshop on Middleware for Grid, Cloud And E-Science pp 1-6*, Dec-2010.

[30] Beloglazov and R. Buyya, "Energy Efficient Allocation of Virtual Machines in Cloud Datacenter," *10th IEEE/ACM International Conference on Cluster,Cloud and Grid Computing*, pp. 577-578, May 2010.

[31] W. Voorsluys, J. Broberg, S. Venugopal and R. buyya, "Cost of Virtual Machine Live Migration in Cloud Computing : A Performance Evaluation," in *1st International Conferance on Cloud Computing 254-265*, 2009.

[32] J.Sonnek, J. Greensky, R. Reutiman and A. Chandra, "Starling: Minimizing Communication Overhead in Virtualized Computing Platforms using Decentralized Affinity-Aware Migration," in *39th International Conference on Parallel Processing*, Sep 2010.

[33] Beloglazov and R. Buyya, "Optimal Online Deterministic Algorothm and Adaptive Heuristic for Energy and Performance Efficient Dynamic Consolidation of Virtual Machines in Cloud Datacenter," in *Wiley InterScience pp 1-24*, 2010.

[34] S. Chakravorty, C. Mendes and L. Kale, "Proactive Fault Tolerance in MPI Applications via Task Migration," in *IEEE International Conference on High Performance Computing pp 1-12*, 2006.

[35] A. G. Delavar, M. J. M. B. Shabestari and m. K. Talebi, "RSDC (Relaible Scheduling Distributed In Cloud Computing)," in *International Journal Of Computer Science Engineering and Applications*, June 2012.

[36] S. Parsa and R. Entezari-Maleki, "RASA: A New Task Scheduling Algorithm in Grid Environment," in *World Applied Sciences Journal 7*, 1995.

[37] D. M. Dakshayini and D. H. S. Guruprasad, "An Optimal Model for Priority based Service Scheduling Policy for Cloud Computing Environment," in *International Journal of Computer Applications Volume 32-No. 9*, October 2011.

[38] S. Ghanbari and M. Othman, "A Priority based Job Scheduling Algorithm in Cloud Computing," in *International Conference on Advance Science and Contemprory Engineering*, 2012.

[39] E.-S. T. El-kenawy, A. I. El-Desoky and M. F. Al-rahamawy, "Extended Max-MIn Scheduling using Petri Net and Load Balancing," *International Journal of Soft Computing and Engineering*, vol. 2, no. 4, September 2012.

[40] S. Ambike, D. Bhansali, J. Kshirsagar and J. Bansiwal, "An Optimal Differentiated Job Scheduling System for Cloud Computing," *International Journal of Engineering Research and Applications*, vol. 2, no. 2, pp. 1212-1214, April 2012.

[41] M. S. Selvaranil and D. G. S. Sudhasivam, "Improved Cost-Based Algorithm for Task Scheduling in Cloud Computing," *IEEE*, vol. 4, no. 3, pp. 31-34, 2010.

[42] A.-E.-B. M, Design and Analysis of Relaible and Fault Tolerance, 2005.

[43] W. M. P. V. Aalst, A. H. Hofstede, B. Kiepuszewski and A. P. Barros, "Workflow Patterns Technical Report," Eindhoven University of Technology, 2000.

[44] D. Georgakopoulos, M. Hornic and A. Sethi, "An Overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure," in *Kulwer Academic Publishers*, Boston, 1995.

[45] S. Hwang and C. Kesselman, "A Flexible Failure Handling Framework for the Grid," in *12th IEEE International Symposuim on High Performance Distributed Computing*, Seattle, Washington, USA, 2003.

[46] J. Yu and R. Buyya, "A Taxonomy of Workflow Management System for Grid Computing," in *Journal og Grid Computing* , 2006.

[47] J. H. Abawajy, "Fault-Tolerance Scheduling Policy for Grid Computing Systems," in *18th International parallel and Distributed Processing Symposium*, Santa Fe, New Mexico USA, 2004.

[48] DAGMan Application, [Online]. Available: www.cs.wisc.edu/condor/manual/v6.4/2. [Accessed 12 January 2013].

[49] Q. A. K. H, Gnansambandam and N. Sharma, "Towards Automin Workload Provisioning for Enterprise Grid and Cloud," in *10th IEEE/ACM International Conference on Grid Computing*, Banf, Canada, October 2009.

[50] C. RN, R. R, R. CAFD and B. R., "Cloudsim: A Toolkit for Modeling and Simulation Of Cloud Computing Environment and Evaluation of Resourse Provising Algorithm," *Software: Practice And Experience,* vol. 41, no. 1, pp. 23-50, 2011.

[51] R. N. Calheiros, R. Ranjan, C. A. D. Rose and R. Buyya, "Cloudsim: A Nowel Framework for Modeling and Simulatoin of Cloud Computing Infrastructure And Services," in *Grid Computing and Disrtubuted System Labouratory*, The

University of Melbourne, Australia, 2009.

[52] R. Buyya and M. Murshed, "Gridsim: A toolkit for the Modeling and Simulation of Distriduted Management and Scheduling for Grid Computing," in *Concurrency And Computation: Practice and Experience (CCPE)*, 2002.

[53] "Cloud Simulation," [Online]. Available: www.cloud-simulation-framework.wikispaces.asu.edu/.

[54] S. Gupta and R. Buyya, "Network CloudSim: Modelling Parallel Applications in Cloud Simulations," in *4th International Conference on Utility and Cloud Computing*, 2011.

[55] R. I. Levin and D. S. Rubin, Statics for Management, Pearson Education India.

List of Publications

1. Atul Kumar Ramotra, Anju Bala, "Task Level Scheduling In Cloud Computing Using X-Bar Chart", International Journal of Advance Research in Computer Science and Software. (Communicated).