An efficient aproach for load balancing on cloud using hybrid method

Gurpreet Kaur[1], Shivani Ahuja[2]
IET Bhaddal, Ropar

Abstract— Cloud computing, a framework for enabling convenient, and on-demand network access to a shared pool of computing resources, is emerging as a new paradigm of large-scale distributed computing. It has widely been adopted by the industry, though there are many existing issues like Load Balancing, Virtual Machine Migration, Server Consolidation, Energy Management, etc. that are not fully addressed. Central to these issues is the issue of load balancing that is a mechanism to distribute the dynamic workload evenly to all the nodes in the whole cloud to achieve a high user satisfaction and resource utilization ratio. In cloud Computing, Load Balancing is essential for efficient operations in distributed environments. To allocate and balance the load of the resources among the various components and nodes load balancing is required. Load balancing aims to optimize resource use, maximize throughput, minimize response time, and avoid overload of any single resource. In this paper, the proposed technique has been implemented using Java Programming and simulate the algorithms on CloudSim. The Main contribution of CloudSim is to provide a holistic software framework for modeling Cloud computing environments and performance testing application services. And, the proposed algorithm gives better results as compared to existing technique. The results were also analyzed using various performance parameters such as Energy Consumption, Response time, Total Execution time, throughput, makespan and turnaround time.

Keywords—*Cloud Computing; Self-Healing; Multi-Tenancy; CloudSim; Load Balancing.*

I. INTRODUCTION

Cloud computing is a new technology and it is becoming popular because of its great features. In this technology almost everything like hardware, software and platform are provided as a service. A cloud provider provides services on the basis of client's requests. An important issue in cloud is, scheduling of user's requests, means how to allocate resources to these requests, so that the requested tasks can be completed in a minimum time and the cost incurred in the task should also be minimum. In case of Cloud computing services can be used from diverse and wide spread resources, rather than remote servers or local machines. There is no standard definition of Cloud computing. Generally, it consists of a bunch of distributed servers known as masters, providing demanded services and resources to different clients known as clients in a network with scalability and reliability of datacenter. The distributed computers provide on-demand services. Services may be of software resources (e.g. Software as a Service, SaaS) or physical resources (e.g. Platform as a Service, PaaS) or hardware/infrastructure (e.g. Hardware as a Service, HaaS or Infrastructure as a Service, IaaS). AmazonEC2 (Amazon Elastic Compute Cloud) is an example of cloud computing services [2].

Cloud computing has recently become popular due to the maturity of related technologies such as network devices, software applications and hardware capacities. Resources in these systems can be widely distributed and the scale of resources involved can range from several servers to an entire data center. To integrate and make good use of resources at various scales, cloud computing needs efficient methods to manage them [4]. Consequently, the focus of much research in recent years has been on how to utilize resources and how to reduce power consumption. One of the key technologies in cloud computing is virtualization. The ability to create virtual machines (VMs) [14] dynamically on demand is a popular solution for managing resources on physical machines. Therefore, many methods [17,18] have been developed that enhance resource utilization such as memory compression, request discrimination, defining threshold for resource usage and task allocation among VMs. Improvements in power consumption, and the relationship between resource usage and energy consumption has also been widely studied [6,10]. Some research aims to improve resource utilization while others aim to reduce energy consumption. The goals of both are to reduce costs for data centers. Due to the large size of many data centers, the financial savings are substantial.

1.1 Characteristics of Cloud Computing

1. Self-Healing

Any application or any service running in a cloud computing environment has the property of self-healing. In case of failure of the application, there is always a hot backup of the application ready to take over without disruption. There are multiple copies of the same application - each copy updating itself regularly so that at times of failure there is at least one copy of the application which can take over without even the slightest change in its running state.

2. Multi-tenancy

With cloud computing, any application supports multi-tenancy - that is multiple tenants at the same instant of time. The system allows several customers to share the infrastructure allotted to them without any of them being aware of the sharing. This is done by virtualizing the servers on the available machine pool and then allotting the servers to

multiple users. This is done in such a way that the privacy of the users or the security of their data is not compromised.

3. Linearly Scalable
Cloud computing services are linearly scalable. The system is able to break down the workloads into pieces and service it across the infrastructure. An exact idea of linear scalability can be obtained from the fact that if one server is able to process say 1000 transactions per second, then two servers can process 2000 transactions per second.

4. Service-oriented
Cloud computing systems are all service oriented - i.e. the systems are such that they are created out of other discrete services. Many such discrete services which are independent of each other are combined together to form this service. This allows re-use of the different services that are available and that are being created. Using the services that were just created, other such services can be created.

5. SLA Driven
Usually businesses have agreements on the amount of services. Scalability and availability issues cause clients to break these agreements. But cloud computing services are SLA driven such that
when the system experiences peaks of load, it will automatically adjust itself so as to comply with the service-level agreements. The services will create additional instances of the applications on more servers so that the load can be easily managed.

6. Virtualized
The applications in cloud computing are fully decoupled from the underlying hardware. The cloud computing environment is a fully virtualized environment.

7. Flexible
Another feature of the cloud computing services is that they are flexible. They can be used to serve a large variety of workload types - varying from small loads of a small consumer application to very heavy loads of a commercial application.

1.2 Cloud Computing Application Architecture

We know that cloud computing is the shift of computing to a host of hardware infrastructure that is distributed in the cloud. The commodity hardware infrastructure consists of the various low-cost data servers that are connected to the system and provide their storage and processing and other computing resources to the application. Cloud computing involves running applications on virtual servers that are allocated on this distributed hardware infrastructure available in the cloud. These virtual servers are made in such a way that the different service level agreements and reliability issues are met. There may be multiple instances of the same virtual server accessing the different parts of the hardware infrastructure available. This is to make sure that there are multiple copies of the applications which are ready to take over on another one's failure. The virtual server distributes the processing between the infrastructure and the computing is done and the result returned.

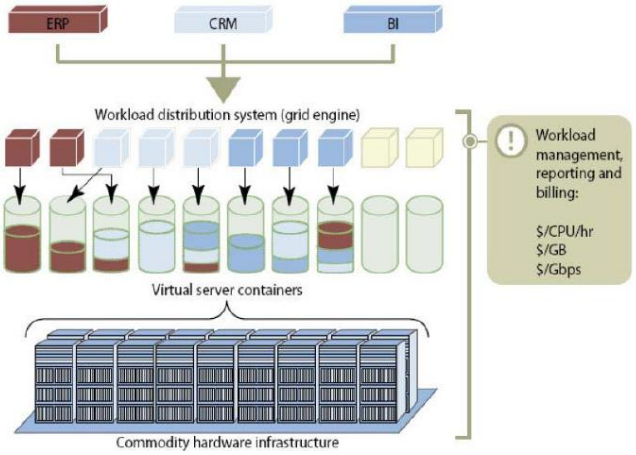

Figure 1 shows the basic Cloud computing application architecture

There will be a workload distribution management system, also known as the grid engine, for managing the different requests coming to the virtual servers. This engine will take care of the creation of multiple copies and also the preservation of integrity of the data that is stored in the infrastructure. This will also adjust itself such that even on heavier load, the processing is completed as per the requirements. The different workload management systems are hidden from the users. For the user, the processing is done and the result is obtained. There is no question of where it was done and how it was done. The users are billed based on the usage of the system - as said before - the commodity is now cycling and bytes. The billing is usually on the basis of usage per CPU per hour or GB data transfer per hour.

1.3 Cloud types

Together with virtualization, clouds can be defined as computers that are networked anywhere in the world with the availability of paying the used clouds in a pay-per-use way, meaning that just the resources that are being used will be paid. In the following the types of clouds will be introduced.

1.3.1 Public Clouds
A public cloud encompasses the traditional concept of cloud computing, having the opportunity to use computing resources from anywhere in the world. The clouds can be used in a so-called pay-per-use manner, meaning that just the resources that are being used will be paid by transaction fees.

1.3.2 Private Clouds
Private clouds are normally datacenters that are used in a private network and can therefore restrict the unwanted public to access the data that is used by the company. It is obvious that this way has a more secure background than the traditional public clouds. However, managers still have to worry about the purchase, building and maintenance of the system.

1.3.2 Hybrid Clouds

As the name already reveals, a hybrid cloud is a mixture of both a private and public cloud. This can involve work load being processed by an enterprise data center while other activities are provided by the public cloud. Below an overview of all three cloud computing types is illustrated.

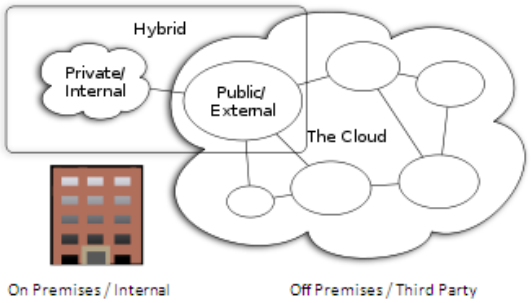

Figure 2: Cloud Computing Types

1.4 Load Balancing on Cloud Computing

With the increasing popularity of cloud computing, the amount of processing that is being done in the clouds is surging drastically. A cloud is constituted by various nodes which perform computation according to the requests of the clients. As the requests of the clients can be random to the nodes they can vary in quantity and thus the load on each node can also vary. Therefore, every node in a cloud can be unevenly loaded of tasks according to the amount of work requested by the clients. This phenomenon can drastically reduce the working efficiency of the cloud as some nodes which are overloaded will have a higher task completion time compared to the corresponding time taken on an under loaded node in the same cloud. This problem is not only confined only to cloud but is related with every large network like a grid, etc.

Load balancing in large distributed server systems is a complex optimization problem of critical importance in cloud systems and data centers. Load balancing algorithms are classified as static and dynamic algorithms. Static algorithms are mostly suitable for homogeneous and stable environments and can produce very good results in these environments. However, they are usually not flexible and cannot match the dynamic changes to the attributes during the execution time. Dynamic algorithms are more flexible and take into consideration different types of attributes in the system both prior to and during run-time [2]. These algorithms can adapt to changes and provide better results in heterogeneous and dynamic environments. However, as the distribution attributes become more complex and dynamic. As a result, some of these algorithms could become inefficient and cause more overhead than necessary resulting in an overall degradation of the services performance.

1.5 Types of Load balancing algorithms

Depending on who initiated the process, load balancing algorithms can be of three categories as given in [4]:

- Sender Initiated: If the load balancing algorithm is initialized by the sender.
- Receiver Initiated: If the load balancing algorithm is initiated by the receive
- Symmetric: It is the combination of both sender initiated and receiver initiated Depending on the current state of the system, load balancing algorithms can be divided into 2 categories as given in [4]:
- Static: It doesn't depend on the current state of the system. Prior knowledge of the system is needed.
- Dynamic: Decisions on load balancing are based on current state of the system. No prior knowledge is needed. So, it is better than static approach.

Here we will discuss on various dynamic load balancing algorithms for the clouds of different sizes.

1.6 Dynamic Load balancing algorithm

In a distributed system, dynamic load balancing can be done in two different ways: distributed and non-distributed. In the distributed one, the dynamic load balancing algorithm is executed by all nodes present in the system and the task of load balancing is shared among them. The interaction among nodes to achieve load balancing can take two forms: cooperative and non-cooperative [4]. In the first one, the nodes work side-by-side to achieve a common objective, for example, to improve the overall response time, etc. In the second form, each node works independently toward a goal local to it, for example, to improve the response time of a local task. Dynamic load balancing algorithms of distributed nature, usually generate more messages than the non-distributed ones because, each of the nodes in the system needs to interact with every other node. A benefit, of this is that even if one or more nodes in the system fail, it will not cause the total load balancing process to halt, it instead would affect the system performance to some extent. Distributed dynamic load balancing can introduce immense stress on a system in which each node needs to interchange status information with every other node in the system. It is more advantageous when mos of the nodes act individually with very few interactions with others.

In non-distributed type, either one node or a group of nodes do the task of load balancing. Non-distributed dynamic load balancing algorithms can take two forms: centralized and semi-distributed. In the first form, the load balancing algorithm is executed only by a single node in the whole system: the central node. This node is solely responsible for load balancing of the whole system. The other nodes interact only with the central node.

In semi-distributed form, nodes of the system are partitioned into clusters, where the load balancing in each cluster is of centralized form. A central node is elected in each cluster by appropriate election technique which takes care of load

balancing within that cluster. Hence, the load balancing of the whole system is done via the central nodes of each cluster [4]. Centralized dynamic load balancing takes fewer messages to reach a decision, as the number of overall interactions in the system decreases drastically as compared to the semi-distributed case. However, centralized algorithms can cause a bottleneck in the system at the central node and also the load balancing process is rendered useless once the central node crashes. Therefore, this algorithm is most suited for networks with small size.

1.7 Policies or Strategies in dynamic load balancing

There are 4 policies [4]:

1. **Transfer Policy:** The part of the dynamic load balancing algorithm which selects a job for transferring from a local node to a remote node is reffered to as Transfer policy or Transfer strategy.
2. **Selection Policy:** It specifies the processors involved in the load exchange (processor matching)
3. **Location Policy:** The part of the load balancing algorithm which selects a destination node for a transferred task is referred to as location policy or Location strategy.
4. **Information Policy:** The part of the dynamic load balancing algorithm responsible for collecting information about the nodes in the system is referred to as Information policy or Information strategy.

1.8 Goals of Load balancing

As given in [4], the goals of load balancing are:

1. To improve the performance substantially
2. To have a backup plan in case the system fails even partially
3. To maintain the system stability
4. To accommodate future modification in the system

1.9 CloudSim Toolkit

Clouds enable platform for dynamic and flexible application provisioning, by exposing data center's capabilities as a network of virtual services. So, users can access and deploy applications from anywhere in the Internet driven by demand and QoS requirements.

1.9.1 Why CloudSim?

Not possible to perform benchmarking experiments in repeatable, dependable, and scalable environment using real-world Cloud. Considering that none of the current distributed system simulators offer the environment that can be used for modeling Cloud, we present CloudSim. **Main contribution**: A holistic software framework for modeling Cloud computing environments and performance testing application services.

1.9.2 Defining CloudSim

- CloudSim is an extensible simulation toolkit.
- Described as a framework for modeling and simulation of both single and inter-networked (federation of clouds) clouds.
- HP and other leading organizations and also many universities around the world are using CloudSim for:
  - Cloud resource provisioning,
  - Energy-efficient management of data center resources,
  - Optimization of Cloud computing
  - Research activities.

1.10 CloudSim Architecture

Figure 3 shows the layered implementation of the CloudSim software framework and architectural components. At the lowest layer is the SimJava discrete event simulation engine [6] that implements the core functionalities required for higher-level simulation frameworks such as queuing and processing of events, creation of system components (services, host, data center, broker, virtual machines), communication between components, and management of the simulation clock. Next follows the libraries implementing the GridSim toolkit [9] that support high level software components for modeling multiple Grid infrastructures, including networks and associated traffic profiles, and fundamental Grid components such as the resources, data sets, workload traces, and information services. The CloudSim is implemented at the next level by programmatically extending the core functionalities exposed by the GridSim layer. CloudSim provides novel support for modeling and simulation of virtualized Cloudbased data center environments such as dedicated management interfaces for VMs, memory, storage, and bandwidth. CloudSim layer manages the instantiation and execution of core entities (VMs, hosts, data centers, application) during the simulation period. This layer is capable of concurrently instantiating and transparently managing a large-scale Cloud infrastructure consisting of thousands of system components. The fundamental issues such as provisioning of hosts to VMs based on user requests, managing application execution, and dynamic monitoring are handled by this layer. A Cloud provider, who wants to study the efficacy of different policies in allocating its hosts, would need to implement his strategies at this layer by programmatically extending the core VM provisioning functionality. There is a clear distinction, a thin layer on how a host is allocated to different competing VMs in the Cloud. A Cloud host can be concurrently shared among a number of VMs that execute applications based on user-defined QoS specifications.

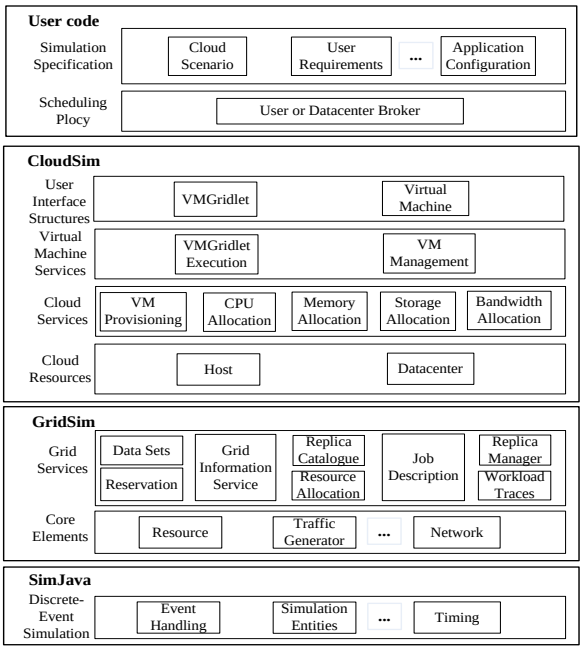

Figure 3 Layered CloudSim architecture.

The top-most layer in the simulation stack is the User Code that exposes configuration related functionalities 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. A Cloud application developer can generate a mix of user request distributions, application configurations, and Cloud availability scenarios at this layer and perform robust tests based on the custom Cloud configurations already supported within the CloudSim. As Cloud computing is a rapidly evolving research area, there is a severe 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, application benchmarking based on execution contexts. By extending the basic functionalities already exposed by CloudSim, researchers would be able to perform tests based on specific scenarios and configurations, hence allowing the development of best practices in all the critical aspects related to Cloud Computing.

1.11 Paper Motivation

Cloud computing is a vast concept. Many of the algorithms for load balancing in cloud computing have been proposed. Some of those algorithms have been implemented in this paper. The whole Internet can be considered as a cloud of many connection less and connection-oriented services. So, the divisible load scheduling theory for Wireless networks described in [9] can also be applied for clouds. The performance of various algorithms has been studied and compared.

II. LITERATURE REVIEW

Load balancing in the cloud computing environment has an important impact on the performance. Good Load balancing makes cloud computing more efficient and improves user satisfaction. There have been many studies of load balancing for the Cloud environment.

Atyaf Dhari et al. 2017, proposed Load Balancing Decision Algorithm (LBDA) to manage and balance the load between the virtual machines in a datacenter along with reducing the completion time (Makespan) and Response time. Findings: The mechanism of LBDA is based on three stages, first calculates the VM capacity and VM load to categorize the VMs' states (Under loaded VM, Balanced VM, High Balance VM, Overloaded). Second, calculate the time required to execute the task in each VM. Finally, makes a decision to distribute the tasks among the VMs based on VM state and task time required. Improvements: We compared the result of our proposed LBDA with Max- Min, Shortest Job Firsthand Round Robin.

Mohammad Goudarzi et al. (2017) evaluated the efficiency of the proposed solution using both simulation and testbed experiments. The evaluation study demonstrated that proposal can outperform existing optimal and near-optimal counterparts in terms of weighted execution cost, energy consumption and execution time. Due to nowadays advances of mobile technologies in both hardware and software, mobile devices have become an inseparable part of human life. Along with this progress, mobile devices are expected to perform various types of applications.

Muhammad Baqer Mollah et al. (2017) presented the main security and privacy challenges in this field which have grown much interest among the academia and research community. Although, there are many challenges, corresponding security solutions have been proposed and identified in literature by many researchers to counter the challenges. We also present these recent works in short. Furthermore, we compare these works based on different security and privacy requirements, and finally present open issues. The rapid growth of mobile computing is seriously challenged by the resource constrained mobile devices. However, the growth of mobile computing can be enhanced by integrating mobile computing into cloud computing, and hence a new paradigm of computing called mobile cloud computing emerges.

M. Vanitha et al. (2017) proposed, involving a well-organized use of resources, which is known as the dynamic well-organized load balancing (DWOLB) algorithm. This is a powerful algorithm for reducing the energy that is consumed in cloud computing. Cloud computing is used in almost all domains today. Through the use of cloud-based applications, it has become easier for an internet user to make use of the services and re- sources that are widely available. The cloud service provider undertakes to deliver all the subscribers' requirements as per the service level agreement (SLA). These resources must be well-protected since they are used by many subscribers.

Weidong Cai et al. [1] in 2016, presented the proposed load balancing approaches in Map Reduce aim at optimizing task execution time, whereas disk space is not considered. In the research, a new scheme which consists of modified K-ELM and NSGA-II is proposed. Corresponding experiment results have shown that our method can assign tasks evenly, and effectively improve the performance of a cloud system. Map Reduce is a popular programming model widely used in distributed systems. With regard to large-scale applications, e.g. home energy management in a city, online social community etc., load-balancing becomes critical affecting the performance of distributed computing.

Oshin Sharma et al. [2] in 2015 provided a clear view of various energy management techniques used for mobile devices and performance analysis of various cloud computing techniques for energy efficient devices. For this, we have given brief introduction and our vision in the field of green computing. In addition to this, they performed performance analysis of various load balancing techniques based on six different cases. For energy efficient mobile devices, cloud computing is very much essential by providing storage and performing computations in the network. With the help of cloud computing many devices can connect over internet and can access the resources at anytime from anywhere.

Ebin Deni Raj et al. [3] in 2015 described Big Data and parallel computing are used extensively for processing large quantities of data, structured, semi structured or totally unstructured. MapReduce and Hadoop are used for the parallel data processing of these kinds of data. Various scheduling policies are used for MapReduce scheduling which is discussed in detail and a new scheduling technique Two Phase Scheduling Policy (TPSP) based resource allocation for MapReduce is implemented and the efficiency is verified.

R.R. Kotkondawar et al. (2014) described a Cloud computing is the most recent technology in today's world of computing and it overcomes deficiencies of traditional ways of computing. Cloud computing is a new way of providing the essential services to cloud users on "Pay As You Go" basis. Cloud computing provides different features like on demand access, flexibility, instant response, pay per use etc. to customers. In order to provide all these features to cloud users, cloud computing systems must be structured and managed efficiently to provide the Quality of Services (QOS) to users. Various technological concepts such as abstraction and virtualization are used that hides the implementation details from an average cloud user. Cloud load balancing plays a very important role in providing all the cloud features to users which is the main topic of interest in our research. Different architectures apply altogether different load balancing algorithms. The research includes the Study of different approaches of effective management of cloud systems. The study includes load balancing approaches in different system architectures like Centralized, Distributed and Cluster based architecture. Finally, various algorithms have been compared based on the different parameters like response time, efficiency and throughput etc.

Saeed Javanmardi et al. [5] in 2014, presented a hybrid job scheduling approach, which considers the load balancing of the system and reduces total execution time and execution cost. We try to modify the standard Genetic algorithm and to reduce the iteration of creating population with the aid of fuzzy theory. The main goal of this research is to assign the jobs to the resources with considering the VM MIPS and length of jobs. The new algorithm assigns the jobs to the resources with considering the job length and resources capacities. They evaluate the performance of their approach with some famous cloud scheduling models. The results of the experiments show the efficiency of the proposed approach in term of execution time, execution cost and average Degree of Imbalance (DI).

Tom Guérout *et al.* [6] in 2014, discussed analysis of studies on Clouds modeling, Clouds scheduling, and actual SLAs of SaaS providers. Based on these analyses, they proposed a Cloud architecture modeling that includes the DVFS, but especially a modeling of Clouds Quality of Service parameters. This list contains definitions, measurable and reusable metrics for non-functional parameters. The defined QoS metrics are measurable and reusable in any scheduling approach for Clouds. The use of these QoS models is done through the performance analysis of three scheduling approaches considering four QoS parameters. In addition to the energy consumption and the Response Time, two other QoS parameters are taken into account in different virtual machines scheduling approaches. The evaluation is done through simulations, using two common scheduling algorithms and a Genetic Algorithm (GA) for virtual machines (VMs) reallocation, allowing us to analyze the QoS parameters evolution in time. Simulation results have shown that including various and antagonist QoS parameters allow a deeper analysis of the intrinsic behavior and insight of these three algorithms. It aims to allow a better analysis of Clouds QoS, and allows being closer to Cloud providers needs while keeping a green approach.

Bernardetta Addis *et al.* [7] in 2014 proposed a new optimization framework for the management of the energy usage in an integrated system for Cloud services that includes both service centers and communication networks for accessing and interconnecting them. The optimization framework considers a PaaS scenario where VMs serving an application can be allocated to a set of SCs geographically distributed and traffic load coming from different world regions can be assigned to VMs in order to optimize the energy cost and minimize CO2 emissions. Numerical results, on a set of randomly generated instances and a case study representative of a large Cloud provider, shows that the availability of green energy have a big impact on optimal energy management policies and that the contribution of the network is far from being negligible.

Rajesh Gorge Rajan et al. [8] in 2013 have the investigated the different algorithms proposed to resolve the issue of load balancing and task scheduling in Cloud Computing. They discussed and compared these algorithms to provide an overview of the latest approaches in the field. Load

Balancing is essential for efficient operations in distributed environments. As Cloud Computing is growing rapidly and clients are demanding more services and better results, load balancing for the Cloud has become a very interesting and important research area. Many algorithms were suggested to provide efficient mechanisms and algorithms for assigning the client's requests to available Cloud nodes. These approaches aim to enhance the overall performance of the Cloud and provide the user more satisfying and efficient services.

Suriya Begum et al. [9] in 2013 have described the random arrival of load in such an environment can cause some server to heavily loaded while other server is idle or only lightly loaded. Equally load distributing enhance performance by transferring load from heavily server. Efficient scheduling and resource allocation is a critical characteristics of cloud computing based on which performance can be estimated. It is required to Distribute the dynamic load workload evenly across all the nodes to achieve the high user satisfaction and resource utilization ration by making sure that every computing resource is distributed efficiently and fairly.

Argha Roy et al. [10] in 2013 presented the researcher proposed the idea of dynamic load balancing. Researcher concluded that dynamic load balancing is a technique to use the cloud computing in efficient manner. The virtual machine algorithm used in the approach can automatically monitor the load balancing with the use of load balancer. The researcher has avoided the data migration, Job migration and static load balancer.

Meysam Orouskhani et al. [11] in 2013 proposed the improved CSO algorithm namely "Adaptive dynamic cat Swarm Optimization"The Paper described the addition of a new adaptive interia weight to velocity equation and use of adaptive acceleration ratio.The Proposed CSO take less time to converge and can find best solution in less iteration.

Amir Nahir et al. [12] in 2013 presented the approach which is based on creating several replicas of each job and sending each replica to a different server. Upon the arrival of a replica to the head of the queue at its server, the latter signals the servers holding replicas of that job, so as to remove them from their queues. They show, through an analysis and simulations, that this scheme improves the expected queuing overhead over traditional schemes by a factor of 9 (or more) under various load conditions. In addition, we show that our scheme remains efficient even when the inter-server signal propagation delay is significant (relative to the job's execution time). They provided heuristic solutions to the performance degradation that occurs in such cases and show, by simulations, that they efficiently mitigate the detrimental effect of propagation delays. Finally, they demonstrated the efficiency of proposed scheme in a real-world environment by implementing a load balancing system based on it, deploying the system on the Amazon Elastic Compute Cloud (EC2), and measuring its performance.

Parveen Patel et al. [13] in 2013 proposed an approach that shows that Layer-4 load balancing is fundamental to creating scale-out web services. We designed and implemented Ananta, a scale-outlayer-4 load balancer that runs on commodity hardware and meets the performance, reliability and operational requirements of multi-tenant cloud computing environments. Ananta combines existing techniques in routing and distributed systems in a unique way and splits the components of a load balancer into a consensus-based reliable control plane and a decentralized scale-out data plane.

Isam Azawi Mohialdeen [14] in 2013 has discussed, the behavior of four job scheduling algorithms, namely: Random, Round-Rubin (RR), Opportunistic Load Balancing and Minimum Completion Time have been investigated and examined in a Cloud computing environment. Based on the results, it can be also concluded that there is not a single scheduling algorithm that provides superior performance with respect to various types of quality services. This is because job scheduling algorithms needs to be selected based on its ability to ensure good quality of services with reasonable cost and maintain fairness by fairly distribute the available resources among all the jobs and respond to the constraints of the users.

Prof. Dr. Jayant et al. [15] in 2013 described cloud computing is a distributed computing model in which everything from software to infrastructure is provided as a service like utility computing. Job scheduling is one of the cores and challenging issues in cloud computing. The decisions like when to allocate hardware resources to the tasks has become the main issue in cloud computing. Job scheduling algorithm is an NP- completeness problem which play key role in cloud computing. They presented the surveys of the current job scheduling algorithms under cloud environment and summarize some method to improve the performance.

Mohamed Abu Sharkh et al. [16] in 2013 proposed a new model to tackle the resource allocation problem for a group of cloud user requests. They included the provisioning for both data center computational Resources and network resources. The model is implemented with the objective of minimizing the average tardiness of connection requests. Four combined scheduling algorithms are introduced and used to schedule virtual machines on data center servers and then schedule connection requests on the network paths available. Of the four methods, the method combining Resource Based Distribution technique and Duration Priority technique have shown the best performance getting the minimum tardiness while complying with the problem constraints.

Hwa Min Lee et al. [17] in 2013 described Cloud computing has become a new computing paradigm that has huge potentials in enterprise and business. Green cloud computing is also becoming increasingly important in a world with limited energy resources and an ever-rising demand for more computational power. To maximize utilization and minimize total cost of the cloud computing infrastructure and running applications, resources need to be managed properly and virtual machines shall allocate proper host nodes to perform the computation. In this paper, we propose performance analysis-based resource allocation scheme for the efficient allocation of virtual machines on the cloud infrastructure. They experimented the proposed resource

allocation algorithm using CloudSim and its performance is compared with two other existing models.

Wei Chen et al. [18] in 2013 proposed a closed-loop approach is proposed for optimizing Quality of Service (QoS) and cost. Modules of monitoring and controlling data centers are required as well as the application feedback such as video streaming services. An algorithm is proposed to help choose cloud providers and data centers in a multi-cloud environment as a video service manager. Performances with different video service workloads are evaluated. Compared with using only one cloud provider, dynamically deploying services in multicloud is better in aspects of both cost and QoS. If cloud service costs are different among data centers, the algorithm will help make choices to lower the cost and keep a high QoS.

Dhinesh Babu L.D. et al. [19] in 2013 Load balancing of no preemptive independent tasks on virtual machines (VMs) is an important aspect of task scheduling in clouds. Whenever certain VMs are overloaded and remaining VMs are under loaded with tasks for processing, the load has to be balanced to achieve optimal machine utilization. They proposed an algorithm named honey bee behavior inspired load balancing (HBB-LB), which aims to achieve well balanced load across virtual machines for maximizing the throughput. The proposed algorithm also balances the priorities of tasks on the machines in such a way that the amount of waiting time of the tasks in the queue is minimal. They have compared the proposed algorithm with existing load balancing and scheduling algorithms. The experimental results show that the algorithm is effective when compared with existing algorithms. Our approach illustrates that there is a significant improvement in average execution time and reduction in waiting time of tasks on queue.

Zhen Xiao et al. [20] in 2013 described Cloud computing allows business customers to scale up and down their resource usage based on needs. Many of the touted gains in the cloud model come from resource multiplexing through virtualization technology. In this paper, we present a system that uses virtualization technology to allocate data center resources dynamically based on application demands and support green computing by optimizing the number of servers in use. We introduce the concept of "skewness" to measure the unevenness in the multidimensional resource utilization of a server. By minimizing skewness, we can combine different types of workloads nicely and improve the overall utilization of server resources. They developed a set of heuristics that prevent overload in the system effectively while saving energy used. Trace driven simulation and experiment results demonstrate that our algorithm achieves good performance.

Navendu Jain et al. [21] in 2013 introduced a novel pricing and resource allocation approach for batch jobs on cloud systems. In our economic model, users submit jobs with a value function that specifies willingness to pay as a function of job due dates. The cloud provider in response allocates a subset of these jobs, taking into advantage the flexibility of allocating resources to jobs in the cloud environment. Focusing on social-welfare as the system objective (especially relevant for private or in-house clouds), they constructed a resource allocation algorithm which provides a small approximation factor that approaches 2 as the number of servers increases. An appealing property of our scheme is that jobs are allocated non-preemptively, i.e., jobs run in one shot without interruption. This property has practical significance, as it avoids significant network and storage resources for check pointing. Based on this algorithm, we then design an efficient truthful-in-expectation mechanism, which significantly improves the running.

Dr. Hemant S. Mahalle et al. [22] in 2013 presented a Clouds are high configured infrastructure delivers platform, software as service, which helps customers to make subscription for their requirements under the pay as you go model. Cloud computing is spreading globally, due to its easy and simple service-oriented model. The numbers of users accessing the cloud are rising day by day. Generally, cloud is based on data centers which are powerful to handle large number of users. The reliability of clouds depends on the way it handles the loads, to overcome such problem clouds must be featured with the load balancing mechanism. Load balancing in cloud computing will help clouds to increase their capability, capacity which results in powerful and reliability clouds.

Kumar Nishan et al. (2012) proposed an algorithm for load Distribution of workloads among nodes of a cloud by the use of Ant Colony Optimization (ACO). This is a modified Approach of ant colony optimization that has been applied from the perspective of cloud or grid network systems with the Main aim of load balancing of nodes. This modified algorithm has an edge over the original approach in which each ant build their own individual result set and it is later on built into a complete solution. However, in this approach the ants continuously update a single result set rather than updating their own result set. Further, as they know that a cloud is the collection of many nodes, which can support various types of application that is used by the clients on a basis of pay per use. Therefore, the system, which is incurring a cost for the user should function smoothly and should have algorithms that can continue the proper system functioning even at peak usage hours.

Klaithem Al Nuaimi et al. (2012) have investigated the different algorithms proposed to resolve the issue of load balancing and task scheduling in Cloud Computing. They discussed and compared these algorithms to provide an overview of the latest approaches in the field. Load Balancing is essential for efficient operations in distributed environments. As Cloud Computing is growing rapidly and clients are demanding more services and better results, load balancing for the Cloud has become a very interesting and important research area. Many algorithms were suggested to provide efficient mechanisms and algorithms for assigning the client's requests to available Cloud nodes. These approaches aim to enhance the overall performance of the Cloud and provide the user more satisfying and efficient services.

Zheng Hu et al. (2012) introduced the failure and recovery scenario in the current Cloud computing entities and propose a Reinforcement Learning (RL) based algorithm to

make job scheduling in the current computing Cloud fault tolerant. We carry out experimental comparison with Resource-constrained Utility Accrual algorithm (RUA), Utility Accrual Packet scheduling algorithm (UPA) and LBESA to demonstrate the feasibility of proposed approach.

Andre Martin et al. (2011) presented a new fault tolerance approach based on active replication for Stream Map Reduce systems. Presented approach is cost effective for cloud consumers as well as Cloud providers. Cost effectiveness is achieved by fully utilizing the acquired computational resources without performance degradation and by reducing the need for additional nodes dedicated to fault tolerance.

Dr.G.Sudha et al. [28] in 2010 proposed a scheduling approach in cloud employs an improved cost-based scheduling algorithm for making efficient mapping of tasks to available resources in cloud. This scheduling algorithm measures both resource cost and computation performance, it also improves the computation/communication ratio by grouping the user tasks according to a particular cloud resource's processing capability and sends the grouped jobs to the resource. The objective of this research is to schedule task groups in cloud computing platform, where resources have different resource costs and computation performance. Due to job grouping, communication of coarse-grained jobs and resources optimizes computation/communication ratio. For this purpose, an algorithm based on both costs with user task grouping is proposed.

Ahmed S. Ghiduk [29] in 2010 described a Search-based optimization technique (e.g., hill climbing, simulated annealing, and genetic algorithms) have been applied to a wide variety of software engineering activities including cost estimation, next release problem, and test generation. Several search-based test generation techniques have been developed. These techniques had focused on finding suites of test data to satisfy a number of control-flow or data-flow testing criteria. Genetic algorithms have been the most widely employed search-based optimization technique in software testing issues. Recently, there are many novel search-based optimization techniques have been developed such as Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), Artificial Immune System (AIS), and Bees Colony Optimization. ACO and AIS have been employed only in the area of control-flow testing of the programs. The author aims at employing the ACO algorithms in the issue of software data-flow testing. They presented an ant colony optimization-based approach for generating set of optimal paths to cover all definition-use associations (du-pairs) in the program under test.

III. RESEARCH PROBLEM FORMULATION AND METHODOLOGY

In a cloud environment, there may be any number of host machines and each host machine has different-different load due to virtual machines as per the client's demand. The load of a host machine may be of various types such as CPU load, Memory load, Storage load and Network related load etc. If the load of any host machine exceeds its capacity then it affects its efficiency. In runtime, any client application service may change their resource (CPU, RAM, Storage and Bandwidth etc.) demand and this causes the host system to be imbalanced. If this imbalanced situation occurs due to overloading then system is balanced using load balancing techniques by distributing the extra workload to the whole clouds host heaving light loads. This helps to improve the overall performance of the cloud system.

Load Balancing is defined as a process of making effective resource utilization by reassigning the total load to the individual nodes of the collective system and thereby minimizing the response time of the job. Load Balancing algorithms are classified as Static and Dynamic algorithms. Static algorithms are most suitable for homogenous and stable environments. However, they cannot match the dynamic changes to the attributes during execution time. Dynamic algorithms take into consideration different types of attributes in the system both prior to and during run time. These algorithms can adapt to changes and provide better results in heterogeneous and dynamic environments.

In the past, a number of load balancing algorithms have been developed specifically to suit the dynamic cloud computing environments such as INS (Index Name Server) algorithm[A], WLC (Weighted Least Connection) algorithm[B], DDFTP (Dual direction Downloading algorithm from FTP servers)[C], LBMM (Load Balancing Min-Min) algorithm[D], ACO(Ant Colony Optimization) algorithm[E] and Bee-MMT(Artificial Bee Colony algorithm- Minimal Migration time)[F]. We are going to use the PSO (Particle Swarm Optimization) algorithm for load balancing in dynamic cloud environments as particle swarm has already get better results than genetic and ACO in grid computing[G]. Performance of Particle Swarm Optimization has also been approved better in distributed system [12]. In the proposed research, the bat optimization algorithm for task scheduling and load balancing on cloud computing will be implemented. The proposed algorithm will also compare with the load balancing decision algorithm for evaluation purpose. The results of the proposed work will be analysed on the basis of makespan, execution time and response time.

Objectives

The key objective of this research work is to optimize the performance of the cloud architecture. Overloaded nodes across the server and storage side often lead to performance degradation and are more vulnerable to various failures. To remove this limitation the load must be migrated from the overloaded resource to an underutilized one without causing harm and disruption to the application workload. Objectives for this research work are:

1.) To study and understand the task shcduling and load balancng approach on cloud.
2.) To implement exising load balancing decision algorithm and proposed bat optimization on cloud environment.

3.) To analyze the behavior of the proposed algorithm on the basis of following parameters:
   (a) Execution Time
   (b) Response Time
   (c) Makespan

Metrics for Load Balancing in Clouds

Various metrics will be considered in load balancing techniques in cloud computing are discussed below

1. **Throughput** is used to calculate the no. of tasks whose execution has been completed. It should be high to improve the performance of the system.
2. **Overhead Associated** determines the amount of overhead involved while implementing a load-balancing algorithm. It is composed of overhead due to movement of tasks, inter-processor and inter-process communication. This should be minimized so that a load balancing technique can work efficiently.
3. **Fault Tolerance** is the ability of an algorithm to perform uniform load balancing in spite of arbitrary node or link failure. The load balancing should be a good fault-tolerant technique.
4. **Response Time** is the amount of time taken to respond by a particular load balancing algorithm in a distributed system. This parameter should be minimized.
5. **Resource Utilization** is used to check the utilization of re-sources. It should be optimized for an efficient load balancing.
6. **Scalability** is the ability of an algorithm to perform load balancing for a system with any finite number of nodes. This metric should be improved.
7. **Performance** is used to check the efficiency of the system. This has to be improved at a reasonable cost, e.g., reduce task response time while keeping acceptable delays.

In cloud computing, load balancing is required to distribute the dynamic local workload evenly across all the nodes. It helps to achieve a high user satisfaction and resource utilization ratio by ensuring an efficient and fair allocation of every computing resource. Proper load balancing aids in minimizing resource consumption, implementing fail-over, enabling scalability, avoiding bottlenecks and over-provisioning etc.

Input: Required parameter for cloudlets and virtual machines are taken from user.

Output: Improves energy efficiency and load balancing at cloud with better response time, data processing time and throughput.

Input from user

Existing Method: Load Balancing Decision Algorithm will be implemented for task scheduling on cloud

Proposed Method: Bat Optimization will be implemented on cloud for task scheduling and load balancing

Behavior of the proposed method with be tested analyzed by using various parameters

Integration of Cloudsim with Eclipse for implementing the load balancing algorithms

CloudSim is a framework developed by the GRIDS laboratory of University of Melbourne which enables seamless modelling, simulation and experimenting on designing Cloud computing infrastructures. CloudSim is a self-contained platform which can be used to model data centers, service brokers, scheduling and allocation policies of a large scaled Cloud platform. It provides a virtualization engine with extensive features for modelling the creation and life cycle management of virtual engines in a data center. CloudSim framework is built on top of GridSim framework also developed by the GRIDS laboratory.

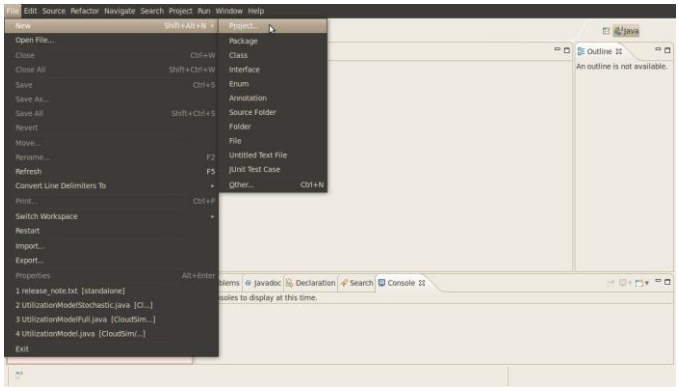

Figure 4. shows the wizard to select new java project

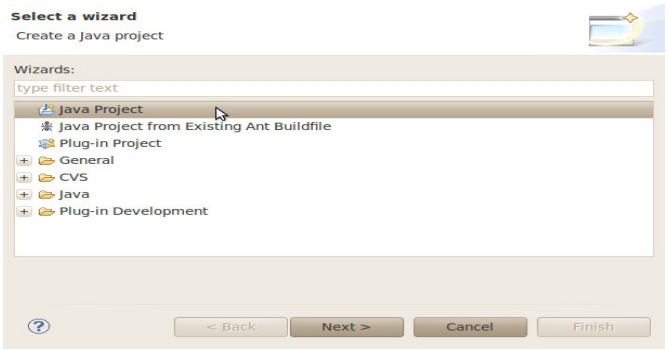

Figure 5 shows the select a wizard to enter the name of the java project

In the "Create a Java Project" window, fill the field "Project name" with CloudSim. Then select "Create project from existing source". In the "Directory" field, select the directory extracted from the CloudSim package. Then, select "Finish" to complete project creation.

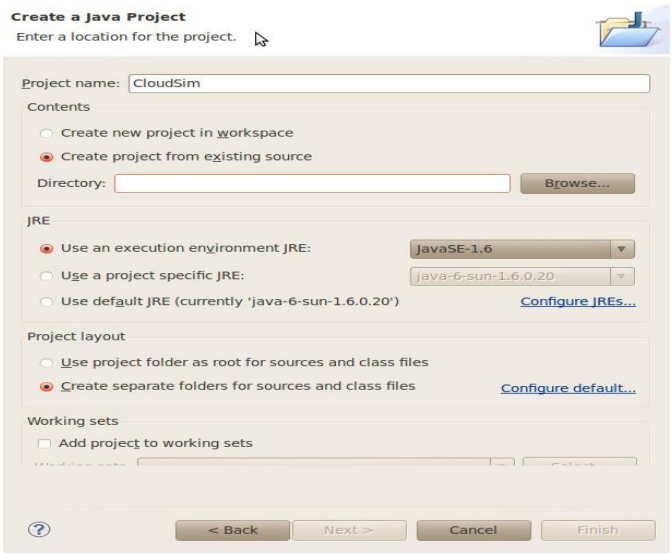

Figure 6 shows the select a wizard to enter next after entering the name of the project

After these steps, CloudSim we can navigate through CloudSim packages, and develop our own simulations using CloudSim.

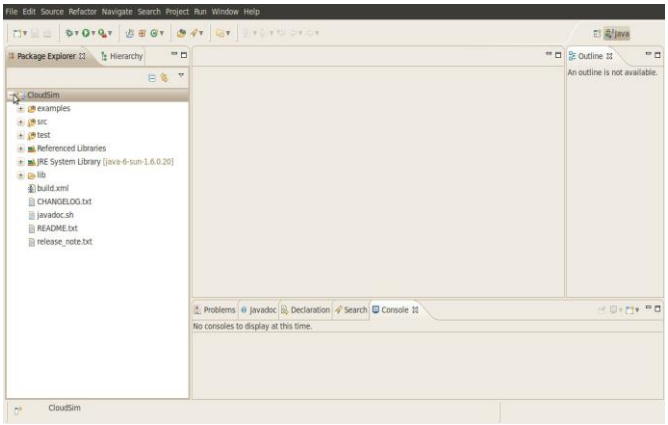

Figure 7 shows the final working snapshot of CloudSim integration with Eclipse

IV. RESULTS AND DISCUSSION

[1] Setup server_config.xml

[2] Initialize the Tomcat Server for project execution

[3] Send Request for the execution of the project

[4] Then select the specific algorithm for its execution

[5] Simulation Results using Tomcat Server

[6] CloudSim Results

[7] Output Tables

[8] Graphical Charts

The implementation steps are eloborated as below:

Setup server_config.xml

In this research work we are using five servers having their different IDs, names, IP address, speed and RAM. The number of jobs can be increased or wane as per the requirement. As by increasing the number of jobs the speed of server has to be increased so that it cannot affect the overall performance of the system. The parameters of job are id, requestType, arrival Time and length. The arrival time should be in increasing order.

Initialize the Tomcat Server for project execution

To run the project on server Tomcat Server should be initialized. The Tomcat server is available in different versions. In this Research work Tomcat v6.0 server is configured. The steps for configure Tomcat v6.0 Server on Eclipse interface is following as:

a) Right click on project and then click on run on server.

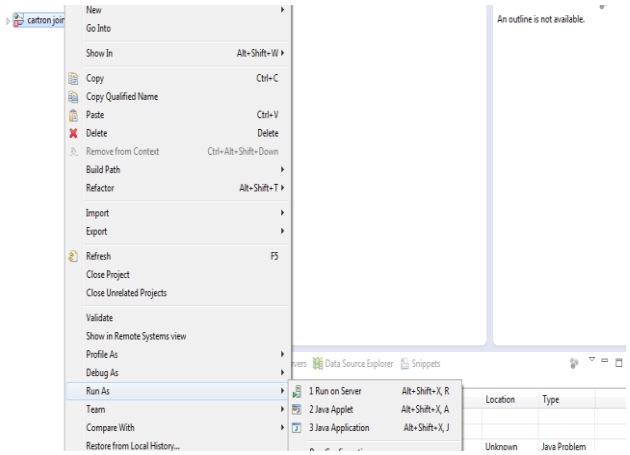

Figure 8 project execution screen.

b) Under apache select tomcat 6

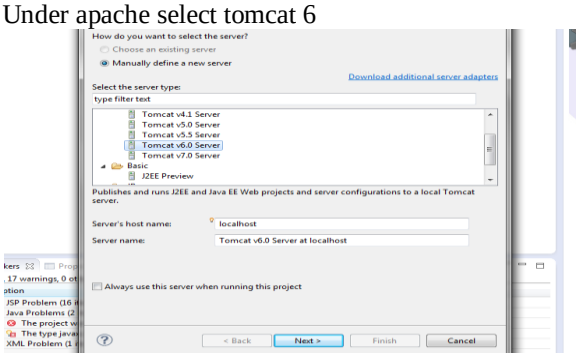

Figure 9 selection of tomcat v6.0 server.

c) Select apache install directory

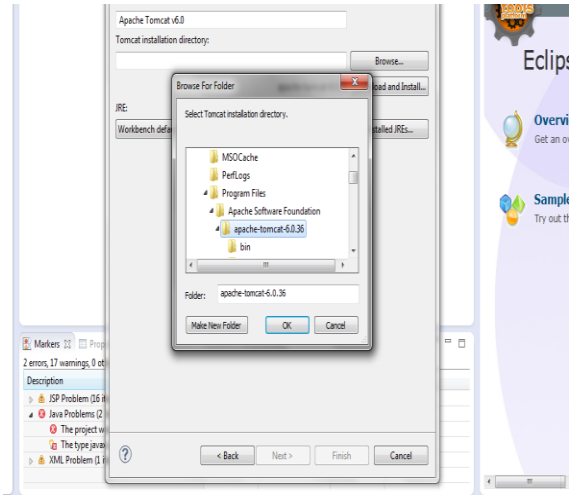

Figure 10 installation of apache-tomcat server from directory

Request for the execution of the project

After the program execution, URL is generated automatically like http://localhost:8080/ loadBalancingandTaskScheduling and fill this on crome browser to get the desired results. It is shown in figure 11.

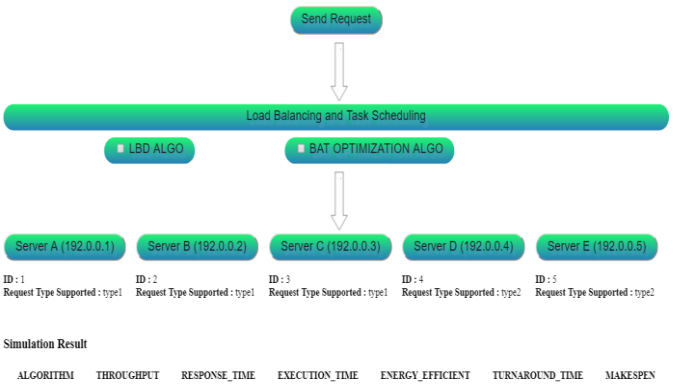

Figure 11 execution of the project.

Simulation Results using Tomcat Server

(i) When algorithms are selected for load balancing, simulation results are as depicted in fig.12.

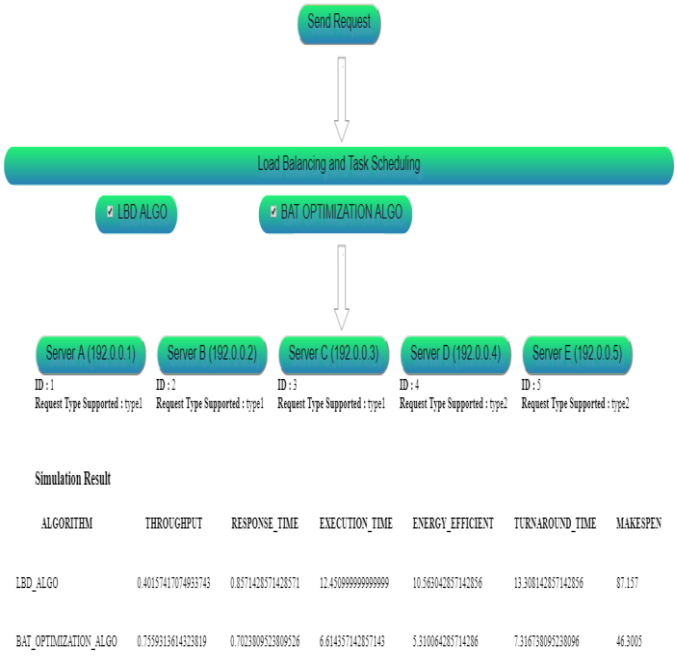

Figure 12 Simulation results of Load Balancing Decision Algorithm & Bat Optimization algorithm

Output Tables

The output tables shows the values for different parameters like throughput, response time,execution time and energy consumption when a particular technique is used for dynamic load balancing with the distinct number of jobs. The different tables are drawn for three dynamic load balancing techniques and then for a particular number of jobs these are compared and the experimental results show that our proposed model gives the best results in terms of energy consumption, execution time, response time and throughput. The tables are shown as below:

Output Table of Load Balancing Decision Algorithm: this algorithm used for web services and systems called as Join-Idle-Queue load balancing algorithm. It facilitates large scale load balancing with distributed dispatchers. In each dispatch firstly load balancing algorithm idles the processors for the availability and then does allotment of the task to processors in such a way that reduces the queue length at each server. This algorithm remove the load balancing work from critical path of request processing which helps in effective reduction of the system load. Join idle queue is the technique for load balancing which unveil the information about different parameters like Throughput, Response time, Execution time, Energy consumption with distinct numbers of jobs as shown in table 1.

Table 1 Simulation results for Load Balancing Decision Algorithm for different number of jobs.

S.No	No. of Jobs	Parameters Name					
		Throughput	Response Time	Execution Time	Energy Consumption	Turnaround Time	Makespan
1.	5	0.4051	0.8571	12.45	10.56	13.081	87.157
2.	6	0.42	1.00	14.27	12.09	15.27	85.65
3.	7	0.423	1.2	16.53	13.97	17.73	82.65
4.	8	0.4829	1.3	16.62	14.51	18.81	81.657
5.	9	0.5444	1.4	16.87	15.21	19.73	80.52

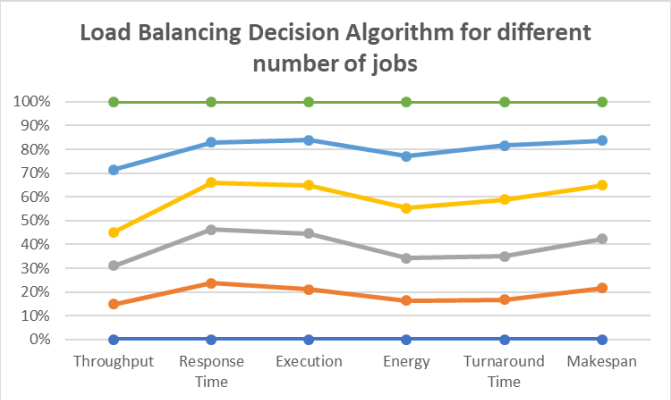

Figure 13: Load Balancing Decision Algorithm for different number of jobs

Bat Optimization Algorithm: This load balancing algorithm works on the principle of grouping similar ones and working on them group wise. The performance of the system is enhanced with high resources thereby increasing the parameter outcome using the algorithm. This algorithm is degraded with an increase in system diversity. A node initiates the process and selects another node called the matchmaker node from its neighbors, satisfying the criteria that it should be a different type than the former one.

Table 2 Simulation results for Bat Optimization for different number of jobs.

S.No	No. of Jobs	Parameters Name					
		Throughput	Response Time	Execution Time	Energy Consumption	Turnaround Time	Makespan
1.	5	0.7559	0.7023	6.6143	5.3100	7.316	46.30
2.	6	0.819	0.6667	7.3258	5.8432	7.992	43.955
3.	7	0.7236	0.5892	6.3527	6.8211	10.472	48.369
4.	8	1.3330	0.5012	6.001	7.1027	10.021	40.021
5.	9	1.4521	0.5032	5.021	7.4521	8.0213	35.021

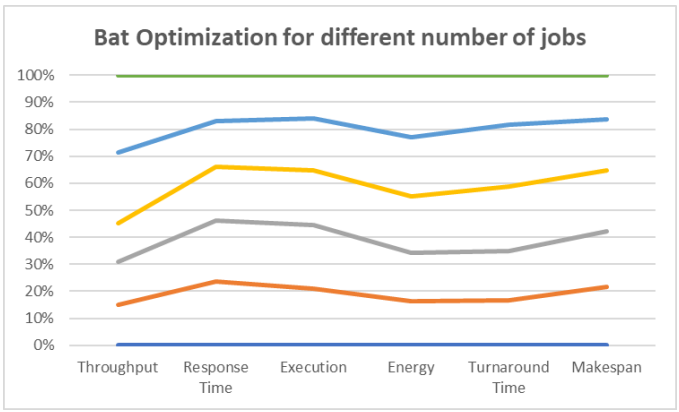

Figure 14: Bat Optimization for different number of jobs

Comparison Tables:

All three algorithms are implemented and compared on CloudSim tool for energy efficiency and load balancing. Table 3 depicts the result of different parameters for five jobs. During the comparison, Vector dot technique is counted as best model for producing the good results according to the user requirements.

Table 3 Different algorithms are compared for 5 Jobs.

Algorithms	No. of Jobs	Parameters Name					
		Throughput	Response Time	Execution Time	Energy Consumption	Turnaround Time	Makespan
Load Balancing Decision Algorithm	#5	0.4051	0.8571	12.45	10.56	13.081	87.157
Bat Optimization	#5	0.7559	0.7023	6.6143	5.3100	7.316	46.30

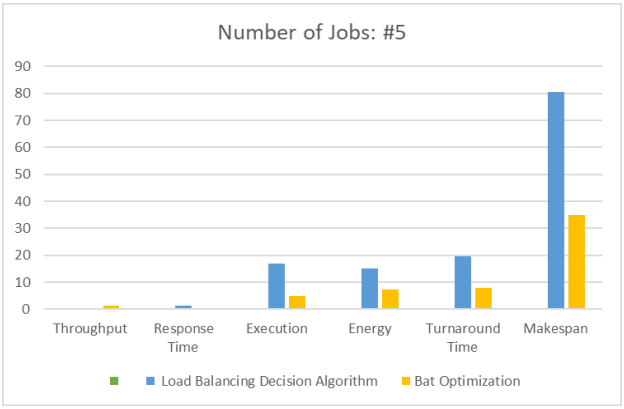

Figure 15: Comparison between Load Balancing Decision Algo with Bat Optimization for 5 Jobs

Table 4 Different algorithms are compared for 06 Jobs.

Algorithms	No. of Jobs	Parameters Name					
		Throughput	Response Time	Execution Time	Energy Consumption	Turnaround Time	Makespan
Load Balancing Decision Algorithm	#6	0.42	1.00	14.27	12.09	15.27	85.65
Bat Optimization	#6	0.819	0.6667	7.3258	5.8432	7.992	43.955

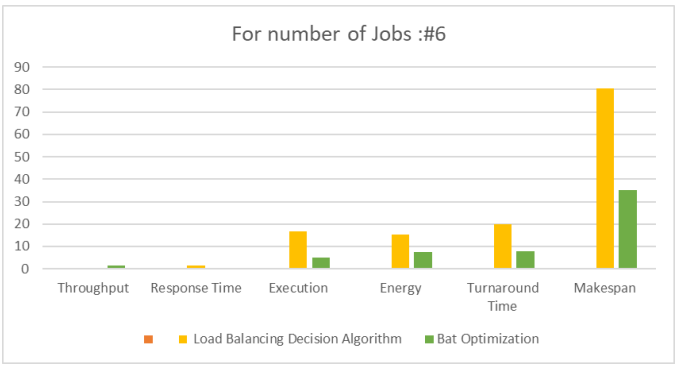

Figure 16: Comparison between Load Balancing Decision Algo with Bat Optimization for 6 Jobs

Table 5 depicts the result of different parameters for 7 jobs

Algorithms	No. of Jobs	Parameters Name					
		Throughput	Response Time	Execution Time	Energy Consumption	Turnaround Time	Makespan
Load Balancing Decision Algorithm	#7	0.423	1.2	16.53	13.97	17.73	82.65
Bat Optimization	#7	0.7236	0.5892	6.3527	6.8211	10.472	48.369

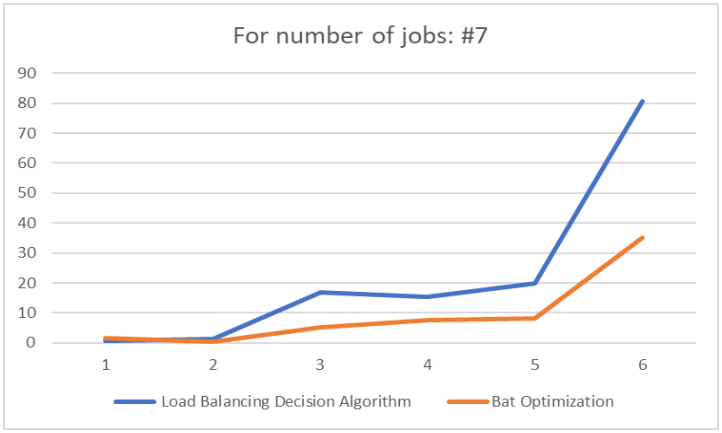

Figure 17: Comparison between Load Balancing Decision Algo with Bat Optimization for 7 Jobs

Table 6 depicts the result of different parameters for 9 jobs.

Algorithms	No. of Jobs	Parameters Name					
		Throughput	Response Time	Execution Time	Energy Consumption	Turnaround Time	Makespan
Load Balancing Decision Algorithm	#9	0.5444	1.4	16.87	15.21	19.73	80.52
Bat Optimization	#9	1.4521	0.5032	5.021	7.4521	8.0213	35.021

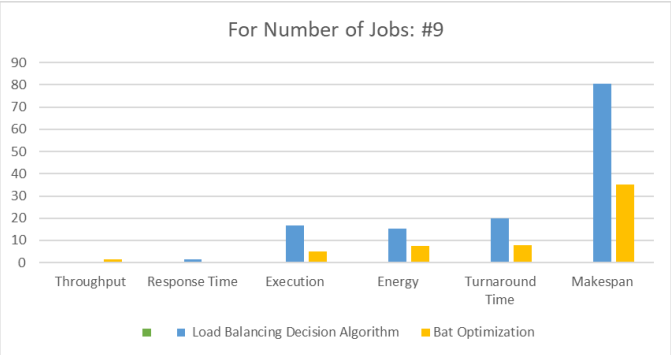

Figure 18: Comparison between Load Balancing Decision Algo with Bat Optimization for 9 Jobs

All two algorithms are compared for energy efficiency and load balancing. The results show that Bat Optimization is better because which consumes less energy and all the tasks are executed in less time with no delay. It concluded that Bat Optimization is best in the energy efficient technique in cloud computing.

V. CONCLUSION AND FUTURE SCOPE

In recent years, energy efficiency has emerged as one of the most important design requirements for modern computing systems, ranging from single servers to data centers and Clouds, as they continue to consume enormous amounts of electrical power. Apart from high operating costs incurred by computing resources, this leads to significant emissions of CO2 into the environment. For example, currently, IT infrastructures contribute about 2% of the total CO2 footprints. Unless energy-efficient techniques and algorithms to manage computing resources are developed, its contribution in the world's energy consumption and CO2 emissions is expected to rapidly grow. It has been shown that proper load balancing of computing resources can lead to a significant reduction of the energy consumption by a system, while still meeting the performance requirements. A relaxation of the performance constraints usually results in a further decrease of the energy consumption. Load balancing that is required to distribute the excess dynamic local workload evenly to all the nodes in the whole Cloud to achieve a high user satisfaction and resource utilization ratio. In the research work we have proposed and implemented a Bat optimization on cloud environment using CloudSim Toolkit. And compared it with the load balancing decision algorithm. The results show that proposed technique is much better than the existing load balancing methods in terms of Response time, Execution Time, and Throughput. We also concluded that Bat optimization technique consumes less energy than Central Load Balancer.

Future Work

Cloud Computing is a vast concept and energy efficiency plays a very important role in case of Clouds. There is a huge scope of improvement in this area. We have implemented only two dynamic load balancing algorithms. But there are still other approaches that can be applied to balance the load and energy consumption in clouds. The performance of the given algorithms can also be increased by varying different parameters. We can also move our research work on any Private Cloud for the Security and further enhancements.

VI. REFERENCES

[1] Atyaf Dhari and Khaldun I. Arif , "An Efficient Load Balancing Scheme for Cloud Computing" in the Indian Journal of Science and Technology, Vol 10(11), March 2011.

[2] Amir Nahir , Ariel Orda, Danny Raz "Schedule First, Manage Later: Network-Aware Load Balancing" in the Proceedings IEEE INFOCOM, 2013

[3] H.Jamal, A.Nasir, K.Ruhana, K.Mahamud and A.M. Din, "Load Balancing Using Enhanced Ant Algorithm in Grid Computing", Proceedings of the Second International Conference on Computational Intelligence, Modelling and Simulation, pp. 160-165,2010.

[4] Isam Azawi Mohialdeen,2013, Comparative Study Of Scheduling Algorithms In Cloud Computing Environment .Journal of Computer Science, 9 (2): 252-263, 2013 ISSN 1549-3636 © 2013 Science Publications.

[5] Jianzhe Tai Juemin Zhang Jun Li Waleed Meleis Ningfang Mi "ARA: Adaptive Resource Allocation for Cloud" in the proceeding IEEE 2011

[6] Kumar Nishant, Pratik Sharma, Vishal Krishna, Nitin and Ravi Rastogi "Load Balancing of Nodes in Cloud Using Ant Colony Optimization" in the 14th International Conference on Modeling and Simulation, IEEE 2012

[7] Klaithem Al Nuaimi, Nader Mohamed, Mariam Al Nuaimi and Jameela Al-Jaroodi "A Survey of Load Balancing in Cloud Computing: Challenges and Algorithms" in the proceeding IEEE 2012.

[8] Mohammad Goudarzi, Mehran Zamani, Abolfazl Toroghi Haghighat, "A fast hybrid multi-site computation offloading for mobile cloud computing" in the Journal of Network and Computer Applications, Elsevier 2017.

[9] Muhammad Baqer Mollah, Md. Abul Kalam Azad, Athanasios Vasilakos, "Security and privacy challenges in mobile cloud computing: Survey and way Ahead" in the Journal of Network and Computer Applications, Elsevier 2017.

[10] Mohamed Abu Sharkh, Abdelkader Ouda, and Abdallah Shami, "AResource Scheduling Model for Cloud Computing Data centers" in the proceeding IEEE 2013.

[11] M.sudha, M.Monica:"Investigation on Efficient Management of Workflows in Cloud Computing Environment", International Journal of Computer Science and Engineering (IJCSE), Volume 02, Number 05, August 2010, Pages 1841- 1845.

[12] Parveen Patel ,Deepak Bansal, Lihua Yuan, Ashwin Murthy, Albert Greenberg " Ananta: Cloud Scale Load Balancing" in SIGCOMM, August12–16,2013, HongKong,China.

[13] Prof. Dr. Jayant. S. Umale, Miss. Priyanka A. Chaudhari, "Survey on Job Scheduling Algorithms of Cloud Computing" International Journal of Computer Science and Management Research , 2013.

[14] Parin.V.Patel ,Hitesh.D.Patel, Pinal.J.Patel, "*A Survey Of Load Balancing In Cloud Computing*" IJERT, Vol.1,Issue 9,November 2012.

[15] P.Salot, "*A Survey of various Scheduling algorithm in cloud computing Environment*" ,IJRET ,Volume:2,Issue:2,Feb 2012.

[16] Shiva Razzaghzadeh, Ahmad Habibizad Navin, Amir Masoud Rahmani, Mehdi Hosseinzadeh, " Probabilistic modeling to achieve load balancing in Expert Clouds" ElsevierB.V 2017.

[17] S.Maguluri, R.Srikant, and L.Ying, "Stochastic Models of Load Balancing and Scheduling in Cloud Computing Clusters," IEEE INFOCOM 2012 Proceedings.pp.702- 710,25-30Mar,2012

[18] Vanitha, P. Marikkannu (2017).Effective resource utilization in cloud environment through a dynamic well-organized load balancing algorithm for virtual machines. Elsevier Ltd.

[19] Saeed Javanmardi , Mohammad Shojafar, Danilo Amendola, Nicola Cordeschi "Hybrid Job scheduling Algorithm for Cloud computing Environment" published in SpringerVerlag Berlin Heidelberg 2014.

[20] Suriya Begum, Dr. Prashanth C.S.R, "*Review of load balancing in cloud Computing".* 10, Issue 1, No 2, January 2013.

[21] Rajesh Gorge Rajan and V.Jeyakrishnan *"A Survey on Load Balancing in Cloud Computing Environments"*Vol-2.Issue-12,December 2013.

[22] Tom Guérout, Samir Medjiah, Georges Da Costa, Thierry Monteil (2014)," *Quality of service modeling for green scheduling in Clouds*", In Elsevier.

[23] Saeed Parsa and Reza Entezari-Maleki," RASA: A New Task Scheduling Algorithm in Grid Environment" in World Applied Sciences Journal 7 (Special Issue of Computer & IT): 152-160, 2009.Berry M. W., Dumais S. T., O'Brien G. W. Using linear algebra for intelligent information retrieval, SIAM Review, 1995, 37, pp. 573-595.

[24] S. K. Garg, C. S. Yeob, A. Anand asivamc, and R. Buyya,"Environment-conscious scheduling of HPC applications on distributed Cloud-oriented data centers", Journal of Parallel and Distributed Computing, Elsevier, Vol. 70, No. 6, May 2010, pages 1-18.

[25] Suresh M., Shafi Ullah Z., Santhosh Kumar B.," An Analysis of Load Balancing in Cloud Computing", International Journal of Engineering Research & Technology (IJERT),Vol. 2 Issue 10, October – 2013, ISSN: 2278-0181.