LOAD BALANCING ALGORITHMS IN CLOUD COMPUTING USING MODERN HYBRID TECHNIQUES

Abstract—Cloud Computing is the next generation of computation. It provides software or hardware resources to the client on demand. Cloud Computing is distributed computing, storing, sharing and accessing data over the Internet. As Cloud Computing is growing rapidly and clients are demanding more services and better results, the data processing and storage amount is growing day by day in the cloud environment. The load on the cloud is increasing tremendously with the development of new applications. Clients in a distributed environment generate requests randomly in any processor. This leads to an uneven distribution of overall work on cloud resources i.e., some of the processors are overloaded and some of them are under loaded. Therefore, load balancing is one of the central issues in cloud computing. The objective of load balancing is to ensure that no single node will be overloaded and used to distribute workload among multiple nodes. It helps to achieve high performance, minimum response time and high resource utilization. Different models and algorithms for load balancing in cloud computing have been developed with the aim to assign the client's requests to available cloud nodes. Hybrid Techniques combine different approaches to obtain optimized and better results than traditional algorithms. Hence this paper presents the different hybrid and optimized algorithms to resolve the issue of load balancing in Cloud Computing. Advantages and Issues regarding each technique have been discussed briefly. Also different metrics associated with load balancing have been discussed.

Keywords—Cloud Computing, Load Balancing, Virtual machine, Load Balancer, Hybrid Algorithm.

I. INTRODUCTION

Cloud computing is a type of Internet-based computing which is the extended version of parallel computing, distributed and grid computing. It is a newly emerging technique that relies on sharing computing resources rather than having local servers or personal devices to handle applications. This type of computing provides different services such as servers, storage and applications which are delivered to an organization's computers and peripherals through the Internet [1]. It also provides conventional data storage, software, platform and online usage of various resources with the help of virtual machines. The intended users can efficiently use these services provided by cloud infrastructure paradigm and can pay for the requested services. Cloud computing applies traditional supercomputing, or high-performance
computing power and to do this, it uses networks of large groups of servers which run low-cost consumer PC technology with specialized connections to spread data-processing tasks across them. This shared IT infrastructure contains large pools of systems that are linked together. To make effective use of cloud computing services scheduling algorithms are needed. These algorithms are applied by cloud resource manager which manages to dispatch chores to cloud resources in an optimal fashion. The main aim of scheduling algorithms is to minimize the execution time and response while leading to maximum throughput. Virtualization is the technique which is often used to maximize the power of cloud computing. In today’s scenario there is an incredible increment in demand for higher computations. Due to this reason there is an uneven and heavy workload on cloud resources. So, we needed a technique to ensure that there is well distributed load on each device in cloud platform and to fulfill this requirement load balancing emerged. Load balancing is a technique which helps in balancing the load on various devices and cloud resources [2][3].

II. LOAD BALANCING IN CLOUD COMPUTING

Cloud computing is an integrated concept of parallel and distributed computing which shares resources like hardware, software, and information to computers or other devices on demand. The random arrival of load in cloud environment can cause some server to be heavily loaded while other server is idle or only lightly loaded. Equally load distributing improves performance by transferring load from heavily loaded server. Cloud computing uses the concepts of scheduling and load balancing to migrate the tasks to under-utilized VMs for effectively sharing the resources. Efficient scheduling and resource allocation is a critical characteristic of cloud computing based on which the performance of the system is estimated.

Virtual machine (VM) is an execution unit that acts as a foundation for cloud computing technology. The VMs in the cloud computing environment share resources like processing cores, system bus, and so forth [5]. The computing resources available for each VM are constrained by total processing power. In this model of environment the job arrival pattern is unpredictable and also the capabilities of each virtual machine vary from one another. Hence, load balancing becomes a critical task leading to a poor system performance and maintaining stability. Thus, it becomes imperative to develop an algorithm which can improve the system performance by balancing the work load among virtual machines. There are various load balancing algorithms available, such as Static Load Balancing Algorithm - round robin, weighted round robin, Equally Spread Current Execution (ESCE) Algorithm and Dynamic Load Balancing Algorithm - Ant Colony algorithm, Honey bee foraging algorithm and Throttled algorithm.

Static Load Balancing Algorithm - Static algorithms are appropriate for systems with low variations in load. In static algorithms the traffic is divided evenly among the servers. This algorithm requires a prior knowledge of system resources the performance of the processors is determined at the beginning of the execution, therefore the decision of shifting of the load does not depend on the current state of system. However, static load balancing algorithms have a drawback in that the tasks are assigned to the processor or machines only after it is created and that tasks cannot be shifted during its execution to any other machine for load balancing.

Dynamic Load Balancing Algorithm - In dynamic algorithms the lightest server in the whole network
system is searched and preferred for balancing a load. For this real time communication with network is needed which can increase the traffic in the system. Here, current state of the system is used to make decisions to manage the load. Dynamic algorithms respond to the actual current system state in making load transfer decisions. Since current state of the system is used to make dynamic load balancing decisions, processes are allowed to move from an over utilized machine to an underutilized machine in real time dynamic.

An easy way to comply with the conference paper formatting requirements is to use this document as a template and simply type your text into it.

Fig. 1 Load Balancer in Cloud Computing [13]

III. RELATED WORK

Saeed Javanmardi et al. (2014) [15] proposed a Hybrid Job Scheduling algorithm using Genetic Algorithm and Fuzzy Theory. The proposed algorithm assigns the jobs to the resources with considering the job length and resources capacities. This Algorithm has been implemented on the CloudSim and the results show that this hybrid approach outperformed other algorithms.

Zhanghui Liu et al. (2012) [14] proposed a PSO-based techniques to balance the load in VMs (virtual machines) of Cloud Computing Environment. The author proposed a new task scheduling model to optimize the time for task execution and utilization of resource.

Kumar Nishant, Pratik Sharma et al. (2012) [16] proposed an algorithm for load distribution of workloads among nodes by the use of modified Ant Colony Optimization (ACO). The main benefit of this approach is detections of overloaded and underloaded nodes.

IV. HYBRID LOAD BALANCING APPROACHES

The key objectives of load balancing are to distribute the work load among various VM’s optimally, to improve the performance and to maintain system stability [12]. Thus, to meet all the objectives of load balancing, a lot of research has been done on load balancing. Various algorithms such as Round-Robin, Throttled, Min-Min, and swarm based techniques etc. have been used to solve the problem of load balancing.

In this paper we will be discussing various hybrid techniques as provide better and optimal results than traditional algorithms.

A. Hybrid Multiple Agent Genetic Algorithm for Load Balancing

The main objective of this algorithm is to obtain better results than traditional genetic algorithm. Genetic Algorithm is a search heuristic, based on evolutionary algorithm process of natural selection. GA mainly uses to repeat three steps - selection, crossover, and mutation [7]. In MAGA, each individual MAGA is considered an agent, capable of sensing, changing, and impacting its environment autonomously, thus possessing its own characteristics. The genetic operators in case of MAGA include: neighborhood competition operator, neighborhood orthogonal crossover operator, mutation operator, and the self-
learning operator. Among these operators, the neighborhood competition operator realized the operation of competition among all agents; the neighborhood orthogonal crossover operator achieved collaboration among agents; the mutation and self-learning operators accomplished the behavior that agents exploit their own knowledge [6]. Results show that this algorithm is better than Min_min scheduling in terms of CPU utilization and memory load balancing when weighting parameters are adjusted. It provides better and optimized results in handling high-dimensional function problems. This hybrid approach for achieving load balancing in cloud computing is thus more effective than GA. Algorithm execution flow is shown as follows:

Step 1: Randomly generate \( L_{size} \) agents, and initiate \( L_0 \), and then update \( Best_0 \), assuming \( t ←0 \).

Step 2: Execute neighborhood competition operator for each agent in \( L_t \), and then obtain \( L_{t+1/3} \).

Step 3: If \( U(0, 1) < P_c \), then apply neighborhood orthogonal crossover operator into each agent in \( L_{t+1/3} \) to generate \( L_{t+2/3} \).

Step 4: If \( U(0, 1) < P_m \), then applies the mutation operator into each agent in \( L_{t+2/3} \), and then achieves \( L_{t+1} \).

Step 5: Determine the \( CBest_{t+1} \) from \( L_{t+1} \) and apply the self-learning operator into \( CBest_{t+1} \).

Step 6: If \( \text{Energy} (CBest_{t+1}) > \text{Energy} (CBest_t) \), then assume \( Best_{t+1} ← CBest_{t+1} \); otherwise, \( Best_{t+1} ← Best_t \) \( ← CBest_{t+1} ← Best_t \).

Step 7: If it meets the termination conditions, output \( Best \) and terminate; otherwise, set \( t ←t+1 \) and then resume at Step 2. \( L_t \) represents the \( t \)th generation agent network, and \( L_{t+1/3} \) and \( L_{t+2/3} \) is the middle generation agent network of \( L_t \) and \( L_{t+1} \). \( Best_t \) is the optimal agent among \( L_0, L_1, ..., L_t \), and \( CBest_t \) represents the optimal agent among \( L_t \). The parameters, \( P_c \) and \( P_m \), are preset, and represent the executive probability of neighborhood orthogonal crossover operator and mutation operator, respectively.

B. Hybrid VM Load Balancing Algorithm

The proposed approach is a combination of Throttled (TVLB) and ESCE (Equally Spread Current Execution) which is also called as AVLB (Active VM Load Balancing) algorithm. Throttled Load Balancing Algorithm is a dynamic approach. Data Center Controller (DCC) is assigned with requests from various clients and DCC further assigns a task to VM Load Balancer so that request is provided to the VM that has less workload or can handle workload easily. TVLB Algorithm maintains an index table of virtual machines as well as their states (AVAILABLE or BUSY) [11]. AVAILABLE state indicates that the virtual machine is idle/free and ready for cloudlet allotment, where BUSY state indicates that the current virtual machine is busy in execution of previous cloudlets and is not available to handle any new cloudlet request. This current load state of a VM helps in taking decision whether to allocate cloudlets to virtual machines or not. In case of ESCE, a job queue keeps all the requests that VMs need for the execution and allot them to the VMs that are idle or free. It also maintains the list of cloudlets allocated to each virtual machine. This allocated cloudlet list helps in determining whether a VM is overloaded or underloaded at particular moment of time. On the basis of this information, VM load Balancer moves some load from overloaded VMs to the VM having minimum number of cloudlets, so as to maintain proper distribution of load among virtual machines [8]. This Hybrid Approach is found to be better than individual TVLB and ESCE algorithms with fair distribution of load, reduced average response time and reduced data.
processing time. Hybrid VM Load Balancing Algorithm is defined as below:

**Input**- Userbases/Cloudlets UB1, UB2,... UBn. - Available VMs VM1, VM2, VM3,...VMn within data center.

**Step1:** Hybrid VM Load Balancer maintains a list of VMs, their states (AVAILABLE/BUSY) and allocated cloudlets list. Initially state of every VM is AVAILABLE and allocated cloudlet list is empty.

**Step 2:** Data Center Controller gets cloudlet requests from cloud clients.

**Step 3:** Data Center Controller asks the Hybrid VM load Balancer for available VM.

**Step 4:** Hybrid VM load Balancer do
a) Find the next available VM using VM State List.
b) Check if present allocation count is less than maximum VM list length and length of virtual machine list is greater than zero, then allocate the VM.
c) Determine the current cloudlet load on every VM.
d) Return vmId of those VM which have minimum load.

**Step 5:** Hybrid VM Load Balancer allocates the cloudlet over available VM.

**Step 6:** If a VM get overloaded then hybrid VM Load balancer moves some workload on VM that have minimum workload

**Step 7:** The DCC get the reply of sent cloudlet & then allots a waiting request from job pool to hybrid VM Load Balancer.

**Step 8:** continue with step 4.

**Output**- Userbases/cloudlets are allocated on the available VMs and completed with minimum response time and processing time at DC.

C. Hybrid Ant Colony and particle swarm optimization for VM scheduling with load balancing

This hybrid approach combines ant colony and particle swarm optimization (ACOPS) to solve scheduling in Virtual machines so that VM’s are assigned to servers and resource usage is best utilized. ACOPS uses historical information to predict the workload of new input requests to adapt to dynamic environments without additional task information [9]. Every time the request arrives, the initial step of ACOPS is Pre-reject in which the algorithm checks for the remaining memory of each server and will find the maximum amount of remaining memory. When the memory demands of request exceed the maximum remaining memory, the request will be rejected before scheduling. Search operator is used to construct the solutions for all ants. PSO operator is applied to improve the search result. In the next step, the Evaluation operator is used to estimate the scheduling score of each ant. It will find the best solution and update the global best solution. In the next step, global pheromone updating is applied. This algorithm can serve requests for CPU, memory and Disk Utilization. Due to pre-reject step, computation time is reduced. PSO operator further reduces the computing time and improves the scheduling result. Thus, ACOPS ensures better load balancing than individual Ant Colony and Particle Swarm optimization Algorithms [9]. The ACOPS algorithm is defined as below:

Every time requests come;

**do** Pre-reject;

**if** (There is any request needing to be scheduled.) **then**

Initialization;

**while** (Termination test) **do**

Search;

PSO operator;

Evaluation;

Pheromone update;

**end**
**D. Hybrid approach using Divide-and-conquer and throttled algorithm for load balancing**

This algorithm combines the methodology of Divide-and-conquer and Throttled Algorithm (DCBT) which schedules the incoming requests to available VMs efficiently and ensures that there is no starvation of the requests. This hybrid approach consists of two algorithms: Pass I and Pass II. According to this algorithm [10] requests from different clients are provided to the available Request Handlers (RH) and Virtual machines (VM). In the initial step, this algorithm checks for the availability of RH’s and VM’s and divides the requests accordingly using divide and conquer approach. In the next step, the incoming requests are assigned to the different RH’s and VM’s. Load Balancer keeps track of the current status of each RH or VM and verifies that the current request should only be assigned to the RH and VM which has not been used recently. This algorithm ensures that the load is distributed in an optimized way and no resource is idle thus leading to maximum resource utilization and minimum execution time thus leading to high performance. The algorithms for Pass I and Pass II are as follows:

I) **Algorithm: Pass I**

1. **Step 1:** Find the number of tasks in queue for a period 't'

2. **Step 2:** Find the number of available RH in a real time distributed system.

3. **Step 3:** Tasks are divided continuously by the number of available RH in a real time distributed system.

4. **Step 4:** After the division, check remainder == number of RH then, assign tasks to the RH for execution

5. **Step 5:** Step 1 to 4 iterates continuously in coordination with pass II until it finishes the execution.

II) **Algorithm: Pass II**

1. **Step 1:** Initially Load Balancer assigns the tasks to the available Request Handlers or Virtual Machines.

2. **Step 2:** For the next assignment, Load Balancer checks for the availability of RH.

3. **Step 3:** Next task is allocated to available RH or VM, iff

   1. RH or VM should be free.
   2. Assigned RH or VM should be used for the previous assignment.

4. **Step 4:** Step 1 to 3 are repeated in coordination with Pass I until all the tasks are executed completely.
## V. Summary of the Reviewed Techniques

### TABLE I

<table>
<thead>
<tr>
<th>Technique</th>
<th>Authors</th>
<th>Key Objective</th>
<th>Advantages</th>
<th>Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hybrid Multi-Agent Genetic Algorithm</td>
<td>Zhu, Song, Liu, Gao, Cheng et al (2011) [6]</td>
<td>To achieve better load balancing as compared to traditional Genetic Algorithm and Min_min Algorithm. To Handle high-dimensional function optimization problems.</td>
<td>Superior Performance than traditional Genetic Algorithm. It can achieve effective load balancing of both CPU and memory as compared to Min_min Algorithm. Single-point failure rate of MAGA is much smaller than Min_min.</td>
<td>To achieve higher performance in cloud, influence of memory will need to be ignored, thus providing load balancing for CPU only.</td>
</tr>
<tr>
<td>Hybrid VM Load Balancing Algorithm</td>
<td>Mamta Khanchi, Sanjay Tyagi et al (2016) [8]</td>
<td>Efficient and effective distribution of load as compared to existing Throttled and Active VM Load Balancer</td>
<td>Reduced Average response time, Request servicing time of data center, Overall response time and processing time as compared to Round-Robin, Throttled and Active VM Load Balancer Algorithms.</td>
<td>Overall data center processing time is not much improved as compared to Active VM Load Balancing.</td>
</tr>
<tr>
<td>Hybrid Ant Colony and particle swarm optimization for VM scheduling with load balancing</td>
<td>Cho, Tsai, Yang et al (2014) [9]</td>
<td>To solve the VM scheduling problem using hybrid approach To predict the workload of new input requests to adapt to dynamic environments without additional task information.</td>
<td>Provide VM services for CPU, Memory and Disk utilization Reduced Execution time, response time and computation time Obtain shorter make span while maintaining higher load balancing and lower rejection rate</td>
<td>Although this algorithm can provide high load balancing, make span is shorter in single scheduling.</td>
</tr>
<tr>
<td>Hybrid approach using Divide-and-conquer and Throttled algorithm for load balancing</td>
<td>Shridhar G.Domanal and G. Ram Mohana Reddy et al (2015) [10]</td>
<td>to reduce the total execution time of the tasks and thereby maximizing the resource Utilization.</td>
<td>Improves the efficiency of execution time and resource utilization Tasks are fairly distributed by the load balancer Distributes incoming jobs among all VM’s equally</td>
<td>Deadline constraints are not considered.</td>
</tr>
</tbody>
</table>
Table II
Load Balancing Metrics in Reviewed Techniques

<table>
<thead>
<tr>
<th>Reference</th>
<th>Response Time</th>
<th>Throughput</th>
<th>Performance</th>
<th>Resource Utilization</th>
<th>Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mamta Khanchi, Sanjay Tyagi et al (2016) [8]</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Cho, Tsai, Yang et al (2014) [9]</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Fig 2: Load Balancing Metrics in Reviewed Techniques

VI. CONCLUSIONS

Virtualization technology in cloud computing provides good support to achieve aim of cloud computing like higher resource utilization, elasticity, reducing IT cost to handle temporary loads as well as cloud computing. Load balancing is an important issue which affects the utilization of resources and performance of the system in cloud environment. Hence, in this paper we have surveyed various hybrid and optimized techniques to solve the problem of load balancing. The vital part of this paper is comparison of reviewed techniques on the basis of advantages and limitations for these algorithms. These algorithms are also compared on the basis of metrics like response time, throughput, performance, resource utilization and speed.

VII. REFERENCES

[2] Sun Microsystems, Inc.”Introduction to Cloud


