# Chapter 1 Introduction

*In this chapter we introduced our thesis overview, our work, motivation, objective and organization. In section 1.1 we discussed about thesis introduction; in section 1.2 we discussed about work; in section 1.3 we discussed about our thesis motivation; in section 1.4 we discussed about our thesis objective; in section 1.5 we discussed about the whole thesis paper organization; in section 1.6 we should give a short discussion about this chapter.*

### Introduction

Cloud computing is experiencing a rapid development both in academia and industry; it is promoted by the business rather than academic which determines its focus on user applications. This technology aims to offer distributed, virtualized, and elastic resources as utilities to end users. It has the potential to support full realization of ‘computing as a utility’ in the near future [1]. Users can host different kinds of applications on the cloud ranging from simple web applications to scientific workloads [2]. With the support of virtualization technology [3, 4], cloud platforms enable enterprises to lease computing power in the form of virtual machines to users. Because these users may use hundreds of thousands of virtual machines (VMs) [5], it is difficult to manually assign tasks to computing resources in clouds [6, 7]. So, we need an efficient algorithm for task scheduling as well as balance the load in the cloud environment. A good task scheduler should adapt its scheduling strategy to the changing environment and the types of tasks. Therefore, a dynamic task scheduling algorithm, such as Ant Colony Optimization (ACO) [8, 9], is appropriate for clouds. Load balancing [10] is a process of dividing and distributing large processing jobs among different processing units to enhance the overall performance of the system. The task of load balancing is to improve both resource utilization Applied Mathematics, Computational Science and Engineering and job response time by avoiding overloading or under loading of any specific node in a distributed system environment, thereby achieving the Service level objectives [11]. Load balancing helps in the fair allocation of computing resources to achieve a high level of user satisfaction and proper use of resources. High resource utilization and proper load balancing help minimize resource consumption. This helps to implement fault tolerance, scalability and avoid difficulties [12]. Load balancing is a method that has helped networks and resources, to provide maximum throughput with minimal response time. Load balancing is performed at two levels in cloud computing [13]

In this paper, a Modified Ant Colony Optimization (MACO) for cloud Load balancing is proposed with the support of virtualization technology. The main goal of MACO is to enhance the performance of ACO algorithm.

### 

### Related Work

Different researchers have been worked on task scheduling as well as load balancing. In this section we focused on Ant Colony and Modified Ant Colony algorihm for better work performance in cloud environment from reference section.

Cloud computing and its applications have come into wide use such as security, iot and vehicular ad hoc networks it is important for cloud users to provide an effcient task scheduling technique since cloud computing is based on the pay as you go pricing model [14].

Load balancing ensures that not all tasks are given to one VM and overloading it. Load balancing techniques must consider the amount of load each VM has by considering the amount of load in each VM, performance of the system, and distribution of load among all available VM [15]. In other words, load balancing algorithms allocate surplus task from an overloaded VM to other VMs [16].

Task Scheduling is a Non-Polynomial Complete problem. Thus, several heuristic approaches and meta-heuristic approaches have been explored in the past to resolve this problem. They have been applied into cluster, grid or cloud environment with appropriate alterations. Min-min is a popular heuristic-based scheduling algorithm that assigns tasks to resources that executes them in the fastest possible time. Etminani et al. [17] presented another scheduling heuristic using two algorithms: Max-Min and Min-Min for task scheduling in grid environment.

Maheswaram et al. [18] designed a dynamic mapping for a set of independent tasks and is applied on heterogeneous systems. Tasks that have the minimum expected time for completion are deduced and assigned to the corresponding machine.

Several meta-heuristic algorithms like Simulated annealing (SA), Genetic algorithm (GA), Particle swarm optimization (PSO) and Ant colony optimization (ACO) etc. are used for scheduling. Ant algorithms are one of the most popular examples of swarm intelligence systems. It has already been applied to solve a number of complex problems, such as task allocation in grid environment [19 and 20].

Fidanova et al. [21] proposed scheduling algorithm on grid computing based on ACO using Monte Carlo technique. It targets in achieving the search for best tasks scheduling for Grid. Similar work has been done in [14] for task scheduling based on ACO but by using weighted Directed Acyclic Graph (DAG) corresponding to dependent task/workflow applications.

The work proposed by Lorpunmanee et al. [22] proposed the scheduling problem by introducing a model based on dynamic information on grid using ACO algorithm. Load balancing under tri-level cloud computing was introduced in [23] by combining Load Balancing Min-Min (LBMM) and Opportunistic-LoadBalancing (OLB) to maintain the load balancing of system and enhance efficiency for execution.

In [24] researchers proposed a Stochastic hill climbing approach for the load balancing to maximize the optimization of resources on cloud computing. They have presented it as soft computing-based load balancing approach. They showed the performance of algorithm with the help of simulation and found that this algorithm gives minimum response time than First Come First Serve (FCFS) and Round Robin (RR).

In [25] authors proposed a Weighted Signature based load balancing (WSLB) algorithm that minimizes response time to the user. They described using simulation on Cloud Analyst tool that WSLB outperforms the existing Round Robin, Equally Spread Current Execution (ESCE) load and Throttled algorithms in terms of response time.

Authors in [26] proposed dynamic load balancing algorithm which is based on the concept of least frequently used mechanism. They described that data center processing choice is made upon virtual machine. They compared the performance of algorithm with existing Round Robin and Throttled algorithm and found that it gives better response time. Authors suggest that in order to get better performance in cloud computing this algorithm can be combined with priority based and least recently used algorithm.

Kokilavani et al. [27] proposed a MinMin (LBMM) algorithm primarily for load balancing which enhances resource utilization and reduces makespan. This has been done in two phases: the former phase executes the conventional Min-Min algorithm, while the later reschedules of tasks for effective resource utilization.

### Motivation

Many of the previous works mainly focus on how balance the load in cloud environment dynamically. Mostly the paper showed a wonderful performance in balancing the load using their developed algorithm inspired from the base algorithm “Modified Ant Colony Load Balancing Algorithm”. We want to develop such an algorithm, for that case we take in concern the task scheduling of various load, response time of virtual machines, datacenter processing time. We have shown our improvement result of our modified algorithm and also discussed how we can further improve our algorithm for better performance in future.

### Thesis Objective

The objective of this paper is to propose a load balancing algorithm aims to distribute the dynamic workload smoothly to all the hosts in the cloud to gain an improvement in both the utilization of resources and the speed of execution time. It allocates the incoming tasks to all available VMs. In order to achieve fairness and avoid congestion, the proposed algorithm allocates tasks to the least loaded VM. This leads to a reduction of the overall response time and the processing time of hosts. In the proposed algorithm, variation of processing time of VM is the key limiting factor during the task allocation process because it avoids underutilization and over utilization of VMs.

The main objective of our proposed work is,

* To distribute the workload among server to main the load.
* To improve resource utilization, minimum completion time and improve Performance of system.
* To optimize the resources while maintaining the budget with less response time and execution time.

### 

### 1.5 Organization of the Thesis

In this section we discussed about the organization of the thesis.

This chapter (CHAPTER 1: Introduction) presents an overview of the background of our work such as related work, motivation and our objective.

CHAPTER 2: Literature Review presents an overview of thesis literature, a clear concept of Cloud Computing, Load balancing, Different types of cloud services, About load balancing algorithms and previous works of others researcher in this field.

CHAPTER 3: System and Architecture represents has the contents about the problem statement, mathematical representation of the problem’s solution. Moreover, it also describes about our work flow and proposed method.

CHAPTER 4: This chapter is the most important chapter as it contains all important chapter as it contains all the details information about our work, work implementation process and how we have implemented it.

CHAPTER 5: In this chapter we have our output of work, we have shown validation of work and also compared our proposed algorithm with other base algorithms and also provide our decision in which way we have done a better job.

CHAPTER 6: Conclusion is the last chapter in this paper. These chapter represents a clear discussion about all the workflows with results analysis. Then a short description about the future work availability in these research fields. Finally, a short conclusion presented as ending of our work.

###### Discussion

This is the introduction chapter and this chapter just introduces about our thesis, previous

related work done by different authors and our goals. Also, it shows the blueprint of our work

**Chapter 2 Literature Review**

*In this chapter we introduced our thesis literature, Cloud Computing, task scheduling and load balancing description. In section 2.1 we discussed about literature introduction; in section 2.2 we discussed thesis literature as Cloud Computing, Load Balancing; in section 2.3 we discussed about different types of algorithms and benefits of using these algorithms in cloud load balancing process; in section 2.4 we should summarize the chapter,*

### Introduction

A thesis is a statement in a non-fiction or a fiction work that a writer intends to support and prove. To understand our thesis work, literature review is important. Our work is all about compute the overall at auto scaling, task scheduling and finally get response time of the virtual machines. Our thesis work “Efficient Load Balancing In Cloud Environment With Modified Ant Colony Algorithm” named. To understand these first we have to understand these concepts. We discussed about these literatures.

### Cloud Computing

Our thesis named “Efficient Load Balancing In Cloud Environment With Modified Ant Colony Algorithm” is a work of developing a algorithm for balancing the load in cloud environment under the research area of Cloud Computing. In this section we should discuss about Cloud Computing, Load Balancing, Different types of Cloud Services, Load balancing methods.

#### Overview of Cloud Computing

Cloud computing is an emerging technology for providing effective and efficient computational services to many organizations worldwide. Services offered by the Cloud can be Software as a Service (SaaS), Platform as a Service (PaaS) or Infrastructure as a Service (IaaS) which can be offered in private, public or hybrid Clouds. Cloud computing incorporates concepts of parallel and distributed computing to provide shared resources; hardware, software and information to computers or other devices on demand. These are emerging distributed systems which follows a “pay as you use” model. The customer need not buy the software or computation platforms. With internet facility, the customer can use the computation power or software resources by paying money only for

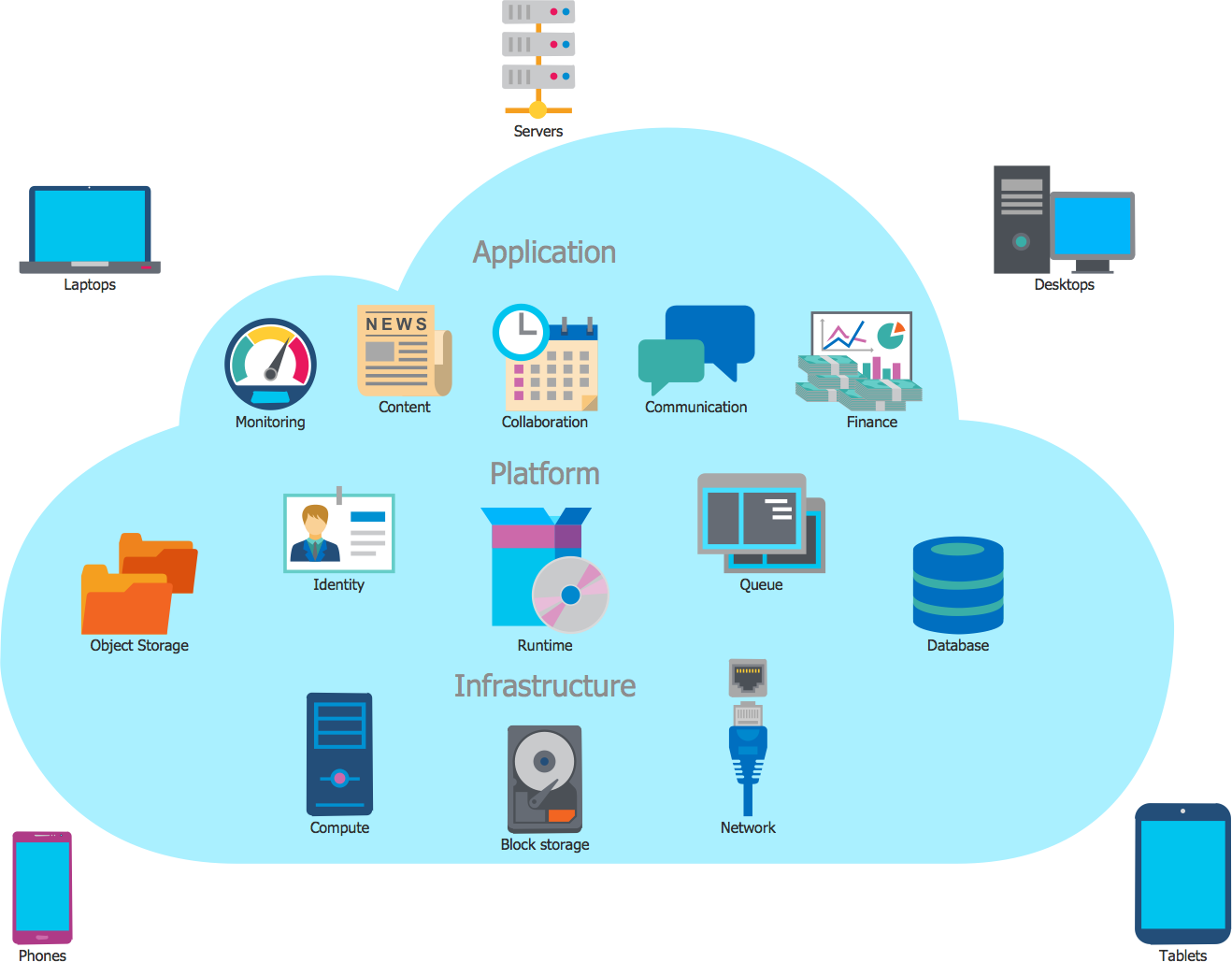


Figure 2. 1: Over View of Cloud Computing

###### Types of cloud computing deployment models

In cloud computing four types of deployment models are available-

1. Private Cloud: A private cloud is a cloud computing model which provides a cloud- based environment in which applications and resources can be access by the specified client only.
2. Public Cloud: It is a computing structure that is offered by the third party over the public internet which makes resources and application available for any user who wants to use them.
3. Community Cloud: A community cloud is an alliance effort which provides an infrastructure shared among many organizations that means the applications can be shared between them.
4. Hybrid Cloud: A hybrid clouds is a mix-up of two cloud that is a public and private cloud.

###### Types of Cloud service model

The Cloud provides three types of services models, these services modes are shown in the below figure:

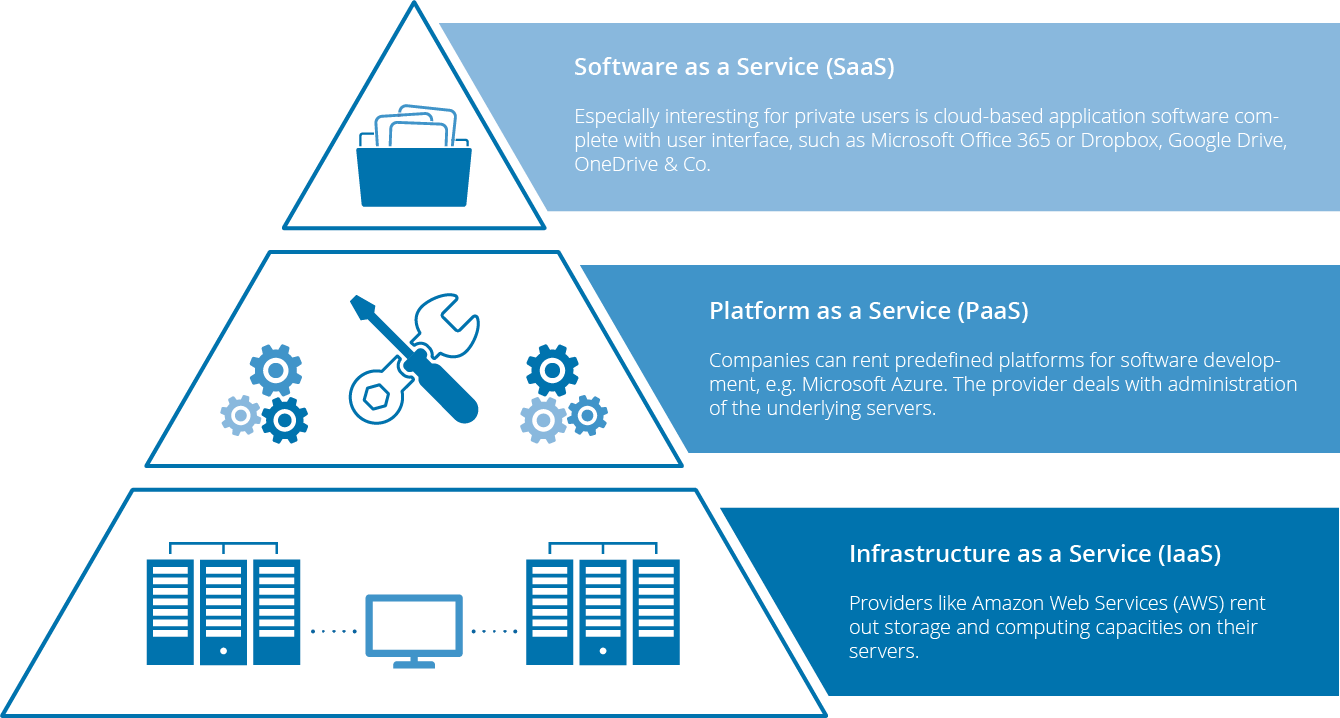


Figure 2. 2: Cloud Service Model

The models are described below- Software-as-a-Service

SaaS model provides the next features:

* access to the software from anywhere;
* payment for software usually include technical support;
* platform responsibilities managed by provider;
* sass applications are customizable and can be integrated with other business;
* anyone can pay monthly or only for the services you actually use;
* the provider performs maintenance and update of the software.

Platform-as-a-Service

PaaS model provides the next features:

* Simple. Cost-effective development and deployment of apps;
* anyone can use automated scalability of resources and do not need to allocate them manually;
* anyone can manage the services he/she develops and the provider manages all the rest.
* Significant reduction in the amount of coding needed

Infrastructure-as-a-Service

IaaS model provides the next features:

* anyone can rent all the tools (hardware and others) you need and use it how you need;
* anyone can receive administrator rights inside rented hardware;
* the MSP guarantees the availability of resources allocated to you;
* the provider manages all hardware issues.

###### Benefits of cloud computing

###### High Speed

Cloud computing allows you to deploy your service quickly in fewer clicks. This faster deployment allows you to get the resources required for your system within fewer minutes.

###### Back-up and restore data

Once the data is stored in a Cloud, it is easier to get the back-up and recovery of that, which is otherwise very time taking process on-premise.

###### Mobility

Employees who are working on the premises or at the remote locations can easily access all the cloud services. All they need is an Internet connectivity.

###### Unlimited storage capacity

The cloud offers almost limitless storage capacity. At any time you can quickly expand your storage capacity with very nominal monthly fees.

###### Collaboration

The cloud computing platform helps employees who are located in different geographies to collaborate in a highly convenient and secure manner.

**Cost Savings**

Cost saving is the biggest benefit of cloud computing. It helps you to save substantial capital cost as it does not need any physical hardware investments. Also, you do not need trained personnel to maintain the hardware. The buying and managing of equipment is done by the cloud service provider.

###### Automatic Software Integration

In the cloud, software integration is something that occurs automatically. Therefore, you don't need to take additional efforts to customize and integrate your applications as per your preferences.

###### Reliability

Reliability is one of the biggest pluses of cloud computing. You can always get instantly updated about the changes.

###### Strategic edge

Cloud computing offers a competitive edge over your competitors. It helps you to access the latest and applications any time without spending your time and money on installations.

###### Quick Deployment

Last but not least, cloud computing gives you the advantage of rapid deployment. So, when you decide to use the cloud, your entire system can be fully functional in very few minutes. Although, the amount of time taken depends on what kind of technologies are used in your business

###### Open research issues in Cloud Computing

Cloud computing is the fastest growing technology. That is why so many researchers are doing research on this filed. Open Research issues in cloud computing are listed below:

* + **Data Security**: Security is the greatest challenge or issue of cloud computing according to International Data Corporation (IDC). When we save our data or run our software into others hard disk using others CPU appears to be very risky. Organization’s data and software face serious risk of security issues like data loss, phishing, botnet etc.
  + **Load Balancing:** The second major open research issues in cloud computing is load balancing. In this issue many research is work for maintaining a load in virtual machine. Load balancing means distributed the upcoming request among the multiple servers to maintain the load.
  + **Disaster Recovery:** Disaster is a suddenly accident for short time but its caused large amount of damage in society. Disaster Recovery issues are data centers down during disaster, Data Backup, Cost, Failure detection and security**.**

**Availability of Service:** Availability refers to the ability of a user to access information or resources in a specified location and in the correct format. Simply availability mean services are available anytime and anywhere. The issues are protecting the confidentiality and integrity of data, Security and identify single point of failure.

### Cloud Load Balancing

In this section we discussed about Cloud Load Balancing, Types of Load Balancing process, Goals of Load Balancing and Different types of load balancing algorithms.

#### Overview of Load Balancing

Load Balancing is a computer networking method to distribute workload across multiple computers or a computer cluster, network links, central processing units, disk drives, or other resources, to achieve optimal resource utilization, maximize throughput, minimize response time, and avoid overload. Thus, load need to be distributed over the resources in cloud-based architecture, so that each resource does approximately the equal amount of task at any point of time.

Basic need is to provide some techniques to balance requests to provide the solution of the application faster. Cloud vendors are based on automatic load balancing services, which allow clients to increase the number of CPUs or memories for their resources to scale with increased demands. This service is optional and depends on the client’s business needs.

It has helped networks and resources, to provide maximum throughput with minimal response time. Cloud platforms enable enterprises to lease computing power in the form of virtual machines. Because hundreds of thousands of virtual machines (VMs) are used, it is difficult to manually assign tasks to computing resources in clouds. So, we need an efficient algorithm for load balancing in the cloud environment.

Load Balancing is one of the major issues related to cloud computing, the load may represent a CPU capacity, memory, network load etc. It is necessary to distribute the load equally among the nodes in a network. This results in agile and efficient performance of the system. Thereby it avoids heavily loading or under loading of nodes in a network.

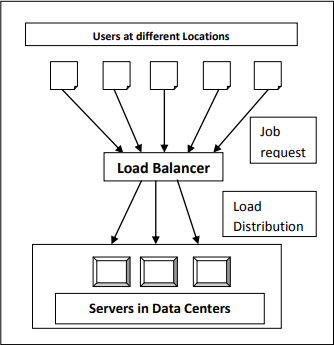


Figure 2.3. 2: Cloud Load Balancer

#### Metrics for Load Balancing in Cloud

Various metrics considered in existing load balancing techniques in cloud computing are discussed below-

* + **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.
  + **Resource Utilization** is used to check the utilization of re-sources. It should be optimized for an efficient load balancing.
  + **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.
  + **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.
  + **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.

#### Load Balancing on the basis of Cloud Environment

Cloud computing can have either static or dynamic environment based upon how developer configures the cloud demanded by the cloud provider.

**Static Environment:** In static environment the cloud provider installs homogeneous resources. Also the resources in the cloud are not flexible when environment is made static. In this scenario, the cloud requires prior knowledge of nodes capacity, processing power , memory, performance and statistics of user requirements. Round Robin algorithm provides load balancing in static environment.

**Dynamic Environment:** In dynamic environment the cloud provider installs heterogeneous resources. The resources are flexible in dynamic environment. In this scenario cloud cannot rely on the prior knowledge whereas it takes into account run-time statistics. The requirements of the users are granted flexibility (i.e. they may change at run time). Algorithm proposed to achieve load balancing in dynamic environment can easily adapt to run time changes in load. Dynamic environment is difficult to be simulated but is highly adaptable with cloud computing environment**.** LBMM (Load Balancing Min-Min) algorithm uses for resource allocation in dynamic environment.

#### Goals of Load Balancing Algorithms

The goals of load balancing are to:

* + For better performance.
  + Manages resources efficiently.
  + For achieving Stable System state.
  + For accompanying further modifications.
  + For construction of fault tolerant system
  + To have the ability to adjust itself in accordance with any modifications
  + Utilizes each the systems resources as efficiently as possible.

#### Different types of Load Balancing Algorithms

* + **Round Robin Algorithm:** In this algorithm, the processes are divided between all processors. Each process is assigned to the processor in a round robin order. The process allocation order is maintained locally independent of the allocations from

remote processors. Though the work load distributions between processors are equal but the job processing time for different processes are not same. So at any point of time some nodes may be heavily loaded and others remain idle. This algorithm is mostly used in web servers where Http requests are of similar nature and distributed equally.

* **Equally Spread Current Execution Algorithm**: Equally spread current execution algorithm process handle with priorities. it distribute the load randomly by checking the size and transfer the load to that virtual machine which is lightly loaded or handle that task easy and take less time , and give maximize throughput. It is spread spectrum technique in which the load balancer spread the load of the job in hand into multiple virtual machines.
* **Throttled Load Balancing Algorithm**: Throttled algorithm is completely based on virtual machine. In this client first requesting the load balancer to check the right virtual machine which access that load easily and perform the operations which is give by the client or user. In this algorithm the client first requests the load balancer to find a suitable Virtual Machine to perform the required operation.
* **BEE FORAGE BEHAVIOUR:** The artificial bee colony formula (ABC) algorithm supported the intelligent forage behavior of honey bee swarm and was planned by Karaboga in 2005. The formula is totally galvanized by natural forage behavior of honey bees.
* **ANT COLONY OPTIMIZATION:**nt algorithm’s is a multiagent approach to difficult combinational optimization problems. Example of this approach is travelling salesman problem (TSP) and the quadratic assignment problem (QAP). These algorithms were inspired by the observation of real ant colonies. Ant’s behavior is directed more to the survival of the colonies. They not think for individual.

#### Challenges in Cloud Computing Load Balancing

Here we discuss the challenges to be addressed when attempting to propose an optimal solution to the issue of load balancing in Cloud Computing:

**Spatial Distribution of the Cloud Nodes:** Some algorithms are designed to be efficient only for an intranet or closely located nodes where communication delays are negligible However, it is a challenge to design a load balancing algorithm that can work for spatially distributed nodes.

**Storage/ Replication:** A full replication algorithm does not take efficient storage utilization into account. This is because the same data will be stored in all replication nodes. Full replication algorithms impose higher costs since more storage is needed.

**Algorithm Complexity:** Load balancing algorithms are preferred to be less complex in terms of implementation and operations. The higher implementation complexity would lead to a more complex process which could cause some negative performance issues.

**Point of Failure:** Controlling the load balancing and collecting data about the different nodes must be designed in a way that avoids having a single point of failure in the algorithm. Some algorithms (centralized algorithms) can provide efficient and effective mechanisms for solving the load balancing in a certain pattern. However, they have the issue of one controller for the whole system. In such cases, if the controller fails, then the whole system would fail. Any Load balancing algorithm must be designed in order to overcome this challenge.

.

### Background Study

Load balancing mechanism distributes the work across multiple computing resources to utilize them effectively and at the same time eliminating a condition during which bound nodes are over loaded whereas others are beneath loaded.

Tasks that have the minimum expected time for completion are deduced and assigned to the corresponding machine.

Kokilavani et al. [12] proposed a MinMin (LBMM) algorithm primarily for load balancing which enhances resource utilization and reduces makespan. This has been done in two phases: the former phase executes the conventional Min-Min algorithm, while the later reschedules of tasks for effective resource utilization The work proposed by Lorpunmanee et al. [13] proposed the scheduling problem by introducing a model based on dynamic information on grid using ACO algorithm.

Load balancing under tri-level cloud computing was introduced in [14] by combining Load Balancing Min-Min (LBMM) and Opportunistic-Load Balancing (OLB) to maintain the load balancing of system and enhance efficiency for execution. Nowadays, it is important to consider multiple objectives simultaneously

Load balancing is a method that has helped networks and resources, to provide maximum throughput with minimal response time. Cloud platforms enable enterprises to lease computing power in the form of virtual machines. Because hundreds of thousands of virtual machines (VMs) are used, it is difficult to manually assign tasks to computing resources in clouds [15]. So, we need an efficient algorithm for task scheduling in the cloud environment.

.

# Chapter 3

**System and Architecture**

*In this chapter we introduced our thesis Methodology, and Problem Statement. In section 3.2 we discussed about problem statement, and mathematical presentation of the problem; in section 3.3 we discussed about overview of our proposed work 3.4. we should give a short discussion of this chapter.*

### Introduction

Load Balancing is the process to divide the workload among different available resource in cloud and equalize with virtual machine to achieve the performance of system. Sometimes several resources are heavy loaded, lightly loaded or idle for execution of tasks. There for task are removed from heavily loaded resources and rescheduled them to lightly loaded resources. We have proposed a modified ant colony algorithm for minimizing the overall response time.

### Problem Statement

Until recently the main works on load balancing assumed solid nodes. Many instances of Cloud computing, as outlined herein, wherever dynamic and heterogeneous systems are necessary to produce on demand resources or services. The Amazon EC2, dynamic load balancing is handled by replicating instances of the precise middleware platform for internet services. This is often achieved through a traffic analyzer that tracks the time taken to method a shopper request. New instances of the platform are started once the load will increase on the far side predefined thresholds. Therefore, combos of rules impose the circumstances and answer for load balancing. Because the systems increase in size and quality, these rule sets become unwieldy and it should not be potential to take care of a viable observation and response cycle to manage the procedure work. In short, the dimensions of those systems could exceed the capabilities of connected meta systems to take care of a sufficiently agile and with efficiency organized load balancing (or general management) rule set. Once such a lot of management rules are outlined at intervals a system, there are probably to be conflicts amongst the rules; interactions and impact are normally terribly tough to research. A load balancing system is needed that self regulates the load at intervals the Cloud’s entities while not essentially having to possess full information of the system. Such self-organized regulation could also be delivered through distributed algorithms.

### Proposed Approach

In the natural world, ants of some species (initially) wander randomly, and upon finding food return to their colony while laying down pheromone trails. If other ants find such a path, they are likely not to keep travelling at random, but instead to follow the trail, returning and reinforcing it if they eventually find food.

Over time, however, the pheromone trail starts to evaporate, thus reducing its attractive strength. The more time it takes for an ant to travel down the path and back again, the more time the pheromones have to evaporate. A short path, by comparison, gets marched over more frequently, and thus the pheromone density becomes higher on shorter paths than longer ones. Pheromone evaporation also has the advantage of avoiding the convergence to a locally optimal solution. If there were no evaporation at all, the paths chosen by the first ants would tend to be excessively attractive to the following ones. In that case, the exploration of the solution space would be constrained. The influence of pheromone evaporation in real ant systems is unclear, but it is very important in artificial systems.

The overall result is that when one ant finds a good (i.e., short) path from the colony to a food source, other ants are more likely to follow that path, and positive feedback eventually leads to

many ants following a single path. The idea of the ant colony algorithm is to mimic this behavior with "simulated ants" walking around the graph representing the problem to solve

In our proposed work consider a minimum completion time of the task in each virtual machine. In existing ant colony algorithm only consider the virtual machine which has a minimum number of high priority task. In our modified ant colony algorithm, first we count the number of tasks and then sort them according their size in ascending order and then allocate them to vm and second is minimum completion time of task on each virtual machine.

#### Flow Diagram of Ant Behavior Approach

Following figure 3.3 shows the flow diagram of the proposed algorithm. In this diagram, shows where the proposed methodology is used in cloud computing.

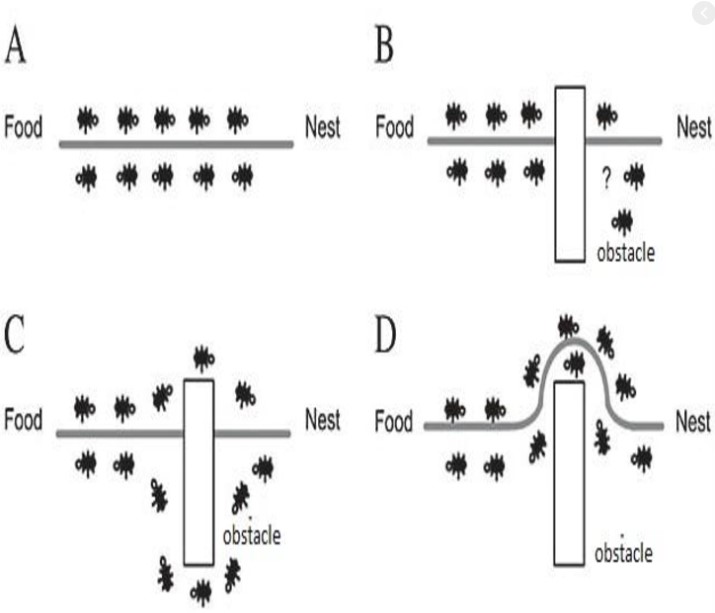


Fig: Behavior of ant colony

* + 1. **The algorithm of Ant Colony is shown as below:**

|  |
| --- |
| **Algorithm: Modified Ant colony behavior inspired load balancing algorithm** |
| Step 1: Initialize the pheromones of all VMs. |
| Step 2: Place all ants at the starting VMs randomly |
| Step 3: Every ant chooses the VM for the next task according to formula. |
| Step 4: When an ant completes its tour, update the pheromone according to formula |
| Step 5: If all the ants end their trip, continue to Step6; otherwise, repeat Step3 |
| Step 6: Nc = Nc + 1, calculate the makes pan of each ant and reserve the current optimal solution |
| Step 7: Judge if it satisfies the iterative condition Nc > Ncmax, If it satisfies, end the iteration and output the best solution, else return to Step2 until satisfy the iterative condition |

Table 3.2. 4: The Ant colony behavior algorithm

#### 3.3.3 Mathematical Representation

###### Initialize pheromone of VMj At the beginning, ants are distributed on VMs randomly, and then it will initialize the VMj pheromone value based on:

###### (0) = pe\_ × pe\_+ vm\_ ………………..(1)

###### Where pe\_numj is the number of VMj processor, pe\_mipsj is the MIPS (Million Instructions Per Second) of each processor of VMj and the parameter VM\_bwj that is related to the communication bandwidth ability of the VMj.

###### B. The rule of choosing VM for next task The k-ant chooses VMj for next task with a probability that is defined as:

###### 

##### Where

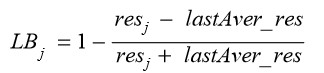
##### • τj(t) is the VMj pheromone value at time t.

##### • EV j is the computing capacity of VMj, it is defined as follows:

##### EVj = pe\_numj × pe\_mipsj + vm\_bwj (2)

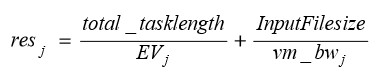
Where pe\_numj is the number of VMj processor, pe\_mipsj is the MIPS of each processor of VMj and the parameter VM\_bwj that is related to the communication bandwidth ability of the VMj.

• LBj is the load balancing factor of VMj, to minimize the degree of imbalance, which is defined as follows:



Where lastAver\_res is the average execution time of the virtual machines in the last iteration of the optimal path, and resj is the expected execution time of the task in the VMj, which is defined as follows:

Processing time of VM:



Where total\_tasklength is the total length of the tasks that have been submitted to VMj, and InputFilesize is the length of the task before execution. • α, β and γ are three parameters that control the relative weight of the pheromone trail, the computing capacity of VMs and the load balancing factor of VMs.

Once some VMs are loading heavy, it becomes a bottleneck in the cloud and it influences the makespan of a given tasks set. Therefore we define the load balancing factor LBj in the ant algorithm to improve the load balancing capability, and the bigger LBj of VMj should be chosen with high probability, that means the comprehensive ability of VMj is power now, and then it is high desirable

C Phenomenon Updating

Let τj(t) be the intensity of VMj pheromone at time t. The pheromone update is given by (7) 

Where ρ ∈ (0, 1] is the pheromone trail decay coefficient. The greater the value of ρ is, the less the impact of past solution is. The value of ∆τj is defined as follows: When an ant completes its tour, the local pheromone updating is applied on the visited VMs, and the value of ∆τj is given by



Where Tik is the shortest path length that searched by ik ant at i-th iteration. When an ant completes its tour, if it finds the current optimal solution, it can lay a larger intensity of the pheromone on its tour and the global pheromone updating is applied on the visited VMs, and the value of ∆τj is given by



Where Top is the current optimal solution, and D is the encouragement coefficient

#### 3.3.2 The pseudo code of our proposed algorithm is as below:

|  |
| --- |
| **begin**  **Initialize the number of ants, datacenter, server,**  **while (has\_task) do**  **Time= estimate\_execution\_time(task)**  **Distance\_matrix = calculate\_distance(task)**  **cost\_matrix = generate\_cost\_matrix(Time,Distance\_matrix,cost\_function)**  **Server=find\_best\_server\_using\_ant\_colony(Time,Distance\_matrix,cost\_matrix)**  **Assign\_task(task,server) If(no\_more\_task) : Wait\_for\_task()**  **Endif**  **If(machine\_is\_idle ) :**  **Server\_is\_Auto\_off();**  **Endif**  **If(task\_completed): relize\_the\_server**  **Endif**  **end while**  **end** |

Fig: pseudo code of load balancing based on ant colony

#### 

#### Summary of my Proposed pseudocode

#### In Ant colony we see that they search food from source to destination by applying updating their pheromone. They use the shortest path to find this food from source to destination like as travelling salesman problem. In my proposed algorithms I used this ants idea for calculating cost for minimum number of task, efficient execution time and auto scaling when task is idle.

#### i) At first initialize a number of tasks, server, ants and datasender.to create two types of datasets one for the server and another for the task and take input from this

#### ii) If have some tasks are free in our system then our system is working. Our system is working in multithreading.

#### iii) Calculating execution time and distance matrix ,cost matrix, depending on task. They are working two different function that is estimate\_execution\_time and calculate\_distance

#### iv) They are working until find\_best\_server\_using\_ant\_colony .it is depending on tasks, execution time, distance matrix and cost matrix

#### v) if no more tasks in this system, then wait for new task until our budget in on.

#### vi) if our budget is low or finished or machine is idle then works on auto scaling method. That’s why our machine is auto off

#### vii) Finally calculating the execution time, number of servers, number of tasks and number of virtual machines.

#### Expected Outcome from the Work

Better utilization of resources and task take minimum completion time so that the performance of system will be improved. Our algorithm finding optimal least underloaded virtual machine for increasing the system performance.

### Discussion

This chapter showed a clear description about the methodology, algorithm, we used for the various analysis of our thesis. Here also showed an overview of Analytical approach, and the Flow Diagram and pseudo code and details.