Situated Agent-Based Simulations at a Large Scale: Load-Balancing

A Project Report Submitted in the partial fulfilment of the requirements for the award of the degree of

Bachelor of Technology

in

Department of Computer Science and Engineering

by

Bhargavi.Chilakala

180030705

G.Divya Sharon

180030928

Under the Supervision of

K.Thirupathi Rao

KONERU LAKSHMAIAH EDUCATION FOUNDATION,

Green Fields, Vaddeswaram- 522502, Guntur(Dist),

Andhra Pradesh, India.

November – 2021

KONERU LAKSHMAIAH EDUCATION FOUNDATION

DEPARTMENT OFCOMPUTER SCIENCE AND ENGINEERING

DECLARATION

The project report entitled **"Load-Balancing for Large Scale Situated Agent-Based Simulations"** is a record of bonafide work of "**Bhargavi.Chilakala(180030705),G.Divya Sharon(180030928)**", **submitted** in partial fulfilment of the requirement for the award of degree in **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE ENGINEERING to the KL University** .

The results embodied in this report have not been copied from any other departments/University/Institute.

Name of the Student	Univ. Reg. No
Bhargavi.Chilakala	180030705
G.Divya Sharon	180030928

KONERU LAKSHMAIAH EDUCATION FOUNDATION DEPARTMENT OFCOMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that this term project report entitled **"Load-Balancing for Large Scale Situated Agent- Based Simulations"** is being submitted by "**Bhargavi.Chilakala (180030705), G.Divya Sharon (180030928)**" submitted in partial fulfilment for the award of degree in **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE ENGINEERING** to the **KL University** is a record of bonafide work carried out under our guidance and supervision.

The results embodied in this report have not been copied from any other departments/ University/Institute..

K.Thirupathi Rao	**Dr. V. HARI KIRAN**
Guide	**Head of the Department**

aSignature of the EXTERNAL EXAMINER

ACKNOWLEDGEMENT

Our sincere thanks to **K.Thirupathi Rao** for the outstanding support throughout the project for the successful completion of the work

We express our gratitude to **Dr.V.HARI KIRAN,** Head of the Department of Computer Science Engineering for providing us with adequate facilities, ways and means by which we are able to complete this term paper work.

We would like to place on record the deep sense of gratitude to the honorable Vice Chancellor, KLEF for providing the necessary facilities to carry the concluded term paper work.

Last but not the least, we thank all Teaching and Non-Teaching Staff of our department and especially my classmates and my friends for their support in the completion of our term paper work.

ABSTRACT

Due to the massive number of agents and interactions in large-scale agent-based simulations, memory and computational power requirements might skyrocket. Distributing the simulator over a computer network to simulate millions of agents is intriguing, but it creates several challenges such as agent allocation and load-balancing between machines. We investigate the most effective methods for automatically balancing the loads amongst machines in large-scale settings in this research. We investigate the performance of two different apps with two alternative distribution methodologies, and our results suggest that some applications can automatically change the loads between machines and achieve a high level of autonomy.

TABLE OF CONTENTS

LIST OF FIGURES:

Chapter 1:

Chapter 3:

1. INTRODUCTION

1.1 CLOUD COMPUTING

- A cloud is a distributed and parallel system made up of interconnected and virtualized computers that are dynamically provisioned and offered as a single or additional computing resource depending on service-level agreements agreed through collaboration between service providers and clients.

- In the most basic terms, cloud computing is when a company employs someone else's computing services (typically over the internet) instead of having to run software on their own systems.

- For example, in our very tiny business, we don't have enough money to hire an IT department and people to manage our core services. We do, however, want to customise a wide range of applications, such as:
  - ✓ Accounting
  - ✓ file storage
  - ✓ shared contacts and calendars
  - ✓ other similar services etc.

  As a result, we use Google Calendar and Dropbox, among other cloud computing applications.

- Because cloud computing allows computing resources to be reduced, instead of 100 corporations each running one server to host some shared calendars, these 100 corporations could use cloud computing services provided by a cloud computing provider, who would only run one server to assist each corporation's calendars. Each of the 100 organisations now pays a portion of the cost of having shared calendars, and the cloud computing provider only needs to run one server to serve them all, resulting in data collection.

1.1.1 Essential Characteristics:

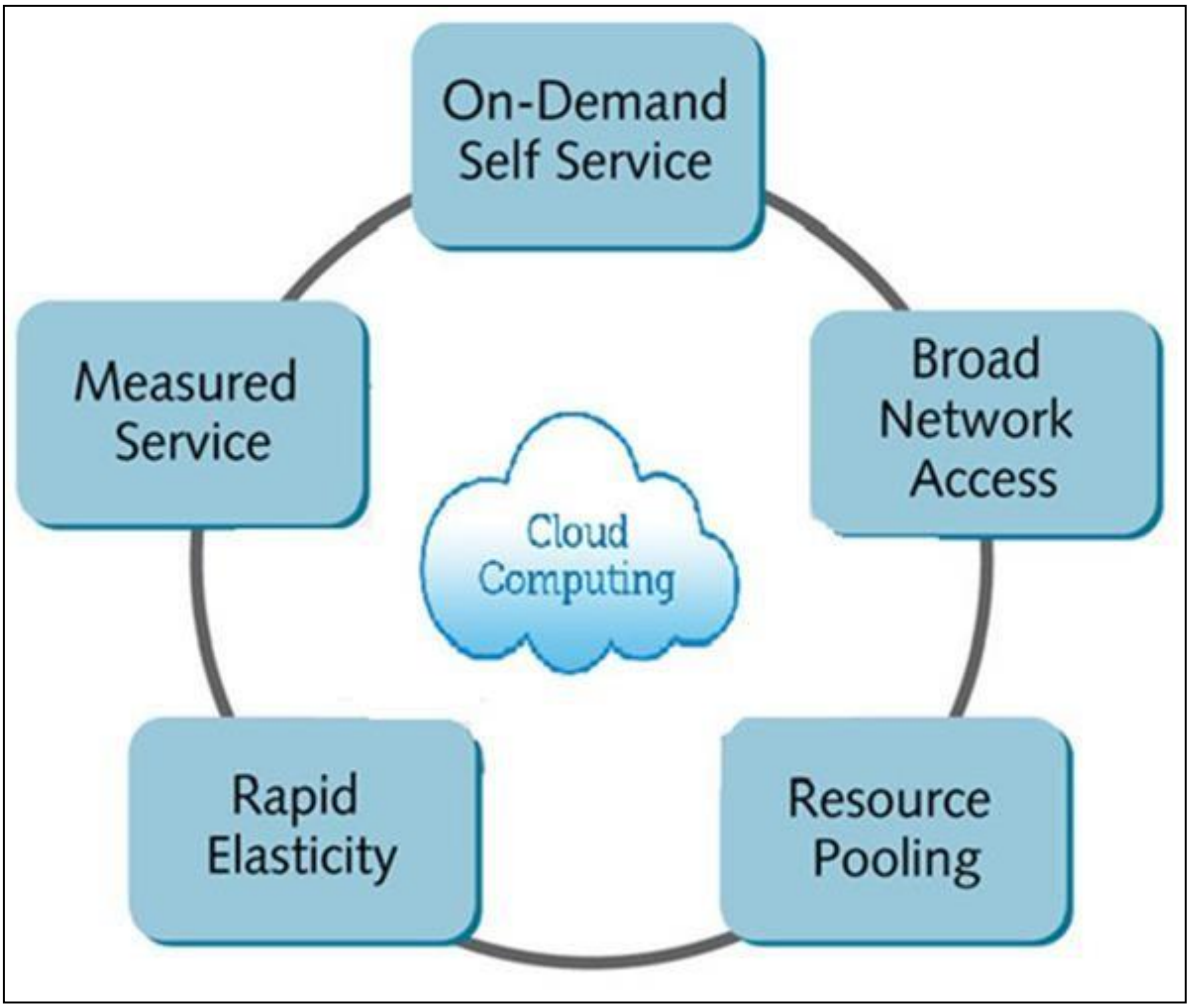


Fig-1.1: Essential Characteristics of Cloud Computing

- **On-demand self-service**: Without requiring human interaction with each specialist organisation, a customer can exclusively configure processing capacities, such as server time and system stockpiling, as desired.
- **Broad network access** : Capabilities can be found on the internet and recovered using common tools that permit the use of blended thin and thick purchaser stages (e.g., tablets, cell phones, workstations and PCs).
- **Rapid elasticity**: Capabilities can be freely provisioned and unlimited, which makes it possible to quickly quantify outside and interior demand. The capacities given for provisioning frequently convey the impression to the client that they are limitless and can be used in any amount at any time.
- **Resource pooling:** Using a multi-occupant show, the supplier's processing assets are linked to help several clients, with diverse physical and virtual assets successively apportioned and reallocated by client request. There is a sense of space freedom in that the client, for the most part, has no control or data over the precise location of the given assets, but may be able to determine position at a higher level of reasoning (e.g., state, nation or datacenter). Capacity, handling, memory, and system transmission capacity are examples of assets.

- **Measured service:** As a result, cloud frameworks use a metering capacity at some level of deliberation appropriate to the type of administration to manage and improve asset consumption (e.g., preparing, capacity, dynamic client records and transmission capacity).

1.1.2 Cloud Computing Architecture:

- Cloud Architecture refers to the various components, such as databases, software capabilities, and applications, that are designed to control the power of cloud resources in order to address business problems. The components of the cloud, as well as the relationships between them, are described by cloud architecture. Typically, these components include
- Front End
- Back End

They communicate with one another across a network, most commonly the Internet. There would be no cloud if it weren't for the network. The Internet, as well as other private or public networks, connect clients with information and information platforms with one another. Clients should have continuous access to information and applications, as well as agile movement between servers and other clouds and true security protocols, thanks to this cloud network layer. The front end is the part of the application that the computer operator or customer sees. The "cloud" unit of the structure is the back end.

Front End

The customer's machine (or computer network) and the application required to access the cloud computing system make up the front end.

Existing Web browsers, such as Internet Explorer or Firefox, are influenced by features such as Web-based e-mail programmes. Customers can access the network through single applications on other platforms. To deal with specific operator or customer issues, Each cloud computing system has its own interface. It can be a desktop or a mobile device at times, and it can also be operated as a thin client or a computing device.

Back End

The back-end data is sent to the cloud. It has all of the resources required to provide cloud computing services. The multiple computers, servers, and data storage devices that make up the "cloud" of computing services are located towards the back end of the structure. A cloud computing system, in theory, might include virtually any computer database you can imagine, ranging from data processing to video games. In most cases, each programme will have its own dedicated server.

The back-end platform component of cloud architecture consists of a large number of computers, data storage devices, and software that may be accessed securely across a community network or through isolated connections. The size and complexity of today's major platforms, as well as their custom development, is staggering.

To put the magnitude of the back-end infrastructure in perspective, Amazon AWS and Microsoft combined have about 4 million servers in hundreds of cloud data centres around the world, while Google's cloud donation most likely has at least 2 million servers. A single AWS data centre will typically house 50,000–80,000 servers and span the equivalent of ten football fields.

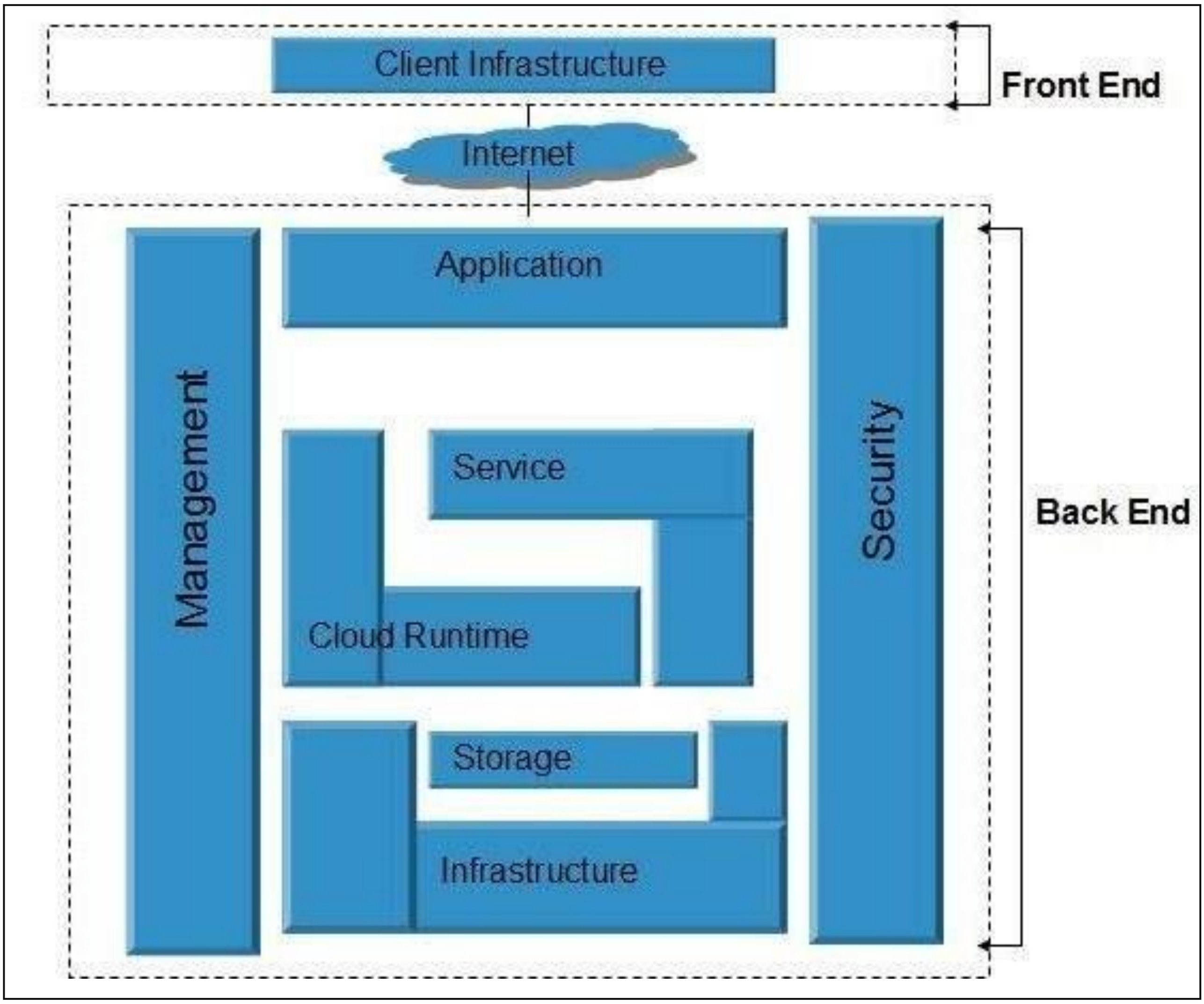


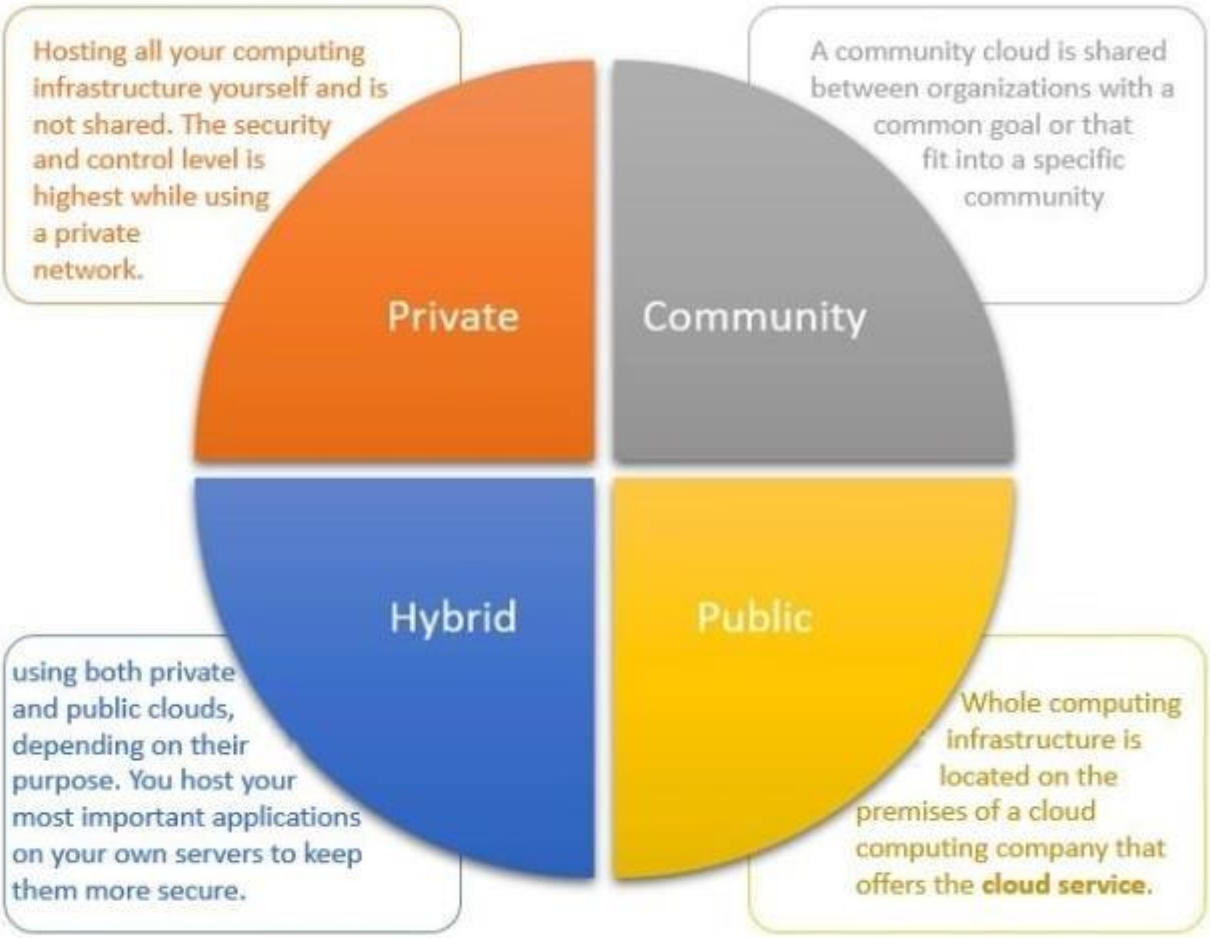
Fig-1.2:Cloud Computing Architecture

1.1.3 TYPES OF CLOUD

Based on a deployment model, we can classify cloud as:

- public,
- private,
- hybrid,
- community.

Fig-1.3: Types of Cloud



Public Cloud:

Google, Amazon, Microsoft, and others provide immediate access to open cloud setups. People in general get framework and administrations through open cloud administrations, and you, or your organisation, get a piece of that foundation and system.

Hundreds or thousands of people share the same assets. Gmail and the University of Illinois Box are examples of open cloud administrations. In April 2017, Google revealed that it reached 1 billion monthly clients. While your email account is protected by a password, the infrastructure on which it is stored is shared by over 1 billion people..

When we talk about open cloud, we mean that the complete registering infrastructure is housed on the premises of a distributed computing company that provides cloud services.

As a result, the client's space remains distinct from him, and he has no physical control over the structure. Because open mists rely on shared assets, they often outperform expectations in terms of execution, but they are defenceless to a variety of attacks.

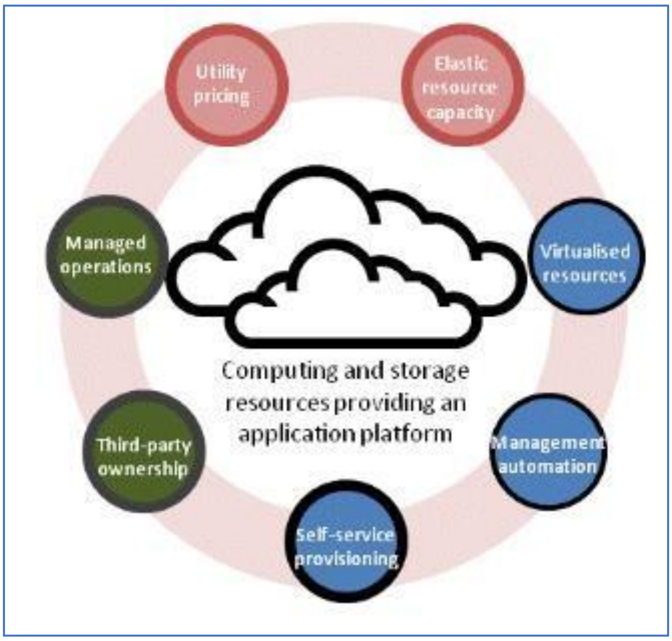


Fig-1.4: Public Cloud

PRIVATE CLOUD:

Private cloud arrangements are devoted to one association or business, and regularly have substantially more particular security controls than completes an open cloud. Numerous restorative workplaces, keeping money foundations and different associations who are required to meet government and state rules for information controls utilize a private cloud.

Utilizing private distributed storage enables them to control very delicate information by meeting directions and industry-based criteria, regardless of whether that be therapeutic records, exchange mysteries, or other ordered data. Private cloud arrangements use framework that is either possessed and controlled by the association, or they can legally require those particular criteria be met by a merchant who deals with the foundation.

Private Cloud gives comparable points of interest of Public Cloud, yet uses submitted, private gear. Private cloud suggests using a cloud establishment (orchestrate) only by one customer/affiliation. It isn't bestowed to other individuals, yet it is remotely found. The associations have a decision of picking an on-present private cloud additionally, which is all the more exorbitant, yet they do have a physical control over the system. The security and control level is most important while using a private framework. Be that as it may, the cost diminishing can be immaterial, if the association needs to place assets into an on-start cloud structure.

With our Private Cloud you'll get:

- Increased abundance
- Decreased provisioning time for new servers
- Saved capital by taking out hardware reinforce contracts

- Quicker pointlessness stood out from encouraging your own one of a kind physical servers
- Use of given, private gear.

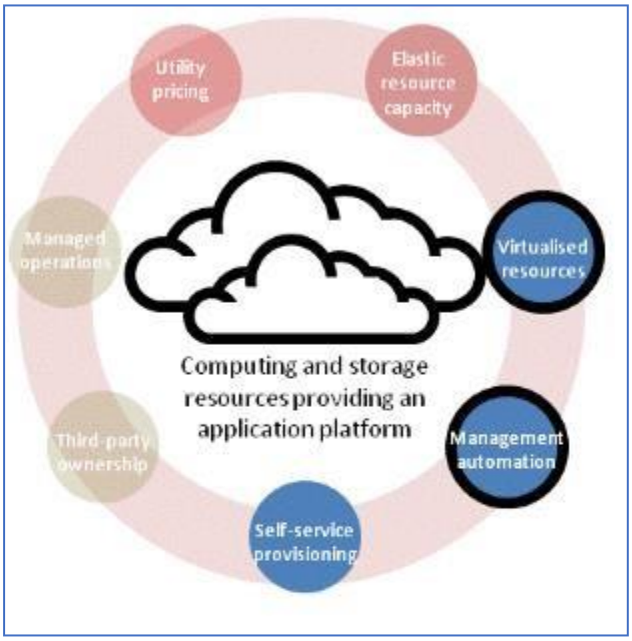


Fig-1.5:Private Cloud

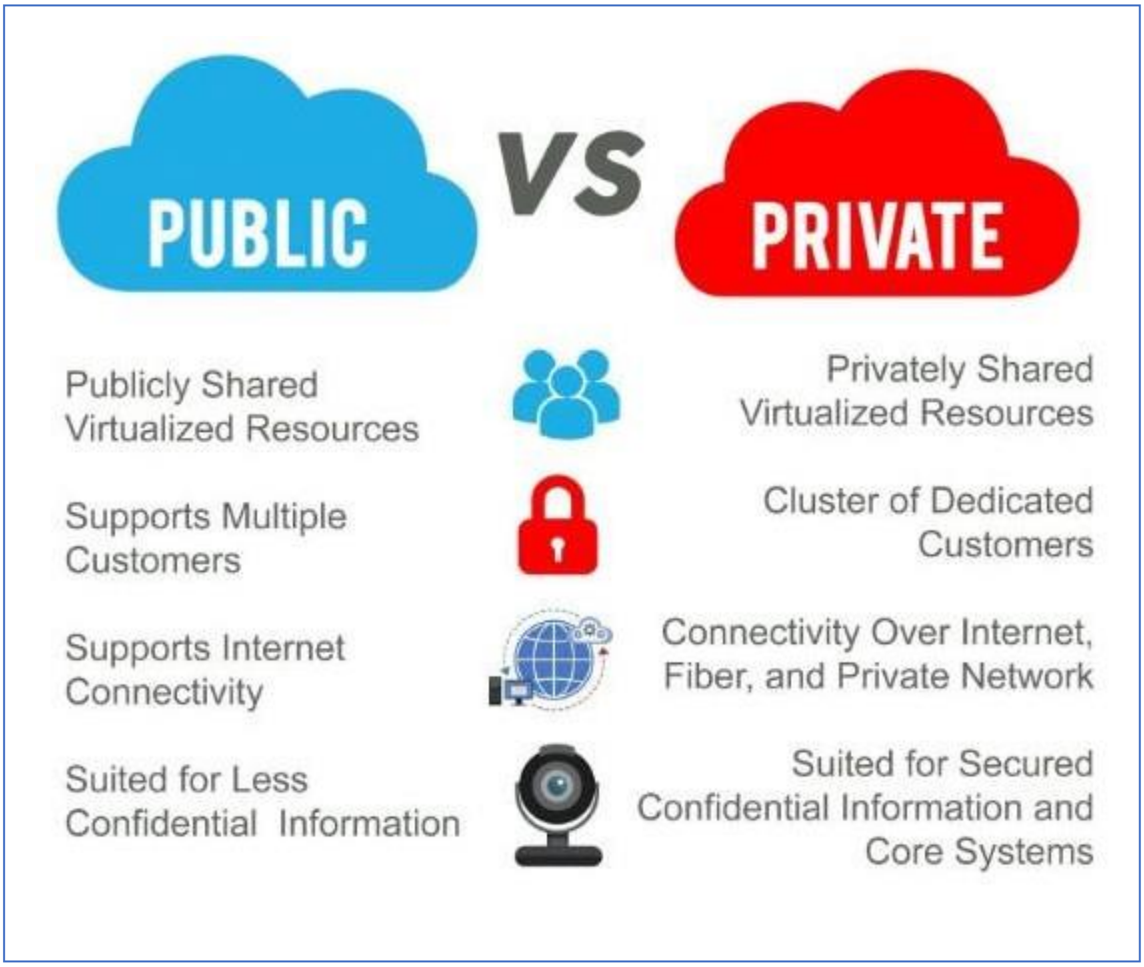


Fig-1.6: Public cloud VS Private cloud

HYBRID CLOUD:

Using both private and open mists, depending on their goal, is what cross breed cloud means. For example, open cloud can be used to communicate with clients while keeping their data safe in a private cloud.

Open and private mists are mixed in half and half cloud arrangements. This is a more complicated cloud setup because the company must deal with multiple stages and figure out where information is stored. An organisation that needs to maintain sensitive data on their private cloud but create more broad, client-facing material on an open cloud is an example of a hybrid cloud configuration.

The great majority associate traditional open cloud advantages with agility and the ability to deal with frequent changes. Regardless, execution challenges may arise for specific information-intensive or high-accessibility outstanding jobs.

COMMUNITY CLOUD:

The term "network cloud" refers to a shared foundation that is used by multiple organisations to communicate information and manage data. For example, a network cloud and a single nation's legislature may coexist. Network mists can be found both on and off the premises. A people group cloud is shared by organisations that share a same aim or are members of a network (proficient network, geographic network, and so forth).

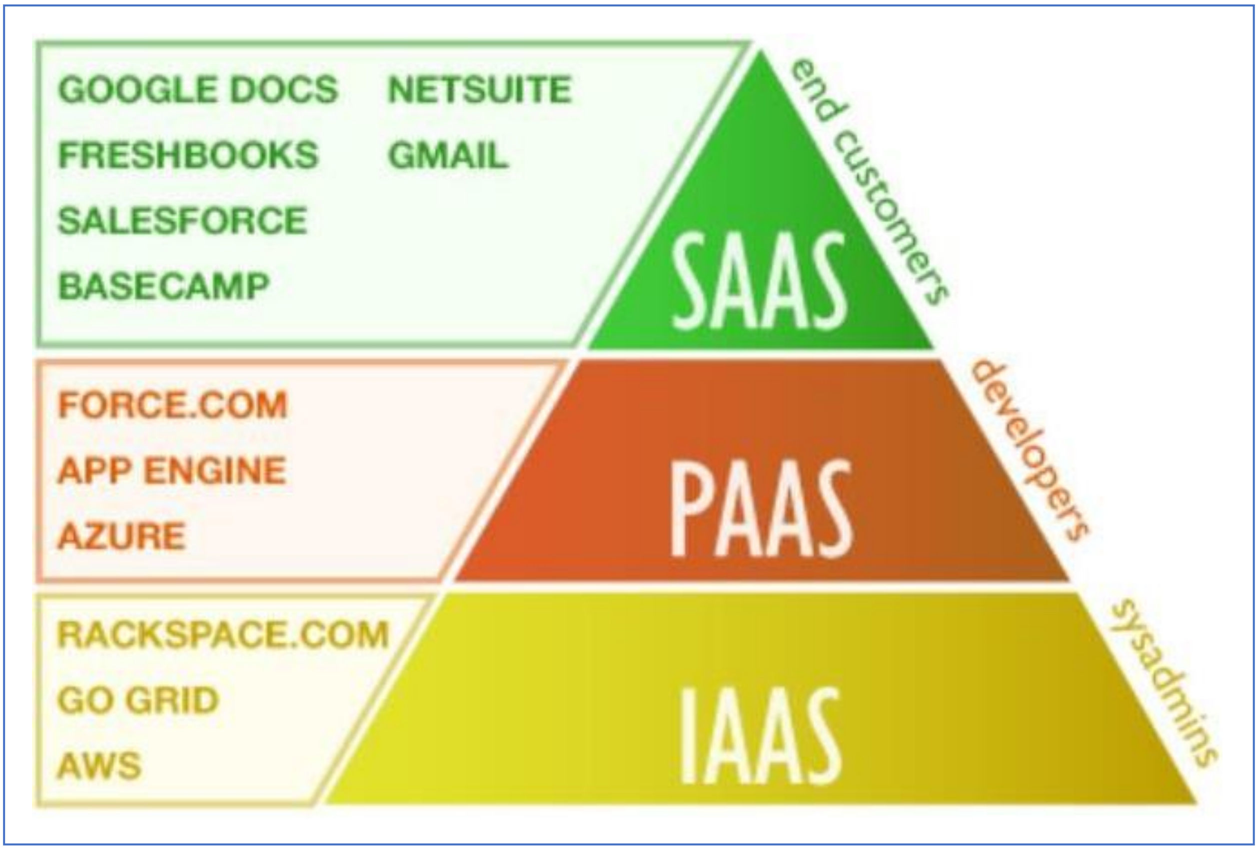
1.1.4 TYPES OF SERVICES:

- In light of the administration put out by the cloud show, we're debating either:
- IaaS (Infrastructure-as-a-Service)
- PaaS (Platform-as-a-Service)
- SaaS (Software-as-a-Service)

The benefit that supplies data storage plates and virtual servers, i.e. the foundation, is the most well-known cloud benefit. Infrastructure-as-a-Service (IaaS) companies include Amazon, Rackspace, and Flexiscale.

When the cloud provides an improvement stage, which contains a working framework, programming dialect execution condition, database, and web server, it is known as Platform-as-a-Service (PaaS). Google App Engine, Microsoft Azure, and Salesforce are all examples of PaaS. PaaS allows working frameworks to be updated and generated on a regular basis, administrations to be obtained from a variety of sources, and programming to be done in groups

Finally, programming as-a-Service (SaaS) denotes that clients can pay per-use access to a variety of programming applications. It is typically far more expensive to purchase allowed projects than it is to purchase unpermitted projects. Similar services include GMail, which is widely used, and Google Docs.


GOOGLE DOCS
NETSUITE
FRESHBOOKS
GMAIL
SALESFORCE
BASECAMP
SAAS
end customers
FORCE.COM
APP ENGINE
AZURE
PAAS
developers
RACKSPACE.COM
GO GRID
AWS
IAAS
sysadmins

Software as a service (SaaS):

Fig-1.7: Types of Services

SaaS is sometimes known as "on-demand programming." Instead of obtaining a license for a certain piece of programming in its whole, SaaS users pay for it on a monthly basis and access it via an Internet program. SaaS provides a wide range of programming options. Businesses' SaaS programming, such as ezTalks Video Conferencing, is utilized in content management, HR management, bookkeeping, ERP, customer relationship management, and other important areas.

The product-as-a-benefit form of distributed computing is rapidly expanding, although the main market for programming as-a-benefit power is client connections.

The cloud-based applications are supplied to the client as an administration on demand in this administration demonstration. It is a singular instance of administration running on remote PCs "in the cloud," owned and operated by others, and connected to clients' PCs through the Internet and, most typically, an internet browser. Long-distance social networking sites like Facebook, Twitter, Flickr, and Google are all examples of SaaS, despite the fact that clients can access the services from any Internet-enabled device.

Benefits of SaaS Solutions

- Rapid Scalability
- Accessibility from any area with Internet
- Eliminates framework concerns
- Custom dimensions of administration contributions
- Bundled upkeep and Support

Platform as a service (PaaS):

Clients of distributed computing can use PCs as stages in PaaS. It also serves as a link between the SaaS and IaaS administrations provided by distributed computing.

A typical PC requires a PC stage that includes both equipment engineering and a programming structure. This is the starting point for a coordinated effort of equipment and programming that includes the working arrangement of the working framework, design, and dialects. Without an actual PC, distributed computing is stage as an administration, which provides customers with everything they need to set up a processing stage.

Facilitation, utilisation, test and application development, and application planning are just a few of the services provided by PaaS. PaaS-enabled cloud specialist co-ops also include a set of features for configuring, testing, implementing, managing, and running applications. Individuals and businesses alike can benefit from all of these services by purchasing a single package that includes all of the offered features.

The stage-as-a-service (PaaS) demonstration is a step up from the product-as-a-service (PaaS) setup, and it gives clients the equipment, organisation, and framework they need to build their own application and programming. A preset mix of working framework OS and application servers from PaaS vendors is used to suit the requirements of users, such as adaptability and sensitivity. Web designers, for example, can use individual PaaS conditions at any point during the process of creating, testing, and finally hosting their destinations.

Benefits of PaaS Solutions

- **Community:** In most PaaS circumstances, a large number of people are working on developing cloud apps. This creates a strong network that will allow you to proceed to the next group.
- **No more redesigning**: The foundation programming does not have to be updated or overhauled. Rather, the PaaS provider is responsible for all overhauls, patches, and routine programming maintenance.
- **Lower cost**: Companies face a lower risk because they don't have to invest upfront in equipment and programming.
- **Simplified sending** – The development team may concentrate on constructing the cloud application rather than worrying about the testing and configuration framework.

Infrastructure as a Service (IaaS):

IaaS (Infrastructure as a Service) is the third type of cloud processing. Various programming and equipment gadgets deliver on a cloud stage, similar to PaaS and SaaS, whereas IaaS provides a virtualized stage. Clients are only shown the dynamic registering stage of the equipment virtualization process, not the physical capacities. As a result of this virtualization, a so-called "virtual machine screen" or "hypervisor" is created, becoming the third element of distributed computing, I. Interface as an administration.

Interface as a Service clients get access to a framework that includes programming, servers, arrange gadgets, and server farm space. The cost incurred by clients' cloud specialist companies is frequently determined by the amount of movement performed by the client and is thus not fixed. In any event, there is a little fee based on the number of offices a client has purchased.

IaaS (Infrastructure as a Service) is a critical registering and storing room provided by a system's institutionalised management. This strategy has simplified the difficult task at hand by combining and allocating server farm space, accumulating frameworks, organising gadgets, servers, and so on. Furthermore, the client has the option of developing and introducing their own working frameworks, programming, and applications.

There are numerous advantages to employing cloud computing technology. Nonetheless, each cloud benefit has its own set of advantages and disadvantages. It is critical that clients carefully weigh the benefits and drawbacks in order to make the best decision possible.

Benefits of IaaS Solutions

- Reduces add up to cost of possession and capital uses
- Users pay for the administration that they need, in a hurry
- Access to big business review IT assets and framework
- Users can scale all over dependent on their necessities whenever

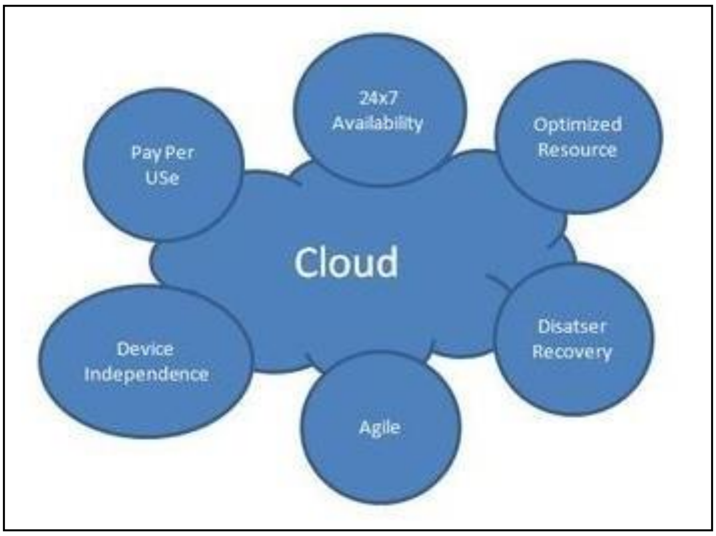
Advantages of Cloud Services

- It just takes you a couple of minutes or hours to set up a cloud benefit application with powerful highlights, which may just cost you a couple of dollars for each seat every month. You can get to the cloud benefit from any processing gadget appended to the web including cell phones, tablets and workstations. In the event that you need to get to an administration, you can do it from anyplace; home, at the air terminal, at the workplace, and so on.
- Cloud administrations are additionally adaptable. As your organization develops, you can expand your membership to the assets you have to run in accordance with your development. The equivalent is the situation when your organization's tasks shrivel. Most cloud suppliers have bundles that enable clients to pick the dimension of administration they require.
- Finally, organizations that receive cloud benefits ordinarily advantage from enhanced effectiveness and lower costs. This gives them an edge over their rivals and can push them to rapidly pick up piece of the pie. For new businesses, cloud administrations can assist them with becoming gainfully rapidly without gambling interests in strong in-house IT foundation, equipment or programming.

Organizations should look into well to comprehend their correct needs and discover suppliers who offer the arrangements that will work for them.

1.1.5 Uses of cloud computing:

Although you do not realize you are probably using cloud computing right now, most of us use an online service to send email, edit documents, watch movies, etc. It is likely that cloud computing is making it all possible behind the scenes.



1. You can ascertain, control, orchestrate, oversee, store information and data by utilizing any gadget generally PC/tablet and versatile. Case of distributed computing innovation: distributed storage, for example, Google Drive, Onedrive, Dropbox and so forth.

2. You can get to the pre-introduced application and framework programming from anyplace on any gadget. It implies cloud facilitating, remotely facilitated server, so you can have your site and the world can get to it from any gadget.

3. You can set up your very own cloud framework and give the entrance to different clients. It implies your PC or authority PCs are open for your clients by HTTPs: or FTPs.

4. Cloud figuring is a propelled utilization of the Internet.

5. Cloud Computing is another plan of action for organizations.

6. Cloud figuring is another approach to educate PCs.

7. Cloud Computing is less tedious than customary employments of PC and web for business.

8. Cloud figuring is expandable according to request.

9. Cloud registering is exorbitant for everyday citizens or people yet less expensive for independent company and enormous organizations.

10. Automatic upkeep and updates.

11. Less utilization of human forces.

12. Cloud figuring innovation is open by SAAS, PAAS, IAAS plans of action.

13. Cloud Follows Pay for What you Use Model

14. 24*7 Availability and Performance Oriented Infrastructure

15. Cloud Technology puts stock in Optimization of Existing Resources

16. Disaster Recovery Plan Help to Restore Data Quickly on Cloud

17. Agile Development on Cloud

18. Organizations Experience Device Independence

19. Cloud Architecture is Exceptionally Flexible

20. SRC (Security, Risk Management and Compliance Management)

21. Downtime and Load Balancing Problems Diminish to a Large Extent

Today an assortment of associations extending from small new companies to government offices is grasping this innovation for the accompanying:

1. Create new applications and administrations and additionally store, back up and recuperate information

2. Host sites and websites

3. Stream sound and video

4. Deliver on interest programming administrations

5. Analyse information for examples

6. Make expectations

1.2 LOAD BALANCING

Load balancing is the act of dynamically and equitably distributing workload across all available nodes in the cloud. By distributing workloads over multiple nodes, the total system performance improves. When resources are not effectively exploited, they might get overheated, resulting in carbon emissions. Carbon emissions can be reduced by appropriately utilising resources.

Some measuring factors that can be used to evaluate load balancing techniques include throughput, performance, scalability, reaction time, resource utilisation, and fault tolerance. These factors allow us to determine whether the given load balancing approach or algorithm is capable of balancing the load.

Every virtual machine in the cloud system may process the same amount of work thanks to good load balancing. As a result, load balancing will be required to increase throughput while reducing response time. It also reduces energy usage, resulting in a cleaner, greener atmosphere. Load balancing reduces energy usage and, as a result, reduces carbon emissions.

Load balancing is the process of evenly distributing resources across users or requests so that no node is overwhelmed or idle. Load balancing is an important part of cloud computing, as it is in all other internet-based distributed computing jobs. In the absence of load balancing, the efficiency of some overloaded nodes can rapidly deteriorate, resulting in SLA violations.

This helps in achieving Green computing. Efficient Load balancing will ensure:

• Uniform distribution of load on nodes.

• Improves overall performance of the system

• Higher user satisfaction

• Faster Response

• System stability

• Reducing carbon emission

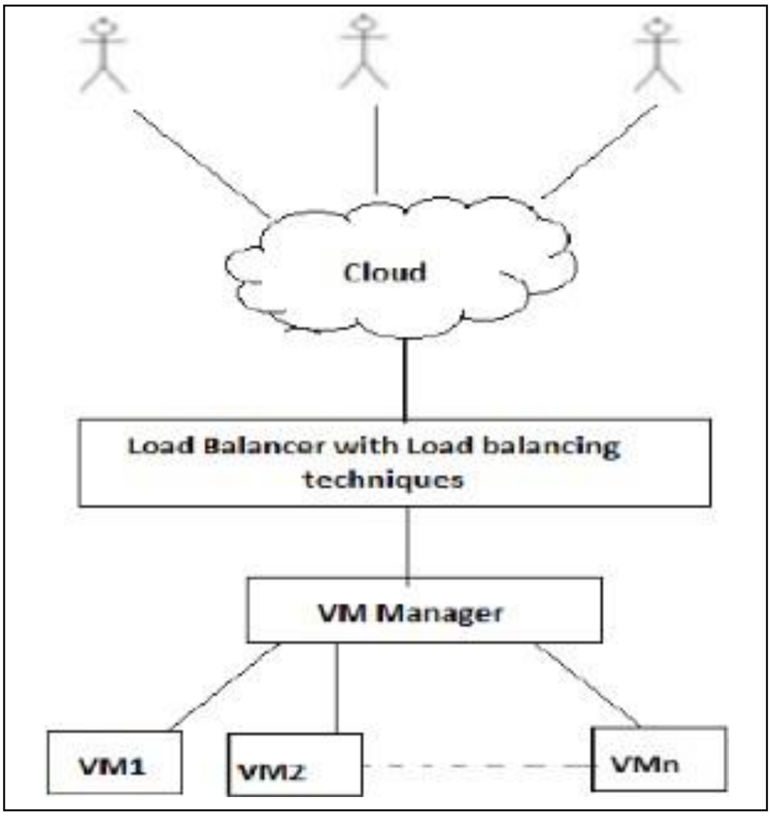


Fig-1.8: General Structure of Load balancing in Cloud Environment

1.2.1 Load Balancing Measurement Parameter

There are some estimation parameters to assess the heap adjusting systems which enable us to check whether the given method is sufficient to adjust the heap or not.

- Throughput: It is the measure of work to be done in the given measure of time.
- Response time: It is the measure of time used to begin satisfying the interest of the client in the wake of enlisting the demand. Calculation ought to limit sit tight time of an assignment for allotment of assets to it.
- Fault resistance: It is the capacity of the heap adjusting calculation that enables framework to work in some disappointment state of the framework. The calculation must guarantee expanded throughput at insignificant cost. On the off chance that a heap adjusting calculation doesn't expand framework throughput, it overcomes its very own motivation. The calculation must guarantee adaptation to internal failure, so that if there should arise an occurrence of an issue in the framework finish stack adjusting component does not quit working.
- Scalability: It is simply the capacity of the calculation to scale itself as per required conditions.
- Performance: It is the general check of the calculations working by thinking about precision, cost and speed.
- Resource usage: It is utilized to keep a beware of the use of different assets.
- Reliability: The calculation must be dependable, since process disappointment while exchanging work from one area to other may prompt expanded holding up time and client disappointment.
- Adaptability: Algorithm must be equipped for adjusting the progressively changing client asks for and give errand allotment in insignificant measure of time.

1.2.2 Classification of Load Balancing Algorithms

- Load balancing algorithms come in a variety of forms. Load balancing algorithms are often divided into two groups based on the current condition of the system:

- **Static Algorithm:** Static Algorithms are excellent for small distributed environments with fast Internet speed and negligible communication latency and are good for homogeneous and stable environments.

- **Dynamic Algorithm:** Dynamic Algorithms work well in a heterogeneous context because they focus on lowering communication latency and execution time, making them ideal for large dispersed systems.

Cloud computing obviously belongs in the second group, based on the aforementioned classification. It indicates that in a cloud computing context, balancing load necessitates an emphasis on dynamic load balancing techniques. Process migration in traditional dispersed systems is less expensive because to modest process granularity, whereas process migration in CC environments is costly due to the high granularity of data involved. As a result, in a cloud computing environment, a load balancing algorithm that can adapt to dynamic service demands while offering optimum load balancing is required.

STATICALGORITHMS:

These Algorithms are good for a stable and homogeneous environment. Static algorithms cannot dynamically change the attributes and these are not flexible. Static load balancing algorithms will not check the functionality and state of the previous tasks while assigning tasks to the nodes.

Examples of Static Algorithms are:

a) Round Robin Load Balancing Algorithm (RR)

b) Load Balancing Min-Min Algorithm (LB Min-Min)

c) Load Balancing Min-Max Algorithm (LB Min-Max)

DYNAMIC ALGORITHMS:

Dynamic algorithms will provide better results in dynamic and heterogeneous environments. Dynamic changes to the attributes can be considered by Dynamic algorithms. These algorithms are more complex and flexible. The performance of the system will be improved by selecting the task based on the current state.

Dynamic algorithms can be implemented In two forms. They are:

1. **Distributed System**

   Here, nodes in the system will execute the load balancing algorithm. All the nodes will interact with each other. The task of balancing the loads is distributed among all the nodes. All the nodes can interact among themselves either cooperative or non-cooperative way. It will not stop the functionality.If any node fails in the system.

   i) All node works together in a cooperative distributed system.

   ii) Each node works independently in the non-cooperative distributed system.

2. **Non-distributed System**

   Non-distributed System can be semi-distributed or centralized.

- The central node is responsible for load balancing of the whole system in a Centralized system . All the other nodes interact with this central node. It will stop the functionality If the central node fails. The recovery will not be easy, in case of failure.
- In a semi-distributed system, nodes are grouped to form a cluster. A central node of each cluster will perform the load balancing of the whole system. If a central node of the cluster fails, it will stop the functionality of that cluster only. The load balancing is managed by multiple central nodes.

1.2.3 HIERARCHICAL LOAD BALANCING ALGORITHM

Different levels of load balancing decisions are involved in hierarchical load balancing. Every node's parent Node is in charge of managing or balancing it. Load balancing is the responsibility of the parent node. In both homogeneous and heterogeneous environments, hierarchical load balancing can be applied. Cluster can also be utilised for load balancing in a hierarchical fashion. Clustering is the process of grouping objects that are similar in nature. VMs with comparable properties are grouped together logically. The last level of virtual machines has been reached.

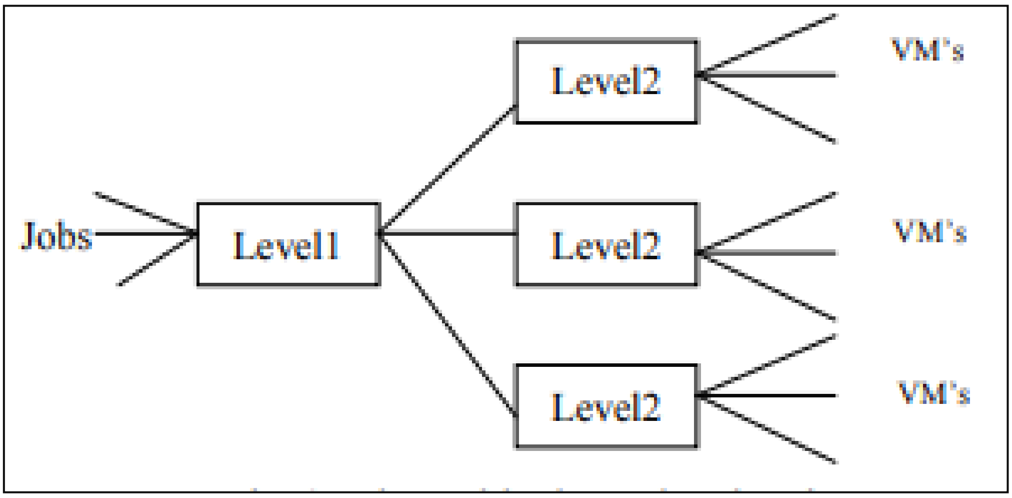


Fig-1.11: Hierarchical Load Balancing

2. LITERATURE SURVEY:

2.1.1 Name of the author:

Ka-Po Chow and Yu-Kwong Kwok

2.1.2 Title of the paper:

On Load Balancing for Distributed Multiagent Computing

2.1.3 Inference:

This paper gives us a concise prologue to Multi-Agent Simulations. Here, the product specialists resemble "merchants" and the communications among operators can be contrasted with the trading of "administrations," simply like a network of individuals working with one another and are generally imagined to have the capacity to perform numerous business exercises for the benefit of people.

A credit-Based Load adjusting model is talked about consequently. It fundamentally manages Selection strategy (which specialist should be moved to different machines) and Location arrangement (which machine the operator should be moved to). In this model every operator has a numerical esteem called credit. The credit shows the inclination of the operator to stay undisturbed on the off chance that relocation is under thought. To a specialist, the higher its credit, the higher its opportunity to remain at a similar machine, which is equal to stating that its opportunity to be chosen for relocation is lower.

A comet calculation is utilized to choose which specialist should be relocated and which machine to pick as a goal for it. In the event that a specialist has a higher calculation remaining task at hand, it will have a higher possibility of being chosen for relocation. After the determination arrangement has figured out which specialist is to be relocated, the area strategy will be in actuality. The area approach initially recognizes which remote operator will play out the most correspondence with the specialist to be relocated. The machine at which this remote operator lives is chosen as the goal machine.

Comet Algorithm reliably exhibits a more grounded ability to decrease the remaining task at hand variety over every one of the hosts in a group. Results demonstrate that for a similar number of hosts, the execution of Comet will be better for a bigger number of specialists.

2.2.1 Name of the author:

Omar Rihawi, Yann Secq, and Philippe Mathie

2.2.2 Title of the paper:

Load-Balancing for Large Scale Situated Agent-Based Simulations

2.2.3 Inference:

This paper gives investigates two fundamental circulation ways to deal with reenact largescale investigated two fundamental circulation ways to deal with reenact largescale arranged MAS applications:

1) Agents dissemination

2) Environment dissemination.

Experience on different classifications of arranged MAS applications demonstrates that a few applications can be circulated efficiently with one appropriation approach more than the other. Undoubtedly, each methodology has a few highlights that enable a few operators to naturally adjust the heap among machines, and that can make some differences in the execution.

Specialist appropriation manages division of the operators list between different machines or N operators for each machine. Each machine handles N specialists to execute their collaborations and speaks with different machines to accomplish operators' objectives, for instance each machine must send data about its specialists and get other data about different specialists too frame different machines. That must be finished with most minimal correspondence costs.

In Environment Distribution, the earth can be partitioned between different machines, each machine holds a little piece of nature with specialists that exist on this part. In this way, each machine registers specialists' discernments and all communications that occurs inside its condition cut. At the end of the day, each machine deals with a piece of nature

Here two applications are significantly examined, and tries have been directed to understand the most ideal conveyance for every application.

The principal application is the rushing conduct Model which shows a directing conduct that is ordinarily seen with flying creatures or fish. In this model, there is just a single sort of operator (e.g. feathered creature), which can push ahead with a gathering of other close-by specialists inside its discernment go.

The second application is the Prey-Predator Model which utilizes the idea of operators with objectives. In this model, there are predators and preys. The predators eat the prey and it passes on the off chance that it doesn't get enough sustenance. The prey kicks the bucket in the event that it gets eaten, else it endures and repeats. Both need to exist together to keep the eco-framework alive.

2.3.1 Name of the author:

Xiaohui Cui and Thomas E. Potok

2.3.2 Title of the paper:

A Distributed Flocking Approach for Information Stream Clustering Analysis

2.3.3 Inference:

Rushing Model comprises of three basic guiding standards that should be executed at each occurrence after some time. Three fundamental tenets include:

(1) Separation: Steering to maintain a strategic distance from impact with different bodies close-by.

(2) Alignment: Steering toward the normal heading and match the speed of the neighbour rush mates.

(3) Cohesion: Steering to the normal position of the neighbour run mates.

The three essential guidelines are adequate to repeat the moving practices of a solitary animal groups feathered creature run on the PC. These three guidelines will in the long run outcome in all bodies in the recreation shaping a solitary rush.

In the Multiple Species Flocking model, notwithstanding the three fundamental activity controls in the Flocking model, a fourth principle, "include likeness rule", is included into the essential activity guidelines of each bodies to impact the movement of the bodies. In light of this standard, the run bid attempts to remain nearby to different bodies that have comparative highlights and avoid different bodies that have divergent highlights.

To accomplish great execution on conveyed figuring, a few issues must be inspected painstakingly when planning a disseminated arrangement. First is the heap balance. It is imperative to keep stack adjusting among preparing hubs to ensure every hub have roughly same outstanding task at hand. Second is the earth states synchronization. It is vital for a circulated usage to build up a synchronization calculation which can look after causality. Third is lessening the correspondence between hubs. That incorporate the correspondence overhead of the earth states synchronization and control message trade between hubs.

3. Theoretical Analysis:

3.1 Situated Agent-Based Simulations at a Large Scale: Load-Balancing:

To mimic expansive scale arranged operator based uses of complex wonders with a few a huge number of operators, we have to convey the reproduction on a PC arrange. That can raise a few issues: operators portion, communications between various specialists from various machines, time administration between machines, stack adjusting, and so on. When we circulate multi-operator frameworks (MAS), operators must be isolated between these machines since they have the most computational expenses in the re-enactment. With this division, specialists should even now have the capacity to deliver their typical practices. Be that as it may, this partition ought to be sufficiently reasonable to keep the balance between machines. In past investigation , we have investigated two fundamental conveyance ways to deal with recreate largescale arranged MAS applications:

1) agents distribution and

2) environment distribution.

We have encountered various classes of arranged applications that can be easily adapted to different circulation approaches. Each approach has its own highlights that allow a few specialists to adjust the heap between machines.

Environment Distribution:

In certain applications, the spotlight can be placed on the natural world instead of the operators. This is especially true for recreations that involve physical imperatives In this case, the earth can be isolated from machines by enclosing them in some sort of natural arrangement. Along these lines, each machine has its own recognitions and associations. At the end of the working day, each machine deals with its piece of the earth.

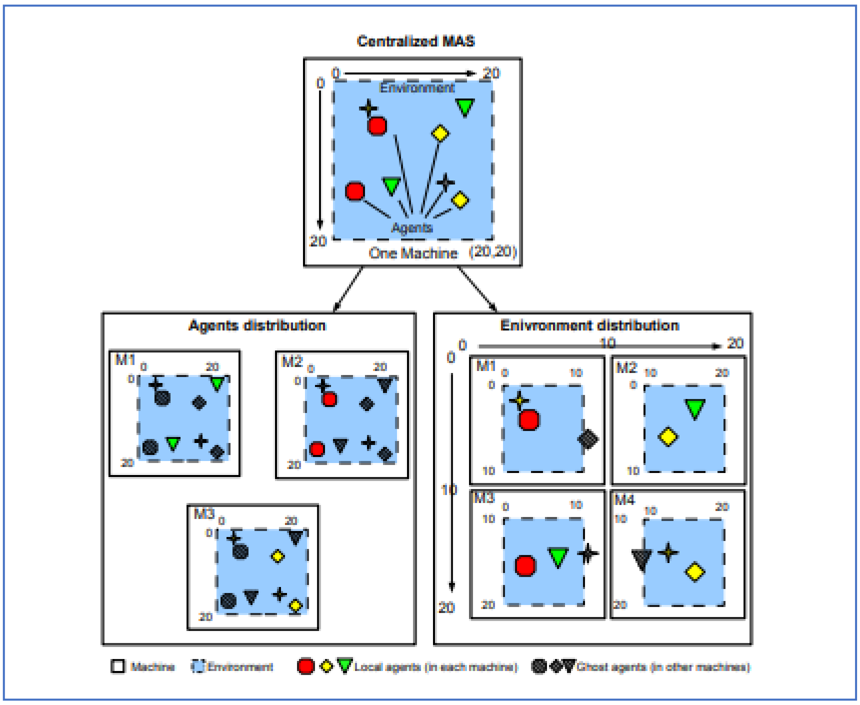


Fig-3.1:agents distribution (left) and environment distribution (right).

- One of the most critical issues that we have is the means by which machines can be programmed in the framework. Data trade between parts should be possible with Ghost region, which allows you to collect data about a region in a neighbouring machine. This zone should be maintained and updated with all messages at each time to permit neighborhood operators from cooperating with each other.
- Each machine has a specific Ghost-territories path that it needs to get around it. It also needs to send Ghost-zone data to other machines. In most cases, the condition conveyance of an arrangement is more reasonable for those who have a typical occupation. Each operator has an equal likelihood to exist in every single exceptional zone on Earth. That implies, operators have a tendency to spread overall condition, which is not confined to one place.

Agents Distribution:

- An operator in a mutli-agent system is a nuclear unit that cannot be divided into littler parts. Regardless of how it might be perceived, it is still a gathering of various entities and practices. Operators are the core segments of the MAS re-enactments. The operators list should be divided among N specialists for each machine. This method would help minimize the dispersion of operators.
- Each machine has its own N operators and speaks to different machines to execute their associations, for instance, if a machine has specialists it needs to collect data about them. This data will help it shape different machines. The machine should have the capacity to store and gather data about different specialists, for instance, neighborhood specialists. It should likewise have the capacity to connect with each other and allow them to collaborate.
- In order to properly implement an environmental distribution system, nature should be isolated from machines and their surroundings. In large scale re-enactments, it can be possible to have more specialists in one player than the others in the same application. For this re-enactment, the stack needs to be adjusted to re-balance between machines and nature parts.

3.1.1 Load Balancing with Environment-Distribution Approach:

There should be an Environment-Distribution (ED) approach where the Earth is divided into various machines. Each machine should have its own piece of Earth. In large scale recreations, it is possible to have more specialists than one player in the Earth. This is because there are usually cases where more are needed in a given application.

For this re-enactment, the stack needs to be re-balanced between machines and nature parts. The concept of an Environment-Distribution (ED) is a way to rationalize the various components of the heap adjusting system. It is a process that involves re-creating the harmony between machines.

There is a good arrangement where one division of the earth has one measurement division and one server that will partition the work among various machines. For them, it's a piece of the earth that comes from a division with a single measurement, to every customer. This division can be adjusted to make it work for different machines. It is difficult to execute for what it is worth on one measurement.

3.1.2 Load Balancing with Agents-Distribution Approach:

If there should be an occurrence of Agents-conveyance approach, the dissemination of information about the machines could be divided among various agents. Each machine has its own set of N specialists that it can handle. These individuals communicate with each other to accomplish their goals. In a few applications, specialists can bite the dust and disappear from the framework. An instrument that re-balances the heap between machines is expected to accomplish these goals.

The most effective way to get around AD approach is by reloading machines that have missing operators. This method will allow the client to identify what N can do next. Even with a settled N, the recreation can still be stretched or enhanced.

This procedure can be utilized to keep the reproduction units from getting overwhelmed with high interchanges. The goal is to make a measure of exchange ventures to adjust the heap. In this scenario, more interchanges can be delivered and executed.

3.2 Benchmark Applications:

This article proposes two fascinating approaches to distribute a reproduction. It shows an examination between two classifications. Different applications have diverse highlights, which are often derived from operators' highlights. For instance, highlights can be short or long amid re-enactments.

Operators can move on little or substantial zone in order to alter the calculation in their re-enactment. Operators can exist in overall condition or can be totaled on a piece of it. The client should have the capacity to pick the type of appropriation that is more appropriate for his application.

For instance, if the life-cycle is shorter than the calculation, then the dispersion approach could be considered futile for some applications. Earth dissemination could be hampered by the accumulation of material and its long life-cycle. This could make it impractical for different applications. For that, one conveyance type can be utilized for a limited number of applications.

Agent features	Category-1	Category-2	...
Life-cycle	Short	Long	...
Movement	Small area	Large area	...
Positioning	Everywhere	Aggregation	...
Reproducing	Yes	No	...

3.2.1 Flocking Behaviour Model:

In nature appropriation case, it can be imagined that a flock of birds would start with one condition and then move on to the next. The optimal solution for this model is to have the same number of operators for each machine. This method would allow the machines to have the same computational expense. The basic marvels of rushing conduct are extremely basic. At first, the recreation should make the winged animals have irregular positions and a bearing on the sky. At that point, all winged creatures endeavour to draw nearer to different flying creatures in their observations, at that point and after a few cycles a few fowls ought to be as one in bigger gatherings.

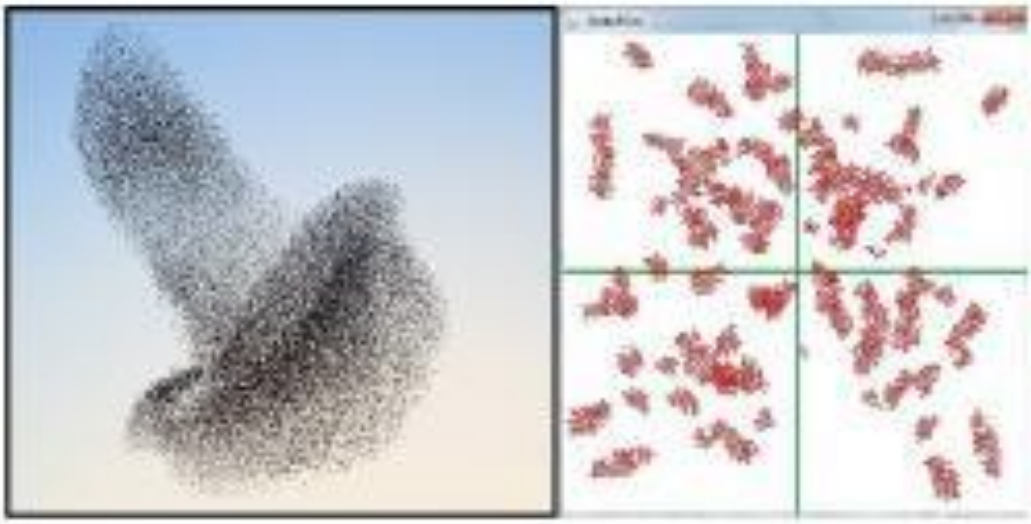


Fig-3.2: Flocking behaviour model (left) and a demo of 4 machines (right)

3.2.2 Prey-Predator Model:

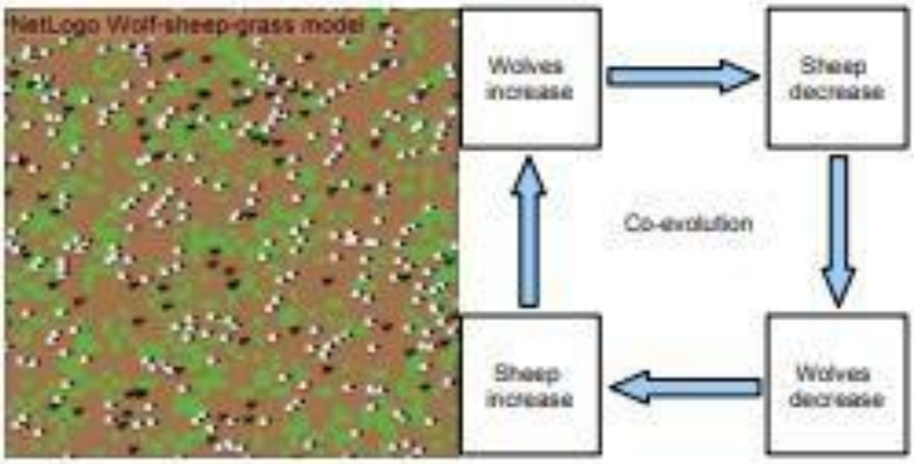


Fig-3.3:Wolf/sheep/grass ecosystem

- The second model is a more intricate version of the rushing conduct show, which is a prey-predator demonstrate. This model utilizes specialists to make goals. A predator is a life form that preys on another creature. The prey is a part of the predator's condition. It needs to get enough nourishment to survive (or preys).
- The predator is a part of the prey's condition. If the prey gets eaten by the predator, it will kick the bucket to avoid getting eaten. The quicker a predator gets nourishment and eats, the more they can reproduce and make more predators. The fastest way to get away from predators is by repeating the same preys over and over again. The two communities have a lot in common to keep the biological community flourishing.
- A case of this model is the fraud committed by Wilensky, which was executed in our system for experimentation. This concept proposes that a wolf-specialist animal that eats a sheep-operator will develop into a grass-specialist animal that eats what's more.Sheep and wolves can have

vitality when they find food. They can then recreate themselves afterwards.

- In typical circumstances, the quantity of wolves and sheep can be relative in a few times of a recreation. In the event that the wolf population increases, the quantity of sheep should also decrease.
- That's when wolves become less effective at hunting and finding food due to the lack of wolves. This is also when they become less likely to attack humans. The sheep can then increase in size, as there are no wolves that attempt to eat them anymore, and The quantity of wolves should also increase as the increasing preys cause the wolves to eat more. There are wolves and sheep in the model as long as the reenactors make it so.
- If we lose all sheep, wolves will never find any sheep to eat, and they should be kicked the bucketIn the event that wolves disappear, the population of sheep would expand interminability. This could be done by developing a wide range of specialists to keep the model alive.

These two applications are picked in light of the way that they convey unmistakable elements. In reality, preys can move amid the reproduction of other animals, though they are still homogeneously acquired over time. In the rushing model, even if the dissemination is homogeneous, the group tends to quickly rise together. At that point, the two applications should have a single primary rush that should show up in the recreation. It should be arranged in a way that enables the exchange of information among the machines.

4.Experimental Investigations:

In order to understand load balancing in Agent-based models, an advanced dynamic load balancing algorithm has been explored. This algorithm consists of two phases.

The suggested system architecture consists of a n number of customers connected to cloud service providers over the internet, with the service provider consisting of virtual machines, a management unit, and a m number of shared pools of resources that we regard to be servers. Agents complete one cycle in two phases at the shared pool of servers.

PHASE-I:

There are two steps in this phase, which are mentioned below:

Step-1:

Iterate through each server in the environment and find out the total number of jobs present in the job queues in all the servers. Using the sum, calculate the average number of jobs each server must have in order to balance the load. This threshold value helps us decide whether a server is overloaded, underloaded or just balanced.

For simplicity, let us consider an environment S consisting of n servers. Therefore

$S = \{S_1, S_2, S_3, S_4 \ldots\ldots S_n\}$

In each server, J_i jobs are present in the S_i's job queue. Initially an agent is initialised randomly at any server. Then it begins iterating through all the servers in order to find the sum.

SUM = $\sum J_i$, where $0 < i < n$ and J_i= Number of jobs at i'th server.

Next, we calculate the average value each server needs to have, in order to be balanced.

AVERAGE = SUM / n, where n = Number of servers

Step-2:

Now, iterate through all the servers and check the if the number of jobs in each server is greater than, less than or equal to the average value and classify them as "Overloaded", "Underloaded" and "Balanced"

respectively. The server status is decided as follows:

```
For each server Si

{

If ( Ji> AVERAGE)

{

Set Server status = "Overloaded"

}

Else if ( Ji< Average)

{

Set Server status = "Underloaded"

}
```

Else

{

Set Server status = “Balanced”

}

In this way, Agent completes Phase-I.

PHASE-II:

In this phase, jobs are transferred from overloaded to underloaded servers, using a data structure (extra_jobs) to distribute them as follows:

For each server Si,

{

If (Server Status== “Overloaded”)

{

Set extra_jobs= Ji – average;

}

Else if (Server Status == “Underloaded”)

{

Add jobs from extra_jobs into the underloaded server till it is balanced.

Update extra_jobs.

}

}

Determine the number of jobs that can be moved to an underloaded server if a server is overloaded, and transfer them; if a server is underloaded, determine the number of jobs that can be received, and migrate jobs from overloaded servers if a server is underloaded. The following formula can be used to calculate the number of jobs that can be broadcast from an overloaded server and the number of jobs that can be received by an underloaded server:

Number of jobs to be Transmitted or Received = $|\text{AVERAGE} - J_i|$

This step will be repeated until the load on all servers is balanced. The agent will balance the load in this manner without interfering with the system's operation.

IMPLEMENTATION OF ABOVE ALGORITHM(PYTHON):

```
import random
s=[[random.randint(1,100),"loaded"] for i in range(5)]
print("The number of jobs present in servers(random) ->     ")
print()
for m in s:
  print(m)
#PHASE-1-STEP-1
k=random.randint(0,4)
print("An agent is initiated at a random server")
print()
print("In this case the random server is: ",k+1,"server.")
print()
sum1=0
print("Starting PHASE-1")
print()
for i in range(k,k+5):
  sum1+=s[i%5][0]
print("After step1 ->")
print()
print("The sum of jobs present at all the servers is: ",sum1)
print()
avg=sum1//5;
print("AVERAGE value is: ",avg)
print()
#PHASE-1-STEP2
for i in range(5):
if(s[i][0]>avg):
s[i][1]="overloaded"
elif(s[i][0]<avg):
    s[i][1]="underloaded"
  else:
  s[i][1]="balanced"
```

```
print("After step2 -> ")
for m in s:
  print(m)
print()
#PHASE-2
extra=0
t=0
ic=k
while(t!=5):
i=ic%5
  if(s[i][1]=="overloaded"):
  extra+=s[i][0]-avg
  s[i][0]=avg
  s[i][1]="balanced"
    t=t+1
elif(s[i][1]=="underloaded"):
    if(extra>0):
      ex=avg-s[i][0]
      if(extra>ex):
        extra-=ex
      s[i][0]+=ex
      else:
      s[i][0]+=extra
      extra=0
      if(s[i][0]==avg):
        s[i][1]="balanced"
        t=t+1
elif(s[i][0]>avg):
      s[i][1]="overloaded"
      else:
      s[i][1]="underloaded"
ic=ic+1
print("After PHASE2: ")
```

```
print("")
for m in s:
  print(m)
print()
print("The extra jobs left -> ",extra)
```

5.EXPERIMENTAL RESULTS:

Output1:

```
The number of jobs present in servers(random) ->

[42, 'loaded']
[17, 'loaded']
[42, 'loaded']
[70, 'loaded']
[6, 'loaded']
An agent is initiated at a random server

In this case the random server is:  4 server.

Starting PHASE-1

After step1 ->

The sum of jobs present at all the servers is:  177

AVERAGE value is:  35

After step2 ->
[42, 'overloaded']
[17, 'underloaded']
[42, 'overloaded']
[70, 'overloaded']
[6, 'underloaded']

After PHASE2:

[35, 'balanced']
[35, 'balanced']
[35, 'balanced']
[35, 'balanced']
[35, 'balanced']

The extra jobs left ->  2
```

Output2:

```
The number of jobs present in servers(random) ->

[80, 'loaded']
[25, 'loaded']
[57, 'loaded']
[19, 'loaded']
[27, 'loaded']
An agent is initiated at a random server

In this case the random server is:  4 server.

Starting PHASE-1

After step1 ->

The sum of jobs present at all the servers is:  208

AVERAGE value is:  41

After step2 ->
[80, 'overloaded']
[25, 'underloaded']
[57, 'overloaded']
[19, 'underloaded']
[27, 'underloaded']

After PHASE2:

[41, 'balanced']
[41, 'balanced']
[41, 'balanced']
[41, 'balanced']
[41, 'balanced']

The extra jobs left ->  3
```

6.DISCUSSIONS OF RESULTS:

In the above implementation of the pseudo code, the random package is imported and is used to generate random number of jobs at each server. For implementation sake, 5 servers have been considered. The value can be scaled as needed. The final output consists of some balanced servers along with the number of jobs remaining to be distributed.

The main drawback of this algorithm is that, some jobs might be left undistributed in order to not overload any of the servers. This can be solved using priority or shortest job first algorithms, in order to assign the remaining jobs to the server that completes the existing job queue the fastest.

7. CONCLUSION AND FUTURESCOPE:

In this paper, we have studied various distributions of the cloud environment (Environmental and Agent-based) and referred two models; Flocking and Predator-Prey Model. Later a dynamic agent-based load balancing algorithm has been observed along with pseudo-code. The implementation shows that a few drawbacks exists, which give a chance for further modifications in future scope.

Job Scheduling algorithms can be used to distribute the remaining jobs based on which server finishes the jobs in the shortest amount of time. The remaining jobs can then be dynamically assigned by the agent , thus ensuring an optimal approach towards the problem.

8. REFERENCE/BIBLIOGRAPHY:

1.Load balancing for agent based simulations – by Omar rihawi,YaanSeeq,Philippe Mathieu

2.Dynamc Load balancing – by Jitendra Grover, Shivangi Katiyar