Resource optimization in Cloud Computing using dynamic load balancing technique

by

Mutasim Rafid
17101304

Jeba Tahsin Hossain
20241066

Mir Nafisa Ali
16301125

Nafisa Tabassum Abonti
17101465

Anika Masud
17101236

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science and Engineering

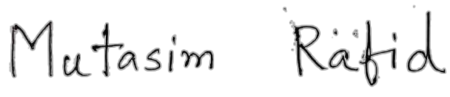
Department of Computer Science and Engineering
Brac University

Declaration

It is hereby declared that

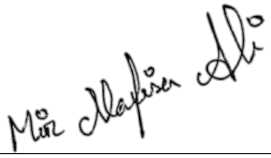
1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

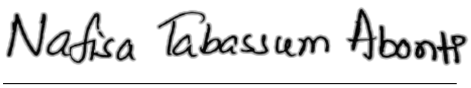


Mutasim Rafid
17101304

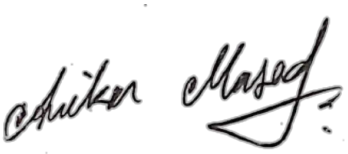
Jeba Tahsin Hossain
20241066



Mir Nafisa Ali
16301125



Nafisa Tabassum Abonti
17101465



Anika Masud
17101236

Approval

The thesis/project titled "Resource optimization in Cloud Computing using dynamic load balancing technique" submitted by

1. Mutasim Rafid (17101304)
2. Jeba Tahsin Hossain (20241066)
3. Mir Nafisa Ali (16301125)
4. Nafisa Tabassum Abonti (17101465)
5. Anika Masud (17101236)

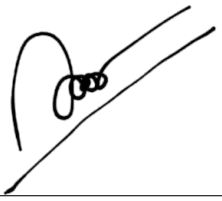
Of Spring, 2021 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. In Computer Science and Engineering on June 02, 2021.

Examining Committee:

Supervisor:
(Member)

Moin Mostakim
Lecturer
Department of Computer Science and Engineering
Brac University

Co-Supervisor:
(Member)



Dr. Muhammad Iqbal Hossain
Assistant Professor
Department of Computer Science and Engineering
Brac University

Program Coordinator:
(Member)

Dr. Md Golam Robiul Alam
Associate Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Sadia Hamid Kazi
Chairperson
Department of Computer Science and Engineering
Brac University

Abstract

In this dynamic era of science and technology, the endless advancement of cloud computing and the tireless usage of different devices now demands more optimized cloud–based services to users over the internet system. Furthermore, most of the cloud computing architectures are implemented in such a way that it only works like one user per one virtual machine (VM) for which, it leads to an extensive wastage of resources. That is why, the urge of dynamic partitioning of applications in cloud servers based on the availability of resources is crucial in achieving the best usage of virtual machines for any computationally intensive applications. Therefore, adapting a load balancing technique that provides an efficient and fair allocation of cloud resources while providing high availability to the end users is a necessity of time. That is why, we worked on dynamic performance optimizing load balancing algorithm "Weighted least connections" and we tried to show why it is the best load balancing algorithm according to our paper's purpose of utilizing one VM for multiple users on cloud servers. We examined our proposed algorithm with our proposed Virtual Machine environment and found that our algorithm and environment is far more efficient than other single VM based cloud architecture.

Keywords: Cloud computing, Virtual machine (VM), Resource optimization, Load balancing technique/algorithm, Weighted least connections

Acknowledgement

Firstly, all praise to the Great Allah for whom our thesis has been completed without any major interruption. Secondly, to our supervisor Mr. Moin Mostakim sir and co-supervisor Dr. Muhammad Iqbal Hossain for their kind support and advice in our work. They helped us whenever we needed help. And finally, to our parents without them throughout sup-port it may not be possible. With their kind support and prayer, we are now on the verge of our graduation.

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation:

By cloud computing we understand the process of the delivery of different computing services over the internet (such as: servers, networking, databases, storage, analytic, intelligence, software etc). To be specific, the whole cloud of computer networking is generalized as the 'Cloud Computing' term. Cloud computing not only means the computing of a PC but it is also running through various mobile applications, AI, IoT devices, Block-chain etc all around us. The presence of the cloud of these super computing services is almost undeniable to experience. According to a Juniper Research survey, the customer and business market for cloud focused mobile apps would have grown to $9.5 billion by 2014 [1]. Despite the growing popularity of mobile cloud computing, fully realizing its potential is difficult due to a number of inherent issues, one of which is a scarcity of resources. The lack of resources has a huge impact on the overall performance of the mobile cloud system. To be more precise, each virtual machine in the cloud assigns its user in such a way that one user uses one virtual machine, and the majority of them are never completely used. This obviously results in a widespread failure of maximum resource utilization, and users are deprived of potentially useful resources whenever they are required. As a result, the authors have proposed a dynamic load balancing strategy or algorithm for the efficient distribution and equitable allocation of cloud resources, which we believe would result in a more usable and beneficial virtual machine for users. We have used the "Weighted least relation" algorithm to enable multiple clients to use a single VM at the same time. In order to maximize resource utilization tasks would be distributed among nodes according to this algorithm. In weighted least connection, users are usually connected using the weighted least link approach depending on the number of active connections. The selections made by this algorithm are also based on server availability. In the weighted least connection algorithm, the system establishes a proportional algorithm for each pool member based on the value the user defines in the connection limit. This proportion, as well as the number of current connections to that pool member, are used to make the load balancing decision [2]. One member, for example, has 30 connections and a connection limit of 100, so his total connection capacity is 30%. Another member, on the other hand, has 30 connections while the connection limit is 300, so its total connection capacity is 10%. The machine will choose the second member in this case. A non-zero relation limit must be defined for all pool members in order for this algorithm to work. This load

balancing method would move a network experience to the pool member or node with the least number of active connections if all servers have the same capacity [2]. We discovered that this algorithm works best in an environment where servers have varying capacities since this paper suggests an algorithm that works on utilized user power of virtual machines. If two servers have the same number of active connections but one has more bandwidth, the BIG-IP system measures the percentage of capacity used on each server as well as uses that percentage in its measurements [2]. However, while this algorithm is perfectly suited for our problem, it does have a disadvantage too. When choosing a pool member or node, the weighted least link method excludes idle connections from the calculations and in the calculation, only active connections are used [2]. Despite this flaw, our team discovered that using weighted least load balancing algorithm rather than other load balancing methods is the best approach for load balancing problem of resource optimization in mobile cloud. Furthermore, the problem of single VMs used by single users and the wastage of remaining cloud storage is also an issue that we can not overlook at this point. In our research, we have seen that most of the cloud servers use a single VM for single users while only 60% of the storage is being used by the users and on the other hand, a potential percentage (40%) of unused server storage is being wasted [3]. To be specific, there are multiple cloud operating systems available in cloud servers. However, there are very few operating systems that actually allow multiple users in a single VM to provide the best service and performance at the same time. Even though Linux is the most widespread and used operating system for cloud servers, the performance, reliability and services of Linux is not up to the mark at all when it comes to allowing multiple users for a single VM. On the other hand, Windows systems by Microsoft do not work for multiple users where this paper's aim is to develop an environment for multiple users so that not a single percentage of the storage goes waste. For all of these reasons, our team has researched and agreed to work on Solaris operating system which not only allows multiple users in a single VM but also gives the best performance compared to Linux and Windows operating systems. That is why, this paper is not only focused to solve the load balancing issue of cloud servers but also aimed to implement a virtual machine that can be accessed by multiple users at the same time in the cloud servers. Now, the aim of this paper is to bring the proposed solution to the test, evaluate the results, and conclude that it is the best solution for load balancing and resource management by implementing the architecture of multiple users and single virtual machines.

1.1.1 Aims and Objectives

Make one virtual machine usable to many users. Apply the required load distribution algorithm. To spread the loads, we used a weighted least load balancing algorithm. Make mobile computing more efficient. Improve the efficiency of cloud computing Making Cloud Computing cost effective. Uses of cloud computing to boost the computing capacity of each cloud connected device. Reducing the cost of cloud computing services.

1.2 Problem Statement

Cloud storage refers to the concept of storing data on the internet, primarily in cloud servers, and then accessing it through the internet when needed. It allows users to use any amount of computing power simply by connecting to the internet at a high speed. Over 70% of IT load is handled by the cloud. In addition, the cloud consumes 30% of IT expenditure. Amazon (AWS), Google Cloud, Microsoft Azure, IBM, and others are among the major tech companies that provide cloud services [23]. As technology advances, the cloud is increasingly being used to store mobile data. In today's world, cell phones, PCs, IoT devices etc. are being used to operate more efficient applications that are used on a daily basis. More resources are required in cloud servers as more powerful applications run on them. However, since cloud servers are being used heavily, PC or mobile phones have fewer resources such as RAM, CPU power, and battery capacity nowadays. Moreover, if more RAM or CPU processing power is used by the users, it kills the most recent application in the stack. Then again, if the user requires the program to re launch, it gets restarted. It wastes time while also consuming more energy. Furthermore, if the power is complete, the device will hang. Even though there are some powerful computers, mobile and IoT devices are available but this is not available for everyone. For example, random general users or people from lower class can not afford these high-end devices. That's why this issue has a significant impact on the cloud based devices.. People's productivity is also being harmed as a result of this problem. Cloud servers, on the other hand, have an enormous amount of resources that can be used more effectively, but we are unable and ignorant of how to make the best use of them. As a result, a significant amount of computing power remains idle while the cost continues to rise. Because of the limited processing capacity of computers and other devices and the widespread use of full computing power, significant problems have arisen. Low battery, slow processing unit, loaded storage, single user for single VM, these are some of the major issues. Since the battery's efficiency degrades when it is charged more, the battery's capacity decreases. Furthermore, people do not have enough computing power when they require it. This is why cloud servers can overcome the limited computing capacity, storage capacity, excessive user demand etc. by using this extra computing power and the existing idle computing power of cloud servers. Moreover, Cloud computing can be used to overcome the limited computing capacity of cloud devices. Each consumer in cloud computing is assigned a single virtual machine. However, the majority of the time, the consumer does not make use of that capability. As a result, a significant amount of capacity is squandered each year. To solve the problem, we used the Weighted least load balancing algorithm in our research paper. This algorithm was used to spread a single virtual machine to different users such that if one user's virtual machine is idle, other users may use it. Furthermore, implementing an architecture that works for multiple users while accessing a single virtual machine and solving the problem of limited storage capacity for one virtual machine is also a need of time now. Since we are experiencing that, the wastage of resources is increasing day by day since most of the virtual servers are providing their users 1 VM per user where a user only gets to use the 60% of their resources. Which means the remaining 40% goes to waste since no user shares their resources with any other user due to the security and privacy issues [3]. For all of the above-mentioned reasons, our team felt the

necessity of looking into this issue and working on the problem so that the problem can be resolved in further research and architecture implementation. According to our research, it has been observed that Solaris is the best suited solution for our problem for multiple reasons. To elaborate, there are some features that Solaris has which are entirely suitable for our architecture of multiple users and single VM. For example, Solaris allows multiple users which Windows does not and Linux ends up providing bad performance and unreliable services. Solaris is a binary compatible system which Linux does not. Furthermore, Solaris has excellent throughput and user scheduling and server management system where, both Linux and Windows have decent throughput, user scheduling and server management system. Moreover, Solaris provides outstanding service while providing reliability and privacy to the users which Linux and Windows have faced some issues with [4]. As long as our main motive is to provide multiple users a single VM with excellent performance while handling user pressure, great throughput and user scheduling and safe or secure service, we have decided to work on Solaris operating system to implement our virtual machine for multiple users. In this study, we're primarily interested in resolving these interconnected issues. This is how cloud computing performance will improve, and we will be able to use it in digital devices to make cloud computing faster and more efficient.

Chapter 2

Related Work

The concept of resource optimization using mobile cloud is not an entirely new strategy. A lot of research has been done to optimize the resources through mobile cloud. Here is some work's summary for better understanding of the concept of resource optimization using mobile cloud.
K.L. Giridas and his team [6] proposed an improved scheme allotment method for cloud computing environments. On their paper progression, they tried to consider bandwidth and processing capability, paired assigned concurrently to every service entreaty and carry back it on an per hour foundation. According to their analysis the owed supplies are bound to each service call and their proposed ORAT method can ask for loss possibility which results in, dropping the total means mandatory, contrasting with the foreseeable issuing method [6]. They also tried to make effective service assignments, where the supply for the tasks are distributed for cloud computing domain by cutting off e-waste and making Information technology as a Green information technology environment. Their purpose of the offering effective resource allotment technique (ORAT) for green cloud computing allocation is to make use of the amount of appeals to which both can result into well maintained resource processing capability and bandwidth because the figure of desired mean operating potentiality does not usually have a fixed interconnection with that of demanded bandwidth and the standard resource assignment cannot be gained if only a single way of type is measured in the variety of a center[6].
Their proposed effective resource allotment technique (ORAT) for green cloud computing is operated in Java utilizing cloudsim which is a software. In this work, they have discovered how a stock of processors can be assigned to users' tasks based on the highest resource utilization technique to catch the implementation of the cloud computing process in Green information technology to other systems that are registered in mainstream languages such as Java. They also ran autonomic tests with several supportive tasks with a persistent figure of tasks conveyed by each consumer. Their suggested effective resource allotment technique (ORAT) for cloud computing in Green information technology can be carried through in CloudSim software, and several execution attributes can be simulated to make the most of the performance of the advanced ORAT in terms of performance executing ability, resource allocation effectiveness, bandwidth[5]. Their paperwork maximum resource allotment technique can take part in to enabling the means allocation between more than one users in the deficiency of huge beg off in resource organization, contrasting with an existing ITTPS method which does not judge the equitable allotment of measures.

Minxian Xu and his team [13] focused on pushing data centers with renewable or green origins of energy that can weight down brown energy usage and as well carbon outrush. They claimed strengthening data centers with these energy initiators is challenging, as they are changing and not found at all times [13]. In this work, they tried to formulate the micro utility management problem as finite Markov Decision Processes (MDP) to utilize endless power use. By actively switching off non-mandatory micro resources and scheduling battery consumption, upon the user's preference, their suggested technique makes a trade-off between the workload implementation and brown expenditure.
They adopted art optimization algorithm which including DMWB (Urged Microservices Without Power supply), SLW (Rolling Window), and BF (Best Fit) as benchmark to quantify efficiency of the system such quantity of microservices, brown power consumption, algorithm reliability, and percentage of green energy consumption. They successfully examined their report traces created from two realistic workstations as well as standard solar data. Simulated investigations reveal that their policy evaluation beat baseline systems by up to 30% when that comes to handling brown power usage and workloads[13].
They studied the effect of varying parameters on the quantity of processed microservices and average throughput by tweaking the compensation function for the Wikipedia traffic from 0 to 1. As hypothesized, the higher the quality, the less microservices are conducted on average, and so less brown power is generated. When the parameter is 0, for example, the strategy runs the maximum estimated amount of microservices when using a certain proportion of brown energy power. When the input is higher than 0, the minimum rate of microservices drops, as does the utilization of brown power. To counterbalance the transaction in the subsequent investigations, they established a figure of 0.5. In practice, production methods have the authority to adjust the value differently to fit into their preferences [13].
Their proposed method is expected to be approximately the exchange between all the volume of microservices including the use of brown power. They would like to design and develop a system prototype implementing the proposed approach in the coming years. They'll also introduce more realistic phenomena to their framework, such as storage self-discharge, dynamic grid electricity costs, and net metering to effectively tackle Markov decision processes using a reinforcement learning approach.
Arthur Geronimo and his colleagues [15] presented a hybrid solution for the data center's cognitive resource provisioning that employs the cloud environment as an action that may deviate to lessen the likelihood of SLA breaches due to unanticipated workload peaks. They presented a dynamic reconfiguration technique for the VMs' features, which they assigned in the data center[7].
To model the university data center environment, they used CloudSim, a cloud modeling program. To mimic a distribution that is both realistic and difficult to implement. To acquire diverse methods, they employed an actual distribution acquired through regulating requests on the institution's websites, as well as a distribution created from a mathematical rhythmic model. The preceding methods make use of real-world settings, with outcomes that are meant to represent reality.
Their composite method is a combination of two existing techniques: the On-Demand (OD) and Additional Resources (SR) methods (SR). It aims to strike a balance between the duo, taking advantage of both sides' capabilities [7]. They advised the systemic approach in circumstances when the resource activation time is

burdensome for SLA fitness, but not when the public cloud is used infrequently due to their course or other factors [7].
Their expanded work would involve developing specification words to express the structure and function of a service, as well as facilitating the flow of information across workloads for cloud modeling, scheduling, and management.
Mazedur Rahman and his team [9] concentrated on superior cloud computing and the extensive usage of handheld phones, which created new demands for cloud-based offerings from smartphone subscribers through the wireless Internet, propelling mobile cloud computing forward (MCC). They attempted to incorporate both Green processing and efficiency in their article, both of which are essential problems in mobile cloud computing. Their work attempted to define the study area and classify topics related to energy conservation in smartphone clouds. Then they looked at the previous findings on these topics. In addition, the study makes judgments and indicates outstanding areas for further investigation.
Their goal is to give new possibilities and challenges in environmental sustainability by utilizing computer, network, and mobile infrastructure to boost resource utilization, sharing, and virtualization while reducing costs and conserving energy with MCC. The rapidly expanding mobile device user base necessitates a strong demand for new mobile cloud infrastructures, solutions, and information technology that address environmental sustainability and reduction, allowing them to benefit from a rich mobile computing technology that significantly affects unified elastic raw materials of various cloud and network technologies toward unbounded functionality, storage, and freedom of movement. Based on the pay-as-you-go approach, it can service a large number of mobile devices anywhere, at any time, through an Ethernet or Internet connection, independent of the surroundings or platforms [9].
They introduced Erdos, an Android OS update that integrates two ways to prolong the battery life of mobile devices. A user-centered proactive resource monitoring system that estimates resource needs based on users' circumstances, habits, and activities that are observed on a regular basis. Flexible resource acquisition, in which the mobile device may use resources from other nearby mobile devices local wireless networks and social networks. Because of its widespread use and low energy consumption, the instigators recommended Bluetooth as the best wireless interface. They claim that delegating chores to a nearby smartphone is another way to conserve energy.They presented an ad-hoc mobile cloud framework in which smartphones delegate duties to nearby mobile devices that seem to be doing so. According to this architecture, a small piece of the task is performed locally on the mobile device, while the remainder is outsourced to adjacent mobile devices, resulting in energy savings. They also engaged in content transfer from the basic cloud computing (CC) data center to local cloud data centers or MCC network infrastructure as another way to cut energy usage. This methodology, according to the researchers, may be used to deal with a variety of knowledge applications, such as distant learning for academic lectures, workshops, and exhibits [8]. Their article focuses on the mobile cloud computing field. The potential applications, problems, and demands in energy conservation in mobile cloud computing will be addressed.
The deployment of a cloud computing environment was recommended by Wassim Itani and his colleagues [9]. Patrons of important cloud services are hesitant and cautious about the integrity of their data housed in the remote cloud, owing to its fast expansion. Their findings were exacerbated in mobile cloud computing archi-

tectures that enable battery-powered wireless and mobile client devices with low energy capabilities. Integrity processes for interpreting and enforcing, according to their findings, should use energy-efficient algorithmic approaches and cryptographic data structures to safely allow dynamic actions on distant data while also taking into account the limits of mobile clients [9].

Their goal is to develop an energy-efficient mechanism for ensuring storage service dependability in mobile cloud computing. The suggested approach used incremental cryptography and trusted computing to provide safe integrity data structures that safeguard customer data while consuming less mobile battery and executing dynamic data activities more efficiently. Their system design has been analytically studied and technically deployed to show the efficiency gains it delivers on mobile clients [4]. They attempted to offer an integrity protocol that consists of a collection of hash and Message Authentication Code (MAC) algorithms that properly enable incremental updates. This implies that if a message with a MAC is modified by adding or deleting a block of data, the progressive function can safely generate an updated MAC value using the embedded block without having to re-generate the messaging MAC value on all the message contents from whatever is in hand. This trusted computing concept is built on the Dyad project's cryptographic coprocessor applications. A digital user receives cloud storage services, a cloud service provider administers and administers the cloud storage facility, and a strong authentication manipulates and supplies tamperproof cryptographic coprocessors for deployment in the remote cloud, according to their system model. Each crypto coprocessor is assigned to a number of different clients and shares with each client a unique shared secret. The initialization phase, the data update phase, and the integrity verification phase are the three phases of their system functioning [9].

A proof of concept deployment employing an emulated cloud computing unit was used to evaluate their system architecture. VMware Workstation 7.1 is used to offer the virtualization layer. The application server is an HP iPAQ Pocket PC with a 300MHz CPU and 64 MB of RAM that runs Windows Mobile 5. They attempted to assess the energy savings by doing fifty typical insertions. . The files ranged in size from 100 KB to 2 MB. According to them, the average execution time of the incremental MAC generation on the mobile device is 37 milliseconds, compared to 420 milliseconds for the standard integrity technique, which builds the file CBC MAC from scratch. This resulted in a decrease of over 90% in processing needs and, as a result, in energy utilization on the smartphone device. Because the incremental and CBC MAC functions employ similar system activities, the processing and energy reductions were proportionate in their situation.

Above all, their study described an energy-efficient security protocol for assuring storage credibility in mobile cloud computing environment, as well as a brief analytical study and an investigational conceptual design to show the efficiency improvements or energy savings [9]. In order to accomplish our objective, we conduct a study of existing approaches and compare their pros and cons. Here is some work's summary for better understanding on the concept of resource optimization using mobile cloud through Q learning and load balancing.

Initially, the Markov Decision Process established a standard framework for defining the decision-making scenario in the surroundings, as per Xu Wang [14]. The future state is decided by the present state and action, and is irrespective of all prior states and actions, according to the Markov property. The value communication

algorithm might be used to solve the Modeling Approach but It necessitated prior understanding of the changeover and reward functions. The Q-learning algorithm does not have a fundamental understanding of how the world functions, but it can learn the policy from its own experience. However, Q-success learning is still dependent on the quality of hand - crafted features. This limitation leads to a limited process scope, as low-dimensional state space and readily hand-engineered characteristics are always required. Now, a progress in computer vision has been made thanks to the convolutional network, which can extract high-standard features from the original increased dimensional inputs.

He noted that the Google Deep mind group released an article in Nature in 2015 called Human level control through deep reinforcement learning, which resulted in a revolution in this field. They showed that a deep Q-network could cope with difficult situations and learn control policies features from the raw pixels data and gaming results [14]. Their method has been used in a number of Atari 2600 games. They show through testing that the new network outperformed all prior algorithms and achieved a level of performance equivalent to that of a competent human game examiner across a collection of 49 games. He used the deep Q-network technique on a few typical RL testing scenarios, including as the Grid-World, inverted Pendulum and mountain car problems, in this thesis. Many reinforcement learning techniques, such as Q-learning and SARSA, have tackled these areas well.

On the Ubuntu platform, he used Torch to create the agent, the Deep Q-Network, using the programming language Lua. Three convolutional layers and two fully-connected layers make up the whole network. The DQN agent analyses four recent sequences at a time throughout the training operation and divides output units for each conceivable action. He built a novel framework for typical RL domains, including certain Grid Word Route planning, Mountain Car Dilemma, and the Inverse Pendulum problem, by incorporating experience replay, reward normalization, and occasionally suspending the target network. BURLAP, an open-source Java library, is used to create an atmosphere. We employ socket programming using TCP/IP protocols since there are linguistic hurdles between Java and Lua. When an action is performed, the environment can return the payout, terminal signal, and frame pixels. He proved that the DQN representative could perform three different tasks with the same architecture without adjusting the settings as a result of the trial [14]. He made an appearance in the journal, which reports on the amazing achievements of the DQN, stating that they have seen tremendous success in the games made by Google. The challenges contained in this multidimensional state space usually present themselves as great difficulties for human beings to resolve. Consequently, in our studies, they find their states to be highly generic, the regular RL domains have wide-dimensional state spaces that have already been extensively expressed by other RL methods. He contrasted the algorithm's results with the algorithm's overall success to standard Q-learning and SARSA. On the basis of the outcomes, SARSA's results, it appears to be the best algorithm for three separate domains with an average of around 100 episodes per domain. Additionally, the DQN was found to be well-suited for more complicated datasets with functions, like those which can't be manually constructed. The reason for this is that the DQN agent still needs more time and resources for image processing is because of this [14].

The paper by Kousik Dasgupta and his team [7] explains how Load Balancing can be used by cloud service providers to spread workloads through cloud services or

server resources. and there are three primary methods of load balancing: centralized, dynamic, and non-predictable; static and non-rhythmic; and non-repeating or irregularly-cycling. An anecdote was shared about Armstrong, who used Cloud Service Providers (CSP) regardless of the actual workload on the machine. It has been shown that some existing scheduling methods, such as Min-Min round robin and FCFS load balancing are still used in literature. They also stated that Yang Xu's technique for data-intensive applications, which uses a new distributed data mining model for balancing data delivery is effective.

Genetic algorithm was employed as a dynamic programming by this team in the construction of the analytical model, which implements natural selection as a virtual strategy to further improve the model's flexibility and allow greater use of simulation and analysis, respectively. Stochastic expansion is compared to two widely used scheduling algorithms, both of which are equivalent algorithms in that regard, as well as a local search algorithm, called Stochastic hill climbing [7].

The paper by Dr. Mahalle and his group [8] identified three load balancing algorithms, which included the Round-robin algorithm, equally distributed current load, and throttled Load Balancing.

Draw in a logarithmic spiral (using a logarithmic scale) during the exam using intervals of half-life time of 1/2 hour. This algorithm is labeled Round Robin such that it follows the concept of distributing resources equally. Each node has an assigned period of time slot, and waits its turn in the sequence. Each node gets an amount of time and a certain amount of interval, all of which are equally separated and shared. Both nodes have a certain amount of time during which they must carry out their respective functions. Contradicting accusations include the fact that the involvement of this algorithm is smaller than the two other algorithms. This is an open-source algorithm known as cloud-expanding (or cloud-expanding) simulation method, which is the algorithm that is employed if the simulation doesn't have any program dependencies. Allocating the workload using a non-strictly ordered round-robin scheduling method ignores the variation in resource loads on various machines [8].

Computation on equally around the servers and actively redistributes the execution load: A load balancer is required to actively redistribute the computation load on the servers to others. Load balancer means getting work requests, handing them out to virtual machines, and handling virtual machine handoff in order to complete them. If there are new jobs that have come in after the last time the balancer scanned the queue, it searches for the new jobs and distributes them among the free virtual servers. Often, the inventory of tasks inside the All Tasks Expanded (also known as the Pending Tasks) is maintained, allowing the identification of the jobs that need to be run to determine which virtual machines are completely finished. A series of experiments using the cloud simulation were conducted to refine the method for this algorithm. Since the name for this algorithm means that it is distributed evenly across various virtual machines, virtual machine contention is a normal [8].

Selectable load balancing: The Selectable algorithm finds the most suitable virtual machine for the assignment. It is creating a list of all virtual machines, where each item in the list corresponds to a specific VM and these items are being allotted to the appropriate computer. If a task is well designed for a certain computer, delegate to it. The work manager has no virtual machines to embrace, so it holds the job until one becomes available. If there are no available, it can hold the task until the

customer can take it.
Hado van Hassel and his colleagues [11] doubt that there is a correlation between stress and task performance; a number of factors other than workload affect it. It's not only the importance that needs to be underestimated, but how it is underestimated that becomes an issue. The consequent policy would be kept true to the ideals of all, which are the same as its parameters. Then, we wouldn't anticipate anything except preference conservation in other cases when all values will be greater than their relation. Therefore, there is evidence that optimism in the face of the unknown may be correct: Furthermore, optimism in the face of confusion has been shown to be correct [11].
To ask for [a particular kind of] information if they are either concentrated or accurate among states or if they are inconsistent and/uniform, irrelevant for national policy making They tested the new DQN algorithm to see whether bias has been overstated in the literature and in reality by doing this and that. DQN uses a deep neural network, which has the capacity to deal with diverse and expressive games. DQN was subjected to a variety of varying and exhaustive evaluation procedures and managed to match human abilities on several Atari 2600 games. This may be called a "the best case" for Q-learning; they believe that, due to the deep neural network's ability to do approximate function learning, the machine's function exploration is capable of preventing the detrimental effects of noise but due to the fact that it is more deterministic, the device's learning capacity has very low error. The subjects exhibited significant error in the anticipated long- and short-term rewards extrapolation (sometimes referred to as DQN) in this low-risk scenario [11].
This paper has five important topics to cover. To begin with, they have explained why the phenomenon of overtraining, which occurs in large-scale problems with known probabilities, will occur in Q-learning even though they are stochastic, and can be tied to Q-intrinsic learning's errors of estimation. Additionally, by evaluating the estimated values on Atari games, the authors have discovered that overestimation and neglect are much more prevalent and extreme in real-world settings than was previously discovered. Second, they've shown that Double Q-learning can be applied at a large scale, and with good effect, helping learning remain more robust and accurate over time. Finally, they suggested a particular implementation of the Double DQN which has access to the current architecture and the existing neural network architecture, but has no additional parameters. Double DQN has shown with real-world, empirical research that it can find better outcomes for modern state-of-of-the-the-the-art policies on the-art computers like the Atari 2600 [11].
Double Q-learning, which was first used in a table format, was shown to be capable of learning arbitrary functions in neural networks and was recently shown to be able to also be able to use other types of neural networks. A new algorithm called the Double DQN was constructed based on these questions This algorithm produces more reliable predictions but yields better results on certain games. DQN's underestimation results in underestimated risks may have caused bad decisions, however it has often benefited some through reducing over and over-expanding. in addition, by increasing the domain size of DQN, we obtain state-of-the-the-art results on the Atari platform [11].
In accordance with the article written by Yousse Fahim and his colleagues [14], VM should be spread in order to enhance service efficiency while using all available tools. The second step in their advance is made up of two distinct phases A prior move

in the implementation plan was to classify missions (missions are identified by the following three letter set of words: Bat, group, level, or another formal algorithm known as Bat Algorithm) and then prioritize them in order of expansion (expanding the most or ascending). The corresponding numbers corresponding to the output levels of mathematical functions indicate how many VMs are needed to be positioned in the data center [12].
Prior to the progress of these studies, they expanded their endeavors to do static and dynamic load balancing and even load testing for modern meta-strategies like this one. To help them find a meta-algorithm that falls into Phase 1, the two, a lot of the program had to use single-variation load balancing approaches, and the software is using (or several) several approaches that have just the one form expectation [12]. Rather than implementing a more stringent algorithm, the load balancer used their own approach to spread their distributed resources and load-balanced workload among the different types of customers, which minimized problems encountered with other more ancient algorithms. The conclusion of their paper contains an analysis of the current state-of-the-art load balancing algorithms, on top of the exploration of previously untested meta-algorithms. To summarize, the third and fourth sections did a decent job of presenting their models, and the data in these sections are backed up by the results in the others [12].
Prior to the progress of these studies, they expanded their endeavors to do static and dynamic load balancing and even load testing for modern meta-strategies like this one. To help them find a meta-algorithm that falls into Phase 1, the two, a lot of the program had to use single-variation load balancing approaches, and the software is using (or several) several approaches that have just the one form expectation [12].
Ram Prasad his team's work [5] on a tight load-balancing algorithm, designed to optimize or reduce various output parameters (like throughput, for example), but through careful observation and evaluation, they discovered that, at scale, different sets of entities on the network such as virtual machines or service instances had to be treated differently (virtual topology depending on the application requirement). The paper took the time to explore various load balancing techniques in large-scale Cloud computing infrastructures. To the company's disappointment, their goal was to provide an assessment and comparative analysis of various distributed algorithms to optimize various cloud parameters, such as throughput, as latency, like average delay, as well as varying approaches for big, small, and infrastructure providers like storage, among clouds. All the various internet-connected services in the "clouds" can be treated as clouds of independent computers that are either using clients or only having slaves, respectively. Thus, in Wireless sensor networks (WSN), as well, "cloud" use can be understood as many computerless systems (Cl) that serve only connections and single systems (Servers) that rely on the many computers (C). They have listed services in the cloud that can be divided into three groups: Software- as a service (SaaS), Platform as a Service (PaaS) and Hardware as a Service (HaaS) or Infrastructure as a Service (IaaS)
A person does his or her work by using various programs on different servers across the Internet in SaaS. The package is used as is and does not require extensive modifications or integration. As far as updates go, the provider does so on a schedule even thus maintaining the overall operation of the infrastructure. The customer would have to cover the time they use the software's usage costs. Software does not have to have all the bells and whistles to be great. It only has to perform just one job

and work independently of other programs. SaaS customers don't even need to have strong programming skills; for many of them, the service's value is the ready-made apps and dependability [5].
All of the software development and application construction tools are supplied by the Internet, but no software is required to operate or manage them. PaaS services provide the design, development, testing, and implementation of applications, but does not include storage or operations in addition to all these other solutions, you may want to look into: database integration, team coordination, web site integration, data stability, and versioning are all factors that must be considered [5].
The term "Infrastructure as a Service" (IaaS) is an abbreviation for hardware as a service (IaaS). All the equipment is "on-hand" or "in-hand" configuration means that the company can be prepared to hand it over to an agency, allowing it to choose and install whatever it wants.
Provides network infrastructure which is "rented" in the form of servers which can be used by the operator and provided without obligation provides a service-oriented and Internet-of-service (IoS) architecture for cloud-based applications, such as high scalability, continuous availability, decreased technological and operational overhead, increased service resilience, as well as well as less requirement for the customer as and knowledge the fact that central to these problems is the creation of an efficient load balancing algorithm [5].
Also, they discussed an algorithm (the actions of honey bees), the Honeybee Foraging Algorithm, which is based on the habits of honey bees, was described as well the other type of bees, the food-finding bees, will do a waggle dance when they have found enough to bring back to the hive. The intensity of this showing depicts how many insects are present or the amount of the food is related to the beehive shows. a swarm of bees then follows behind the foragers to where the food is located and then begins to harvest it.
When they retire, they then retrace their steps to give an impression of how much food remains, they perform a beehive dance, thereby driving the hive more into exhaustion or resulting in further food deprivation. If the number of Web servers rises or declines, utilities are shared to ensure the same percentage of users are being loaded on any one of them, so as to control demand for the online resources. The virtual servers are divided into virtual services (VS), each service queue is assigned to a single virtual server (Virtual Service Queues) The numbers of workers and produce received inputs to every server for every transaction in the system, known as a profit, are calculated through the system of Waggle Dance in the same way. This compensation can be measured as per a metric, according to their study, is how long the CPU spends on the processing of a request. This is an interesting analogy because it is relevant to the operation of a beehive: the dance floor, much like an advertisement board with a beehive, gets covered in honey. The board, along with the counters, thus serves to increase the profitability of the whole colony [5].
Xiao Song [16] and his group, in the Institute of Electrical and Electronic Engineers, Modeling and Simulation society recently published an article on an emerging standard in which it was stated that HLA also has "sluggish steady dynamics". There are not enough load-balancing techniques that would need to switch the main (CPU-bound) host ahead of slower hosts while all the other nodes must wait until the slowest is load-balanced before moving to the next secure domain to allow the other (domain-balancing-heavy) to progress.

In terms of the experiments that had been conducted, the HLA balance theory may be separated into two types. In the discussion on distributed and parallel system load balancing algorithms, they touched on ways to use HLA. For HLA-based networks, load balancing techniques, there are guidelines that can be applied that are essentially mapped out to help spread the processing load, but they don't have any effect on the actual operation of the system because they are merely applicable to the architecture of a shared framework. The primary issues are computation and coordination costs; in load balancing algorithms, they are computations and concentration costs. as far back as I can tell, it has been modeled after these prior works in those three areas: Heuristic, computational, and communication cost [5].
They went on to say that heuristic algorithms are used for complex load balancing because of the difficulty of finding optimal values. This paper has examined the use of different strategies for dynamic load handling in a message passing supercomputer. The technique it uses is simple, choose able heuristics (selecting a node with the least number of devices on it for migration) and assigning different migration processes a low or zero load levels based on the location of the multicomputer nodes. In dynamic simulation algorithms, the genetic algorithm is said to be equivalent to a first fit algorithm and to a random assignment method, which is compared to first fit-sharing. It is also seen in the experiment that the Genetic Algorithm (GA) completes in less time and utilizes more average processor cycles than either protein analysis (PA) or VCSA. In their report, adaptive load threshold is stressed to make the most suitable system expand first and to meet the evolving needs of the most heavily loaded nodes, while the simple approach helps to ensure that less loaded nodes are able to grow even though the system is heavily loaded. the proposed NP-complete minimum degree-min spanning problem is simulated annealing algorithm is applied to the authors use it in their research paper, where they show how to turn a problem that is thought to be an NP-complete into and NP-complete It must refine their approaches based on the specific cases that are used in practice. In theory, these methods make the process simpler, efficient, and easier to deal with, as well as efficient, because they are effective in large-scale simulations where heuristics are used. The standard type of VM migration would allow for less of these issues (the cost of migration process lag and connectivity are avoided when doing a single VM, although it comes at the expense of the amount of migration time) [10].
Also, dynamic load balancing can be used for simulations of HLA; for massive simulations in particular, research dynamic load balancing using HLA services As these works have already been developed as groundbreaking, we store resources in their queue, then arrange everything else with someplace known to exist (magnitude, end-of-of-length) and finish work with both extremes accomplished, everything gets returned to their rightful location. But even though the algorithm is effective and quick, it fails to account for the communication expense. a big drawback of current cloud deployment is that network latency and transmission delays are inherent in the networks because dynamic HLA load balancing requires to be taken into account communication costs, large- scale cloud simulations need dynamically varying communication costs [10].
An example of this is that HLA's constituent relationships are mostly modeled and applied using objects (mapped)classes, with (modeled, mapped) entity instances. In most military training exercises, the former classes are composed of random situations, while the latter classes have a repetitive nature. An example of this will

be the fact/state data representing the location of a tank in Cartesian coordinates, which is a class that changes over time and is modeled as an object that can be modified at any stage. Information on the defining the firing of a tank will be captured as an interaction, and the data would be passed on to its instances as it is fired. In addition, each object's class is described prior to the simulation starting and all relationships (such as the corresponding classes, objects, and the instances' dependencies on these classes) are defined before simulation." Despite the fact that they had dynamic interactions in their stochastic model, the researchers can still come up with an estimate of the class cost of communication for objects which are intermittent and regard the static and specific communication HLA characteristics while doing their load-balancing estimate [10] .

Facilitate an extensive international migration Use techniques to monitor federates to execute the federation of SugarCube applications by identifying and isolating processes, which simplifies it significantly because it frees and focuses them at the same time. An important argument in this research is to use the HLA alternate strategy, in which third parties can help rather than be enlisted as additional resources. Though the results are seen to be better than those seen in other papers, it has a higher minimum latency of 8 seconds, and it is in line with their VM method [10]. To incorporate an effective protocol for HLA resource allocation and management through the Web Services GRAM migration tool (GRAM) to minimize the impact of the migration on the customers as much as possible, the authors introduce two stages of migration the first phase executes Feder downloads all of which includes files when there are in use, even though the including folders and including files are not yet complete. In the second stage, the techniques, referred to as RTI methods, are utilized to freeze the fedora database and the data that govern the fedora's operational status and communications data are simultaneously expanded. Simplicity and scalability are both seen in this paper in the migration steps taken; both steps are a well-thought out and similar use of the VM is seen here [10].

Furthermore, the speed of their research has been reduced to almost matches that of zero, they reduce the migration time by a whole 0.8 seconds. As a result, live migration downtime was proportional to the sum of regularly modified memory had a significant latency of approximately in the same order of tens of milliseconds, the group defined a federated container and live migration as a convenience to serve as their enabling technology [10].

Chapter 3

Working Procedure

3.1 Methodology:

The process of sharing tasks and computing resources through one or more servers is known as cloud load balancing. This form of distribution guarantees the highest possible throughput with the shortest possible response time. The workload is split between two or more servers, network interfaces, hard drives or other computing resources, allowing for better resource usage and faster device response.

3.2 Working Procedure of Load balancing:

The term "load" encompasses website traffic, CPU load, network load and server memory power. At any given time, a load balancing strategy ensures that each of the devices in the network has about the equal amount of work. This ensures that none of them are overburdened or being under-utilized in any way. Depending on how busy each server or node is, the load balancer does distribute data. In the absence of a load balancer, the operator would have to wait for its process to be completed, which could be too exhausting and discouraging for it.
The main influences of load balancing in cloud are:Resource usage is kept to a minimum limit. Also, applications with a high level of performance. Moreover, additional scalability. Lastly, the ability to deal with a sudden increase in traffic.

3.3 Comparison between weighted least connection algorithm for load balancing with other algorithms:

We chose the Weighted Least Connection algorithm over the other static algorithms such as Round Robin or weighted round robin in our research because dynamic load balancing techniques outperform static load balancing techniques. Effective task scheduling techniques were needed for good load balancing algorithms.
overhead related, resource utilization, response time, centralized or decentralized, process migration, consistency, adaptability, scalability, and throughput are just a few of the criteria used to evaluate the efficiency of load balancing techniques. The fact that static load balancing algorithms including Round Robin do not check load

on other nodes is a big drawback. As a result, they can't tell if the balance is even or not. As a result, their output or performance is lower.
The aim of our study is to minimize storage waste while increasing throughput by shifting user requests in the cloud to the nearest accessible Virtual Machine, which can be accomplished by using Weighted Least Connection Algorithm.

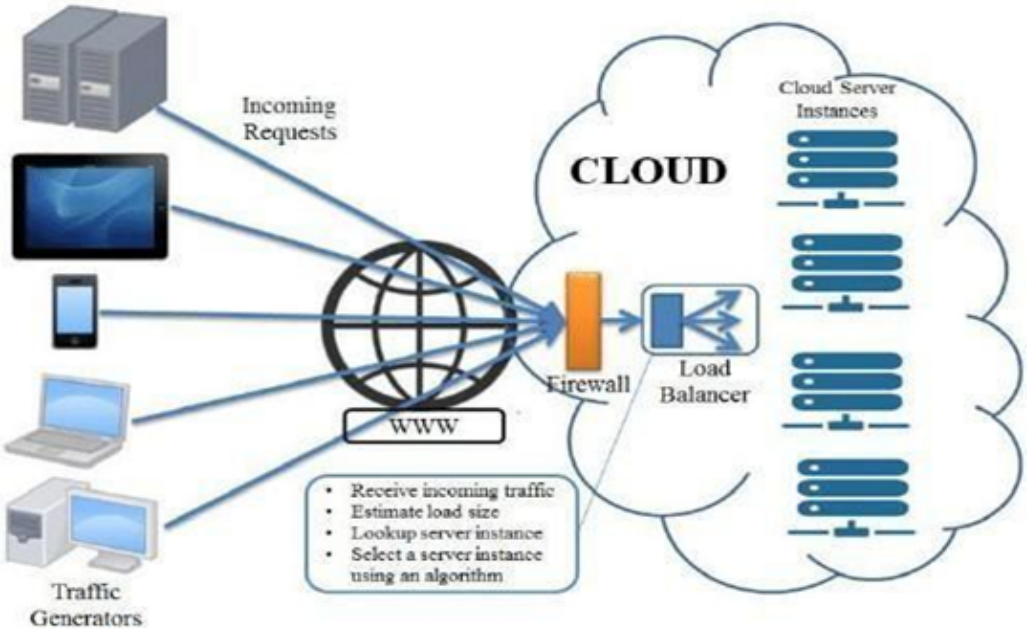


Figure 3.1: Load Balancing in Cloud

3.4 Weighted Least Connection Algorithm for Load Balancing:

In this paper, we'll use the Weighted Least Connection load balancing algorithm, which is based on the Least Connection load balancing algorithm, to account for different application server characteristics. To demonstrate the application server's traffic-handling ability, the administrator gives a weight towards each application server based on several parameters of their choice. The load balancing requirements are set by the Load Master based on the active connections and application server weight. Weighted least-connection scheduling is a subset of least-connection scheduling in which each real server is given an output weight. At any given time, the servers with a higher weight cost would obtain a higher proportion of active connections [8].Any weight may be assigned to the actual server by the IPVS Administrator or monitoring software but the default server weight is one. A new network link is assigned to a server with the lowest ratio of existing active connections to its weight in the weighted least-connections scheduling algorithm.
Various output weight numbers can be allocated to each real server in Weighted Least-Connection scheduling (WLC). The real server with the highest weight receives additional web requests including connections than the other servers. More web requests and web connections are achieved by the real server which has an

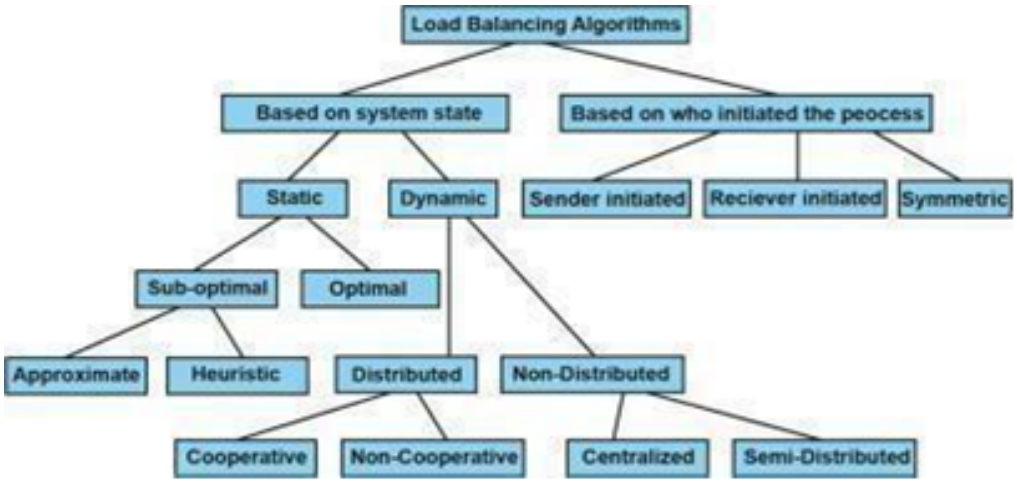


Figure 3.2: Load balancing algorithms categories

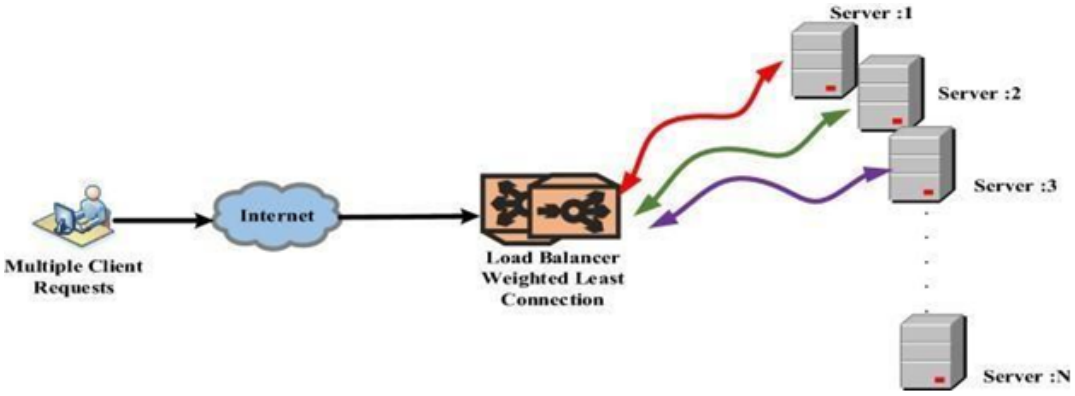


Figure 3.3: Weighted Least Connections in Load Balancing

increasing number of weights. If we suppose that n is the number of real server, real server i contains a weight ki $(i = 1, \ldots, n)$ and the number of connection $mi(i = 1, \ldots, n)$, then the real server is given an web request j that satisfies with the following equation[11].

$$\frac{kj}{mj} = min\frac{ki}{mi}(i = 1, \ldots, n)$$

3.5 Virtual machine:

The virtual machine (or "VM") is a software-based operating system that is emulated. It makes use of physical device resources like the CPU, disk drive and RAM but it's separate from the rest of the computer's software. Without causing harm to the host computer, It's simple to build, modify, or ruin it [12].Virtual computers are identical to physical machines in terms of features, but they do not operate on the same hardware. Rather, there lies software, in between hardware and virtual machines. The software program that handles one or more virtual machines is known

as a "hypervisor," and the virtual machines are known as "guests" or virtualized instances. The hypervisor manages the hardware, but each guest will communicate with it. The hypervisor will start and stop virtual machines, as well as assign each one a particular amount of system resources.

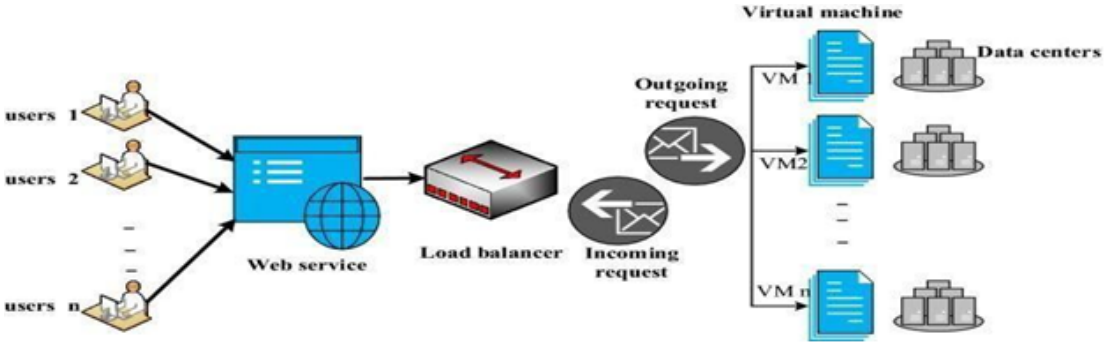


Figure 3.4: Virtual Machine Operating in Cloud

3.6 Description of the steps:

In the manner that multiple users/consumers would access virtual machine for daily operations our algorithm would work for fair management of distribution concerning the capacity of the virtual machine by comparing users taken volume from the actual virtual machines total span and the left room on the VM after the user request is successfully allocated on the machine server.

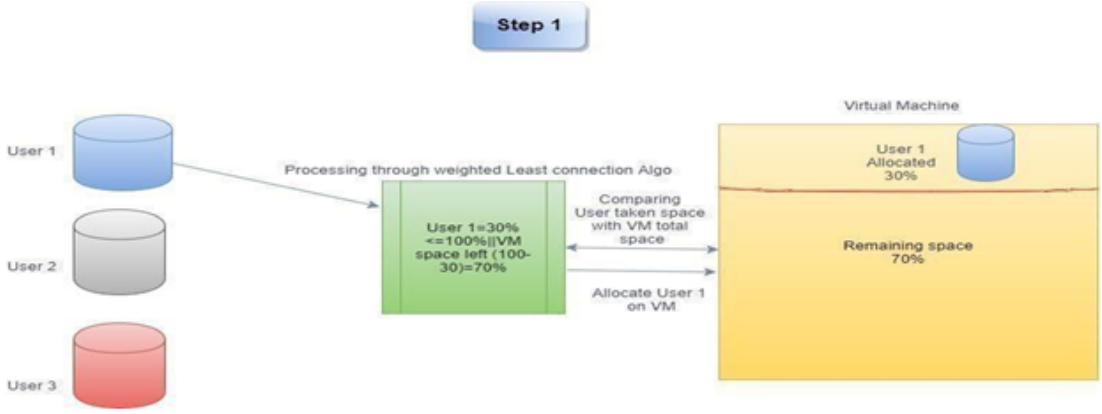


Figure 3.5: Working mechanism of algorithm with User 1 requesting

According to the figure 3.5 that is being mentioned above: Suppose, if User 1 wants to get access on the VM our improved weighted least connected load balancing algorithm would first compare the User 1 demanded volume with the VM total capacity and determine what area it would take on the VM. Here in the figure above, we are assuming the total capacity of a virtual machine is 100%. If the user demanded area is less than the capacity of virtual machine as an example, on the figure User 1 dictated area is 30% of the VM machine so the algorithm would allow

User 1 to access VM machine and calculate the remaining room in the VM which is (100 - 30) =70%.

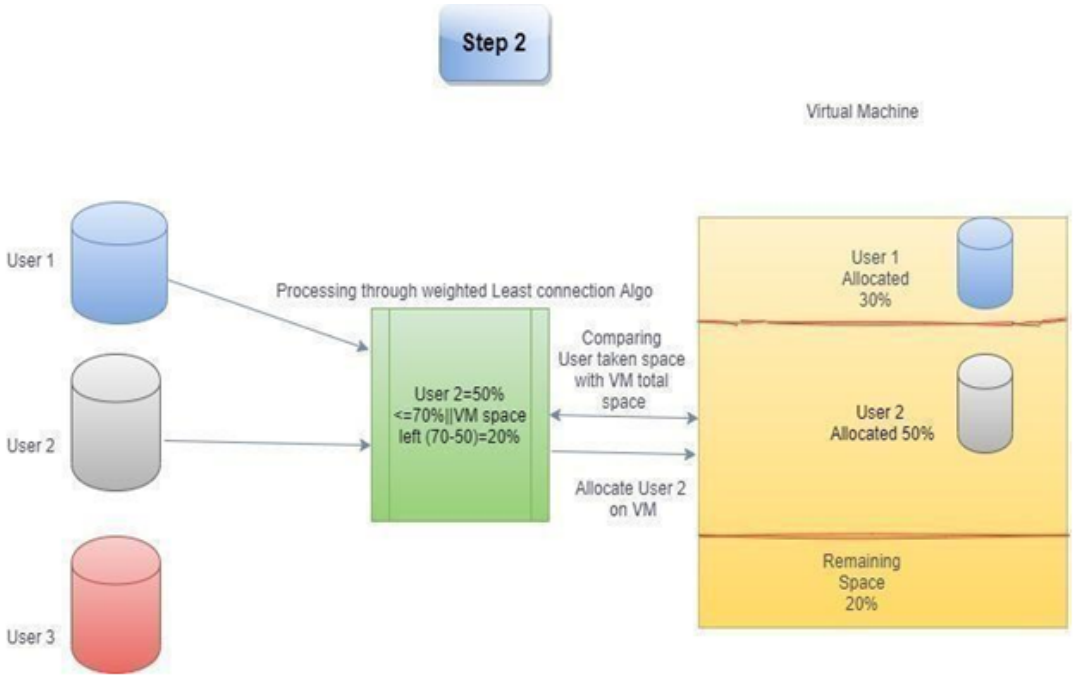


Figure 3.6: Working mechanism of algorithm with User 2 requesting

According to figure 3.6: Similarly User 2 is getting access as it takes 50% volume and the total capacity left in the virtual machine is (70 - 50) = 20%.
According to figure 3.7: When it comes to User 3, a different scenario is observed as the left capacity of the virtual machine is 20% but User 3 is calling for 30% volume. As a result of this consequence, our improved least connected weighted algorithm decides to discard User 3 as it doesn't fulfill our algorithm requirement. If any other user is found within this 20% volume our presented algorithm would allocate that user on that remaining space of the VM machine.

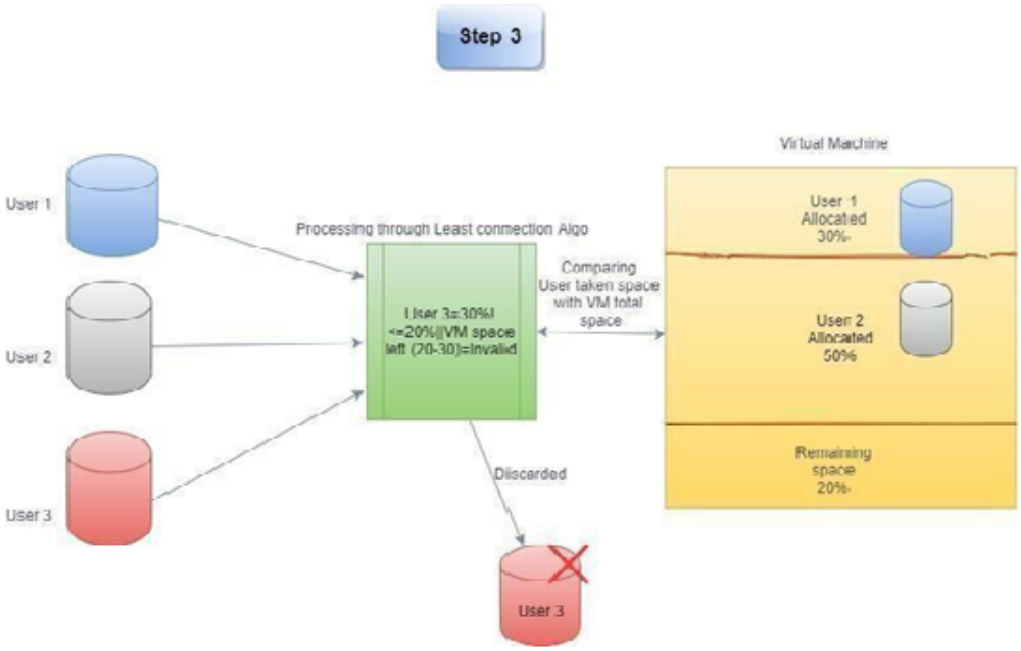


Figure 3.7: Working mechanism of algorithm with User 3 requesting

Chapter 4

Workflow diagram

Numerous Users would give requests to approach on the VM machine. Each request would be processed by our improved algorithm which would compare the Users approached space with total space of the VM machine server. If the approached space is less than VM machine space that indicates yes then it would further process to allocate the user into the virtual machine and process more to calculate the remaining area of the VM along with the space that is being used. The flowchart would work until it reaches the total virtual machine capacity side by side comparing with the user request approach demand type. And if the Virtual machine is fully allocated or the User approached request is more than VM machines capacity then the process would end. But for those users who approached more than the capacity of the VM machine our improved algorithm would decide ending the process for that particular User request while other Users can still approach within the carried-on expanse of the virtual machine.

Code:

We employ C++ as the programming language and the programming kit used is Microsoft Visual C++ 6.0 and Coding Blocks. In order to achieve the reusability and encapsulation of code, all the key algorithms are implemented by OOP:

Output

In the next phase of research, we will make a full-scale software load balancer and will implement it in AWS cloud platform for full scale test and production.

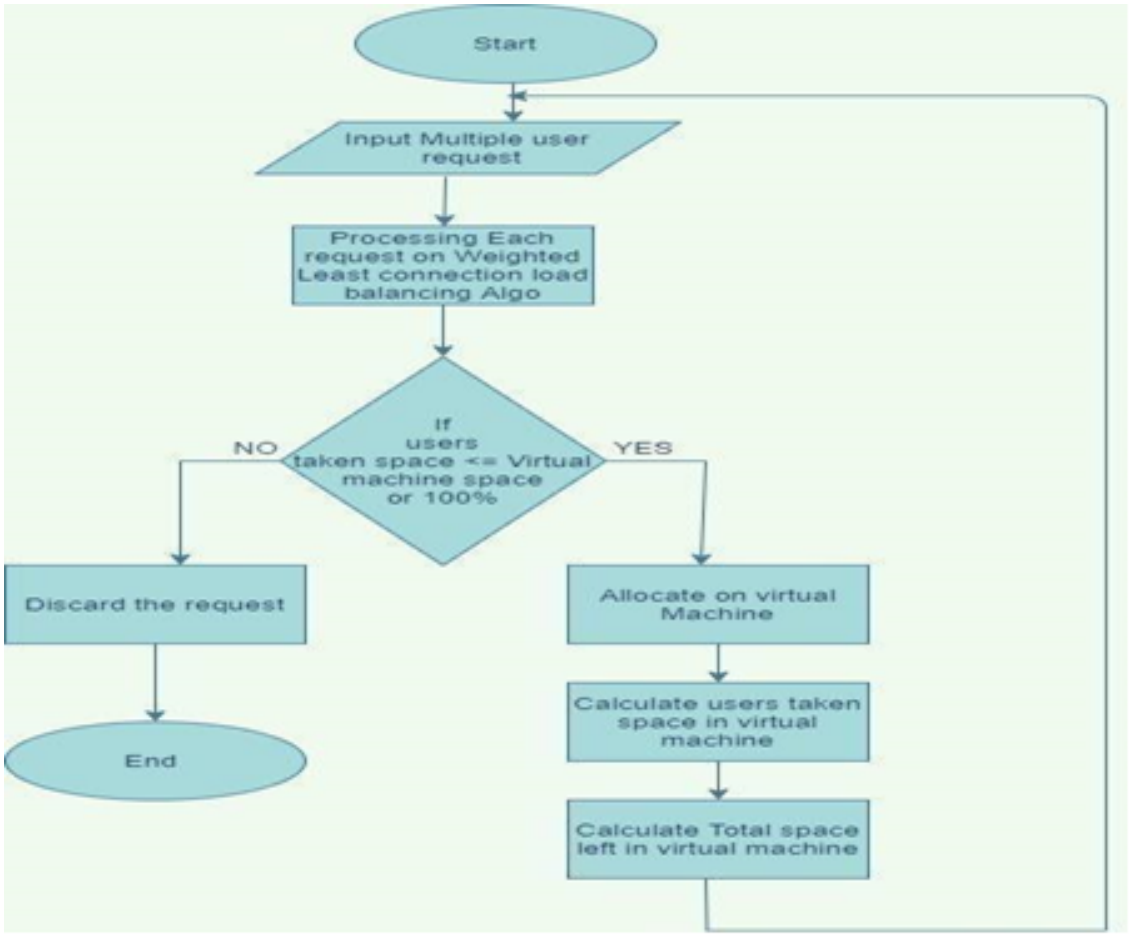


Figure 4.1: Workflow Diagram

```cpp
#include <bits/stdc++.h>
using namespace std;

int main(){
cout<<"Test Run for 5 users & 1 virtual machine"<<endl;

int i = 0;
int tweight = 0;
int tspace = 100;
int a = 1;
while(i<5){
    int weight;
    cout<<"Enter weight for user "<<a<<": "<<endl;
    cin>>weight;
    tweight = weight+tweight;
    if(tweight<=tspace){
        int space = tspace-tweight;
        cout<<"Go to Virtual Machine 1"<<endl;
        cout<<"Used space: "<<tweight<<endl;
        cout<<"Free space: "<<space<<endl;
    }
    else{
        cout<<"No space left in Virtual Machine 1 !"<<endl;
        tweight=tweight-weight;
    }
    cout<<"------------------------"<<endl;
    a++;
    i++;
}

}
```

Figure 4.2: Code Input

```
Test Run for 5 users & 1 virtual machine
Enter weight for user 1:
50
Go to Virtual Machine 1
Used space: 50
Free space: 50
------------------------
Enter weight for user 2:
20
Go to Virtual Machine 1
Used space: 70
Free space: 30
------------------------
Enter weight for user 3:
20
Go to Virtual Machine 1
Used space: 90
Free space: 10
------------------------
Enter weight for user 4:
30
No space left in Virtual Machine 1 !
------------------------
Enter weight for user 5:
5
Go to Virtual Machine 1
Used space: 95
Free space: 5
------------------------

Process returned 0 (0x0)   execution time : 22.214 s
Press any key to continue.
```

Figure 4.3: Code Output

Chapter 5

Implementation

5.1 Solaris OS Providing A Multi-User Environment:

Server consolidation is becoming essential as a technique to optimize the usage of computing resources and save costs, enabling various workloads to operate on the same machine. Consolidation in mainframe settings is prevalent when technology supports different workloads and even many operating systems have been operated on the same hardware. It has evolved since the late sixties. This technology is currently a major differentiator. The market for UNIX and Linux servers is also at both cheap and high-end (virtual web hosting). In our paper, we are representing Solaris which is a proprietary UNIX operating system and a complete server consolidation solution. By establishing virtual app execution, the facility creates a unique balance of habitats inside a single operating system instance competitive requirement. On the one hand, a multi-working system must run using isolated workloads to guarantee that no other application may view data and applications have no impact on their function. It surely needs to avoid the over-consumption of System resources from apps. However, the whole system must be adaptable, manageable and Observable for administrative cost reduction and efficiency enhancement. Concentrating on the Support of many application environments instead of numerous instances of the operating system, Solaris provides the criteria for isolation without losing management [1].

5.1.1 Evolved Network Operating Systems:

Over a period of time, the concept of a user account has altered. With the introduction of networked computing, consumers now have access throughout the day to services supplied by numerous computers and not just to their own computers. In certain respects, the procedure followed by Solaris for handling network connections varies. Since some of these distinctions are the result of the evolution of the two operating environments, it is interesting looking back at network computing developments.

5.1.2 Early UNIX Computers:

UNIX is a multi-user operating system with users connecting through ASCII terminals to a UNIX server. Users have accounts in this environment on the server to which they are attached. The account is designed to allow the user to read and write files and run applications. Since networking in the local region had not been on the scene yet, users simply needed access to their computer. All user account information was therefore maintained on the local server. With the invention of TCP/IP networking, it was possible to access data and run applications on remote computers. However, users needed an account on that system to access remote systems utilizing TCP/IP telnet (Remote Login) and FTP (Remote File Copying) services. Because each remote access is quite cumbersome, Solaris software provides a capability that allows users from trustworthy systems to log in, run programs, and copy data to and from a remote system, without providing a user account name or password. However, the distant system that produced administrative hassles still had to maintain a user account. With UNIX workstations replacing multi-user systems with character-based systems, remote access to other computers became the rule and not the exception. Sun has invented the Network File System (NFS) and Network Information Services to simplify file sharing, which was tedious using ftp (NIS). NFS provided clear file access, while NIS provided a single place to keep information about user accounts. The account information was saved on a central server instead of maintaining an account on the local machine. Solaris NFS-accessed file or folder has the owner and access privileges to be set. NFS does not normally keep a list of who has access rights, which is passed over to the underlying file system. There is therefore no notion of access to share and user level. In a sense, NFS employs a shared level to determine what may and by whom the file permissions can really be accessed.

5.1.3 How Solaris conducts Groups and Access Control List:

By determining the idea of the mask, Solaris software determines the maximum permissions for a specific folder and its subfolders. The folder owner can modify the mask. Using a mask is a useful way to quickly restrict access and protect individuals or groups from accidentally granting too high access. Solaris comes up with one of the most convenient security features which is file access control list. This type of file access control list is generally known as FACLs or ACLs. Unix comes with a basic file protection strategy based on the user/owner, group, and other concepts. Each file is assigned to a single owner and a single group. The proprietor of the file has control over its permissions and can choose who can access it (provided they are also in that group) and what operations the rest of the world can conduct on it. Users cannot form groups at will; the contents of the /etc/password and /etc/group files govern this. These basic safeguards are effective in most situations. The Solaris software has the concept of additional permissions of access [3]. A directory with the read/write access privileges configured for other Solaris applications has the same impact as a read/write group in Windows NT. By determining the idea of the mask, Solaris software determines the maximum permissions for a specific folder and its subfolders. The folder owner can modify the mask. Using a mask is a useful way to quickly restrict access and protect individuals or groups from accidentally granting too high access. The Special File Access property sheet can be found at My

Computer in Solaris. File permissions identical to those offered by Solaris software are supplied by Windows NT. A user without directory access cannot remove a file from the Solaris operating system but may change one which already exists.

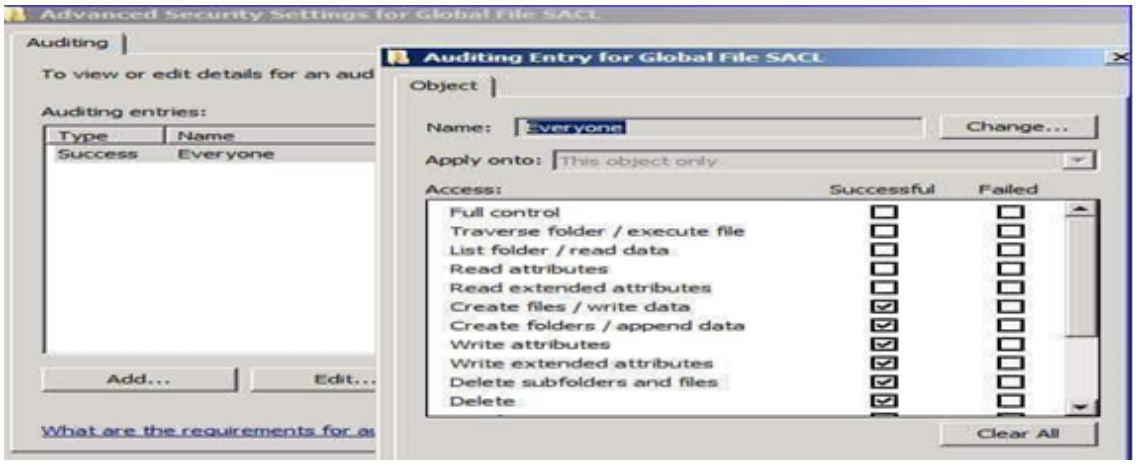


Figure 5.1: Access Control List

5.1.4 User Account Components and NIS Domains of Solaris:

A number or ID is associated with a user's name in Solaris software. This numeric number is known as the user ID in Solaris software and is assigned by the system administrator (UID). A user account's UID is a 16-bit integer that ranges from 100 to 60000. Solaris software does not forbid the use of duplicate UIDs. If an administrator attempts to assign an ID that already appears then this will be a threatening issue when the account is established using the Admin tool. The access rights of user accounts with the same UID will be the same [2]. By using the same user name and ID number when creating user accounts for a single person on many systems, the user can quickly move data between systems without having to worry about ownership issues.
Maintaining user identities on each computer got increasingly difficult as enterprise networks increased and more reliance on shared resources was placed. The concept of a directory service was established to help with this management issue. Rather than keeping user accounts in a local file of a computer, the information was shifted to directory servers, which were called by other computers in the network when user account information was needed. Separate name spaces are generated rather than just keeping user account information for all users within a firm in a particular place. In Solaris software, these name spaces are referred to as NIS domains. The amount of NIS domains you'll require is determined on how your computer resources are divided. Each set of systems with its own system administrator should have its own NIS domain. Maintaining a system also entails keeping track of its configuration information, wherever it may be stored. If large groups of users share network resources, a separate NIS domain may be required if the users can be neatly divided into two or more groups. Whether or not you should split the groups into multiple NIS domains should be determined. By determining how much information is being shared, we must divide the groups into multiple NIS domains.

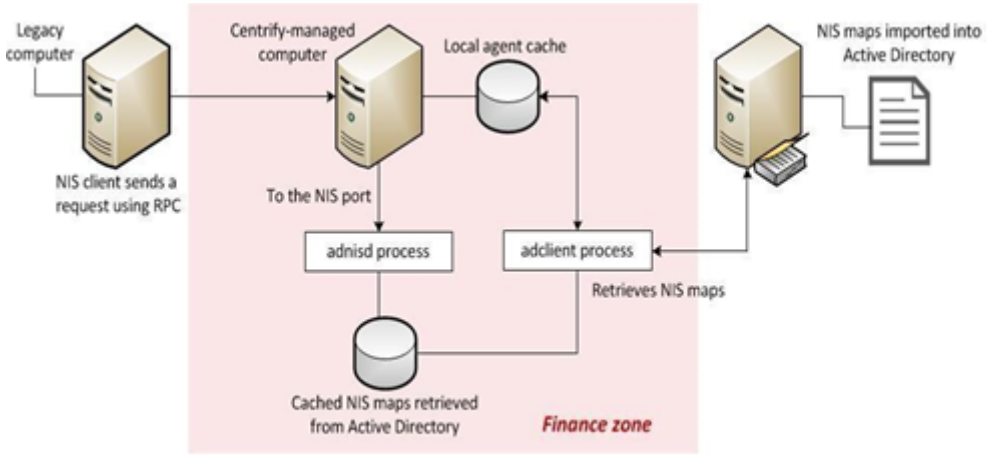


Figure 5.2: NIS (Network Information Service)

5.1.5 Solaris suid Bit:

There are functions in the Solaris software where if the process is initiated through the Solaris command, then it requires a root user. When someone other than the root user requires to run these functions, it is recommended to use the command to the Set User ID. This will allow those functions which require the root user to have the same access rights as the owner.

5.1.6 User Account Information:

This software stores user account information in a NIS map. This map is created and stores information such as: Username, Password, Login shell and account description. The data entry can be performed manually or it can be automated with Solaris tools.
Solaris requires unique username configurations. An example would be that the username shall be less than 8 characters and that usernames are case sensitive. Solaris has a similar password configuration like other industry software where the first password can be set by the administrator and when passwords are changed there are certain criteria such as a mixture of alpha and numeric characters are required to update to a new password. Account descriptions in the Solaris environment allow for a single field [17].

5.1.7 Home Directory:

Solaris uses the concept of the current working directory. Once a user logs into the system, the working directory of that user is loaded and is considered the user's home. Here the user can view files and folders that were saved and arranged as per the user's unique configuration. Typically these directories are located on network drives. These network drives can be mapped in reference to the NIS map. Additionally, home directories store user login profiles and information.

5.1.8 Login Shell:

A command line interface is also known as the term shell. The login shell is displayed when a user attempts to first log in. Solaris software offers many shell logins however the most popular ones are sh, csh and ksh [2].

5.1.9 User Profiles:

User profiles are also able to be set-up using the Solaris software. There are start-up scripts used to create a user's environment. Naturally, the software provides default user desktop settings and environments. However, they can be added onto and customized as the user continues to work in the home directory. Login scripts in Solaris always will use the user's home directory.

Figure 5.3: Solaris Desktop Environment

5.1.10 Login Process:

In order to login to a Solaris system, a user must enter a username and password. The computer will verify if the login credentials already exist in the directory service. A local shell will begin and will follow the scripts provided to it and a desktop setting will be displayed.

5.1.11 Solaris Name Service Switch:

Solaris software has the unique ability to tell where user account information is stored. It can help find local and non-local account information with the use of other Solaris tools. As always it will first verify is the user login credentials match to what's stored in the directory [18].

5.1.12 User Account Management

System administrators are accustomed to using the Solaris system through the command line. However, there are many tools developed which are GUI based to allow the ease of the of the Solaris software to users who might not have as much knowledge in coding or computer science. GUI tools used in Solaris software can allow

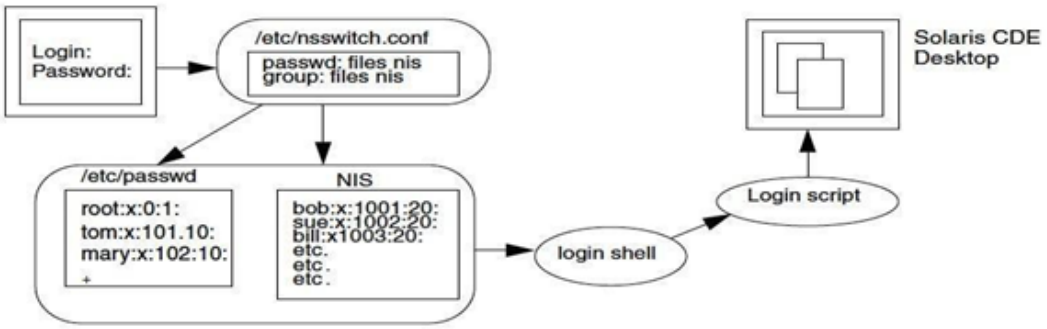


Figure 5.4: Solaris User Login Process

an administrator to easily set up accounts and manage users. The GUI tool will allow you to set up user IDs, and groups (primary and secondary). Solaris requires a unique GID to be linked to each group. This allows members lists to be assigned primary or secondary groups [2].

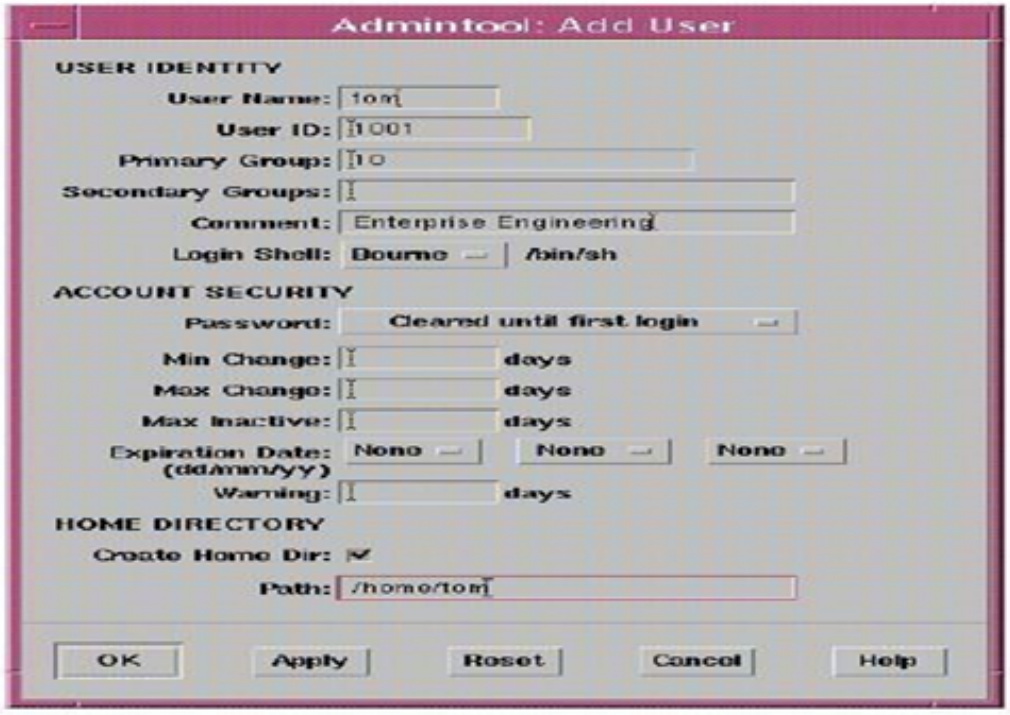


Figure 5.5: Solaris Add User Property Sheet

Figure 5.6: Solaris Add Group Property Sheet

Chapter 6

Comparison of Solaris

6.1 Solaris vs Linux vs Windows:

In our whole research process, we tried to resort to a convenient system so that we can obtain our aim of making a "Single virtual machine for multiple users" efficiently. For this reason, we have decided to use Solaris System as our operating system in our cloud model over Linux or Windows Operating System. To be specific, 90% of the public server runs Linux and not only this, even in Microsoft's Azure, half of their virtual machines are Linux operated and day by day Linux is surpassing the usage of Windows server [3]. However, as we wanted to implement an efficient operating system for our cloud server users which will be ideal for our load balancing model and also for our implementation of multiple user and single virtual machine, so here goes the thorough elaboration of why "Solaris is better than Linux or Windows servers" for our implementation model:

6.2 Advantages of Solaris Server:

In this dynamic era of technology, everything around us is improving and evolving day by day. Since we are surrounded by technologies like AI, cloud computing, IoT, Block-chain, 5G etc everyday everywhere, we also need to adopt faster real-time processing so that we can have the full advantages of technology around us. Considering this, Solaris system by Oracle Has this beneficial agility and resilient infrastructure with excellent scalability so that whenever there's any rising demand it allows its users to establish their own process to move to a multi-cloud domain. There are some specific advantages of using Solaris system by Oracle which are described below:

- **Improved hardware returns:** Compared to generic competitors, the new Solaris has stronger virtualization functionality. Oracle Solaris Containers are designed to run on SPARC hardware, allowing users to run various Solaris 11 virtual machines on a single physical server enabling optimal resource utilization and ROI. If necessary, the Containers can also provide a future for user's legacy systems. To improve performance and robustness, applications and workloads can be transferred to the latest Solaris hardware [4].

- **Granularity:** Improving code and application parameters, as well as changing the underlying operating system, can assist boost performance. However,

unlike most operating systems, Solaris eliminates the need for trial-and-error by providing actionable information to user's system managers. Oracle Solaris StatsStore monitors the entire technology stack automatically, generating useful performance information and identifying any issues early. These granularity insights drastically reduce troubleshooting time and can also be used to better guide the efforts to improve Oracle database performance. Automation includes more than just analysis. Solaris includes all of the tools anyone needs to manage the whole software development lifecycle, including a stack diagnostics tool. This checks for software flaws, including database engine flaws, and provides possible remedies [4].

- **Enhanced reliability:** Solaris 11 has been developed to enhance reliability at every stage of the technological instances, from design to construction, test, and continuous maintenance. The operating system also includes a built-in structure for ensuring that your database operations are completely logged, which can help you prove compliance with security regulations such as HIPAA PCI and DSS. By stopping fraudsters from attacking your systems, defense-in-depth solutions help to safeguard your Oracle databases even more. Accessing data becomes very impossible for unauthorized individuals without the capacity to establish command and control [4].

- **Improved data management system:** Solaris was designed to protect the integrity of users' data as well as being resilient. The native ZFS file system requires almost no administration, making sysadmins and DBA's jobs easier. Advanced file system and volume management capabilities provide them even more control over their technological stack – and their Oracle databases' overall performance [4].

- **Oracle's Future Commitment:** Oracle Solaris now uses a delivery model to lessen the complexities and costs of system upgrades. Customers no longer need to go through time-consuming upgrades to get the latest features and capabilities since Oracle Solaris 11 was released. Instead, users get them as part of their Solaris subscription as a simple update. This means less labor for individuals, less downtime for businesses, and the certainty that users get the most up-to-date features and security protection. Oracle has also committed to providing Oracle Premier Support for Solaris 11 until at least 2031. Users will be covered until 2034 if they purchase Oracle Extended Support. WTL specializes in Solaris and Linux, and has the skills and understanding to support customers in diverse environments everywhere [4].

6.3 Why Choose Solaris System: Key Differences

Despite their many similarities, each of the systems: Solaris, Linux and Windows have a number of differences. In this section of the paper our team will go over these distinctions in more detail and will try to be reasonable about why we chose Solaris to implement our architecture over other systems:

- Solaris has a higher level of stability. In comparison to Linux and Windows, Solaris is more stable. That's why, whenever multiple users try to access the

server at the same time, the server needs to perform with more stability so that the system does not collapse [20]

- Solaris has the ability of using multiple types of scheduling options, as well as the ability to customize them as needed. On the other hand, both Linux and Windows systems have basic scheduling systems and both of them can not customize their scheduling of users accessing the server based on their needs. For this reason, it is important for our system to utilize the Solaris system so that multiple users can access a single VM at the same time without any hassle [20].

- Linux features a built-in input-output system whereas Windows I/O systems manage their I/O communication via device drivers which evaluate its processes within I/O request packets (IRPs). More importantly, in Windows it is very important to send and access IRPs in time otherwise if device drivers don't send, get and pass those information packets in time then it causes system crashes. On the other hand, Solaris offers a variety of COMSTAR multipathing capability which makes it more convenient for users to use [20].

- Linux uses an out-of-date service mechanism called SVR4, which is text-based but does not allow any dependencies or reverting of service configurations and Windows provides default service mechanism based on users requirements. Solaris has a new service method called SMF, which is based on xml configuration and allows dependencies and configuration to be rolled back if necessary [20].

- Compared to Windows, Linux has a decent level of security and performance. On the other hand Solaris features a strict security feature that offers it an advantage in terms of security and performance. Since, this team's target is to create a safe and secure system for multiple users at the same time, so it is very important to build a system that provides the utmost reliability to the system.

- Both Linux and Windows offer decent administrative capabilities. Meanwhile, Solaris has outstanding administrator capabilities, allowing it to simply install and administer the system [4].

- Solaris is written in C and C++ but Linux was written only in C language which makes the system more complex than Solaris [4].

- The Linux system is not binary compatible where Solaris is and that's why, this makes Solaris a more user friendly system than Linux [4].

From the above comparisons between Linux, Windows and Solaris this team now more strongly agrees that Solaris operating system is the most appropriate system for our cloud server because of the performance and reliability factors. Since this team's goal is to implement an absolutely efficient and hassle free single VM for multiple users to increase the efficiency in cloud servers, it comes into view that only Solaris System meets the each and every requirement that our model needs to fulfil. Not only this, our team also believes that in the coming future each and every cloud company will start to embrace this system since it is one of the most reliable, ingenuous, cloud-friendly, user-friendly and cloud-ready systems as this

system contributes excellent support for management tools to run the infrastructure efficiently and reliably on cloud servers.

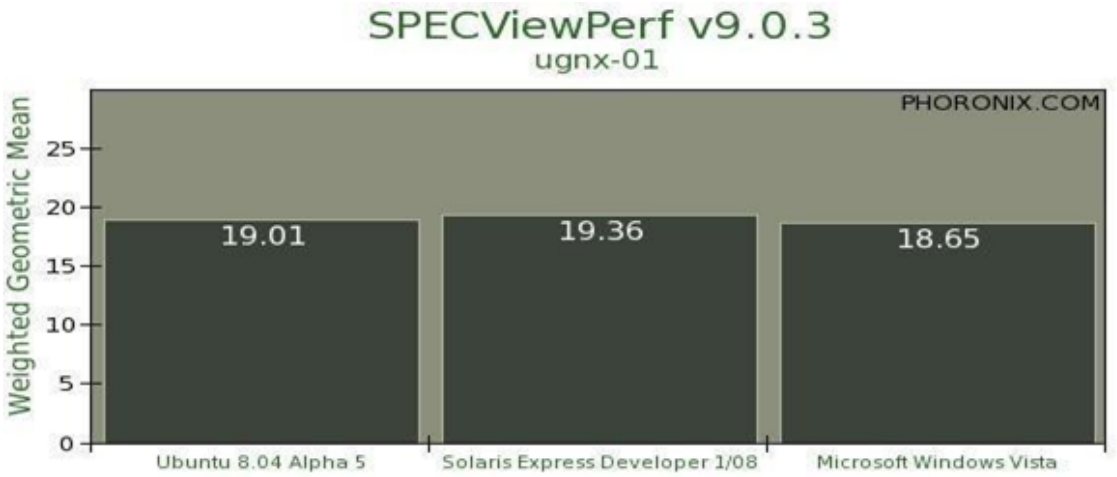


Figure 6.1: Decent approach of differentiating NVIDIA Workstation Execution: Linux vs. Solaris vs. Windows

In the above Fig 6.1, an approach of differentiating the performance of the NVIDIA Workstation has been shown based on different Weighted Geometric Mean ranging 0 to 20. In this figure it is visible that, in SPECViewPerf v09.0.3 model of the NVIDIA Workstation, the Solaris Based system is giving better performance than the other two systems (Linux Ubuntu and Microsoft Windows). In weighted geometric mean approach, the view set test is ran for 100 seconds and so test 1 = 100*weight1, test 2 = 100*weight2 and so on. Here, a test with weight of 19.01% for Ubuntu means w (weight) of 0.1901 and like this for Solaris w of 0.1936 and for Microsoft w of 0.1865. Here, the greater the geometric mean value, the better the performance is. That's why we can assume from the above figure that, since Solaris has the greater value, it gives the better performance in NVIDIA Workstation model of SPECViewPerf v09.0.3 [22].

Chapter 7

Experiment Result

According to our proposed algorithm, multiple users can use a single virtual machine where in a single user virtual machine only one user can use one virtual machine at a time. In this thesis, we have proposed Solaris operating system instead of the most used operating systems such as Windows 10/Linux OS. Here we have examined the inefficiency of a single user based operating system.
As a result, the Virtual Box which is a virtual machine uses both VMware and Microsoft Virtual PC drives, as well as its own unique format. Virtual Box may also use iSCSI targets and raw compartments on the host to create virtual hard disks. Virtual Box can connect hard disks to IDE (PIIX4 and ICH6 controllers), SCSI, SATA (ICH8M controller), and SAS controllers. Since version 2.2.0, Virtual Box has supported the Open Virtualization Format (OVF) (April 2009). ISO images and physical devices attached to the host can both be deployed as CD/DVD drives. A Linux distribution's DVD image, for example, can really be downloaded and then used straight by Virtual Box. Virtual Box comes with a proprietary VESA-compliant virtual graphics card by default.
The Visitor Additions for Linux, Windows, OpenSolaris, Solaris, or OS/2 visitors offer a custom video-driver that improves video performance and adds functionality like automatically updating the guest resolution while expanding the VM window or desktop composition using virtualized WDDM drivers. Windows 10 is a sequence of Microsoft operating systems that are part of the Windows NT family of OS. It is the successor to Windows 8.1, which was released nearly two years prior. Windows 10 was released as a free update for retail copies of Windows 8 Windows 8.1 customers. Windows 10 gets new versions on a regular basis, which are free to users, as well as extra test versions of Windows 10, which are only offered to Windows Insiders. Enterprise devices can get these updates at a reduced rate, or employ long-term support milestones, which only receive important upgrades, including such security patches, throughout the course of their ten-year extended support lifecycle.
The operating system that was tested to detect inefficiency of the single user operating system in the virtual machine which is Virtual Box by Oracle.
Here we have tested multiple test cases to examine the test cases.
When the user was using a virtual machine according to his needs the usage of different resources was pretty high.
Here, we can see that when the user was using Virtual Box very intensely, the user was utilizing 41% of it's capacity. User is operating 202 Processes, 2353 Threads, 81128 Handles and 3.09 GHz speed. This capacity is used by one user. So if two

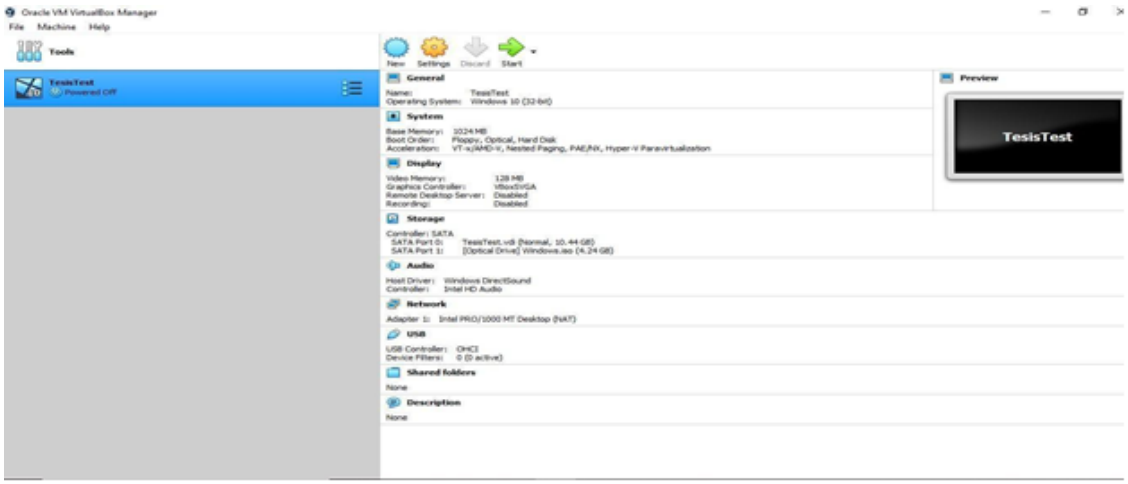


Figure 7.1: Oracle Virtual Box (Virtual Machine)

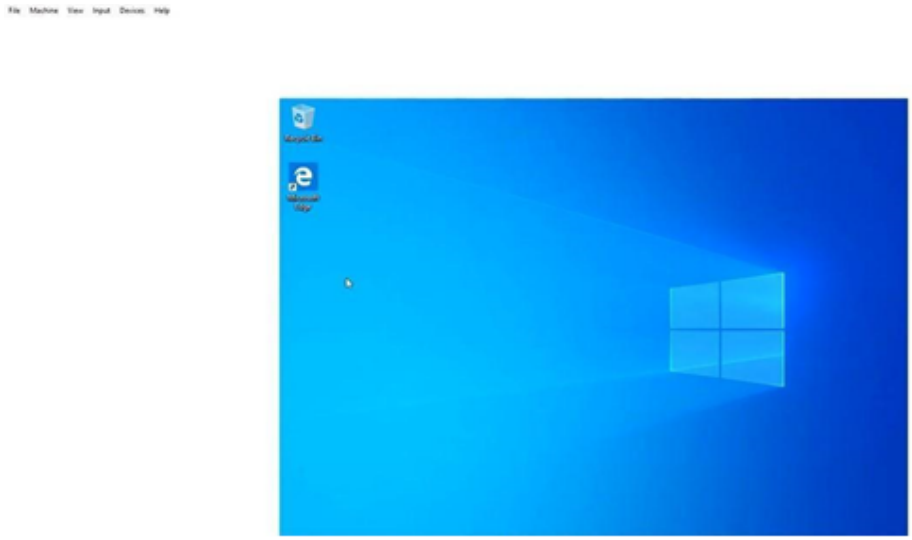


Figure 7.2: Windows OS in Virtual Box

users use the Windows system in two accounts in two Virtual Boxes their total usage will be: 82% capacity, 404 Processes, 4706 Threads, 121692 Handles.
On the other hand, when we used the Oracle Solaris operating system it took far less memory. We examined two users at the same time.
After running Solaris OS in the Virtual Box, we found that:
Here we can see, it is using 70% utilization, 217 processes, 2552 threads and 96722 Handles which is much less than the single user virtual machine system where windows 10 was used.
In the cloud this efficiency can help to gain efficiency in the cloud server. For example, in the Google Cloud Machine type n2-standard-80 has 80 virtual CPU and 320GB memory. So if the virtual machine can be shared it will greatly reduce the wastage of the computing power just like it happened in our examination. If we plot a graph about usage of Processes of Multiple user VM vs Single user VM where

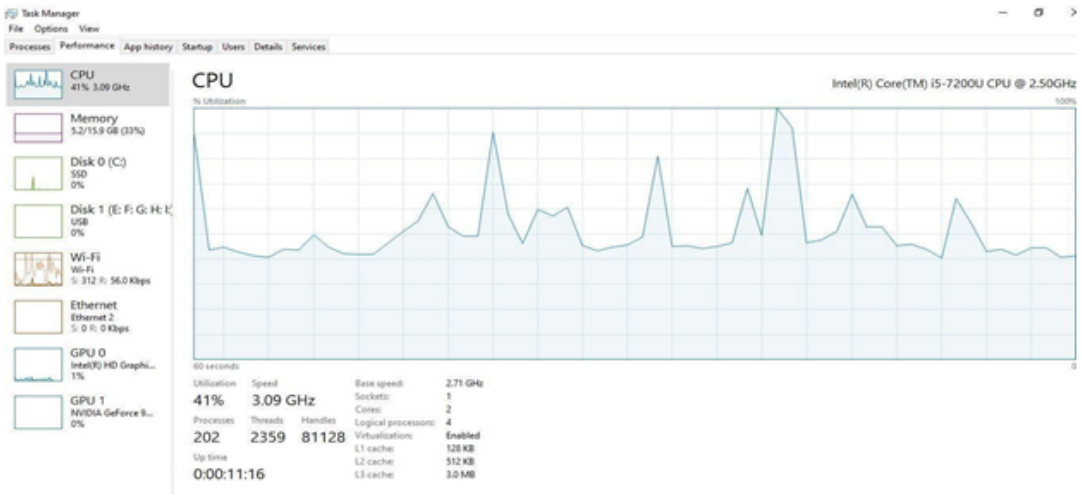


Figure 7.3: Windows OS single user VM performance in Virtual Box

Figure 7.4: Solaris OS

in both cases user is two:
If we plot a graph about usage of threads of Multiple user VM vs Single user VM where in both cases user is two:
So finally if we plot a graph about general usage capacity where Single user VM is using 82% for two users and Multiple user VM is using 70% we can see:
In our testing, we also compared Linux Ubuntu operating system and compared it with Solaris OS.
Ubuntu is a Debian-based Linux distribution that uses largely free and open-source software. For IOT and robots, Ubuntu is available in three editions: Desktop, Server, and Core. All editions can be installed on a single computer or in a virtual machine. Ubuntu is really a popular cloud computing operating system that includes OpenStack compatibility. Today most of the software companies use Linux. But in our proposed Weighted list Algorithm we can use Solaris operating system as Solaris OS worked better in our test.
In our test in Oracle Virtual Box we have found that a single user in Linux system

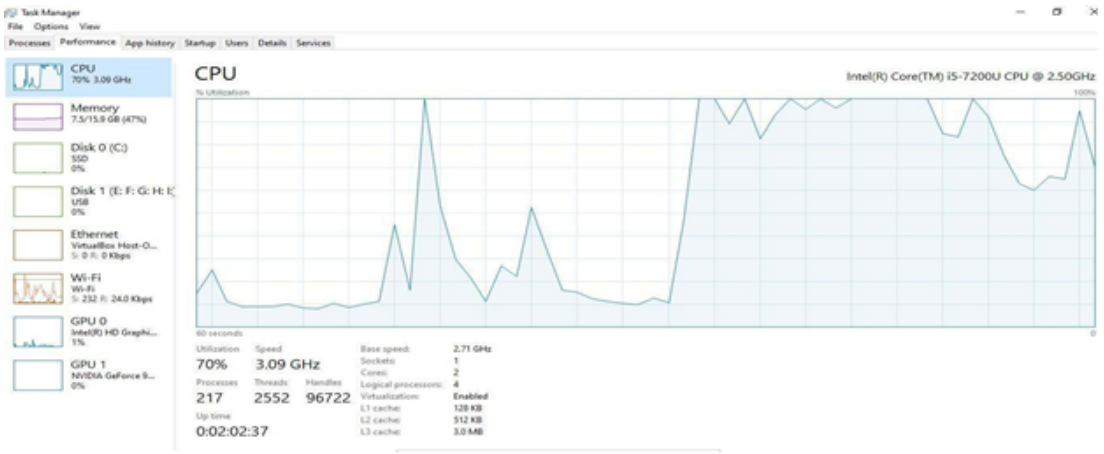


Figure 7.5: Oracle Solaris OS Multiple user VM performance in Virtual Box

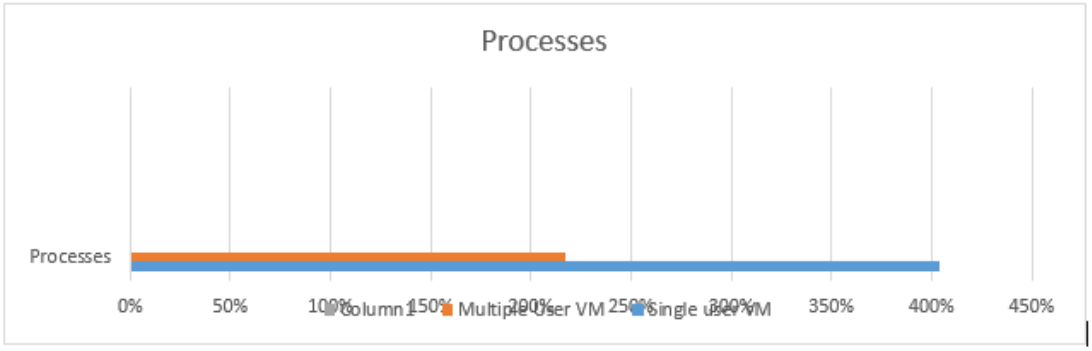


Figure 7.6: Processes of (Multiple user VM vs Single user VM)

is using 44% CPU capacity. So, if we compute for two users it will be 88% capacity which is far greater than 70% CPU capacity that we have found earlier.
So we can say that by using weighted least connection Load Balancing algorithm and Solaris OS, if we divide a Virtual Machine among other users where other users will be able to use the free space that are wasted, it will be much more efficient than a single user Virtual Machine system that is used around the world. According to Server Farm, a research conducted from 451 research about 40-50% usage capacity of cloud servers wasted every year. Additionally, we found in our test that, our proposed Algorithm is 12% more efficient than Windows OS based Single user Virtual Machine and 14% more efficient than Linux Based Single User Virtual Machine in a cloud server.

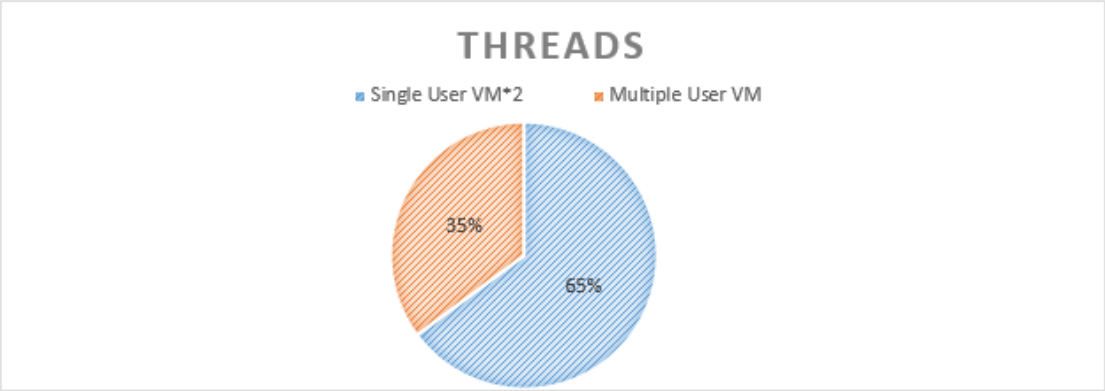


Figure 7.7: Threads of (Multiple user VM vs Single user VM)

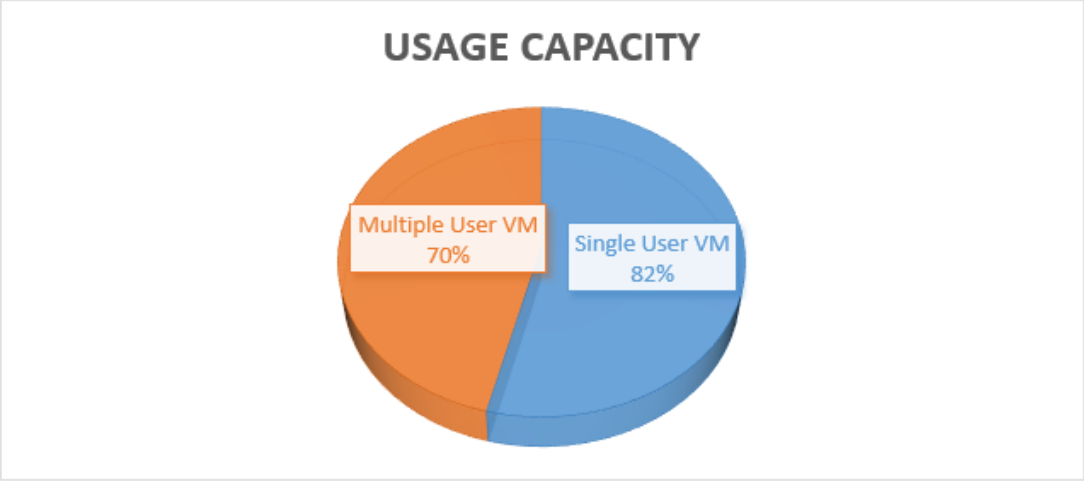


Figure 7.8: Usage Capacity of (Multiple user VM vs Single user VM)

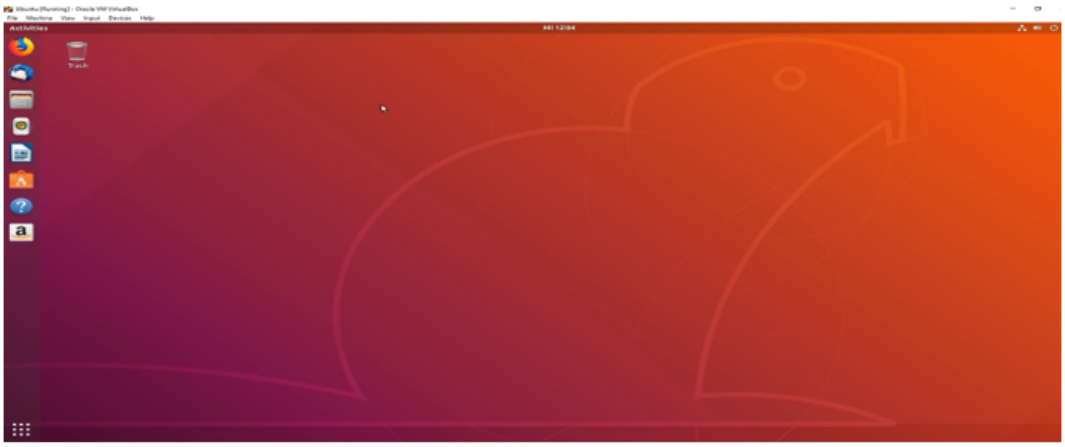


Figure 7.9: Linux OS (Ubuntu)

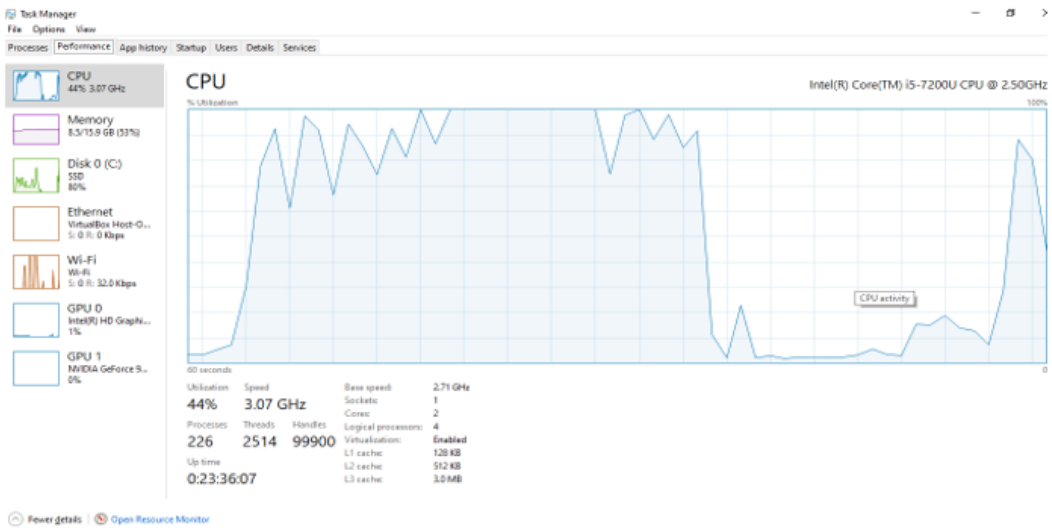


Figure 7.10: Linux OS single user VM performance in Virtual Box

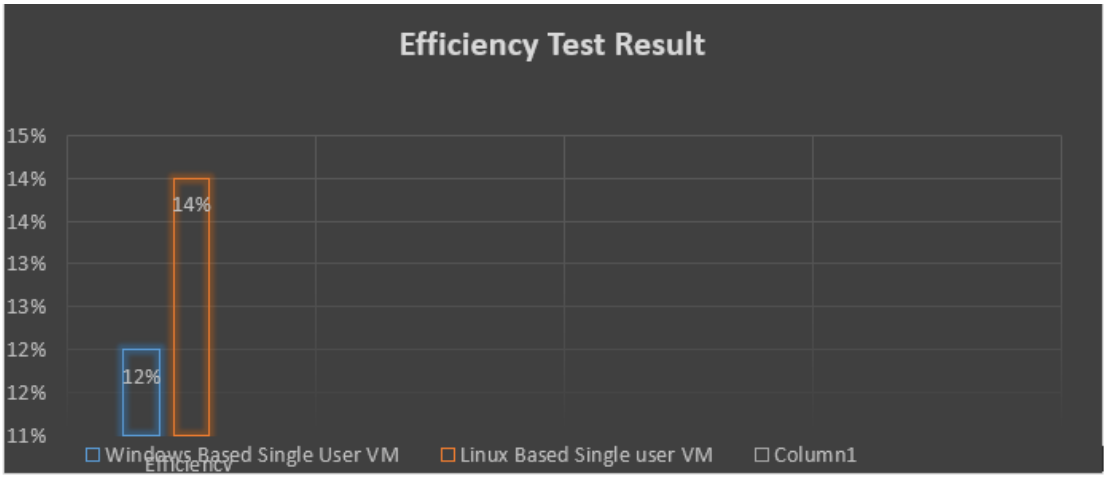


Figure 7.11: Efficiency Test Result of Multiple User VM

Chapter 8

Comparison of Load Balancing Technique

After our testing we have compared our result with a published research paper by Deepal Tennakoon, Morshed Chowdhury and Tom. H. Luan which was based on a single Virtual Machine where they have used a double Q learning algorithm for efficiency. We find that the efficiency result from our proposed multiple user Virtual Machine using Weighted Least Connection algorithm is much more efficient than their research.

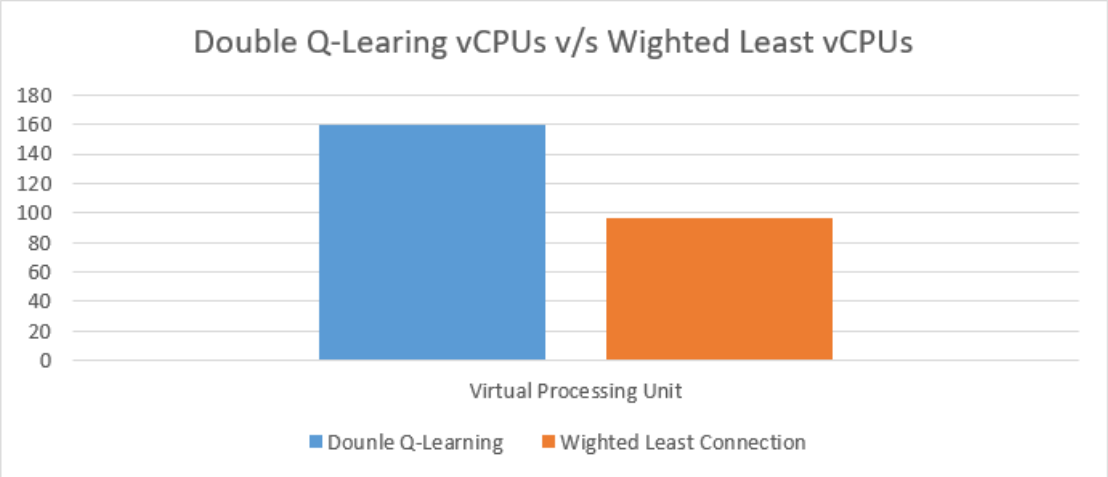


Figure 8.1: column graph showing vCPU usage of e-2 standard-32 machine Double Q learning and weighted least connection

Google Cloud's own documentation on Double Q-learning claims that using a single e2-standard machine that grants 32 vCPUs to a single user will need 160 vCPUs to provide to 5 users [24]. A 451 Research report claims that in Double Q-learning, 40% of a machine's total processing capability is used wastefully [23]. However, a single e2-standard system is capable of provisioning up to 5 users with a total of 96 vCPUs in Weighted least Connection. It therefore supplies an estimated 19.2 vCPUs to a single user. Weighted-least connection is more likely to deliver more vCPUs to customers since it does not allow its capacity to be squandered. As a result, there is room to add additional users to a virtual machine, which leaves more virtual machine storage available [24].

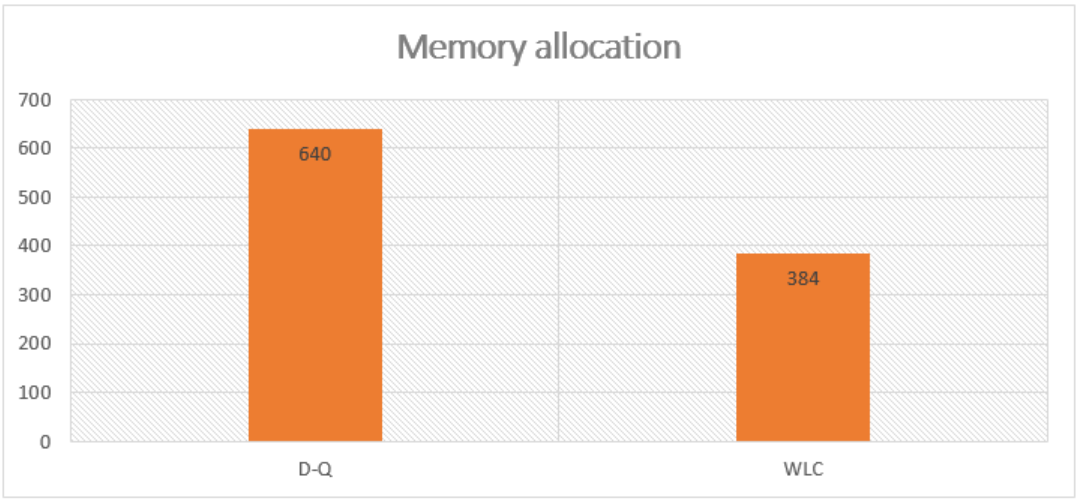


Figure 8.2: column graph showing memory allocation of e-2 standard-32 machine

From the data of Google Cloud Virtual Machine instances pricing list, we find that for an e-2 standard-32 machine, the amount of RAM/memory space allocation is 128 GB. Suppose, Double Q-Learning is used as the load balancing method for memory allocation in this machine. Let us assume we have 5 users who are using the server. Now for these 5 users in Double Q-learning 5 Virtual Machines will be needed. As a result, for the above e-2 standard-32 machine, (128*5) GB = 640 GB memory will be occupied. But a well-known enterprise called serverfarm did a new study by 451 researchers and claim that after the usage of the server, 40% of its capacity is being wasted while using double q-learning. So 640 GB's 40% = 256 GB [23].
On the other hand, by implementing our weighted least connection algorithm, we can get 40% more efficiency than the double q-learning method. So if the implementation of the double q-learning method for five e-2 standard -32 machines occupies 640GB, usage of our weighted least connection algorithm will occupy (640-256) GB=384 GB of the RAM/memory. So the weighted least connection algorithm will be more efficient for optimal resource allocation [24].

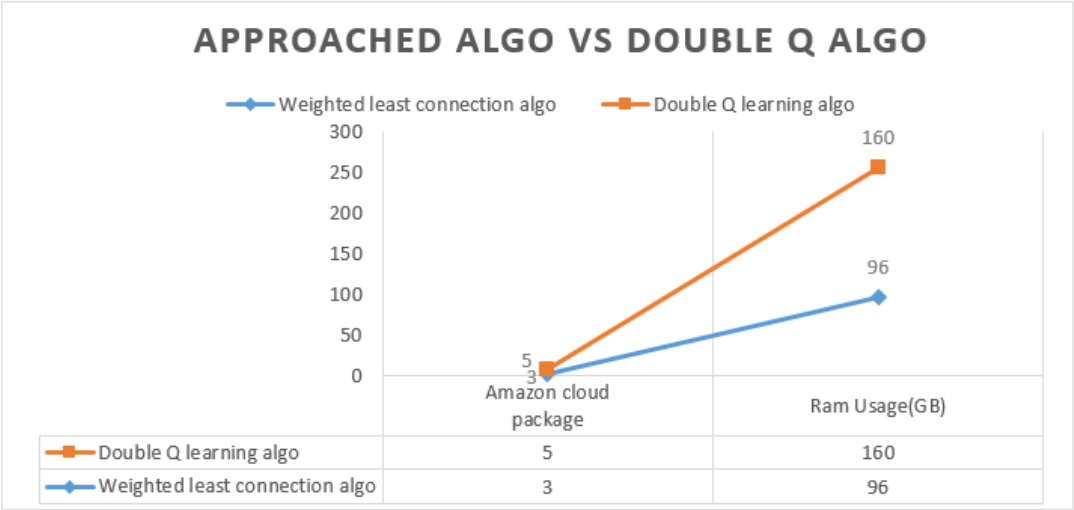


Figure 8.3: Ram Usage Comparison

Now, If we consider an existing cloud package t4g.2xlarge with 32 GB Ram space from Amazon Web Services retained by Amazon and use it for our comparison. Hence, based on the double Q learning algorithm 1 user can use 1 package that eventually would result in 5 user/ client would require 5 packages which indicate 32*5=160GB ram needed, it is obvious 5 clients would not use this amount of space so the rest of the ram space would remain idle. On the other hand, on our approached weighted least connection algorithm users can share a package since our algorithm supports multiple user access observing idle space. Therefore, following this 5 users would need 3 packages which imply 32*3=96GB ram is being used here. If we compare both ram usage in both scenarios 160GB-96GB=64GB ram is being saved from being wasted which signifies 40% of the ram space is being utilized in this process.

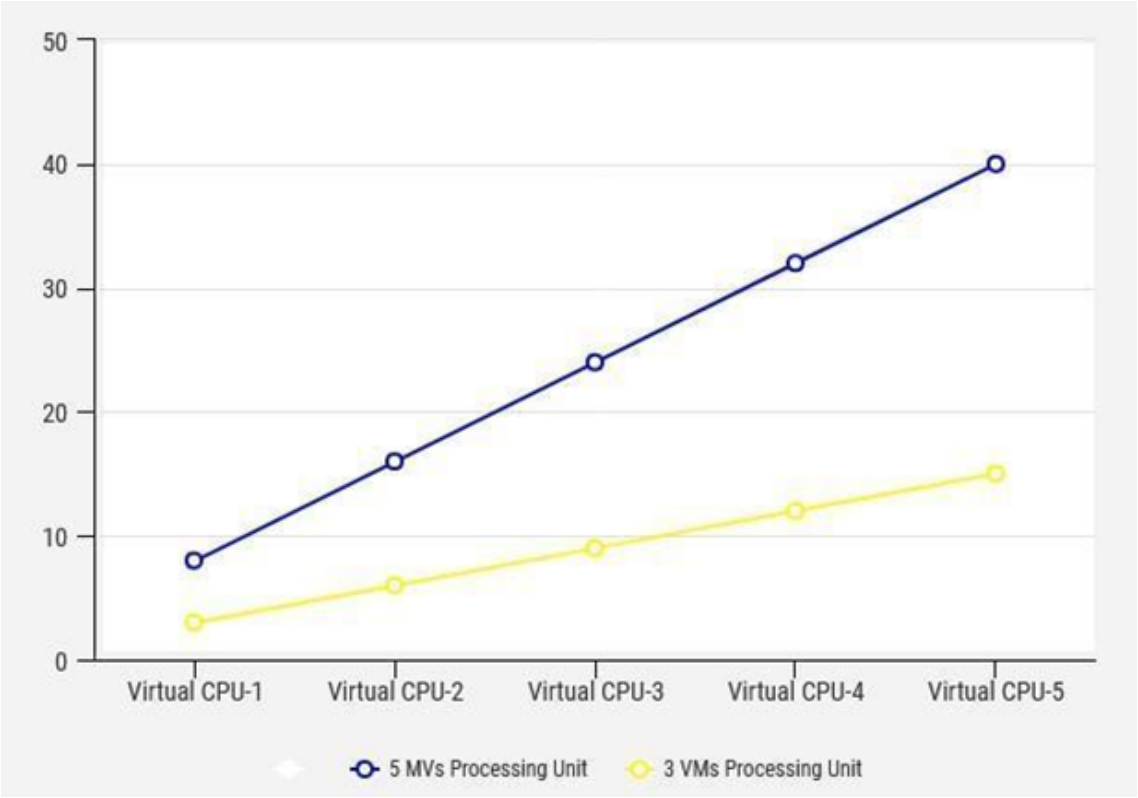


Figure 8.4: Comparison between Double Q-learning based Virtual CPU user process (for 5 VMs) and weighted least connection algorithm executed by our team (for 3 VMs) on Amazon AWS

In the above diagram, we can see that, if we implement Double Q-learning based load balancing algorithm on Amazon AWS's t4g.2xlarge cloud package then according to the comparison paper they are executing their algorithm using 5 MV processing unit because according to their method each user needs one single VM which means total 5*8 = 40 processing units are needed for their algorithm. On the other hand, according to our algorithm that we are implemented there are only 3*8 = 24 processing units are needed since we are utilizing our 3 VMs entirely by implementing them as multiple users and a single VM in the server. Thus, we can save the 40% wastage of virtual machines that are being wasted, according to Romero A. [23] in his recent research.

In the above diagram, we can see that, if we implement Double Q-learning based load balancing algorithm on Amazon AWS's t4g.2xlarge cloud package then according to the comparison paper they are executing their algorithm using 5 MV processing unit because according to their method each user needs one single VM which means total 5*8 = 40 processing units are needed for their algorithm. On the other hand, according to our algorithm that we are implemented there are only 3*8 = 24 processing units are needed since we are utilizing our 3 VMs entirely by implementing them as multiple users and a single VM in the server. Thus, we can save the 40% wastage of virtual machines that are being wasted, according to Romero A. [23] in his recent research.

Chapter 9

Conclusion

In this paper, we tried to present an optimal resource allocation method for load balancing in the cloud servers so that we can resolve the problem of resource management in virtual machines. We have run our code on C++ and observed the simulation result and decided that the problem of load balancing that we are facing everyday on our cloud servers and believe that weighted least connection algorithm is the best suited solution to solve our problem. We also simulated on Oracle VM VirtualBox and observed the different simulation results upon Windows operating system, Linux operating system as well as Solaris operating system. Finally, based on our research observation and simulation result, we have come to the conclusion that Solaris gives the outstanding result as performance, service and security wise which we are intended to provide to our cloud users. In this way, we have attempted a decent approach to an issue that the modern world is currently facing, which is wastage of server resources while load balancing. Finally, this paper also reviews the existing research issues on this subject in current years and what might be the solution method for that. To conclude, we are looking forward to working further on the future trends, challenges and needs in resource optimization in cloud computing servers.

Bibliography

[1] S. Forrest, S. A. Hofmeyr, A. Somayaji, and T. A. Longstaff, "A sense of self for unix processes," in *Proceedings 1996 IEEE Symposium on Security and Privacy*, IEEE, 1996, pp. 120–128.

[2] K. K. Yue and D. J. Lilja, "Dynamic processor allocation with the solaris operating system," in *Proceedings of the First Merged International Parallel Processing Symposium and Symposium on Parallel and Distributed Processing*, IEEE, 1998, pp. 392–397.

[3] S. Fire and E.-L. Midrange, "System administration guide," 2004.

[4] W. Itani, A. Kayssi, and A. Chehab, "Energy-efficient incremental integrity for securing storage in mobile cloud computing," in *2010 International Conference on Energy Aware Computing*, IEEE, 2010, pp. 1–2.

[5] R. P. Padhy, "P goutam prasad rao load balancing in cloud computing system rourkela-769 008," *Orissa, India May*, 2011.

[6] K. Giridas and A. S. Nargunam, "Optimal resource allocation technique (orat) for green cloud computing," *International Journal of Computer Applications*, vol. 55, no. 5, 2012.

[7] K. Dasgupta, B. Mandal, P. Dutta, J. K. Mandal, and S. Dam, "A genetic algorithm (ga) based load balancing strategy for cloud computing," *Procedia Technology*, vol. 10, pp. 340–347, 2013.

[8] J. Kaur and S. Kinger, "A survey on load balancing techniques in cloud computing," *International Journal of Science and Research (IJSR)*, vol. 3, no. 6, pp. 2662–2665, 2014.

[9] M. Rahman, S. Iqbal, and J. Gao, "Load balancer as a service in cloud computing," in *2014 IEEE 8th International Symposium on Service Oriented System Engineering*, IEEE, 2014, pp. 204–211.

[10] X. Song, Y. Ma, and D. Teng, "A load balancing scheme using federate migration based on virtual machines for cloud simulations," *Mathematical Problems in Engineering*, vol. 2015, 2015.

[11] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, 2016.

[12] Y. Fahim, H. Rahhali, M. Hanine, E.-H. Benlahmar, E.-H. Labriji, M. Hanoune, and A. Eddaoui, "Load balancing in cloud computing using metaheuristic algorithm.," *Journal of Information Processing Systems*, vol. 14, no. 3, 2018.

[13] M. Xu, A. N. Toosi, B. Bahrani, R. Razzaghi, and M. Singh, "Optimized renewable energy use in green cloud data centers," in *International Conference on Service-Oriented Computing*, Springer, 2019, pp. 314–330.

[14] S. A. Khoshroo and S. H. Khasteh, "Increase the speed of the dqn learning process with the eligibility traces," *Journal of Control*, vol. 14, no. 4, pp. 13–23, 2021.

[15] G. A. Geronimo, J. Werner, C. B. Westphall, and C. M. Westphall, "Provisioning and resource allocation for green clouds (slides),"