SLA and Cost-aware Cloud Service Provisioning through Matching Game and TOPSIS model

by

Rifat Shahriar Karim
16101328
Shabab Hossain Rhythm
14201053
MD.Rakibul Islam
14301098
Kazi Asif Ahmed Kawwshik
15101029

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 Engineer

Department of Computer Science and Engineering
Brac University
December 2019

Declaration

It is hereby declared that

1. The thesis submitted is 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:

Rifat Shahriar Karim	Shabab Hossain Rhythm
16101328	14201053
MD.Rakibul Islam	Kazi Asif Ahmed Kawwshik
14301098	15101029

Approval

The thesis titled "SLA and Cost-aware Cloud Service Provisioning through Matching Game and TOPSIS model" submitted by

1. Rifat Shahriar Karim (16101328)
2. Shabab Hossain Rhythm (14201053)
3. MD.Rakibul Islam (14301098)
4. Kazi Asif Ahmed Kawwshik (15101029)

Of Fall, 2019 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc in Computer Science and Engineering on December 26, 2019

Examining Committee:

Supervisor:
(Member)

Dr. MD. Golam Robiul Alam
Associate Professor
Department of Computer Science and Engineering
BRAC University

Head of Department:
(Chair)

Dr. Mahbubul Alam Majumdar
Professor and Chairperson
Department of Computer Science and Engineering
Brac University

Abstract

Cloud service delivery is based upon Service Level Agreements (SLA) reflecting a customer and service provider signature agreement setting out the terms of the agreement, Providers must use resources efficiently to minimize the costs of provisioning services. Therefore, strategies are required that take into account multiple SLA parameters and efficient resource allocation. Recent work takes different strategies with single SLA parameters into consideration. These approaches are, however, restricted to simple workflows and to single tasks. The preparation and execution of service requests that take into account several SLA parameters such as required amount of CPU, storage, memory and price are still unresolved research challenges. In our studies we have found matching game approach can be useful in dealing with multiple criteria of cloud resources which can be allocated fairly and efficiently. The emphasis of this paper is the implementation of an SLA provisioning algorithm for game theoretical resources that takes into account the user equality and the use of resources for both. To rank the clients on the basis of their requirements against the resources provided by the clouds service and cloud services against the requirements of clients we have used TOPSIS algorithm. In our thesis, the TOPSIS implementation tests on a 4-VM cluster demonstrate how best this algorithm can be managed in a fair comparison with the Jain's Fairness Index analysis.

Keywords: TOPSIS, Gale-Shapley, Matching game, Resource Allocation, Game Theory, SLA.

Acknowledgement

Firstly, all praise to the Great Allah for whom our thesis have been completed without any major interruption.

Secondly, thank our supervisor, Dr. MD. Golam Robiul Alam who helped us to coordinate our thesis with innovative ideas and his efforts made this thesis easy and informative for us.

And also to our parents, without their support it may not have been possible to complete our studies. With their kind support and prayer we are now on the verge of our graduation.

Finally, a special thanks to the Thesis committee for taking the time to review and evaluate our thesis as a part of our undergraduate program.

Table of Contents

List of Figures

List of Tables

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

AHP Analytical Hierarchy Process

$IaaS$ Infrastructure as a Service

$MCDM$ Multiple Criteria Decision Making

$PaaS$ Platform as a Service

PSO Particle Swarm Optimization

QoS Quality of Service

$SaaS$ Software as a Service

SLA Service Level Agreement

$TOPSIS$ The Technique for Order of Preference by Similarity to Ideal Solution

VM Virtual Machine

Chapter 1

Introduction

1.1 Motivation

Our prime drive is to conduct the research solely relied on enhancing the field of Cloud service provisioning focused on matchmaking . With the rapid advancement of technology, newer and better devices are hitting the market left, right and center. But people from all demography of the world cannot upgrade their devices so frequently. This was predicted by Gordon Moore back in 1965. In his perception which is known as Moore's Law he stated that, the number of transistors on a microchip doubles every two years, though the cost of computers is halved. This is still applicable in these modern times [24]. So there is a chance that a device might become obsolete over the years. Applications running on the devices also get updated frequently. They become more resource heavy and consume more energy with each update. So any method that will help the device overcome this problem is a must for this day and age. This is where cloud computing comes in. Cloud computing services like AWS and Google Cloud provide cloud resources so that a user can run his or her application on cloud servers and get the result on their device. Cloud computing also reduces energy consumption [18] . In our research, we took the matching game approach to match users to servers depending on the criteria of the users and also the capacity of the servers by using TOPSIS and Gale-Shapley Algorithm. This will help the resource constrained device find the server which is optimal for the device's criteria and offload the task to the server. We assume that our research will help computational offloading designs of future services to appro-

priate clients by using Gale-Shapley Algorithm to match the users to servers for optimal resource allocation.

1.2 Problem Statement

The main issue with cloud service provisioning is that different strategies deal with SLA parameters into consideration. But sometimes this does not benefit both parties. And there is no matchmaking for the users and VMs. Also they only provide high end systems which are sometimes not properly utilized. People who need small amount of resource are bound to a fixed price for the services. This also creates resource wastage and bottle-necking problems. In this thesis, we aim to provide a solution that will address these problems

1.3 Research Objectives

The main objective of our research is to ensure that users get matched with cloud resources which can fulfill their requirements. Firstly, we would want to optimize the resource allocation so that the clients get the optimum amount of resources they need. We will also increase the fairness of resource allocation by using both of their preferences. Secondly, we want to maintain our SLA for all the interaction between client devices and cloud resource pairings. Lastly, we want to reduce resource wastage and minimize the cost of the clients so that they get the most resources at the least amount of cost.

1.4 Thesis Contribution

Currently, most of the cloud resource providers satisfy their user requirements. But as far we know this sometimes create some trade-offs which does not allow low resource devices to participate in cloud services. This also creates bottleneck which affect performance and waste resources. In our thesis, we have suggested a system which will take several parameters from the users and the VMs and run a ranking algorithm in a cloud broker. On the basis of these rankings, the cloud resource provider would run a matching game and match clients to the VMs accordingly.

This will open up possibilities for low-end systems to join and provide resources as they are required. This will help to reduce wastage and bottleneck. Moreover, cloud resource providers give these services at a fixed price whereas our aim is to give the users flexibility to choose a price range to get their desired resources and by that range our algorithm will look for the nearest prices and match them accordingly. This would reduce the cost to avail the services. As a result, pairings of clients and VMs would be created which will have optimized resource allocations at the least amount of cost. In summary, our main contribution is

• To create a ranking for both users and VMs based on their criteria with TOPSIS algorithm.

• To run a matching game based on those rankings for optimal resource provisioning.

1.5 Thesis Orientation

The alignment of our thesis paper is as follows. Chapter 2 features the work gathered for our research and existing models. It also discusses about analysis about background information and the methods we need to learn related to our project. Chapter 3 introduces the proposed model of our experiment with dataset and the algorithms used. Chapter 4 explains the result of our project and explains our work with related discussion. Chapter 5 concludes and summarizes the thesis.

Chapter 2

Literature Review and Related Works

2.1 Literature Review

Various examinations have proposed new mathematical models, including optimization theory [20], matching game theory [23], an analytic hierarchy process (AHP) [25], and fair resource allocation [12], to capture the user perceived QoS. Game theory has recently been used to solve problems of resource allocation in cloud computing. Ye and Chen are researching non-cooperative load balancing games and the problem of virtual machine placement [21]. We concentrate on the nature of Nash equilibrium and the search for an optimal allocation method is of little concern. Also another research suggest a method to research both non-cooperative and cooperative games for the distributed resource allocation problem in the federated cloud [17]. We demonstrate that there is a greater incentive for providers to add capital to the cooperative allocation game. Nevertheless, their work models resources as one kind, while our research considers the question of allocation in multi-resource environments. Data sharing collaboration is also one of the fundamental problems for cloud resource allocation. For example, a fair scheduler for Hadoop, which divides resources as fixed-size partitions or slots, has been studied by many works so far [29]. Another common equal policy is max-min equality, which aims to optimize each user's minimum asset. Waldspurger strengthens this strategy by offering a weighted max-min equity template to help such policies which take

into account various factors such as priority, reservation and deadline [4]. Recently, several methods to measure equality have been proposed [13]. However, most of them are simply investigating the fairness of the problem of allocating a single type of resource. Another research studied the issue of equitable allocation of multiple types of resource allocation [16]. We present a dominant approach to resource equality which addresses the issue by measuring each user's dominant share. Cloud computing not only allows users to move their data and computing to a remote location with minimal impact on system performance, but also allows users to easily access a cloud computing platform to display their data and receive computing at anytime and anywhere [9]. Cloud computing aims to provide cheap and easy access to observable and billable computational resources compared to other paradigms such as data delivery, network computing, and so on. Therefore, another research [15] proposed a new approach to dynamic autonomous resource allocation which in computes clouds through decentralized analysis of multiple parameters. Cloud computing aims to provide cheap and easy access to observable and billable computational resources compared to other paradigms such as data distribution network computing, and so on. The tasks are spread around various computational nodes in a cloud computing environment. In order to assign cloud computing resources, nodes with spare computing power are found, and network capacity, line performance, response time, task cost, and resource allocation reliability are evaluated. Cloud computing service quality can therefore be defined through resources such as cpu usage, full time, task costs, and reliability, etc. Multi-criteria-based approach called the TOPSIS algorithm is applied with PSO in order to obtain an efficient optimal solution for scheduling tasks. Other targets include enhancing cloud metrics, i.e. MakeSpan, time of execution, time of transmission and cost of processing. To confirm its accuracy, job quality is compared with pre-existing algorithms. The ultimate goal of the project is to boost the cloud quality QoS.

2.2 Game Theory

Game Theory is a mathematical concept that deals with the formulation of a correct strategy that will allow the person or entity (i.e. the player) to successfully solve the challenge when faced with a complex challenge. It was developed on the premise that there is a strategy that will allow one player to' win' for whatever circumstances, or for whatever' game' [14]. Typically, every Game has three components: a set of players, a set of possible actions for each player, and a set of utility functions that map action profiles into real numbers. The set of players is denoted in this situation is I, where I is a set of finite values

I = 1, 2, 3, ..., I.

The set of possible actions that player I can take for each player is denoted by Ai, and A, which is denoted as the space of all the action profiles is equal to:

$A = A_1 * A_2 * A_3 * A_4 * A_5 * \ldots\ldots.. * A_I \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$ (2.2)

Based on certain important features, games can be divided into different categories. The language used in game theory is vague, so it is possible to use different terms in different sources for the same definition. A game can be classified as a one-player game, two-player game or n-player game, depending on the number of players in the game. The Nash equilibrium is a concept of game theoretical solution that is normally used in economics. John Nash introduced Nash equilibrium in 1950 and emerged as one of game theory's fundamental concepts [31]. Nash equilibrium is a solution theory of a game involving two or more players in which each player is supposed to know the other players ' equilibrium strategies and no player has anything to benefit from modifying his own strategy only [32]. Players must know exactly what their opponents are going to choose [11]. Players should not base themselves on the assumption that all players are rational to do so. They concentrate, if such information is available and 8 accurate, on the basis of statistical information on previous game playing situations. To game theory, the most basic assumption is rationality. It implies that each player is motivated to increase their own payoff, i.e. each player seeks to maximize their own usefulness. In order to meet the increasing demands of various quality of service (QoS) requirements, a range of priority-based mechanisms have been introduced to achieve a higher level of service quality. When delivering QoS for multi-class products, pricing is an important issue. Pricing sys-

tems include usage-based pricing schemes, where users are billed for traffic volume and/or the length of time of their sessions. Pricing based on usage represents costs and is fair to users with different requirements for network use. Over recent years, in these research areas, game theory has been found to be a useful tool. As game theory suggests that every player is trying to maximize their utility, users (as players) will try to maximize their outputs while also minimizing the cost. On the server side servers (as players) will try to maximize the number of tasks completed while minimizing the time for completion and resources used. Some models have been already introduced regarding this problem. Another study [2] shows that if there is additional allocation capacity in a non-cooperative network prior to the operational phase, the optimal allocation is to allocate additional resources to the connection with the highest initial capacity. Another study [3] demonstrates with a mathematical model that there is no allocation function in a change that can optimize any Nash Equilibrium Pareto. Park et al. [6] proposed another two-class model for users with heterogeneous QoS preferences, where there is no one-to - one set user-service relationship. Nash Equilibria's existence conditions are given in their model. We conclude that control schemes in the Nash sense could not assess the proneness to consumer demands stabilization. A' resource-plentiful' framework, however, may show strong convergence to Nash Equilibria in the two-class model. An auction-based bandwidth allocation mechanism called the Progressive Second Price Auction at the edge of the network is proposed by Semret et al. [5] [7] .
We can use game theory to help determine the most likely outcomes whenever we have a situation with two or more players involving known payouts or quantifiable consequences. A basic game contains

• Game: Any set of circumstances that has a result dependent on the actions of two of more decision-makers (players).
• Players: A strategic decision-maker within the context of the game.
• Strategy: A complete plan of action a player will take given the set of circumstances that might arise within the game.
• Payoff: The payout a player receives from arriving at a particular outcome. The payout can be in any quantifiable form, from dollars to utility.

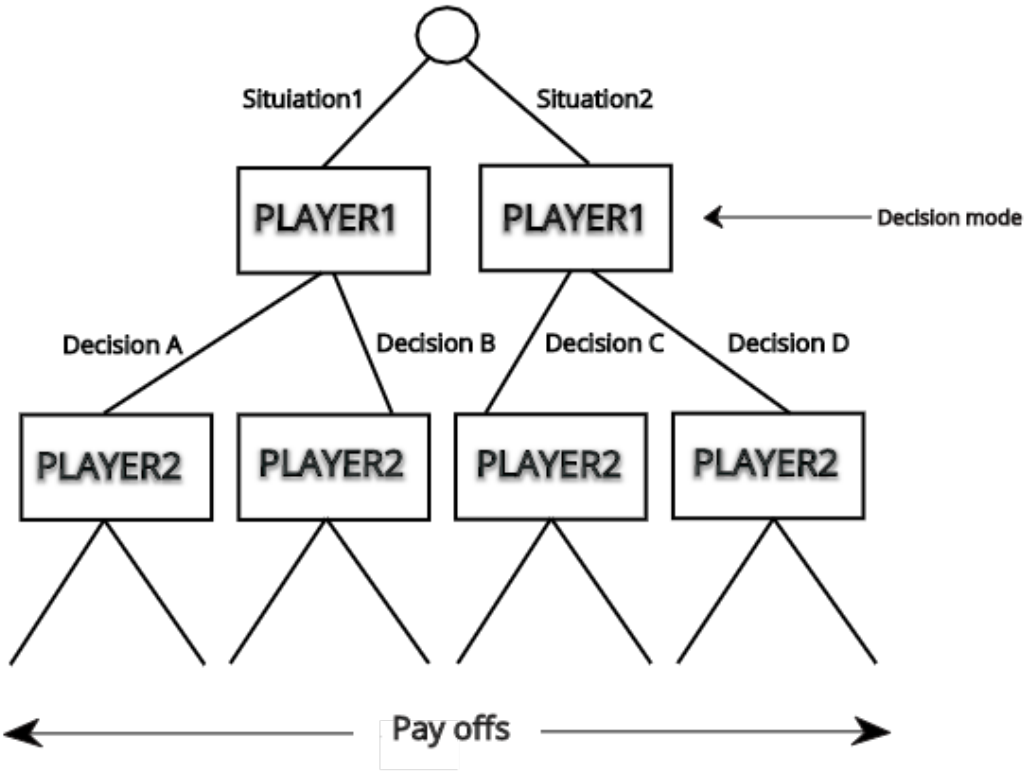

Figure 2.1: Illustration of Game Theory

In this figure Player 1 is the user and Player 2 is the system. Depending on the user's decision, the payoffs would change accordingly.

2.3 Cloud Computing

Cloud computing isn't a new concept; it comes from previous distributed, large-scale computation. However the fast revolution in the area of computer science and data technology will be a technology for subversion and cloud computing. Cloud computing is also a new way of manually configuring business computing. In the near future it will be used extensively. Cloud computing's central concept reduces the user's storage burden. Consumers ultimately connect increasing kind of services, processing and application development tools over the Internet via varied appliances, including PCs, notebooks, mobile phones and PDAs. All of these cloud computing services are available. Cloud computing technology benefits include cost savings, high accessibility and easy scalability.

Cloud Computing Features:
Similar to other technology paradigms, cloud computing offers a variety of different features and benefits. In this section, it is briefly described.

• Cloud computing offers resources and services for consumers on requests. Data Scalability and On-demand infrastructure. The resources can be calculated through multiple data centers.

• Service Quality (QoS)-Cloud computing will maintain the quality, availability, and memory capacities of QoS for applications of hardware or CPU capacity.

• Autonomous system — autonomous systems are operated transparently to the consumer by cloud infrastructure systems. Apps and software may be reconfigured and stored in clouds automatically, based on the specifications of the user.

• Cloud applications are stand-alone and can be accessed through well-defined interfaces like web services and web browsers. User-centric applications.

• Cloud Computing – No front-line expenditure is required. Pricing. There is no provision for capital expenditure. Consumers may pay for access resources and capabilities, or pay for services and capabilities.

Components of Cloud Computing:

• Client Computers: The end user can communicate with the cloud on the computers of the customer.

• Distributed Servers: The computers are located to various locations, but they behave as if they operate together.
• Data Centers: Data Centers are server infrastructure.

Services of Cloud Computing:
• **Software as a Service (SaaS)**: Application is regarded as a software to hold request as a platform on the web. The client will easily access it via the internet [19] instead of downloading the application on his computer. It frees the user from the complex software and hardware management. No software or hardware must be bought, maintained and updated by SaaS users. The only thing that the user needs to be connected to the internet and then access is very easy. For example, Google Apps, etc. Microsoft Office 365.

• **Platform as a Service (PaaS)**: Consumers are equipped with a development ecosystem or framework as a PaaS system where customers can use their respective applications and code. The user is free to build his own software on the network of the provider [19]. Service as a service provider provides the operating system and database server a predefined structure in order to achieve requirements with management capabilities. For example, LAMP, J2EE, Ruby and others. LAMP (Linux, Appache and PHP).

• **Infrastructure as a Service (IaaS)**: IaaS provides a large number of computer services in the form of on-demand space, network, operating system, hardware and processing appliances. Users of IaaS can use an extended network, like the Internet [19], to access services. For example, by signing on to the IaaS system a user may create virtual machines.

2.4 Multiple-criteria Decision Making

Multiple-criteria decision analysis (MCDA) or Multiple-criteria decision making (MCDM) is a sub-disciplinary and full-scale field of strategic analysis devoted to the development of mathematical and analytical methods that enable the empirical assessment of a finite number of alternatives to decisions by a single decision-maker or team under a finite number of success criteria [8]. MCDM refers to the testing, prioritization, ranking or choice of alternatives under typically separate, contradictory or opposing attributes.

Determining the characteristics is very important for MCDM because they play a very significant role in the decision-making process. Many approaches have been developed to address related issues, but a major problem with MCDM is that different techniques may produce different results for the same problem.

There are three major steps in utilizing any MCDM technique:

a. Determine the criteria and alternatives in question.

b. Attaching numerical values or weights to the relative importance of the criteria and the effect on these criteria of the alternatives.

c. Process the weights to determine each alternative's ranking.

Of different MCDM techniques like AHP, WSM, WPM, ELECTRE, TOPSIS etc. We have chosen the TOPSIS method for ranking both servers and clients.

2.5 TOPSIS (Technique for Order Preference by Similarity to Ideal Solution)

TOPSIS (the Technique for Order Preference by Similarity to Ideal Solution) was developed as an alternative to the ELECTRE method [10]. The fundamental concept of this approach is that in a geometrical sense the chosen alternative should have the shortest distance from the ideal solution and the farthest distance from

the negative-ideal solution. TOPSIS believes that each feature tends to increase or decrease usefulness monotonically. Therefore, the ideal and negative-ideal solutions can be easily located. To determine the relative closeness of alternatives to the ideal solution, the Euclidean distance technique is used. By contrasting these relative distances, the preferred order of alternatives is thus given. The TOPSIS method can be summarized by the following diagram:

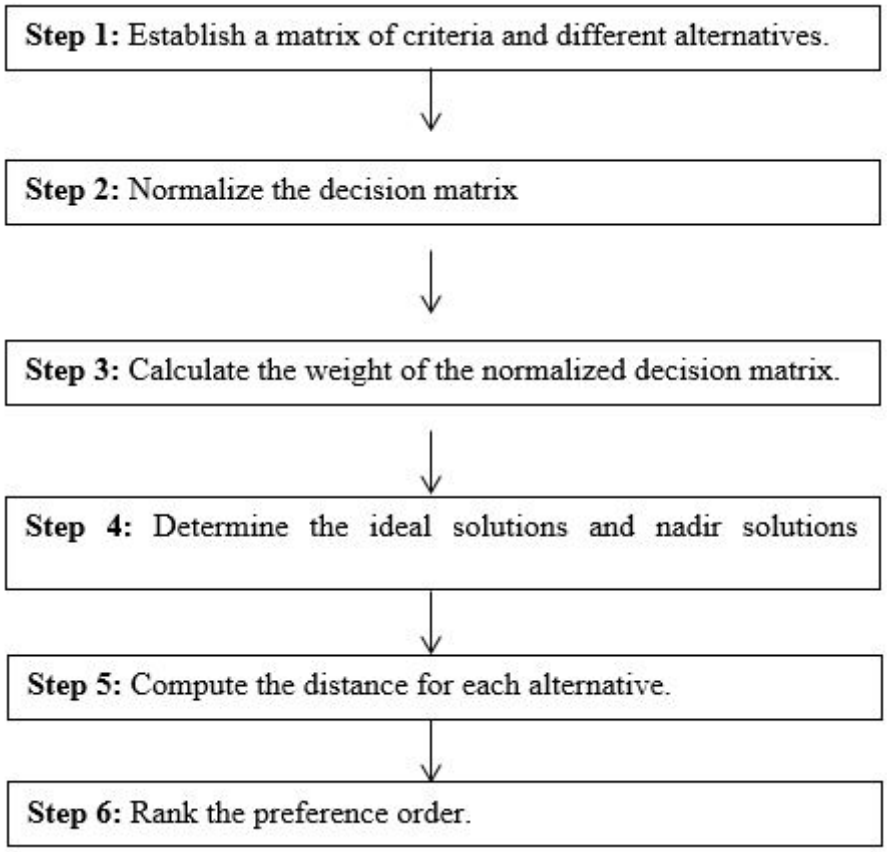

Figure 2.2: TOPSIS method [1]

2.6 Stable Matching

Many algorithms are concerned with matching patterns. The stable matching algorithm is an algorithm used to find a solution to the problem of stable position. By using this algorithm, a bipartite graph shows the match between stable and optimal

component, whether optimal on the client's side or optimal on the server's side. For example, both men and women are to be applicants in the Stable Marriage Question, and another form is to be the applicant's recipient. The type of applicant collection will be the optimal measurement category. If the man acting as a applicant, the produced stable pair will be optimal relative to men; if the position of the applicant changes, it will also apply. The male function will be used as an applicant to ensure that the results of this algorithm yield an optimal stable pair compared to men. In this research men are clients who request resources and women are servers who provide or accept the request or proposal and provide the resources.

2.7 Gale-Shapley Algorithm

In 1962, David Gale and Lloyd Shapley initiated a matching experiment to assign a set of Stable Marriage Problem pairs. The goal is to find an optimal pair of instances of X and Y [27]. Gale and Shapley developed the Gale-Shapley Algorithm to use specific rules to pair some n X objects with n Y objects. The first object is described as n men, and the second object is n women in an arranged marriage in which both parties have preferential lists against each other's gender. Gale-Shapley suggests to each woman a law of every man n. Every man has an alternate pair and free in the execution of algorithms, but each woman has to pair.

This algorithm successfully matches the client with the offered VM. The result of this test is stable so that the optimal pair formed. The Gale-Shapley algorithm determines the pair based on the weight value of each client and worker. If the position of the couple changes, this means there is a new candidate who is more qualified. The algorithm has a preference table that contains interest between both parties, both clients, and workers. Any installation made between clients and workers in a set can be said to be the allocation of stable pairs. But not all preference list is a potential stable partner. A stable pair will be determined in the last round. A condition will result in a separate pair because a certain client rank is higher than the one previously occupying that position. Although engagement occurs, clients

who do not have a worker will occupy an existing position. The more preference lists, the more employees to occupy certain positions.
The following pseudo code is Gale-Shapley algorithm process [27].

```
function stableMatching {
   Initialize all w ∈ W and c ∈ C to free
   while ∃ free client c who still has a worker w to propose to {
         w = first worker on c's list to whom w has not yet proposed
        if w is free
              (c, w) become engaged
        else some pair (c', w) already exists
          if w prefers m to c'
           c' becomes free
           (c, w) become engaged
          else (c', w) remain engaged
        }
}
```

As seen in the figure below, if the worker to whom the client proposes is free and unengaged, the likelihood is determined. This is because we aim to reduce the possibility of a rogue couple being formed. If rogue pairs are formed, they must be corrected by using the conditions for stable matching.

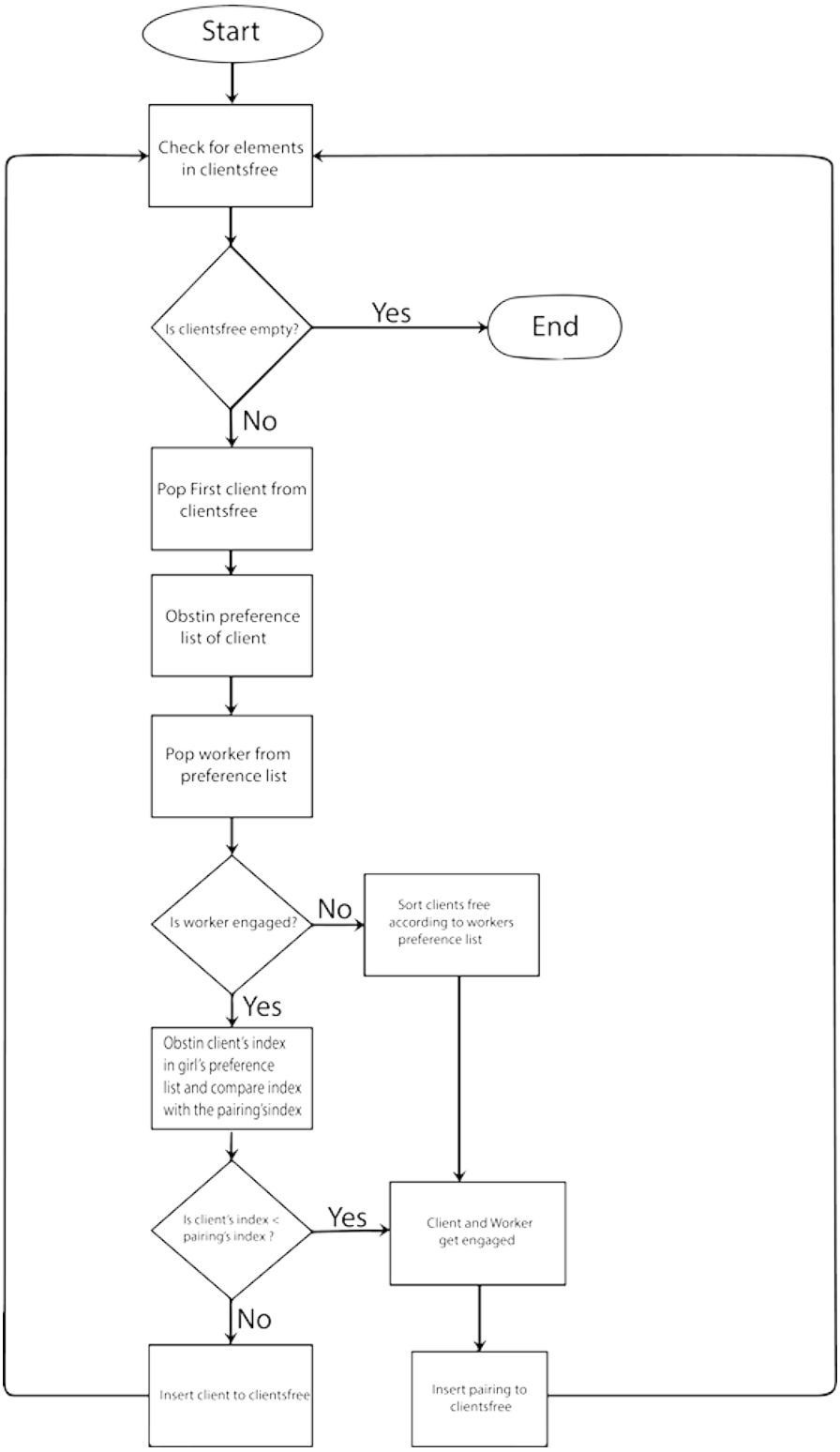

Figure 2.3: Working flow chart of Gale-Shapeley Algorithm

2.8 Docker and Swarm

Docker is a set of platform-as-a-service (PaaS) products that use operating-system-level virtualization to deliver software in packages called containers. Containers are isolated from one another and bundle their own software, libraries and configuration files; they can communicate with each other through well-defined channels. All containers are run by a single operating-system kernel and are thus more lightweight than virtual machines [28].

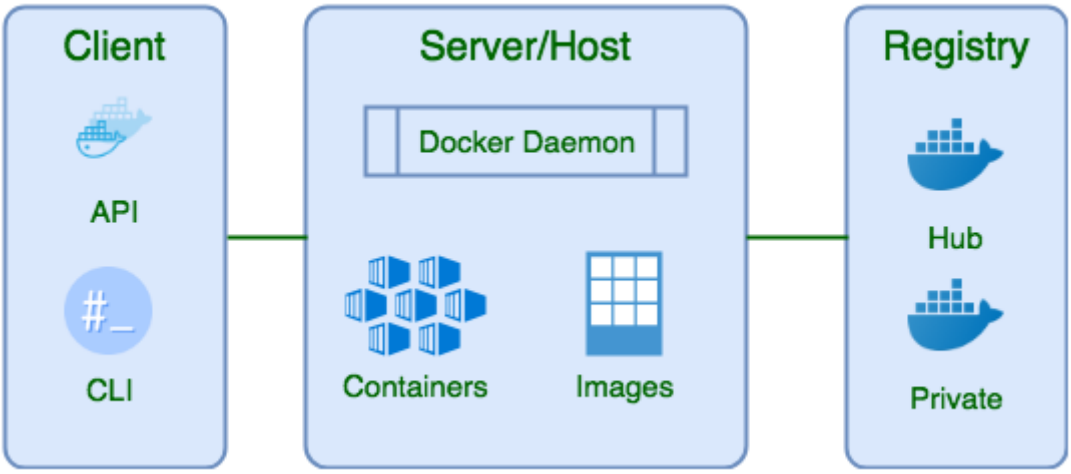

Figure 2.4: Docker model [28]

A Swarm [28] is made up of multiple Docker hosts running in swarm mode and serving as administrators (managing membership and delegation) and workers (managing swarm services). A host of a given Docker may be a manager, a worker, or both. Once you create a system, you identify its optimal state (number of replicas, network and storage assets at your fingertips, external ports that the service exposes.

One of the key benefits of swarm systems over standalone containers is that you can change the configuration of a system, including the networks and volumes to which it is connected, without the need to restart the server manually. Docker changes the configuration, stops the service tasks with the outdated configuration and creates new ones that suit the configuration you want.

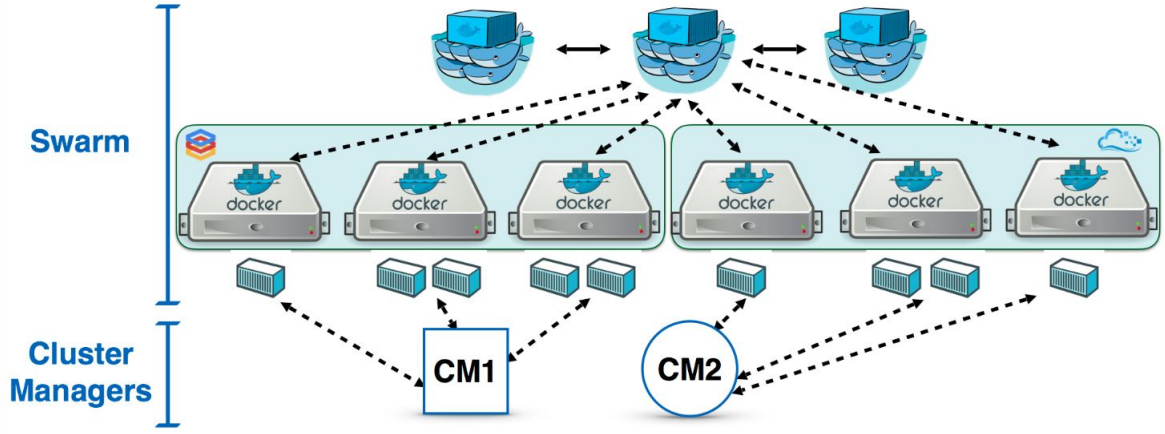

Figure 2.5: Cluster Created using Docker Swarm [28]

2.9 Cloud Broker

The main features of the cloud are self-service on-demand, large network access, resource pooling, strong elasticity, and calculated services[26]. Cloud computing is a phenomenon that is evolving. Several researchers have been working on cloud computing in various aspects such as architecture, SLAs, load balancing, security, third-party participation, cost management, and service quality (QOS) etc.Users also store sensitive information in the cloud, but the provider may not be sure. Because of the risks of service quality failure and the likelihood of malicious insiders in the cloud, working with "single cloud" provider is expected to become less popular with clients. This reveals the Multi cloud environment's importance. Here a third party would benefit both the users and the cloud. This third party is cloud broker. A cloud broker is an individual or business third party that serves as an intermediary between a cloud computing service's purchaser and that service's sellers [30]. The task of the broker may be simply to save the buyer's time by investigating products from various vendors and providing information to the customer on how to use cloud computing to support business objectives. This broker provides cloud vendors and their users that offer different services such as availability, aggregation, integration, performance management, security, etc.

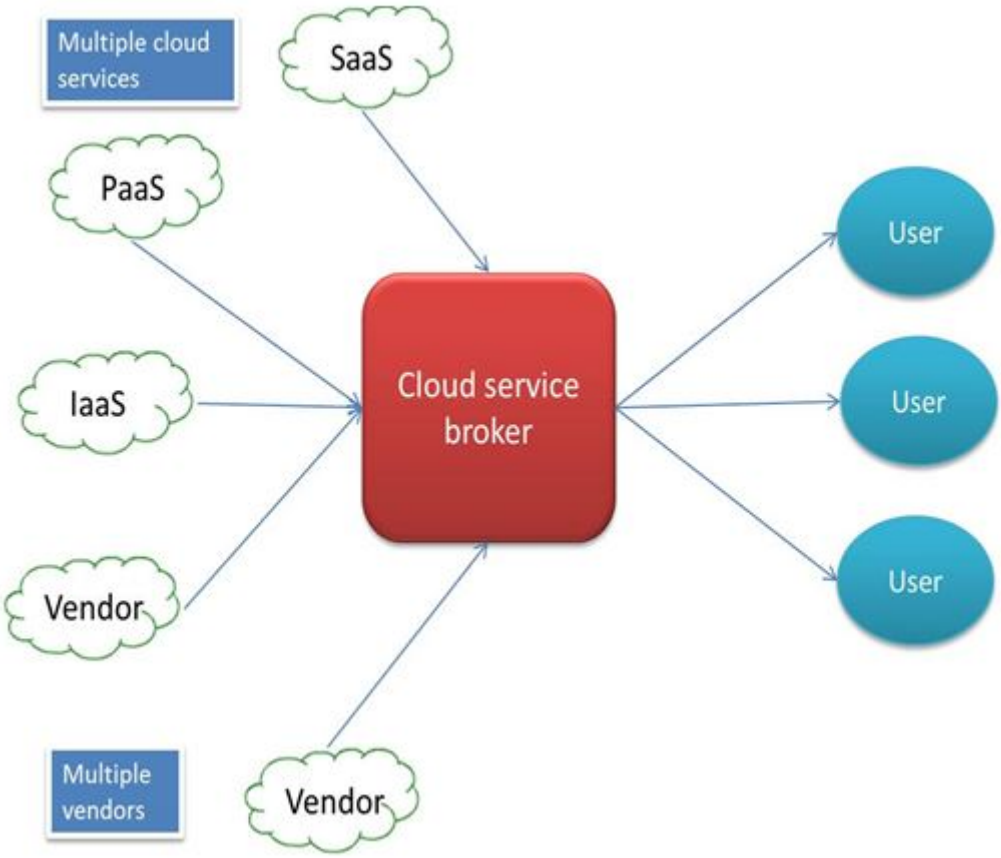

Figure 2.6: Cloud Broker Model [22]

Chapter 3

Proposed Model

For our proposed model, we have used Docker as the container of the VMs, TOPSIS for both client and server side are run on the cloud broker which generates the rankings. Then the rankings are sent to the manager who runs the matching game. For this we used the Gale-Shapley Algorithm.

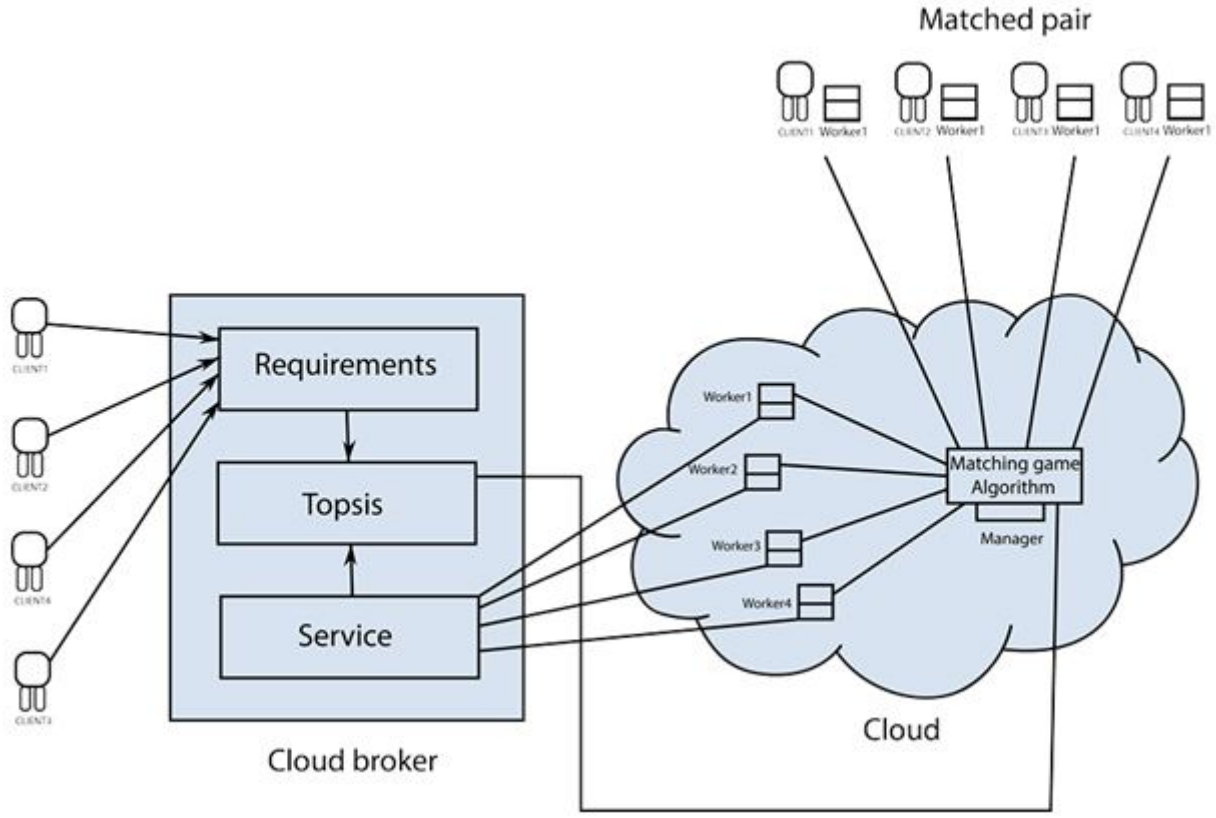

Figure 3.1: Proposed Model

For our model, the clients send their requirements to the cloud broker. In the figure the elements on the left are clients. The middle part is the cloud broker and the right side is the cloud. Firstly, the clients send their requirements to the cloud broker. In our experiment, we named the VMs as workers. The workers also send their specifications to the broker. The broker then runs the TOPSIS algorithm and generates the list of preferences for both clients and workers. Then the list is sent to the manager who runs the matching game and matches the clients with the workers.

3.1 SLA definition

We assume that SLA offered by the system to clients and vm nodes is based on four parameters H_{sla}, R_{sla}, Cp_{sla}, F_{max} where H_{sla} is maximum or required amount of HDD or hard disk resource that a client required and R_{sla} and Cp_{sla} is the maximum or required amount of RAM and CPU of a client. Here H_{all}, R_{all}, Cp_{all} is respectively the allocated resource of a client which will be always $\leq$H$_{sla}$, R_{sla}, Cp_{sla} respectively to fulfill our SLA . There is a tolerable Constant which we can be measured by Jain's fairness of allocation where we can calculate the fairness of allocation in percentage . Here The allocation of resources is H_{all}, R_{all}, Cp_{all} divided by required resources H_{sla}, R_{sla}, Cp_{sla}.

$$Normalizedx_i = \frac{T_i}{O_i} = \frac{H_{all}}{H_{sla}}, \frac{R_{all}}{R_{sla}}, \frac{Cp_{all}}{Cp_{sla}} \tag{3.1}$$

$$FairnessIndex, F_{all} = \frac{(\sum_{y=1}^{n} x_y)^2}{(n * \sum_{y=1}^{n} y^2)} * 100 \tag{3.2}$$

Where y = (1,2,3....n) number of resource

The index ranges from 0-1 and multiply with 100 make it to 0 to 100 percent. Our fairness allocation threshold is F_{max} and the fairness measure by the client resource after allocation is F_{all} . Here, $F_{all}\leq$F$_{max}$ to fulfill our SLA. If F_{all} exceeds the limit

of F_{max} then the corresponding matched pair will be discarded and will join the next session of pairing. The session slots are calculated.

$$T = \frac{j}{u} \tag{3.3}$$

where (j =1,2,3,....M) , is the number of vms are available and u= 4 the chunk we consider here. That's how we can ensure the resource allocation optimization and our SLA.

3.2 Problem formulation

We now describe how to determine optimal virtual allocation over a slot T. starting from the problem constant and variables.

Constants:

• u is the rage of a slot T where in every slot T= j/u

• y is the number of the resource criteria , in our case we consider HDD, RAM,CPU. In can be changed in other locations where the number of criteria can be different. Y is essential for calculating the F_{all}.

Variables :

• T_i is the resource that allocated for a client in any resource criteria $T_i = (H_{all}, R_{all}, Cp_{all})$

• O_i is the resource that the requirement of a client in any resource criteria $O_i = (H_{sla}, R_{sla}, Cp_{sla})$

• c_i is the cost of matched client or allocated cost.

• i and j are number of client and VM nodes respectively

• Z is a non-negative number which varies from 0 to 1 to calculate whether a matched pair is allocated or not $G(c,z)=\sum_{j=1}^{v_{max}} \sum_{i=1}^{U} c_{i,j} * z_{i,j}$

• $V_{max} is the total number of vm.$

• U is the total number of clients

Objective Function :

To minimize the cost of all the clients as their preference of choices in pricing sector, we propose a function where we calculate the cost of the pair after allocation. Here

$$G(c,z) = \sum_{j=1}^{v_{max}} \sum_{i=1}^{U} c_{i,j} * z_{i,j} \tag{3.4}$$

Here, C is the cost of corresponding vm after allocation and z is either 0 or 1 . our objective is to minimize this cost as the clients preference of costing.

$$minG(c,z) = \sum_{j=1}^{v_{max}} \sum_{i=1}^{U} c_{i,j} * z_{i,j} \tag{3.5}$$

$$H_{all} \leq H_{sla} \tag{3.6}$$

$$R_{all} \leq R_{sla} \tag{3.7}$$

$$Cp_{all} \leq Cp_{sla} \tag{3.8}$$

Subject to:

$$v_{all} \leq v_{max} \tag{3.9}$$

$$v_{max} = \sum_{j=1}^{v_{max}} z_j \tag{3.10}$$

$$\sum i = \sum j \tag{3.11}$$

$$F_{all} \leq F_{max} \tag{3.12}$$

Here (3.5) is our main objective function. Equations (3.6),(3.7),(3.8) and (3.9) are inequalities. Equation (3.11) states that, the number of clients and workers must be equal.

3.3 Algorithm Matching Game

while (worker are free)

1 Take 4 clients and workers to form a cluster and generate separate matrix for them

2: rankClients = clients.TOPSIS();

3: rankWorkers = workers.TOPSIS();

4: for (each client)

5: preferenceClients = TOPSIS(rankClients, rankWorkers);

6: for (each worker)

7: prefernceWorkers = TOPSIS(rankWorkers, rankClients);

8: matchingGame (preferenceClients, prefernceWorkers)

9: Match Clients to workers and give resources;

10: end if (workers are busy)

The positive ideal solution consists of all the strong criteria achievable values, whereas the negative ideal solution consists of all the worst criteria achievable values. In the TOPSIS method, precise scores are used in the formation of a decision matrix and standardized decision matrix that each alternative receives from all the criteria [1]. The step-wise procedure for implementing TOPSIS is presented as follows:

Step 1: Generate the criteria matrix

$$A_{n*n} = \begin{bmatrix} a_{11} & ... & a_{1n} \\ . & & . \\ . & & , \\ a_{n1} & ... & a_{nn} \end{bmatrix} \tag{3.13}$$

Step 2: Create a normalized decision matrix with criteria.

$$r_{ij} = \frac{x_{ij}}{\sqrt{\sum_j^J x_{ij}^2}} \tag{3.14}$$

Where j=1,2,3,. . . . . .,J; i = 1,2,3,. . . . . ,n

and Where x_{ij} and r_{ij} are the original and normalized decision matrix score.

Step 3: Build the weighted standard decision matrix by multiplying the weights

w_i of assessment criteria with the standardized decision matrix r_{ij}.

$$v_{ij} = w_i * r_{ij} \tag{3.15}$$

where j=1, 2, 3 ,. . . . . .,J; i = 1,2,3,. . . . . .,n

Step 4: Determined the positive ideal solution (PIS) and negative ideal solution (NIS)

$$A^+ = (v_1^+, v_2^+, v_3^+, \ldots\ldots\ldots, v_n^+) \tag{3.16}$$

Where $v_i^+ = max(v)_{ij}$ if j ϵJ;min(v)$_{ij}$ if j $\epsilon J^- A^- = (v_1^-, v_2^-, v_3, \ldots\ldots\ldots, v_n^-)$(3.17)Where $v_i^- = min(v)_{ij}$ if j ϵJ;max(v)$_{ij}$ if j ϵJ^-

Step 5: Calculate the separation steps between PIS and NIS for each alternative

$$d_{ib} = \sqrt{\sum_{j=1}^{n}(v_{ij} - V_j^+)^2} \tag{3.18}$$

where, i = 1,2,3,. . . .,n

$$d_{iw} = \sqrt{\sum_{j=1}^{n}(v_{ij} - V_j^-)^2} \tag{3.19}$$

where, i = 1,2,3,. . . .,n

Step 6: Calculate the relative closeness co-efficient of each alternative to the ideal solution.

$$C_i = \frac{d_{iw}}{(d_{iw} + d_{ib})} \tag{3.20}$$

Step 7: Depending on the decreasing values of the coefficient of closeness, alternatives range from the most desirable to the worst. By this we can rank the preference order.

By this TOPSIS algorithm we generate the client's preference and worker's preference in the cloud broker and after that send this to the cluster manager where matchmaking algorithm works. We used Gale-Shapley algorithm to match clients to worker nodes based on the TOPSIS calculated preference list of both clients and workers. Figure 3 illustrates our matchmaking algorithm. After getting the pairs we calculated Jain's fairness index. It shows how fairly the resources where allocated. If

$F_a ll F_m ax then the matched pair is stable and ful fills our SLA. Otherwise discard the pair. Some test cases$

3.4 Preprocessing

Firstly, we took 4 clients' requirements from the dataset. Because our system model is based on 4-VM and in Equation (13) states that the number of clients and workers has to be equal. Then we set the criteria for the clients (max, max, max, min) for the four parameters which are (HDD, RAM, CPU, Maximum Preferred Price). As for the workers, the criteria are (max, max, max, max). We created two matrixes and sent them to the broker. The broker ran the TOPSIS algorithm and then sent the results to the manager.

Chapter 4

Result and Analysis

4.1 Results

After running the TOPSIS function on both clients and workers, we plotted the TOPSIS score of both client and workers. In x-axis we showed the number of workers and in y-axis we plotted the TOPSIS scores.

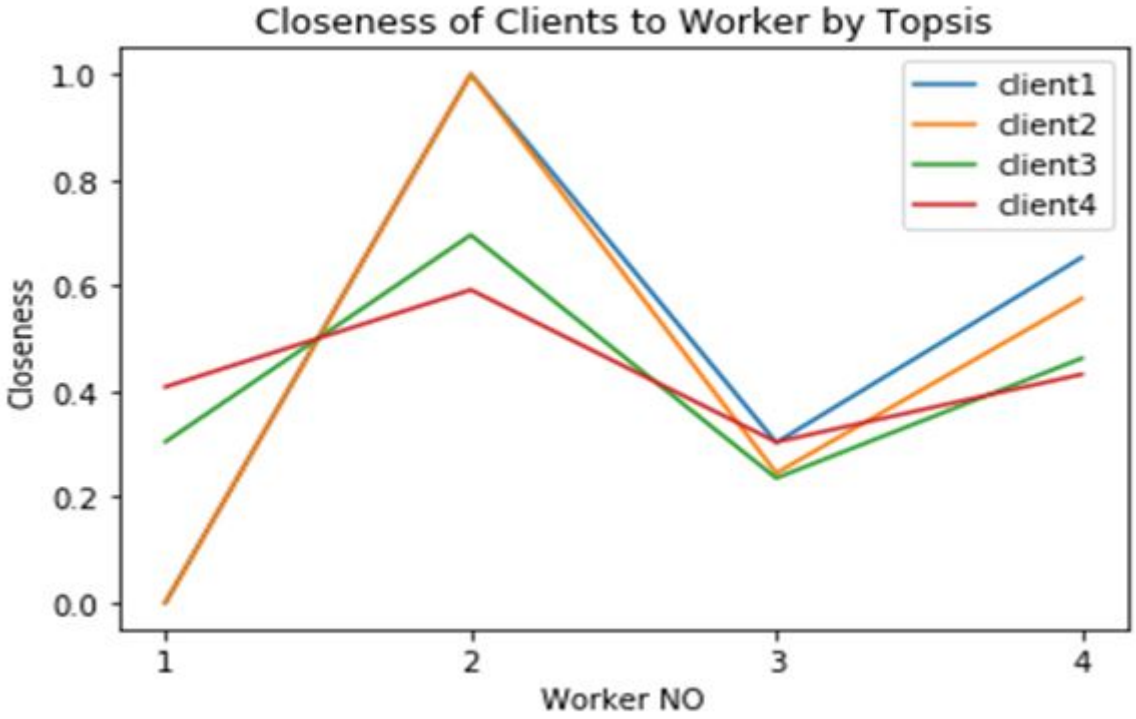

Figure 4.1: Client TOPSIS score

The closeness value is the distance from the current situation to the ideal situation. As we can see in Figure 8, the higher the closeness value of clients to workers is, the

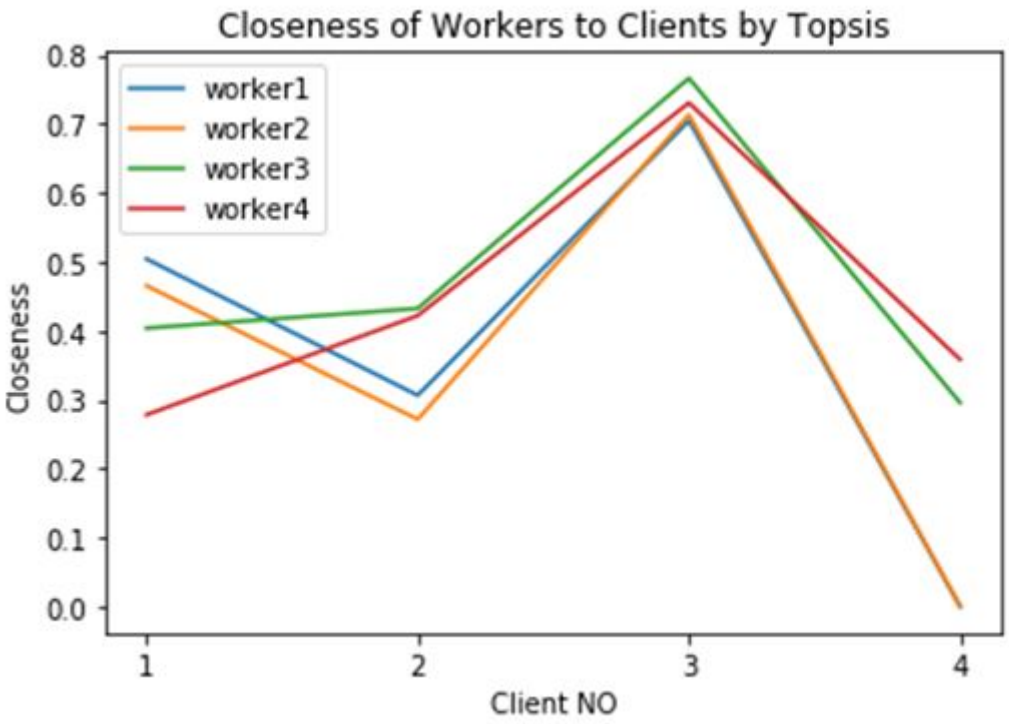

Figure 4.2: Worker TOPSIS score

lower the distance is from the ideal situation. Based on this value the rankings of Clients are made.

For the worker TOPSIS score we plotted the worker to client TOPSIS score against number of clients.

Again in Figure 9, the higher the closeness value of workers to client is, the lower the distance is from the ideal situation. Based on this value the rankings of rankings are made.

Now we get the tables with rankings.

Client ID	HDD	RAM	CPU	Maximum preferred Price	RANK
Client 1	1000	32	4	30	1
Client 2	2000	8	3	20	3
Client 3	5000	16	3	25	2
Client 4	1000	4	2	10	4

Table 4.1: Client Ranking Table

From 4.1 we get the above table which shows the rank of the clients against workers. Here Client id are the client number, HDD, RAM, CPU are the requirements of clients. Furthermore, here Price column is the maximum preferred range of price of clients'.

Worker ID	HDD	RAM	CPU	Maximum preferred Price	RANK
Worker 1	1000	4	2.5	10	4
Worker 2	3000	16	3.4	30	1
Worker 3	1000	8	3	17	3
Worker 4	2000	16	3	20	2

Table 4.2: Worker Ranking Table

From 4.2 we get the above table which shows the rank of the workers against clients .Here Worker ID are the worker number ,HDD, RAM,CPU are the specification of clients. Furthermore, here Price column is the fixed prices of workers.

Engagements

Client 1 and Worker 2

Worker 2 dumped Client 1 for Client 3

Client 1 and Worker 4

Client 2 and Worker 1

Client 4 and Worker 3

Couples

Worker 1 is engaged to Client 2

Worker 2 is engaged to Client 3

Worker 3 is engaged to Client 4

Worker 4 is engaged to Client 1

So the final matched sets are (1,2),(2,3),(3,4),(4,1)

4.2 Analysis

During the Analysis phase we used Jain's Fairness index to observe how fairly the resources where allocated.

Measured throughput $T_i = (T_{HDD}, T_{RAM}, T_{CPU})$ Optimal throughput $O_i = (O_{HDD}, O_{RAM}, O_{CPU})$

$$FairnessIndex = \frac{(\sum_{y=1}^{n} x_y)^2}{(n * \sum_{y=1}^{n} y^2)} \tag{4.1}$$

The index ranges from 0-1.

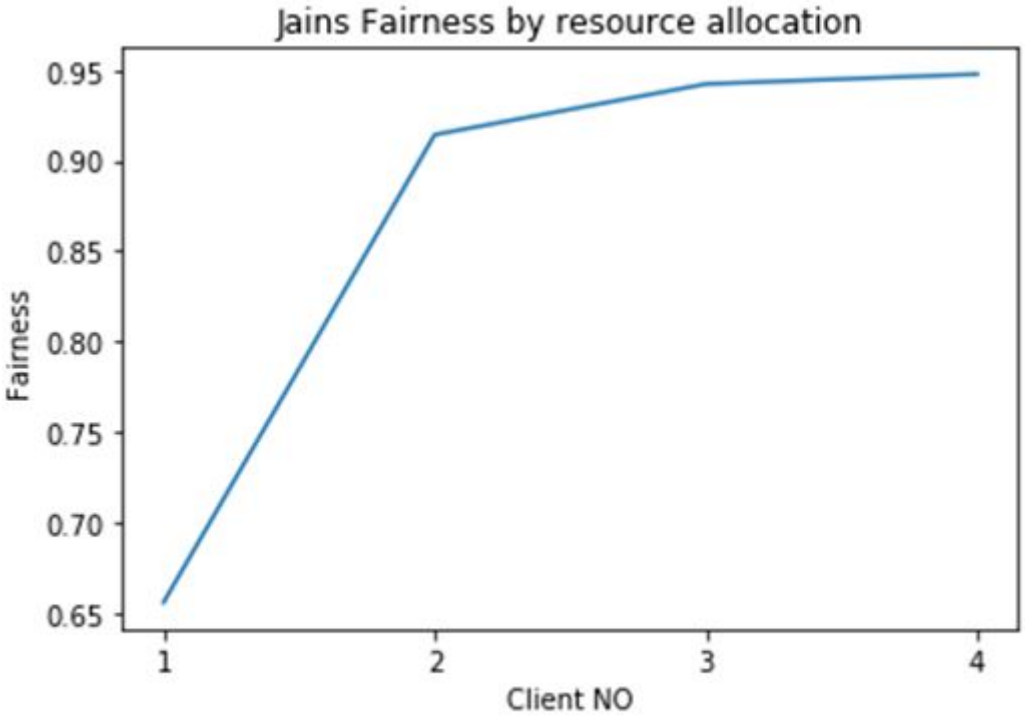

Figure 10: Jain's Fairness Index

Figure 4.3: Calculated Jain's Fairness Index

In 4.3 we can see that the clients' fairness ar 65%, 92%, 94% and 95%.

Now we compared our results with the traditional Min-Max approach.

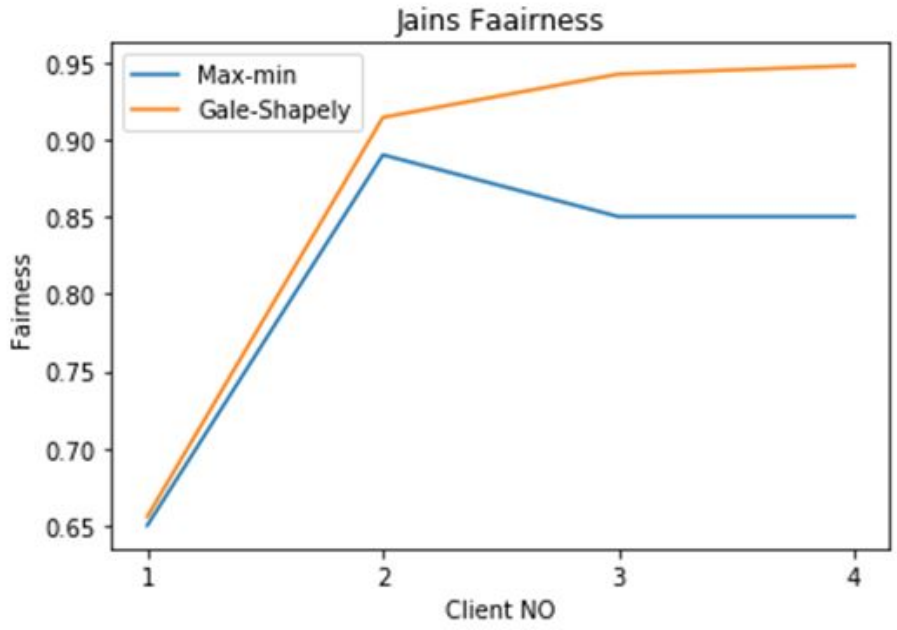

Figure 4.4: Comparison of Max-min vs Gale-Shapley

Here we compared our Allocation with Max-min Allocation. We can see that our Allocation if more Fair and efficient. Max-min allocation is used by most service providers. We can see that our method results in better allocation and less wastage.

Now we will compare the cost of the clients maximum preferred price and allocated price after matching.

(1).png (1).png (1).png (1).png (1).png (1).png (1).png (1).png

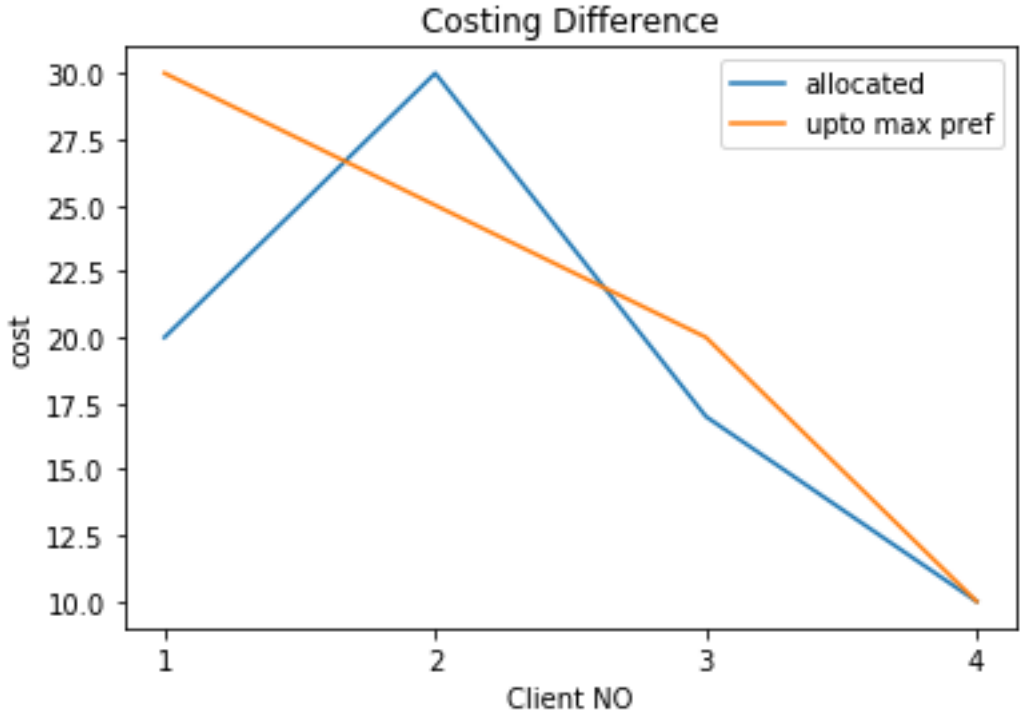

Figure 4.5: Cost Comparison

We can see that only 1 out of 4 clients allocated price is more than their preferred price.

Chapter 5

Conclusion

We investigated the issue of SLA provisioning in cloud computing in this paper. In order to implement SLA we used TOPSIS and Gale-Shapley algorithm, we consider multiple types of resources such as CPU, memory, and virtual machine level storage. For each physical server, the algorithm promotes not only equal resource allocation for users, but also efficient use of resources.

The problem of SLA provisioning in resource allocation is modeled as a finite extensive game with perfect information, and our approach leads to a decision on the Nash equilibrium.

If jobs have machine preferences, future work could usefully study the fairness-usage trade-off. Another direction is to consider the problem of allocation within the priorities of the work situation. In addition, we plan to explore how to use this game theoretical provisioning with resource allocation with multiple resource providers affect the future of cloud computing.

Bibliography

[1] C.-L. Hwang and K. Yoon, "Methods for multiple attribute decision making", in *Multiple attribute decision making*, Springer, 1981, pp. 58–191.

[2] Y. A. Korilis, A. A. Lazar, and A. Orda, "Architecting noncooperative networks", *IEEE Journal on selected areas in communications*, vol. 13, no. 7, pp. 1241–1251, 1995.

[3] S. Shenker, "Making greed work in networks: A game-theoretic analysis of switch service disciplines", *Networking, IEEE/ACM Transactions on*, vol. 3, pp. 819–831, Jan. 1996. DOI: 10.1109/90.477727.

[4] C. Waldspurger, "Lottery and stride scheduling: Flexible proportional-share resource management", Nov. 1996.

[5] N. Semret, R. Liao, A. Campbell, and A. Lazar, "Market pricing of differentiated internet services", Feb. 1999, pp. 184–193, ISBN: 0-7803-5671-3. DOI: 10.1109/IWQOS.1999.766494.

[6] K. Park, M. Sitharam, and S. Chen, "Quality of service provision in noncooperative networks with diverse user requirements", *Decision Support Systems*, vol. 28, pp. 101–122, Mar. 2000. DOI: 10.1016/S0167-9236(99)00078-0.

[7] N. Semret, R. Liao, A. Campbell, and A. Lazar, "Pricing, provisioning and peering: Dynamic markets for differentiated internet services and implications for network interconnections", *Selected Areas in Communications, IEEE Journal on*, vol. 18, pp. 2499–2513, Jan. 2001. DOI: 10.1109/49.898733.

[8] F. A. Lootsma, *Multi-criteria decision analysis via ratio and difference judgement*. Springer Science & Business Media, 2007, vol. 29.

[9] B. Hayes, "Cloud computing", *Commun. ACM*, vol. 51, no. 7, pp. 9–11, Jul. 2008, ISSN: 0001-0782. DOI: 10.1145/1364782.1364786. [Online]. Available: http://doi.acm.org/10.1145/1364782.1364786.

[10] Y.-J. Wang, "Applying fmcdm to evaluate financial performance of domestic airlines in taiwan", *Expert Systems with Applications*, vol. 34, pp. 1837–1845, Apr. 2008. DOI: 10.1016/j.eswa.2007.02.029.

[11] B. Bruin, "Overmathematisation in game theory: Pitting the nash equilibrium refinement programme against the epistemic programme", *Studies in History and Philosophy of Science*, vol. 40, pp. 290–300, Sep. 2009. DOI: 10.1016/j.shpsa.2009.06.005.

[12] T. Lan, D. Kao, M. Chiang, and A. Sabharwal, "An axiomatic theory of fairness in network resource allocation", *Proceedings - IEEE INFOCOM*, vol. 1, Jun. 2009. DOI: 10.1109/INFCOM.2010.5461911.

[13] ——, "An axiomatic theory of fairness in network resource allocation", *Proceedings - IEEE INFOCOM*, vol. 1, Jun. 2009. DOI: 10.1109/INFCOM.2010.5461911.

[14] O. Raoof and H. Al-raweshidy, "Theory of games: An introduction", in. Sep. 2010, ISBN: 978-953-307-132-9. DOI: 10.5772/46930.

[15] Y. Yazir, C. Matthews, R. Farahbod, S. Neville, A. Guitouni, S. Ganti, and Y. Coady, "Dynamic resource allocation in computing clouds using distributed multiple criteria decision analysis", Jul. 2010, pp. 91–98. DOI: 10.1109/CLOUD.2010.66.

[16] A. Ghodsi, M. Zaharia, B. Hindman, A. Konwinski, S. Shenker, and I. Stoica, "Dominant resource fairness: Fair allocation of multiple resource types", in *Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation*, ser. NSDI'11, Boston, MA: USENIX Association, 2011, pp. 323–336. [Online]. Available: http://dl.acm.org/citation.cfm?id=1972457.1972490.

[17] M. M. Hassan, B. Song, and E. Huh, "Distributed resource allocation games in horizontal dynamic cloud federation platform", in *2011 IEEE International*

Conference on High Performance Computing and Communications, Sep. 2011, pp. 822–827. DOI: 10.1109/HPCC.2011.116.

[18] F. Chen, J.-G. Schneider, Y. Yang, J. Grundy, and Q. He, "An energy consumption model and analysis tool for cloud computing environments", Jun. 2012. DOI: 10.1109/GREENS.2012.6224255.

[19] H. Yang and M. Tate, "A descriptive literature review and classification of cloud computing research", *Communications of the Association for Information Systems*, vol. 31, no. 1, p. 2, 2012.

[20] A. Elarfaoui and N. Elalami, "Optimization of qos parameters in cognitive radio using combination of two crossover methods in genetic algorithm", *International Journal of Communications, Network and System Sciences*, vol. 06, pp. 478–483, Jan. 2013. DOI: 10.4236/ijcns.2013.611050.

[21] D. Ye and J. Chen, "Non-cooperative games on multidimensional resource allocation", *Future Generation Computer Systems*, vol. 29, pp. 1345–1352, Aug. 2013. DOI: 10.1016/j.future.2013.02.004.

[22] J. Gottschlich, J. Hiemer, and O. Hinz, "A cloud computing broker model for iaas resources", Jan. 2014.

[23] N. Namvar, W. Saad, B. Maham, and S. Valentin, "A context-aware matching game for user association in wireless small cell networks", May 2014, pp. 439–443, ISBN: 978-1-4799-2893-4. DOI: 10.1109/ICASSP.2014.6853634.

[24] M. Guarnieri, "The unreasonable accuracy of moore's law [historical]", *IEEE Industrial Electronics Magazine*, vol. 10, pp. 40–43, Mar. 2016. DOI: 10.1109/MIE.2016.2515045.

[25] B. Kim, J. Cho, S. Jeon, and K. Lee, "An ahp-based flexible relay node selection scheme for wbans", *Wireless Personal Communications*, vol. 89, Apr. 2016. DOI: 10.1007/s11277-016-3284-y.

[26] M. G. R. Alam, M. Munir, M. Z. Uddin, M. S. Alam, T. Nguyen Dang, and C. S. Hong, "Edge-of-things computing framework for cost-effective provisioning of healthcare data", *Journal of Parallel and Distributed Computing*, Sep. 2018. DOI: 10.1016/j.jpdc.2018.08.011.

[27] E. Elviwani, A. P. U. Siahaan, and L. Fitriana, "Performance-based stable matching using gale-shapley algorithm", Jan. 2018. DOI: 10.4108/eai.23-4-2018.2277597.

[28] *"docker documentation"*, https://docs.docker.com/, Accessed: 2019-11-30.

[29] *"scheduling in hadoop,"* http:/www.cloudera.com/blog/tag/scheduling, Accessed: 2019-11-30.

[30] *"what is cloud service brokerage"*, https://www.theresearchpedia.com/research-articles/what-is-cloud-service-brokerage, Accessed: 2019-11-30.

[31] A. Krek, *Analysis of geoinformation pricing using game theory.*

[32] M. J. Osborne *et al.*, *An introduction to game theory*, 3, vol. 3.