A User Relinquishment-based Resource Assignment Scheme to Maximize the Net Profit of Cloud Service Providers

by

Sarabjeet Singh

A thesis submitted to the Faculty of Graduate and Postdoctoral Affairs in partial fulfillment of the requirements for the degree of

Master of Applied Science

in

Ottawa-Carleton Institute for Electrical and Computer Engineering

Department of Systems and Computer Engineering

Carleton University

Ottawa, Ontario, Canada

Thesis Supervisor: Professor Marc St-Hilaire

Abstract

In a cloud federation, by using the pay-as-you-go billing model users can relinquish their services at any point in time and pay accordingly. Therefore, this thesis aims to study the resource assignment problem in the situation where the user relinquishment impacts the net profit of a cloud service provider. As a solution, our study 1) proposes a tool to calculate the net profit which includes income, electricity expenses, and relinquishment loss; 2) compares different ways to predict the user behavior and deduce a better prediction technique based on linear regression; and 3) proposes a relinquishment-aware resource optimization model to estimate the amount of resources based upon the predicted user behavior. Simulations were performed with the CloudSim framework. The results show that instead of blindly assigning resources to users, a cloud service provider with a finite resource pool can gain more by estimating the resources using better prediction techniques.

Acknowledgements

I would like to express my sincere gratitude to Dr. Marc St-Hilaire for his supervision of this dissertation. I am indebted to him for sharing his insightful comments and constant support for this study. I am greatly appreciative to him for having faith in me and investing his time and effort for reviewing and suggesting the changes to improve my research paper and thesis. I would also like to acknowledge Dr. Mohammad Aazam for his initial guidance, long meetings and expert views while he was a Post-Doctoral fellow at Carleton University. I wish him all the best for his future research work.

I am especially grateful to my close friend Gurinderbeer Singh who was always ready to provide me his perceptive views on technical aspects. His role was unique and influencing throughout my degree. I am also thankful to my colleague Qi Hu for taking out his time to help me understand the basics of this project.

I dedicate this thesis to my family as my study would not have been possible without their loving support and unwavering encouragement. Moreover, I am sincerely thankful to all my roommates and Punjabi friends in Canada who helped me forget the toughness of being far away from home and family. Lastly and the most importantly, I thank Sri Guru Granth Sahib ji's 'baani' that made me capable of calmly handling the pressure situations.

Table of Contents

List of Tables

List of Figures

Chapter 1

Chapter 2

Chapter 3

Chapter 4

List of Algorithms

List of Acronyms

Amazon EC2	Amazon Elastic Cloud Compute
AOP	Average Overall Probabilities
AOU	Average Overall Utilization
API	Application Programming Interface
ARIMA	Autoregressive Integrated Moving Average
ARP	Average Relinquish Probability
BaaS	Broker-as-a-Service
CAD	Canadian Dollar
Capex	Capital Expenditure
CPU	Central Processing Unit
CRM	Customer Relationship Management
CSP	Cloud Service Provider
GCP	Google Cloud Platform
GPU	Graphics Processing Unit
HPC	High Performance Computing
IaaS	Infrastructure-as-a-Service
IDC	International Data Corporation
IDS	Intrusion Detection System
IoT	Internet of Things
IP	Internet Protocol
LP	Linear Programming

LR	Linear Regression
MCC	Mobile Cloud Computing
MDP	Markov Decision Process
MIP	Mixed Integer Programming
M/M/1	Markovian Model for single queue
ONP	Overall Net Profit
Opex	Operating Expenses
OPL	Optimization Programming Language
OS	Operating Systems
PaaS	Platform-as-a-Service
QoS	Quality of Service
RACE	Relinquishment-Aware Cloud Economics model
RAM	Rapid Access Memory
RP	Reactive Prediction
SaaS	Software-as-a-Service
SAaaS	Sensing and Auction-as-a-Service
SEaaS	Sensor Event-as-a-Service
SOP	Service Oriented Probabilities
SLA	Service Level Agreement
VM	Virtual Machine
VSaaS	Video Surveillance-as-a-Service
WSBN	Wireless Body Sensor Networks
XaaS	Everything-as-a-Service

Chapter 1: Introduction

Cloud computing is known as the most widespread paradigm of distributed and parallel computing. The core feature of cloud computing is to provide reliable and consistent services to the users. The user requests a service from the Cloud Service Provider (CSP) with certain requirements and deadlines. The prices for these services are negotiated and then finalized into a Service Level Agreement (SLA) [1]. Based on the SLA, the provider leases/rents the amount of resources required to run the service at any place and anytime through the medium of the Internet [2]. The resources that are provided to run a service are housed by a CSP in a facility called data center which includes CPU, GPU, memory, storage [3].

Further, virtualization allows CSPs to dynamically scale the resources as per the demand of the users. The resource scaling is beneficial on both the ends as providers make the money that users save on the hardware [4]. As a result, more and more reputed organizations and users are getting attracted to completely shift their business to the cloud. The biggest example is the recent deal between a very famous application company called Snapchat and Google to rent the entire application development infrastructure through the Google Cloud Platform (GCP) service [5] [6] [7]. Other than that, various reputable organizations and financial institutions are expected to be multi-billion markets for the cloud computing industry [8]. As a result, many academic as well as industry people are showing a great deal of interest to carry out research in this emerging field of study.

Moreover, in the pay-as-you-go model, users have the liberty to give up the allocated resources whenever they want. This leads to the situation in which users overestimate the amount of resources their application requires and relinquish the resources before their scheduled completion time. As a matter of fact, the provider in a cloud federation already reserves the resources for the duration requested by the user and loses a chance to earn from those resources if the user relinquishes before its scheduled end time.

Therefore, it could be beneficial for the CSP to store the users' history based upon their usage of resources in past. Further, that history could be used to predict the user behavior (how likely user would relinquish) and allocate the resources based upon that when user returns for the service. This way, CSPs could maximize their server utilization and minimize their loss. This thesis looks more into this problem by proposing a methodology to analyze the impact of the variability in the user's behavior on the net profit of cloud service providers. It digs more into the techniques to accurately predict the user' behavior and to optimally estimate the resources to maximize the net profit.

In the next section, we elaborate the problem statement related to the allocation of resources in an environment where user can relinquish their resources at any point in time. Then, based on the problem statement, we devise three research objectives followed by the methodology which explains how these objectives will be reached. Finally, we describe our contributions and present an outline of the thesis.

1.1 Motivation and problem statement

Cloud computing is deemed as a revolution in the computing world as it changes Capital Expenditure (Capex) to Operating Expenses (Opex) by applying a pay-as-you-go pricing model. Users request resources for a time period and get charged for that specific period. The study is carried out on the context that if a user relinquishes before its scheduled end time, he only pays for the time the service was used.

Moreover, as mentioned in [9], the customers (end-users) never fully utilize the allocated resources for running their application. Therefore, the application providers overbook the customers and rely on infrastructure providers to reserve and scale the resources to ensure the service availability in case their users would want to fully utilize the allocated resources. For example, if a user requests the on-demand premium video service for one month from Netflix [10], the application provider (Netflix) reserves the resources required through the CSPs (infrastructure providers like Amazon Cloud Services [11]). Amazon reserves the resources for the total duration of the requested service and charges Netflix according to pay-as-you-go model. If the customers of Netflix disconnect or stop using the service, Netflix will relinquish the resources from Amazon (CSP) before schedule end time and pay according to the resources used. This kind of situation is not beneficial for the CSPs. This is because, CSPs cannot reuse the reserved resources over time as the cloud resources are perishable and lose their value if left un-utilized over time [12]. Thus, relinquishment of resources can possibly lead to a loss in revenue as the service provider may also have to deny other customers if the resource pool is contended or fully utilized.

Moreover, the arrival rate of users varies throughout the day leading to variable utilization of the data center servers. Servers may be over-provisioned during peak hours or under-provisioned during non-peak hours. Likewise, when a user relinquishes, the server remains idle until the next request arrives. This idle time could be utilized better (by assigning it to other requests for example) if the provider can accurately guess when the user is potentially going to relinquish. Due to these changeable trends, resources are not adequately utilized in either case. Accordingly, several models, such as [13] [14], have been developed to assign resources to improve utilization. However, they do not cover the economic aspects such as the income generated and the electricity expenses incurred by the provider. Additionally, various techno-economical models have also been proposed to analyze the costs but they do not consider the loss suffered by the provider when users leave before their schedule end time. Hence, it becomes important to have a tool which incorporates all the parameters discussed above and provide a complete cost-benefit analysis on the techno-economical aspects of a cloud service provider.

Further, for the optimal assignment of resources, the CSP must be aware of the user behavior, that is, the duration for which the user would potentially use the resources. To deal with this, various algorithms have been proposed to predict the user behavior by simply computing some sort of average based on the users' history [13] [14]. Although these algorithms produced good results, they are not reliable enough in terms accuracy due to their use of average approximation algorithms to predict the user behavior. To have a steadfast analysis and make better decisions, the accuracy of the predictions needs to be improved. Nowadays, machine learning is considered as one of the best approaches to make predictions in any field of study. Policies based upon prediction through any machine learning algorithms can prove to be handy for optimal resource assignment to maximize the net profit of cloud service provider. Furthermore, in an environment where users can relinquish, this becomes more complex and leads to the subtle stream of interdisciplinary research between computer science and economics. Thus, the scope of this thesis is to study the techno-economical aspects from the infrastructure CSPs' perspective based upon the net profit earned by CSPs through different resource allocation policies. In addition to that, a framework is proposed to predict user behavior and find the optimal amount of resources to be allocated so that the net profit can be maximized. Figure 1.1 shows the general architecture of the cloud used in this study.

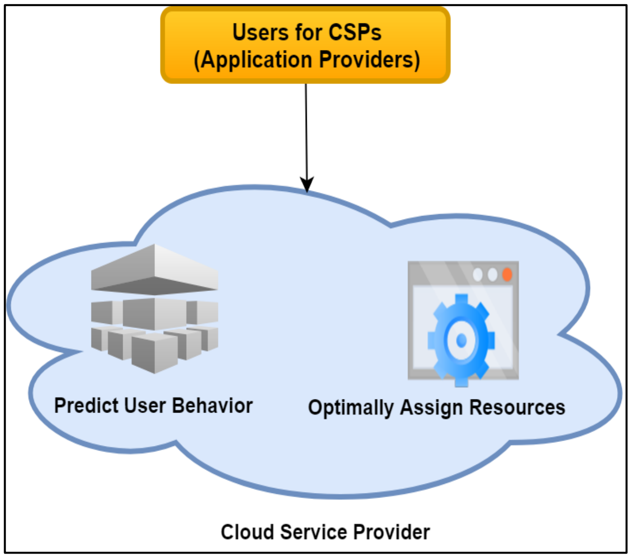


Figure 1.1 Architecture for profit maximization of CSP

1.2 Research objectives

Based on the problem statement described above, we will now present the research objectives. The main goal of this thesis is to develop tools and models to guide the CSPs in the complex task of assigning resources in an environment where users can abruptly leave the system at any point in time. More precisely, we want to address the following three sub-objectives:

a) Develop a model to calculate the total profit earned by cloud providers. Given a set of requests, the model should be able to evaluate the profit and provide a cost-benefit analysis from any resource assignment scheme available in the literature. The model should also account for the loss incurred when users leave the system.

b) Develop an enhanced prediction algorithm which can accurately predict the users' behavior. If better predictions can be made about the user behavior, then resources can be properly allocated.

c) Develop a mathematical model to assign the optimal amount of resources in order to maximize the net profit of the CSPs and at the same time try to entertain as many users as possible.

1.3 Methodology

The work presented in this thesis is built upon the quantitative research methodology by reviewing the literature related to the techno-economical aspects of a cloud service provider. Based upon that, the thesis proposes a novel cost-benefit analysis model that calculates the net profit of cloud service provider. To achieve this objective, we will formulate an equation which calculates the income, electricity expenses and the relinquishment loss incurred at each time. Further, to check its effectiveness, we will compare the net profits of different resource assignment schemes to check the impact of relinquishment of users.

To reach the second objective and improve predictions, we will first look at machine learning algorithms and see how they could be applied to our specific problem. We believe that linear regression, as suggested as future work in [14], could be an appropriate choice to improve the prediction accuracy. To evaluate the accuracy of the predictions, we will compare with other existing algorithms such as the ones proposed in [13] and [14].

The third objective is the culmination of this work and integrates the previous two models/algorithms within a coherent optimization problem. First, we will use the model derived from the first objective (i.e. the model to calculate net profit) and use it as an objective function to be maximized. Then, we will also define other constraints so that the model represents the reality of cloud providers. The prediction algorithm derived from the second objective will also be used. In the end, the goal of the model will be to optimally estimate the amount resources so that net profit is maximized. To evaluate the model, we will run simulation in CloudSim and compare the results with other state-of-art resource estimation techniques.

1.4 Contributions

Based on the objectives and the methodology mentioned above, this section summarizes the contributions of this thesis as follows:

- **Relinquishment-Aware Cloud Economic Model (RACE):** A profit calculation model that considers various parameters such as income, expenses in terms of energy consumption, and relinquishment loss in a cloud environment is proposed. This is a general pricing model that calculates all the above parameters over time for all types of resources being used. The resulting net profit shows the impact when users abruptly leave the system. Based upon its results, the model was presented at the following conference:

SINGH, S., AAZAM, M. and ST-HILAIRE, M., "RACE: Relinquishment-Aware Cloud Economics Model", *24th International Conference on Telecommunication* (ICT 2017), Limassol, Cyprus, pp. 1-6, May 2017.

- **Prediction algorithm based on linear regression to predict user behavior:** User behavior is predicted by the means of linear regression. The prediction results are then compared to previous algorithms proposed in the literature such as [13] and [14]. Results show that predictions based on linear regression are more accurate than the predictions made by previous approaches.

- **Optimization model to maximize the net profit:** The mathematical problem formulation of the cost-benefit analysis is re-used and the profit maximization problem is treated as an optimization problem. Through this, the resources are optimally allocated to user requests. In addition to that, an extra parameter is added to maximize the number of users being processed while simultaneously maximizing the net profit. The model is solved using LP (Linear Programming), which is an eminent theoretical technique that can find the optimal solution to an optimization problem with specific constraints.

1.5 Thesis outline

The rest of the thesis is organized as follows. In Chapter 2, background information is provided to better understand the remaining of the thesis followed by a review of the literature. Chapter 3 is divided into three sections. The first section aims to describe the approach to formulate the cost-benefit analysis model. The second section presents the theory of linear regression along with other techniques to predict the user behavior. The third section presents the mathematical model to optimize the resources to be allocated. Then, the results produced by the contributions from Chapter 3 are explained and presented in Chapter 4. Lastly, concluding remarks and the future work are mentioned in Chapter 5.

Chapter 2: Background and related work

This chapter presents the background and a literature review related to the thesis. More precisely, Section 2.1, discusses the current basic concepts and definitions prevalent in the field of cloud computing. In Section 2.2, the literature review relevant to the profit calculation model is presented. Then, in Section 2.3, basics of machine learning and applications of linear regression are discussed. Section 2.4 provides a survey of various cloud resource estimation techniques based upon user behavior and batch workloads. Finally, Section 2.5 compares the objectives of this thesis with the current state-of-art.

2.1 An overview of cloud computing

Cloud computing has been one of the favorite topic amongst the IT research community since 2007 [1]. Various multinational companies save enormous amounts of the money spent on Operating Systems (OS) and other hardware resources by using cloud services. The major reason being the elimination of buying and maintenance cost of the computing hardware and its storage space. Moreover, cloud computing satisfies its users with an aim of providing on-demand services for which users pay the CSP according to the pay-as-you-go model [2]. The resources are in the form of hardware resources (such as processor, memory, storage) and software (such as OS, applications) which are leased to the user over a network (Internet) [2].

2.1.1 Background of cloud computing

Generally, cloud computing is termed as cloud. The entire process at the backend of requesting and providing services is invisible. Various user interfaces are used to make this process easily operable. The simplest application of this process is the storage service. Users store data on the "cloud" and access it whenever and wherever it is required. At the backend, the data is stored in data centers owned by the CSPs [15]. The CSPs are usually the companies or the vendors that rent the resources publicly or privately using public and private clouds respectively.

In public clouds, any user can request and use the service. Services on the public cloud are free (example Gmail) or the CSP provides services and charges according to pay-as-you-go model. Even if the CSP owns and manages all the data stored by the user, there are lot of security concerns with this type of cloud [16]. A secure option is to use private clouds which are dedicated to only a single organization. The organization is solely dependent for managing the users using the services. Specifically, in-house data centers can be termed as private clouds. These clouds have high Capex and hence are not widely used. However, to mitigate the limitations of both public and private clouds, users prefer to use the hybrid clouds. In hybrid clouds, a part of the infrastructure runs on a private cloud and the other part runs on a public cloud. In this way, these clouds provide more flexibility as compared to private clouds while there is a tighter control over the data which makes it secure for the organizations using it [16]. The key players amongst the pool of cloud computing companies which provide these services are IBM, Amazon, Microsoft, Google and Salesforce [17].

The three types of clouds use the service oriented model that offers "Everything as a Service" (XaaS) [2]. The XaaS model can be elaborated into three major on-demand service models: Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS) [15]. Before discussing these services in detail, a brief description about virtualization is required for better understanding.

Virtualization is a key component which is widely used by the CSPs to offer these services. Authors in [18] even recognize virtualization as the major element in the success of cloud computing technology. In terms of cloud, virtualization allows the hardware resources (such as CPU, GPU, memory and storage) to be duplicated and partitioned into multiple independent Virtual Machines (VMs). A VM is an emulation of a physical machine [19]. It may be viewed as a logical processing environment overlaid on a physical device which acts like an actual computing machine with specified properties of its own CPU and run its own operating system. Further, the hardware can be shared to create multiple VM instances and even VM resources can also be shared amongst each other to assign tasks [20]. With the help of a single server, providers can accommodate several users by virtualizing resources and renting them. Unlike traditional distributed systems, cloud service providers rent these resources using a pay-as-you-go model [15].

Hence, making the most of virtualization technology, the following cloud services are offered:

- Infrastructure-as-a-Service (IaaS): The most widely used type of cloud service is IaaS in which users lease computational resources using various virtualization techniques. This service model provides on-demand virtual infrastructure to compute-oriented users in the form of VMs. The VMs are bought as a replacement of the computational resources in which users install various software to develop their applications on their own end. Amazon EC2 is the most prevalent example of IaaS. Moreover, the VMs are networked by the means of Internet Protocols (IPs) within the data center [21]. Moreover, according to the International Data Corporation (IDC), IaaS is the fastest growing form of cloud service as it best supports the on-demand access to the available resources in the cloud facility of the provider [22].

- Platform-as-a-Service (PaaS): PaaS users rent the higher-level services (say, software development frameworks). The platform uses the computing resources which typically are billed like IaaS products although the infrastructure is abstracted away below the platform [23]. The biggest example is the Google App Engine which provides PaaS services using GCP. Users can easily build their applications and run them even under heavy load [7].
- Software-as-a-Service (SaaS): In this type of service model, CSPs provide the running software and application to the users. SaaS products are available through the medium of web browsers or mobile applications (more prevalent these days). Customers only use the end products without taking care of the infrastructure or the development environment. All social media applications are examples of SaaS services. Other than that, Salesforce provides various Customer Relationship Management (CRM) tools for end users.

Apart from these services, cloud computing resources are characteristically elastic and scalable as well. The provider on its end can scale the resources as per the current resource demand. Moreover, the resources dedicated to each user can also be dynamically increased or decreased depending upon the user’s requirement. For example, if an application deployed on the cloud abruptly exhibits poor Quality of Service (QoS), resources can be scaled up to meet the QoS requirements. Similarly, if a user feels that the resources leased are not being fully used, it can relinquish the desired amount of resources at any time and pay only for the current usage. As discussed earlier, this activity on the providers’ end is not beneficial as they already had reserved the resources for the requested duration of the user. This can lead to a potential monetary loss for the provider. In Section 2.2, we will discuss the work of various researchers that was done on the economic aspects of cloud computing. Before that, we would present recent advancements in the field of cloud computing in the next sub-section.

2.1.2 Advancements in cloud computing

Cloud computing provides flexibility to both the providers as well as the users. This feature has opened many new possibilities for cloud computing companies and the organizations of other areas to collaborate with each other. From Internet of Things (IoT) to e-healthcare, cloud computing is being explored in every possible way. The major reason being the introduction of Mobile Cloud Computing (MCC) [24] which integrates the concept of cloud computing into mobile environment. Consequently, the gigantic amount of data (named Big Data) is being gathered and processed using the services mentioned in the previous sub-section. These three fields have collectively attracted the attention of researchers as a solution to green IT [25] whereas the CSPs eye the revenue generated through it [26].

2.1.2.1 Using cloud computing for IoT and e-healthcare

Talking more on it, *Botta et al.* [27] named the integration of cloud computing and IoT as *CloudIoT*. IoT is basically the network of sensors and devices interconnected with each other based on standard communication protocols [28] [29]. As claimed in [30] that IoT is going to be one of the major technology to bring revolution by 2025, its impact can already be clearly seen in day to day lives. Moving further, though the healthcare is totally a separate field, experts in this field believe that adopting IoT can significantly improve the quality of medical services. Thus, a lot of mobile devices suited for health information delivery are being used to automate the process of collecting and delivering the real-time data [31]. Therefore, e-health has turned out to be one of the widely used applications of IoT which uses Wireless Body Sensor Networks (WBSNs) to communicate with each other [12]. Further, new service oriented paradigms have been by enabled CloudIoT as an extension to XaaS. These are Sensing and Auction-as-a-Service (SAaaS) [32], Sensor Event-as-a-Service (SEaaS) [33], Video Surveillance-as-a-Service (VSaaS) [34] and many more. The communication of IoT devices has become a source of Big Data which made it obligatory to blend IoT with cloud (CloudIoT) to process that data [35]. This blend also has introduced the new resource management techniques associated to optimizing utilization and maximizing the profit on the CSPs' end.

2.1.2.2 Cloud federation

With the growing amount of data going through the internet, the resources from a single provider may not be sufficient enough. As a result, the number of CSPs is also growing to provide computing and storage services. Moreover, with more users migrating to the cloud, CSPs are collectively looking to provide flexible services. Therefore, by adopting an evolutionary approach called cloud federation, CSPs are reshaping their business structures to dynamically scale the resources by interconnecting, cooperating and sharing the resources with each other [36]. Cloud federation includes services from different CSPs technically combined into a single pool of resources. It supports three basic interoperability features at the backend: resource migration, resource redundancy and combination of complementary resource respective services [37]. Not only beneficial for the users, this approach has tempted many CSPs for better utilization of their resources and increasing their profit by processing more users. Figure 2.1 shows an overview of a cloud federation. We use the similar cloud architecture to carry out the study in this thesis.

As can be seen, different CSPs provide their services from different geographical locations. All the user requests they receive and process are also come from various parts of the world. In this situation, it becomes much more complex to manage these requests in

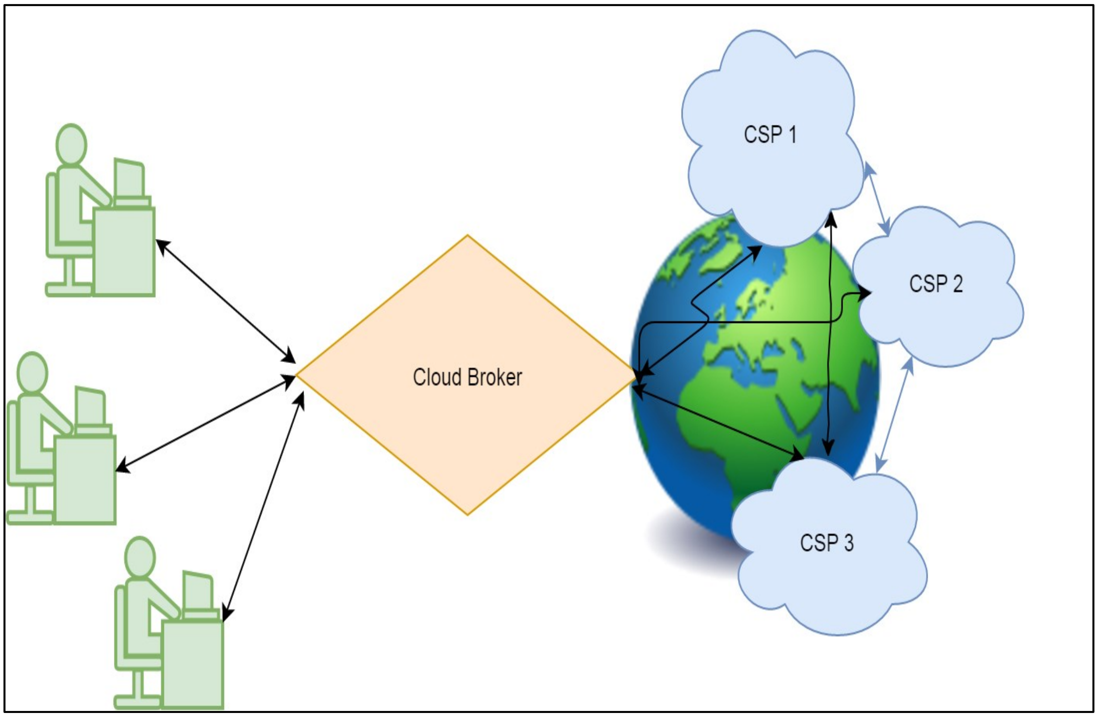


Figure 2.1: Overview of cloud federation process

a cloud federation environment. Therefore, to maintain the quality of service and monitor the provisioning of resources, an administrator is required. Hence, in this evolution of business process, a strong and intelligent system is required to handle and manage the dynamic changes in a cloud federation. To ensure this, CSPs use cloud brokers as a key component to make end to end communications [38]. The broker acts as a centralized control point to manage end to end communication of users and CSPs [39]. The details of the cloud broker are discussed in the following sub-section.

2.1.2.3 Cloud broker

"The broker is a third-party agent used which acts an intermediate between the purchasing and the selling parties to make the negotiations so that both the parties agree with each other for a business agreement [40]*".*

The history of using brokers in computing is not new. The concept has already been used in distributed computing to monitor the jobs being processed at grid sites on the behalf of users [40]. With the advancements in computing, from distributed to cloud and from cloud to federation of clouds, the concept of broker has added a new value to carry out the business in utility computing. By definition, a cloud broker is an entity used by CSPs in a cloud federation environment to make negotiation with the cloud users. Various researchers have proposed their "own" brokers which carry out the resources estimation [13], manage SLA requirements [41], satisfy QoS [42] and negotiate resource prices to help CSPs in a federation to reach an agreement with the users. Moreover, on the users end, brokers help them to discover and compare different services offered by the CSPs and chose the best one in the federation. The cloud broker earns its profit by satisfying

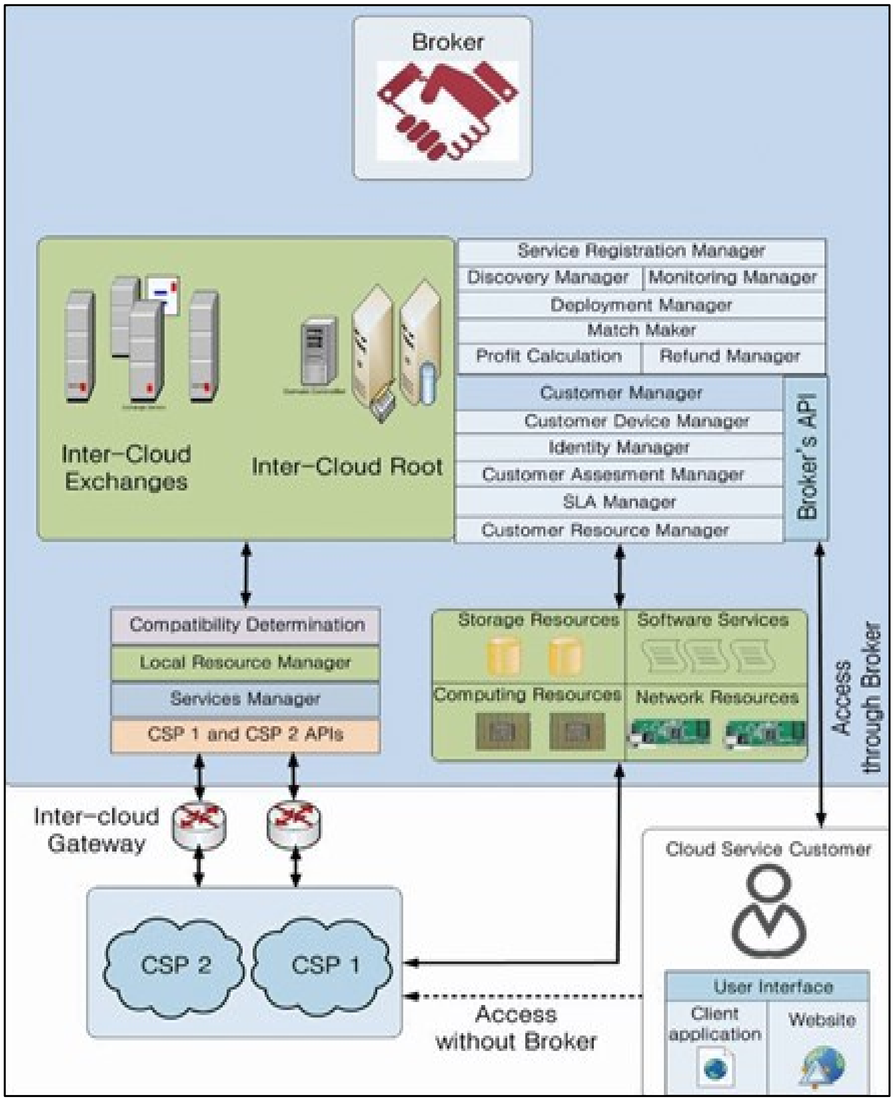


Figure 2.2: Broker's architecture **[13]**

requirements of both parties. It uses methods like data sharing and integration across data sharing services to achieve the best possible deal [2]. Figure 2.2 shows the communication between the user, the CSP and the broker.

As seen in Figure 2.2, the broker has various properties for building the relationship between a CSP and the user. The key properties are briefly discussed below:

- **Compatibility determination:** Checks the compatibility of user request with available resources.
- **Customer manager:** Manages the types of users' devices.
- **Customer assessment manager:** Fetches the history of user from the respective CSP.
- **Customer resource manager:** Estimates the amount of resources to be allocated.
- **Discovery manager:** Discovers CSPs with suitable resources for a use request.
- **Deployment manager:** Deploys infrastructure for CSPs.
- **Identity manager:** Responsible for the admission control.
- **Local resource manager:** Manages the resources required for local processing (broker itself).
- **Monitoring manager:** Monitors the provisioning of resources (deadlines, service times).

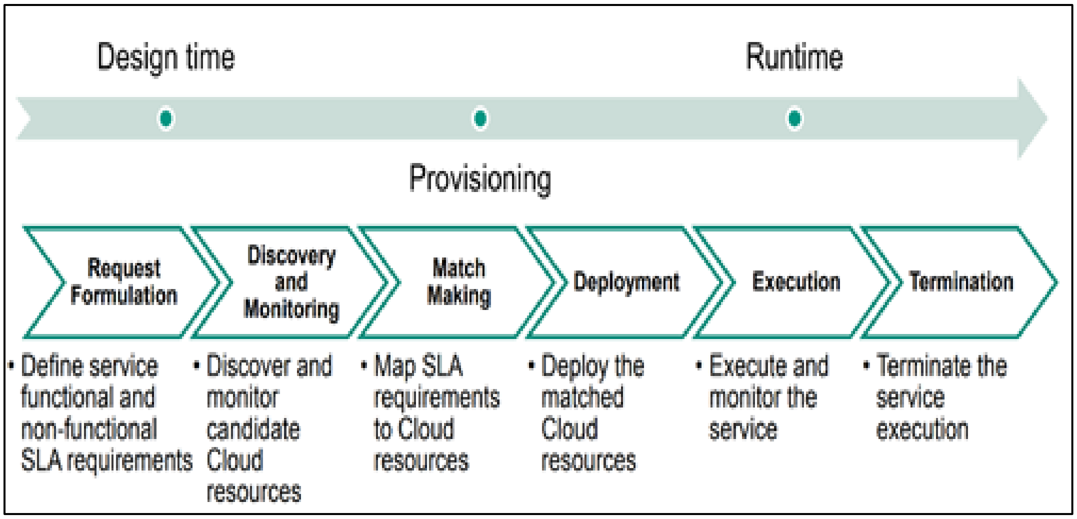

Design time
Runtime
Provisioning
Request Formulation
Discovery and Monitoring
Match Making
Deployment
Execution
Termination
• Define service functional and non-functional SLA requirements
• Discover and monitor candidate Cloud resources
• Map SLA requirements to Cloud resources
• Deploy the matched Cloud resources
• Execute and monitor the service
• Terminate the service execution

Figure 2.3: Lifecycle of a cloud broker **[43]**

- **Match-maker:** Makes use of various algorithms to match the user request with CSPs.
- **Service registration manager**: Registers the service when broker receives a service from cloud provider to allocate it to user.
- **SLA manager:** Finalizes the service level agreement between both the parties.

The cloud broker uses APIs and a standard abstract API to perform the tasks using these properties. The interaction of CSPs and the cloud broker happens through the inter-cloud gateway which is a kind of frontend for CSPs used to manage and monitor the APIs [43]. Further, Figure 2.3 depicts the steps of the service brokering cycle for a cloud.

2.2 Techniques to estimate the net profit in clouds

To analyze the service provider’s earned profit, various revenue calculation methods are proposed that are either based on utilization or the energy consumption of the data centers. For example, the authors in [13] and [14] use the concept of relinquish probability (how likely the user leaves the service before the schedule end time) to prevent the loss associated to relinquishment of users. In [14], Hu proposes a reactive resource management model which allocates resources based on historical records of the customers and current server utilization. Their results showcase an increase in server utilization in a finite resource pool. However, their income-only calculation technique gives a foggy idea on the net profit of the CSP.

Aazam *et al.* [13] introduced a pricing mechanism and a resource allocation method using the concept of relinquishing probabilities. Their work determines the price of the resource according to the history of the user. Additionally, they include the income and refund mechanisms to determine the net profit. The work is majorly done the users who reserve their resources earlier and pay the premium. When they call off the service, the refund is given by the CSP. Their pricing methodology is different but not able to decide whether their model is better in terms of profit margin or not when users call off the service. In [44], Emeras *et al.* made a cost analysis of running a workload on a cloud as compared to processing it on local high-performance computing platforms. A pricing model was proposed to compare the costs of running high performance computing applications on local clusters of an in-house facility with Amazon EC2 instances. The goal of the work was to find the best investment options and cost saving aspects from a CSC's prospective. It showed that it is cost-effective to run a workload demanding high performance computing (HPC) on a local premise as compared to running on a rented EC2 instance. Hence, cost breakdown to analyze the profit from CSP's end is missing. Macias *et al.* [45] introduced a resource manager to fulfill the business level objective of increasing profit and acts as a bidirectional communication channel to share revenue information between the broker and the resource manager. This knowledge is further used by the resource manager to optimize the allocation of resources to increase profit and prevent SLA violations at the same time. The authors calculated revenue as a function of penalty in which the cloud provider pays if it violates the SLA to achieve its business level objective. Nevertheless, this work does not focus on the loss incurred by the cloud provider if the user relinquishes. Likewise, it lacks to model the power consumption which prevents it from being a complete revenue calculation model.

Other than this, Mazzuco *et al.* [46] maximize the earnings of cloud providers by putting forward a method which allocates the resources via energy aware policies. To do so, they calculate the revenue by only considering the electricity cost as the major expense which intensely depends upon the number of actively running servers. The results of the paper suggest that decisions, such as how many servers should be switched on, can have a significant effect on the revenue earned by the provider. Another framework to calculate the total profit was adopted by Xu *et al.* [47] in which they calculate the income at each unit of time using different pricing policies. The pricing is dependent upon the state of the system in which it is running. Hence, the related losses are discussed. Further, using the dynamic pricing, an infinite horizon revenue maximization framework was presented by the authors. An average reward dynamic program was formulated for the infinite horizon case. Its optimality conditions and structural results showed that the revenue could be maximized by increasing the optimal price at low arrival rates.

Cui *et al.* [48] present a pricing model which calculates the revenue as a function of the service performance. Income is measured in the form of reward and electricity expenses and the expected income if the cloud provider fails to execute a task is also considered. Therefore, the revenue calculated in this study reflects the performance errors made by the cloud providers but ignores the loss due to user relinquishment. Further, the study reveals that the system static power plays a critical role in the tradeoff between the desired level of fault-tolerance, profit maximization and energy consumption. Babu *et al.* [49] used a revenue calculation technique which uses cooperation and competition amongst the CSPs as its major metrics. The revenue generated from the competition approach was formulated using game theory. The pricing mechanism is treated as a game where provider guesses the selling price of the requested resource. If the user is satisfied with the price, he is charged with the guessed price else the price keeps on updating according to the state of the market. The results of the study show that for a single cloud, this approach generates higher revenue. Melendez *et al.* [50] introduced the concept of blocking ratio which they use to increase the total resources after a certain threshold of the blocking probability is reached. To scale the resources, they use the term Grade of Service, which is based on the workload and revenue. Workload is the total number of users in the cloud to be supported and revenue is a function of operational cost that is increased by the proposed method. Hence, the total resources are automatically scaled and the extra step of a priori capacity analysis is eliminated.

Further, studies have also been carried out on analyzing the techno-economical aspects of CSPs in a cloud federation. Hadji *et al.* [51] carried out a cost analysis of the CSPs in a cloud federation. The work focuses to opt for a light federation where providers are not forced to engage in a complex and compelling federation level agreement. They suggest that providers should only declare the prices to each other and locally make outsourcing and insourcing of the resources. So, they followed the approach of calculating the income considering the insourcing and outsourcing dimensions. Insourcing means that the user requests is processed in the CSPs own facility while outsourcing stands for the resources shared with the other CSPs in the cloud federation. Further, they estimate the electricity expenses of the CSP just by including the electricity consumed by the number of active servers to jointly serve the insourced and outsourced requests. Although the results of this study maximize the revenue, they do not consider the cases where the incoming rate of the users affects the utilization. Hence, the electricity consumed by the servers during the idle durations is ignored. Besides that, Lampe *et al.* [52] carried out an auctioning approach to increase the net profit of the service provider. To calculate the revenue generated, they consider the income and fixed and operating costs of the physical and virtual machines respectively. Interestingly, the income from a user includes the cost of allocated resources at a given price of resource finalized during the bid. With these parameters, they formulate a pricing model and optimize it to find the optimal price of the resource which can maximize the revenue. Their approach was split into two phases, VM pricing and VM distribution. The results determined the equilibrium prices for all VM types such that the profit from the served bids was maximized. Subsequently, the virtual resources requested in these bids were cost-efficiently distributed across the physical hosts.

2.3 Machine learning and its applications

Machine learning is the domain of computational intelligence which is concerned with the question of how to construct computer programs that automatically improve with experience [53]. This technology is directly related to statistics as the predictions are made by analyzing the historical data. The predictions are made by using various self-learning algorithms. These algorithms improve their computational efficiency over the time series. The time series is a set sequence of observations which are generally ordered in time. The daily life examples of time series are recording the temperature, power consumption, web logs, activity of sales, history of users and many more. The authors in [54] claim that making these algorithms learn complex relationships and patterns from empirical data from time series can help us to make accurate or closer to accurate choices.

2.3.1 Supervised and unsupervised learning

The authors in [55] mentioned that machine learning can be classified into two types: supervised and unsupervised learning. Supervised learning is used to deduce a functional relationship from training data that generalises well to testing data. The supervised learning algorithms are inputted with the class of examples which happened over time, linearly or non-linearly [56]. This type of learning is widely used in the form of regression problems. The regression problems use the relationship between historical inputs and their corresponding outputs to predict the continuous output values. Regression typically involves linear regression, neural networks and support vector machine algorithms to carry out the predictions. On the other hand, unsupervised learning seeks to discover relationships between samples or reveal the latent variables behind the observations. The most widely used unsupervised learning approach is cluster analysis which is used to find hidden patterns in data and form clusters. The clusters consist of similar kind of data defined on metrics such as Euclidean or probabilistic distances [57].

2.3.2 Use of linear regression in cloud computing

Linear regression is a statistical modeling technique used to describe a continuous response variable as a function of one or more predictor variables [58]. Apparently, this technique has been used in various areas to deduce better predictions. In other words, it has opened a new path to merge statistics with computer science. Cloud federation is a new and trending area but is prone to various security issues, says Salman *et al.* [59]. Therefore, they used linear regression to detect and categorize anomalies in Intrusion Detection Systems (IDS) in cloud federation environments. The results showed more than 99%

detection accuracy and categorization accuracy of 93.6%. Belusso *et al.* [60] presented their entire work in Spanish with English only title and abstract. The information in their abstract mentions that they used linear regression to model the process of resources on the CSP end. The work provides a solution which aims to help CSPs to select the best instance to deploy and execute integrations solutions in the cloud. Moreover, linear regression has also proved to be a great tool to carry out green computing. The study of Farahnakian *et al.* [61] predicts the future utilization of the hosts to carry out the VM migration. If linear regression predicted the CPU to be underloaded, then the migration was carried out and the underloaded server was turned off to sleep mode. The simulations carried out in CloudSim showed that 51% of energy can be saved by using this approach in VM migrations. More details about linear regression will be discussed in Chapter 3.

2.4 Resource management in cloud computing

With the emergence of cloud federation, the resource pool has become huge but it is still limited since it must be shared by multiple users at once. No matter which type of cloud is deployed, efficient resource management strategies are needed to harness the power of the underlying resource pool of the cloud. Through effective resource management strategies, providers can increase the profit earned while improving the resource utilization at the same time. There is a rich literature available on the resource management of clouds focusing on various areas. In context to this thesis, we present the studies that have used various predictions schemes and used batch workloads for allocating resources to maximize the net profit.

2.4.1 Resource management in clouds using various prediction techniques

Several researchers have used prediction techniques to estimate the amount of resources to be dynamically allocated. For example, Aazam *et al.* [13] carried out a unique study in which they predict the amount of resources to be allocated to the user based upon its historical usage records. The authors suggest that the relinquish probability of the users must be checked before allocating all the resources that are requested by the users. Based upon the historical relinquish probabilities, they predicted the current relinquish probability which is used to estimate the amount of resources to be allocated. The history of the user is divided into two parts: Service Oriented Probabilities (SOP) and Average Overall Probabilities (AOP). SOP is the probability of using the requested service and AOP is the average probability of using all the services collectively. Both these probabilities are used in an approximation algorithm to make predictions. Variance is an additional parameter used to mitigate the fluctuations in the user behavior. This unique work sets the basis of our study to predict include user behavior as an important parameter for estimating the resources. Moving along, Hu [14] improved the model from [31] by considering an extra parameter (current utilization of the system) before assigning resources. For example, the amount of allocated resources will be greater in case of low server utilization but smaller when the server is over utilized. Overall, this study managed to improve the server utilization but did not mentioned anything about the net profit of the provider.

Putting more weight on the same idea of [13] and [14], Moreno *et al.* [62] says that user behavior is important to study in characterizing the workload of the system. Their results show that the task and the user dimension vary significantly every day. This dynamicity and diversity of users have a momentous impact on the resource utilization and energy costs. Furthermore, they claim that in most of the cases, users overestimate the amount of resources requested. This overestimation impacts the utilization if they end the service before their requested duration. Based upon that, authors tend to improve the utilization and energy efficiency by understanding the relationship between the user and the kind of tasks within a workload. Similar kind of study was presented by Fang *et al.* [63] in which they predict the load using an Autoregressive Integrated Moving Average (ARIMA) model. The resources were allocated according to the normal and sudden spikes in the utilization. For predicted normal workload, the resources were scaled accordingly while the coarse-grained capacity scaling approach was used in case of sudden spikes. The main idea of their study was to propose a prediction framework which can handle sudden hike of the workload during peak times.

Sadeka *et al.* [64] use time-series analysis for adaptive resource allocation. A neural network algorithm was used to predict the future surge in resource demands for proactive scaling of the resources. The future utilization was predicted for a varying sliding window sizes to assure the accurate predictions with respect of time. The accuracy was further tested by the means of error correction methods for neural network algorithms. Resource scaling improved the performance and generated higher profit for the CSP. While most of the prediction studies were generally carried out for the resources like CPUs, memory and storage, Truong-Huu *et al.* [65] proposed an algorithm to handle the uncertainties in the demand of bandwidths to maximize the revenue of the CSP. They modeled the bandwidth allocation problem as a Markov Decision Process (MDP). The demand predictor uses the algorithm based upon Bellman equation and solves the MDP problem to estimate the bandwidth to be allocated for a future time slots. The inputs for the model to make predictions was uncertainty of resource utilization of reservation requests and future demands of on-demand flexible requests. The results showed a 27% surge in revenue as compared to the techniques which do not use the predictions for future demands.

2.4.2 Resource management schemes for batch workloads

The techniques described in this sub-section carry out the resource management in cloud computing for a batch workload to maintain the resource utilization and improve the profit of CSPs. Authors in [66] mention that the batch jobs are generally delay tolerant up to a relatively loose deadline. The deadlines can be met until the fulfillment ratio is guaranteed. The fulfillment ratio is the ratio of execution time of a job to its scheduling duration. The prices charged by the provider are set according to the fulfillment ratio. High prices were charged for high fulfillment ratio (on-demand service) while users pay lesser price for low fulfillment ratio (spot instances). The problem stated in their study addresses the lack of cost effective solution for processing the batch of users with deadline guarantees. As a result, the authors propose a flexible instance which uses service fulfillment ratio as a pricing factor and at the same time guarantees the jobs to be executed within a given deadline. The users are free to decide the fulfillment ratio and they will be charged accordingly. To automatically adapt resource prices to the demand supply relation and to maximize provider revenue, pricing schemes were derived from a well-designed pricing curve and the Nash bargaining solution, respectively. Moreover, the flexible instance allows providers to utilize the ideal resources as users can ask to scale up the resources at a reasonable price. The results showed the increase in utilization which eventually increased the net profit of the provider. Besides, electricity cost has become a big concern of commercial cloud service providers with the rapid expansion of network-based cloud computing. Hence, Li *et al.* [67] addresses the issue of electricity expenses by limiting the delay in response by following the price-sensitive and cooling efficiency-enabled batch computing workload dispatch approach. The results of a Mixed Integer Programming (MIP) -based resource demand management solution show that aggregating the batch requests with similar deadlines can decrease the energy consumption by 30%. In the long run, it adds up to the saving of the CSP and increases the net profit. Furthermore, authors in [68] introduce an online learning algorithm to allocate the resources to address the trade-off between the computation cost of the provider and resources allocated to the batch jobs to maintain the performance. The algorithm incorporates history of spot prices and workload characteristics to allocate the resources to meet the performance requirements. Authors compare their approach with a hindsight (optimal) approach. The aim of the study focuses on minimizing the regret on the CSP end by minimizing the difference between the cumulative performance of the sequence of its decisions and the cumulative performance of the best fixed decision in hindsight. The outcomes showed that the difference between the regret of both approaches for allocating resources to a batch converges to zero with the proposed approach acquiring a single rate-centric policy with fixed bid. This pricing policy is a result of learning from the history of the highly variable spot and bid prices.

2.5 Comparison of proposed research with related work

Different resource management schemes for predicting the future workload, processing batch request and maximizing the revenue of CSPs were discussed in the previous sections. Various researchers in the literature talked about the problem of maximizing the net profit on the CSPs' end by addressing the issues of idle resources being un-utilized.. On the other hand, the studies focused on managing the execution delays have also been proposed in the past to reduce electricity loss and increase net profit while maintaining the performance.

However, in case of using prediction techniques, most of the work is done for predicting the collective workload. A couple of studies worked around predicting user behavior but either failed to improve overall utilization or lacked a mechanism which could maximize the CSPs' net profit. Moreover, in Section 2.2, none of the profit calculation techniques included the loss incurred by the CSP when the user leaves the service before the scheduled end time. Furthermore, the works with batch processing do not carry out prediction of user behavior to allocate resources. Instead, most of the researchers worked in the area of scheduling the jobs after the batch is processed. Additionally, the techniques using the batch workload showed an increase in CSPs' profits by scaling the resources. However, none of the approaches increased the profit by considering the finite capacity environment of a CSP. The reason being, as mentioned in Section 2.1.2.2, even with the federation of clouds, the collective resource pool is still finite. Therefore, it becomes necessary to propose a resource management mechanism which maximizes the net profit in a finite capacity environment. This is the focus of attention for this thesis along with objective of maintaining the market value by maximizing the number of users to be processed in each batch All in all, none of the existing works have dealt with all the aspects of the problems of concern of this thesis..

The key contributions of this study are to propose a complete profit calculation model which includes income, electricity expenses and most importantly the relinquishment loss. Then, using the techniques which can make better prediction of a user behavior, an optimization model is presented to increase the net profit of CSPs which considers the loss incurred when a user leaves before the requested duration. Most of the works related to batch processing entertain the jobs with heterogeneous properties. The issue with this is that individuals with different goals can lead to a situation where the profit of one goes to the disadvantage of the others [69]. Therefore, since the optimal allocation of resources in a finite capacity is a core concern of economics [69] and this thesis as well, the jobs in the batch are assumed to be homogeneous. Hence, the user behavior is made the key component to optimally assign the resources to the users in a batch. In summary, the thesis presents a novel study to overcome the limitations in resource management techniques of current state-of-art that are related to economics of cloud computing.

Chapter 3: Cost-benefit analysis and prediction based resource optimization model to maximize the net profit

As the cloud computing technology is emerging at a significant pace, more and more users are starting to rely on the cloud for day to day operations. Obviously, each user has its own behavior and preferences for their usage of the cloud. Users request resources to be consumed for a specific duration and can potentially relinquish before they fully utilize their resources for their scheduled requested duration. Since cloud providers in a cloud federation practically have a finite resource pool, an important consequence of this variation is that it introduces a vital optimization problem to be addressed regarding the optimal allocation of this finite data center capacity [70].

In this chapter, we first propose a cost-benefit analysis model (called RACE) for calculating the net profit. Then, the linear regression is used to improve predictions based on user history. In the concluding section of this chapter, considering the dynamic nature of user arrivals, the cost-benefit analysis model and the predictions based on user behavior are jointly used to derive an optimization model. The proposed model estimates the optimal amount of resources to be assigned to each user in a batch so that the profit for the CSP is maximized.

3.1 RACE model to analyze net profit

In this section, a new model to evaluate the net profit from the perspective of a cloud provider is proposed.

Since cloud-based computing is a market-driven process, all cloud providers tend to maximize profit by using optimized ways to assign the resources. Due to the perishable nature of resources, they cannot be reused if a user relinquishes. This is due to the fact that in on-demand services, the user does not pay a reservation premium at the start of the service [71]. With SLA providing them liberty to revoke the service at any moment, providers potentially lose some time which could have been allocated to other users. This time is comparable to monetary value as various expenses are still incurred when a user impulsively calls off a service [72]. In the scope of this work, these expenses include the electricity expenses for the time the system is idle. In addition to that, the provider also loses the income that could have been earned if the user had fully used the service. In economic terms, this income is called the expected income or the opportunity cost [73]. Hence, to minimize the opportunity cost loss, proper resource estimation is required. The loss associated to the missed opportunity cost is therefore termed as the relinquishment loss. Figure 3.1 shows the key parameters used to calculate net profit in this study.

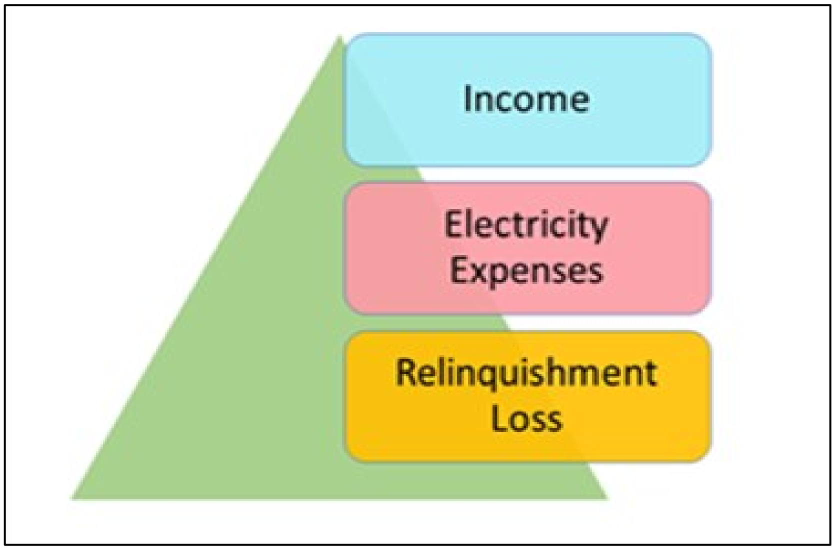


Figure 3.1: Key components used to calculate the net profit

Further, Equation 3.1 formulates the net profit (ȵ) earned by the CSP. The net profit is the difference between the income (ʅ) generated by allocating resources to the users and the expenses (Ɛ) required to operate the data center. Although other models, such as [74], [75] and [76], have been proposed to calculate income and overall profit using electricity and other general operating expenses such as software costs, network costs, they do not consider the loss in the form of opportunity cost specifically incurred from idle/un-utilized server due to user relinquishment. However, the along with the general operating expenses discussed above, the other costs of ownership such as building costs, facilities maintenance cost and employment costs are ignored in this model as the focus of the research is on evaluating the impact of opportunity cost loss due to user relinquishment. Therefore, the main contribution of the model is that it considers the summed-up loss incurred at each time when user relinquishes its service before the scheduled end time. Moreover, the income and electricity expenses at each time, that are dependent upon the current server utilization are also included in the proposed methodology. The proposed tool can be extremely useful for CSPs to evaluate the effect of different resource management schemes as it covers the noteworthy aspects related to income as well as expenses. Once they have a complete picture, they can select a scheme that eradicates the problem of underutilization and maintains high profit margins. In the next section, we present the notation used to expand this general equation to include all the related components.

$$ȵ = ʅ - Ɛ \tag{3.1}$$

3.1.1 Nomenclature

We propose the following formal notation which is composed of sets and input parameters.

Sets

- *I,* set of time slots where time slot $i \in I$
- *J,* set of all resource types j where $j \in J$
- *K,* set of user requests in the system where $k \in K$
- *L,* set of geographical locations of data centers in a cloud federation where $l \in L$

Input Parameters

- p_j, price for service j
- T_j^i, total amount of resources of type j being used in the cloud at time *i*
- C_l, unit price of electricity at location *l* in $/kWh
- P_l, power used to operate all the servers at location *l* when fully utilized
- u_j^i, utilization % of a resource of type j at time *i*
- t_r^k, time at which user *k* relinquishes where $t_r \in I$
- t_e^k, time at which user *k* was supposed to end where $t_e \in I$
- E^k, duration between the relinquishment time of user *k* and the next user arrival
- C_p, total electricity/power expenses
- C_{RL}, total relinquishment loss
- a_j^k, allocated resources of type j to user *k*
- t_c^k, time till which the relinquish loss for user *k* is considered where $t_c \in I$

3.1.2 Income

Most service providers will charge users based on the amount of resources they will consume (i.e. pay-as-you-go). Even though one of the economic interests of cloud computing is to convert Capex into Opex [77], the pay-as-you-go model does not fully benefit the cloud service providers in terms of financial aspects. Practically, the resource pool consists of heterogeneous virtual resources typically including CPU, memory, GPU and storage. To generate revenue from the users requesting resources, the provider estimates the amount of resources using different allocation schemes and provision them accordingly. This implies that distinct types of resources collectively generate revenue over time. Further, if all the user requests are provided the on-demand services, resources are variably utilized over time as users have the liberty to relinquish the service at any time. Thus, the overall income can be calculated by checking the amount of allocated resources at each point in time. Likewise, authors in [78] state that income is the product of the total quantity of resources allocated, the price per unit time usage and the duration for which it is used. Therefore, to calculate the income generated from the allocated resources for a specific duration, we use Equation (3.2) which is the product of the price and the amount of allocated resources at each time unit.

$$\mathfrak{l} = \sum_{j \in J} p_j \sum_{i \in I} T_j^i \tag{3.2}$$

3.1.3 Electricity expenses

Cloud computing companies like Amazon, Google and Microsoft have invested a large amount of money to deploy their data centers in various parts of the world. These data centers contain several servers working 24×7 to make virtual resources available to be leased within or across the organizational boundaries. When the servers are turned on, no matter if they are over utilized, moderately utilized or even if they are idle, there is a certain fixed cost for the constant consumption of electricity to operate them [79].

Moreover, as server utilization and energy consumption are highly coupled, the electricity cost is also dependent upon the amount of resources being used, the number of servers running in the cloud, the unit price of electricity and the power consumed per server [76] [80] [81]. In addition to that, the authors from [82] state that the prices of electricity depend upon the location of the data center as well. Therefore, the geographical location in which the data center is located also plays a vital role in calculating electricity expenses of a cloud provider. Moreover, as a matter of fact, the unit price of electricity charged during different times of the day would also vary at each geographical location which may affect the total electricity expenses of the provider. As discussed earlier, the amount of resources used at each time unit varies resulting in frequent changes in the server utilization. Hence, incorporating all the factors that contribute to the electricity expenses is important. The electricity expenses for the utilized resources of the data center can be derived as:

$$C_p = \sum_{l \in L} (C_l \times P_l) \sum_{j \in J} \sum_{i \in I} u_j^i \tag{3.3}$$

$$\text{where} \quad \boldsymbol{u_j^i} = \frac{T_j^i}{Total\ Resources} \tag{3.3a}$$

Algorithm 3.1 Calculating electricity expenses

Input: Start time, relinquish time/end time and resources allocated to each user

Output: Total Electricity expenses for *I* time units

```
for each user entered in $I$ do
    for all resources $J$ do
        allocatedResourses[i][j] ← Get allocated resources to each user from input
        for start time to relinquish time
            for all locations $L$ do
            resourceUsage [i][l] = sum of allocatedResources
            end for
        end for
    end for
end for
for all resourceUsage do
    Calculate utilization ($u_j^i$) using Equation 3.3a
    electricityExpense[i][l] = $C_l \times P_l \times u_j^i + \frac{2}{3}(1 - u_j^i)$
    sum electricityExpense
end for
```

The point to be noted is that if a system is not fully utilized, the provider still has to pay for the electricity of all the idle servers. According to [83], an un-utilized part of the server costs two-third of the electricity expenses as compared to a server running at its maximum utilization rate. For example, if a server is used at 60%, then the remaining 40% in idle/un-utilized state will use two-third of the total power. Consequently, the cost of electricity incurred over all these idle intervals will add up to the total electricity expenses. Therefore, the total cost of electricity as an expense can be written as:

$$C_p = \sum_{l \in L} (C_l \times P_l) \sum_{j \in J} \sum_{i \in I} \left\{ u_j^i + \frac{2}{3}\left(1 - u_j^i\right) \right\} \tag{3.4}$$

Equation 3.4 provides the total electricity expended paid by the cloud provider for the utilized and idle duration in a cloud federation. The process of calculating the electricity expenses is further explained in the Algorithm 3.1. As described in the Algorithm 3.1, with the given amount of allocated resources to each user, first the utilization at each is calculated which is further used to determine electricity expenses.

3.1.4 Relinquishment loss

Relinquishment loss is a form of opportunity cost loss. This cost is the amount that a CSP is expected to earn if the user had not relinquished. The cloud provider estimates and reserves resources on demand for each user request. However, when a user gives up his resources before the scheduled end time, the expected income after that time is considered as the opportunity cost for which the provider misses the chance to earn [84]. This expected income is termed as relinquishment loss and the approach to calculate this loss is discussed below. The relinquishment loss of the provider practically depends upon the time at which the next user arrives into the system. This is because the system is only

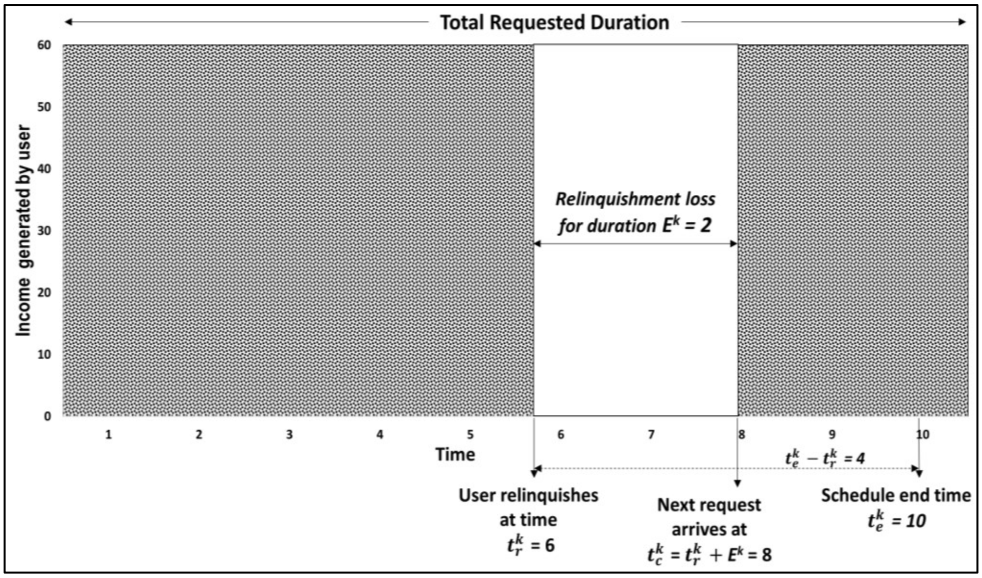


Figure 3.2: Relinquishment loss from a user when $E^k \leq (\boldsymbol{t_e^k} - \boldsymbol{t_r^k})$

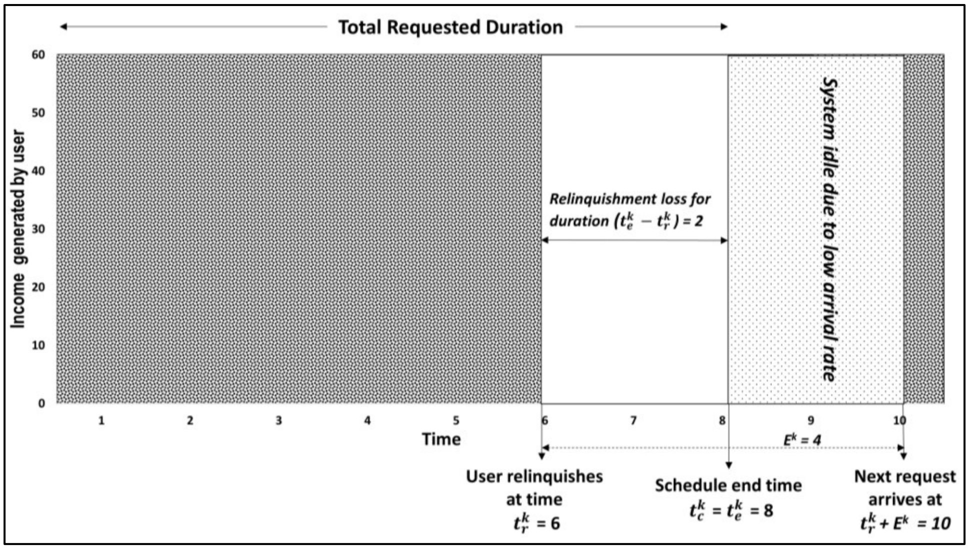


Figure 3.3: Relinquishment loss from a user when $E^k > (t_e^k - t_r^k)$

idle until the next request enters the system and not the duration for which the user was supposed to end its service. This kind of situation generally arises during peak hours when the arrival rate of user requests is high. As seen in Figure 3.2, the relinquishment loss is calculated only for the duration for which the resources released by the relinquished users are actually idle i.e. from time t_r^k =6 to t_c^k = 8. This is estimated according to the duration E^k. It would have led to an over estimation of the relinquishment loss value if the relinquish duration would have been considered from t_r^k to t_e^k instead of considering it till the next user arrives.

On the other hand, the situation may arise where the arrival rate is very low, say during off peaks. In these circumstances, the actual interarrival times (E^k) could be greater than the duration $(t_e^k - t_r^k)$. This is the duration from the time the user relinquishes till the time the user was supposed to end. In this case, if only the interarrival time is taken as the relinquish duration to calculate the loss, it can lead to an over estimation of the loss. As can be seen in Figure 3.3, the user leaves at time 6. If the relinquish duration would have

Algorithm 3.2 Calculating relinquishment loss

Input: Start time, relinquish time/end time and resources allocated to each user

Output: Total relinquishment loss for *I* time units

```
for all users in K do
    for all resources in J do
        allocatedResourses[k][j] ← Get allocated resources to each user from input
        if ($t_e$ [k] – $t_r$ [k]) ≥ (nextUserArrival – $t_r$ [k])
            $t_c$ = nextuserArrrival
        else
            $t_c$ = $t_e$
        end if
            for $t_r^k$ to $t_c^k$ do
                relinquishmentLoss [i][j] ± allocatedResourses[k][j] × $p_j$
            end for
    end for
end for
for all relinquishmentLoss do
    sum relinquishmentLoss
end for
```

been calculated with the approach discussed in Figure 3.2, then it would have led to an over estimation of the loss. Hence, as depicted in Figures 3.2 and 3.3, the minimum value between E^k and $(t_e^k - t_r^k)$ is considered to calculate the relinquishment loss from a user. To make it clearer, refer to Algorithm 3.2 which explains the implementation of how the loss is calculated. Parameter t_c^k represents the time till which the loss is calculated, where $t_c^k = min\ \{(t_r^k + E^k),\ t_e^k\}$. Thus, the total loss incurred due to the relinquishment of can be written as:

$$C_{RL} = \sum_{j \in J} p_j \sum_{k \in K} \sum_{t_r^k}^{t_c^k} a_j^k \qquad \{t_r^k, t_c^k\} \in I \tag{3.5}$$

Therefore, summing up the electricity expenses and the relinquishment loss, the total expenses which impact the net profit become:

$$\Sigma = \; C_p + \; C_{RL} \tag{3.6}$$

3.2 Various approaches to predict relinquish probability

In a cloud computing environment, resources are allocated to users based on demand. Many researchers have focused on the allocation problem based upon various aspects such as VM scheduling [85], VM migration [86], VM consolidation [87], QoS [88]. Many studies related to resource estimation also suggest that the amount of resources to be allocated could be estimated based upon the user's history and the server utilization. To make better resource estimation decisions, the behavior of users should be known. This way, it can be predicted how much resources will be consumed based upon the usage history.

Most of the cloud providers require the user to determine and specify the amount of resources required to process and complete the job. Experienced cloud customers may be able to gather historical data and properly determine the required amount of resources. However, most users may not have data to estimate the resource needs of their applications. Subsequently, these users request much more resources than what is needed for running their applications. For example, it has been observed that many userss tend to purchase 10 times the amount of resources than what is actually needed for the operation of their jobs, resulting in low server resource utilization [89]. As time goes on and users keep using services, the cloud provider stores information about the behavior of the users over time. Based upon their history, users can be categorized into three main categories: 1) loyal (or high-utility) users typically use the maximum portion of their requested duration; 2) average utility users use around half of their requested capacity and 3) disloyal (or low-utility) users typically relinquish shortly after the start of their service. However, according to the pay-as-you-go model, users have the freedom to relinquish resources at any point in time. This is not a beneficial activity for cloud service providers as they reserve resources for the whole requested duration.

Consequently, there is a lot of research performed on the prediction of the current server workload [90] [91] in order to scale resources [92] to increase net profit. However, there is minimal work done on anticipating the behavior (relinquish probability) of a user based upon his usage history. Thus, it is important to examine resource prediction approaches that can improve the prediction accuracy and therefore yield better server utilization and increased revenue. This section focuses on the comparison of different resource prediction methods such as linear regression, Broker-as-a-Service (BaaS) [13] and Reactive Prediction (RP) [14] for different profiles of users. All these methods use the history of the users to make predictions.

3.2.1 Broker-as-a-Service (BaaS) model to predict relinquish probability

Aazam *et al.* [13] predicted the amount of resources to be allocated to run a job requested by a returning user. The relinquish probability is used as a crucial factor to make decisions. To make predictions, the history of the users was used to calculate two metrics: Service Oriented Probability (SOP) and the Average Overall Probability (AOP). SOP is the average of the relinquish probabilities of a specific service while AOP is the average of the relinquish probabilities across all services. The following equation was used in their study to predict the user behavior:

$$r = ((1\text{-}SOP) - \sigma^2) * (1 - AOP) \quad (3.7)$$

Where *r* is the predicted probability of user to use the resources. Therefore, *r'* could be regarded as the predicted relinquish probability of a user where,

$$r' = 1 - \{((1\text{-}SOP) - \sigma^2) * (1 - AOP)\} \quad (3.8)$$

3.2.2 Reactive Prediction (RP) model to predict relinquish probability

Hu [14] proposed a different model to estimate the amount of resources to be allocated. His model is motivated by the model discussed in Sub-section 3.2.1 with slight changes to it. According to the history of the user and the current server utilization, he predicts the amount of resources needed to complete the job. Suppose the user has history length of *m*, therefore in order to estimate the amount of resources to be assigned, he predicts the relinquish probability of the user for $m{+}1^{th}$ time by the following equation, where *z* is the predicted relinquish probability

$$z = \frac{2}{3}\, SOP + \frac{1}{3}\, AOP \quad (3.9)$$

The logic behind this formulation is that the average based on the currently requested service (SOP) is a better indicator as it is based on the history of the same service.

3.2.3 Predicting user behavior using linear regression

Machine learning is a form of artificial intelligence [55] which uses different algorithms that continuously improve themselves using the input fed to them. Linear regression is one of the branches of machine learning and a form of supervised learning which involves the prediction of the value of a continuous variable based on one or more continuous variables. It generally quantifies the relationship between several input feature variables and a corresponding linear output variable. It is assumed that the output variable *y* is linearly related to the various input feature variables $\{x_1,..,x_p\}$, where *p* is the number of features used to make a single prediction output *y* . Equation 3.10 represents the testing sets for linear regression algorithm where *m* represents the length of the training set.

$$\mathbf{X} = \begin{vmatrix} x_{ip} \ \dots x_p \\ : \\ x_{mp} \ \dots x_{mp} \end{vmatrix} \quad \text{and} \quad \mathbf{Y} = \begin{vmatrix} y_i \\ : \\ y_m \end{vmatrix} \tag{3.10}$$

In our comparative study, *p=1* is considered as we use a single feature: history length of a single user. Univariate linear regression (linear regression technique for a single feature) is used to predict output *y* (relinquish probability) related linearly to feature *x*. In this algorithm, the results are predicted by mapping the previous continuous outputs to the inputs. In other words, the input is mapped to some continuous function to obtain a hypothesis, as shown in Figure 3.5 which helps to predict the desired output. For example, given a data set about the model year (feature) of different cars, one may want to predict the price of the car. Predicting price as a function of the model year is a continuous output.

To understand it better, we need to go through the cost function *J* of linear regression algorithm. For accurate predictions, the best possible slope (hypothesis) is required to be fit into the graph containing the training set where the Y-axis represents the previous outputs and inputs are plotted on the X-axis. Equation 3.11 below represents the

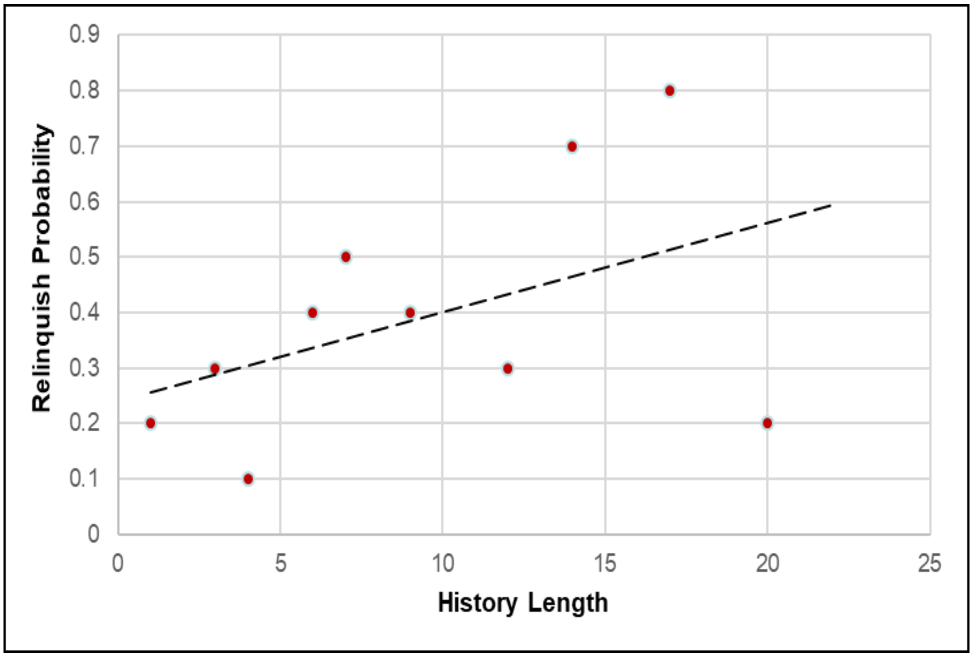


Figure 3.4: An example of data showing the relationship between relinquish probability and history length of a user returning for 21st time

dependent variable *y* and the independent variable *x,* where *x* and *y* are the elements of vectors *X* and *Y* respectively (as shown in Equation 3.10) and a_0 is the intercept point on the Y-axis and a_1 is the slope of the hypothesis in Figure 3.4

$$y = a_0 + a_1 x \tag{3.11}$$

In Equation 3.11, we can choose arbitrary values for coefficients a_0 and a_1, say a_0' and a_1'. With the initially chosen values of a_0 and a_1, arbitrary value of *y* in Equation 3.12 becomes $h_a(x_i)$ which represents the inaccurate hypothesis function obtained by the initial iteration. This value needs to be minimized further to get an accurate hypothesis.

$$h_a(x_i) = a'_0 + a'_1 x \tag{3.12}$$

The accurate value of $h_a(x_i)$ is obtained by minimizing the cost function $J(a_0, a_1)$. The cost function is a squared error function whose curve is in the form of a bowl with a single global minimum. The coefficients are solved by a least square method which estimates the best fitting straight line or hypothesis. The values of the hypothesis are the minimized error values as compared to the actual data. The cost function J is as follow:

$$J(a_0, a_1) = \frac{1}{2m} + \sum_{i=1}^{m} (h_a(x_i) - y_i)^2 \tag{3.13}$$

where m is the length of the training set (history) of the input feature. Here $(h_a(x_i) - y_i)^2$ is the squared difference between the predicted value ($h_a(x_i)$) from Equation 3.12 and y from Equation 3.11. This function is minimized to find the minimum values of a_0 and a_1

There are two methods to minimize the cost function J, namely Gradient Descent and Normal Equation method. Both methods are useful to solve the hypothesis function in their own way. The gradient descent method is generally used when we have multiple features p and primarily larger length of history m, where $m > 100k$ [93]. A multiple iterative process with desired learning rate is carried out to obtain global minimum for the function in this case. To figure out the correct learning rate, the algorithm is initially required to run for a few times [93]. Whereas, the normal equation method is efficient for a smaller history as it requires lesser computational time. Practically, we consider that users returning to the cloud providers will have history $m < 100k$. Therefore, the Normal Equation method is used in our study.

Normal equation method: In this minimization approach, optimal values of a are calculated analytically. Instead of carrying out many iterations, the derivative of the cost function presented in Equation 3.13 with respect to a is directly equated to zero to minimize the residual sum of squares [94]. Assume that the input with length $\{x_i \ldots . x_m\}$is loaded in a vector matrix X and the corresponding output in vector matrix Y. Therefore, after solving Equation 3.13, the least square solution of the normal equation becomes equivalent to:

$$a = (X^T X)^{-1} X^T Y \tag{3.14}$$

Here, X^T is the transpose of the matrix $X$and $(X^T X)^{-1}$ gives the inverse matrix vector where,

$$\mathbf{X} = \begin{vmatrix} 1 & x_i \\ 1 & \vdots \\ 1 & x_m \end{vmatrix} \quad and \quad \mathbf{Y} = \begin{vmatrix} y_i \\ \vdots \\ y_m \end{vmatrix}$$

An extra column of ones is added into the X vector because there must be at least two features to solve the normal equation matrix multiplication [93]. Hence, adding a column of ones as an additional feature would not affect the end predicted results. In the context of this study, for an incoming user returning for $m+1^{th}$ time, matrix X is loaded with the m number of times the user has returned (history length) for $\{x_i \ldots . x_m\}$. The results are depicted in the next chapter of this thesis.

3.3 Proposed optimization model for resource estimation

In this section, a mathematical model is proposed to increase the net profit of cloud providers in the context where users can leave the system at any point in time. The model is derived from the cost-benefit analysis model (RACE) that we presented in Section 3.1. As a reminder, RACE is used to calculate the overall net profit earned by the cloud providers using different resource allocation schemes. For the proposed optimization model, the parameters used to calculate the net profit in Section 3.1 are reused in context of estimating the optimal resources for the users with known history. The history represents the user behavior and based upon that the model estimates the amount of resources that should be assigned to maximize the net profit.

Equation 3.1 calculates the net profit of the cloud provider with user relinquishment being the principal factor which affects the resulting net profit. In the following sections, the RACE model is used to derive a profit function, denoted *P(r),* which is used to estimate the amount of resources to assign to a user request to maximize net profit. The resources are estimated based upon the predicted relinquish probability done by the linear regression model discussed in the previous section.

3.3.1 Extended nomenclature

To create the model and solve it, we extend the notation from Section 3.1 with the following additional information.

Set

- *N,* set of user requests in an incoming batch where $n \in N$

Input Parameters

- r^n, predicted value of the relinquish probability of the user n through linear regression
- D^n, requested duration by user n

Constants

- R_j^n, amount of resources requested by user n for service j
- V, remaining capacity of the cloud when an incoming batch is processed

Decision Variables

- a_j^n, amount of allocated resources of service j to a request n of the batch
- x_j^n, binary variable such that $x_j^n=1$ if and only if request n in the batch is allocated resource for service $j \in J$, otherwise $x_j^n = 0$ if the request is not processed

3.3.2 Objective function

The aim of the profit function presented in Equation 3.15 is to maximize the net profit by allocating the optimal amount of resources while considering the current available capacity of the resource pool, the relinquish probability, and the arrival rate of the requests. The decision variable in this problem is the amount of resources to be allocated to each incoming request (a_j^n). To fulfill this requirement, the model estimates the income, electricity expenses and relinquishment loss by using the predicted value of the relinquish probability of the user using linear regression.

$$\boldsymbol{P}(\boldsymbol{r}) = \sum_{j \in J} \{p_j \,(1 - r^n)D^n\}\, a_j^n \; - \sum_{l \in L}(C_l \times P_l) \sum_{j \in J} \; [\{(1 - r^n)D^n \, u_j^n \} \tag{3.15}$$

$$+ \{\frac{2}{3}\left(1 - u_j^n\right) r^n \;\; D^n\}\,] \; - \sum_{j \in J} \; D^n \, p_j \, a_j^n$$

$$\text{where} \quad u_j^n \; = \frac{a_j^n}{Total\ resources}$$

3.3.2.1 Discussion on the components of profit function $\boldsymbol{P}(\boldsymbol{r})$

The first term of the objective function in Equation 3.15 calculates the expected income generated for each user. When the users arrive with the amount of requested resources for a duration D^n and relinquish probability r^n, batches of user requests are created before they are processed. As seen in Equation 3.16, the model uses the price of each resource (p_j), relinquish probability and the requested duration of the users in the batch and processes all requests in the batch at once.

$$Income = \sum_{j \in J} (1 - r^n)D^n \, p_j \, a_j^n \qquad\qquad n \in N \tag{3.16}$$

At the same time, the predicted electricity expenses (*EE)* in Equation 3.17 is for the utilized duration is and calculated using the predicted usage duration $(1 - r^n)D^n$ while the expected electricity consumption for the idle duration is calculated for the duration $r^n \, D^n$.

$$EE = \sum_{l \in L} (C_l \times P_l) \sum_{j \in J} \left[\{(1 - r^n)\, D^n\, u_j^n \} + \{\frac{2}{3}(1 - u_j^n)\, r^n\, D^n \} \right] \quad n \in N \tag{3.17}$$

As far as the relinquishment loss generated from a given request is concerned in Equation 3.18, the model uses the predicted duration ($r^n\, D^n$) for which the user would not use the service.

$$Relinquishment\ Loss = \sum_{j \in J} p_j\, a_j^n\, r^n\, D^n \qquad n \in N \tag{3.18}$$

Moreover, the provider would always want to process the maximum number of requests as the users may eventually become loyal in future. Therefore, the sum of the all binary variables (x_j^n) should also be maximized where x_j^n is assigned the binary value of *one* if user in a batch is processed. This signifies that the maximum number of users should be accommodated along with maximizing profit. Hence, the objective function of the model becomes:

$$\max\ (\ \mathrm{P(r)} + \sum_{n \in N} x_j^n\ ) \qquad j\ \in\ J\,, x\ \in \{0{,}1\} \tag{3.19}$$

3.3.3 Constraints

The objective function shown in Equation 3.19 is subject to different constraints expressed in Equations 3.20, 3.21 and 3.22:

$$\sum_{n \in N} a_j^n \leq V \qquad j \in J \tag{3.20}$$

$$a_j^n \leq R_j^n \qquad j \in J, \; n \in N \tag{3.21}$$

$$a_j^n \geq x_j^n * R_j^n * 0.2 \quad x \in \{0,1\}, \; j \in J, n \in N \tag{3.22}$$

Equation 3.20 specifies that the sum of all allocated resources for the entire batch must be less than or equal to the current resource capacity (V) in the system. In addition to this, it is important that on the verge on maximizing profit, resources must not be overestimated. Therefore, Equation 3.21 specifies that the allocated resources should not be greater than the amount of what was requested by the user. Moreover, there might be a case where a disloyal user is assigned a very small amount of resources (or even no resources). To avoid this, a relaxation is introduced in Equation 3.22 to specify that if the request is processed, that is, if binary variable $x_j^n = 1$, then the amount of allocated resources a_j^n must be greater than or equal to at least 20% of the amount of requested resources. Based upon the equations and the constraints mentioned above, the model finds the optimal amount of resources to be allocated (a_j^n) to each user request in the batch. The point to be noted is that the calculations of the model are majorly dependent upon the user behavior as all other input parameters such as amount of requested duration and resources, and the arrival rate of the users are constant for each incoming request. Hence, to achieve the objective, an optimization model is generated and results are presented in the next chapter.

Chapter 4: Performance evaluation

This chapter presents the simulation and result analysis of the study carried in Chapter 3. More precisely, this chapter is divided into four different sections. In the first section, we present the general simulation environment and the modifications in the tools that were made to reach our objectives. Then, the next three sections present the simulation results of the RACE model, the prediction model and the optimal model respectively.

4.1 Simulation environment

The simulations of all the proposed algorithms were implemented in Java using the CloudSim toolkit [95]. The tools (simulator and computer) used during the simulations are outlined in Table 4.1. The simulations are carried with fixed resource pool of the CSP in an environment where users can relinquish. Our results are deduced based upon the pricing of utilizing an “On-Demand windows-based general purpose t2.medium vCPU” from Amazon Web Services in Canada (Central) on a usual business day [11]. Besides, the prices of electricity vary during different times of the day. Therefore, the mean price of the electricity of summers in Ontario is considered [96].

Table 4.1: Simulation Environment

Operating System	Ubuntu 16.04 LTS
Processor	Intel i7 - 3.6GHz
Memory	16 GB
Simulator	CloudSim 4.0
Implementation Language	Java

4.1.1 Using CloudSim

All the simulations, unless otherwise mentioned, are performed using CloudSim 4.0. CloudSim is a library which contains the basic entities of a cloud computing environment such as virtual machines, data centers at various geographical locations, cloudlets, users, hosts and all the other computational resources required for the simulation of cloud scenarios. In other words, CloudSim is a discrete event-based simulator which provides a model of cloud computing hardware to run simulations by simply changing the behavior of various cloud computing components at no external or setup costs.

All the components in the CloudSim library are modeled in the form of fundamental java classes whose code can be modified to run the simulations with desired input parameters. New allocation strategies and pricing methods can also be added by modifying the existing classes or even adding new classes into the package *org.cloudbus.cloudsim* which contains all the classes. These classes can be instantiated in the main class which uses the package *org.cloudbus.cloudsim.core* to initiate and maintain the simulations. Therefore, to evaluate our proposed work, we modify the existing classes for our proposed resource allocation strategy and add a new class which calculates the net profit of the cloud service provider. The details of the working of existing classes, modified classes, and newly added classes in terms of our work are as follows:

User – This class is outsourced from the work done by Hu in [14]. This class originally does not come by default into the CloudSim package but was added to model a cloud customer (typically a SaaS provider) who submits requests with the desired resources to run an application.

Cloudlet – The object of the cloudlet class instantiates the tasks/jobs submitted by the user request. Each user may require multiple cloudlets to be created. Each cloudlet contains user ID, name, and history which specifies the computational requirements of the user request. The user IDs are further used to re-route the response of the cloud service provider. To keep it simple, each user submits a single cloudlet respectively which is processed by the cloud service provider. In case of a batch request, the single cloudlet models a single batch.

Datacenter – This class models the core infrastructure services (typically IaaS providers) which are provided to the cloud customers by the cloud computing giants like Amazon, Microsoft Azure, etc. The infrastructure is provided by the cloud service provider by encapsulating the required number of hosts or servers based upon their hardware configurations. These total resources of each host are summed up to be used as the total capacity of the cloud service provider. Several data centers with distinct locations can also be created using this class. For simplicity, our simulations consider a single data center at an individual location.

Datacenter Broker – This class represents the data center broker. The data center broker is a multitasking entity which provides services from mapping the user requests with different data centers to accomplishing the negotiations between the user and the host created in the mapped data center regarding the pricing and amount of resources to be allocated. It also performs the matchmaking between the cloudlets and the virtual machines containing the amount of resources allocated to process the respective cloudlet. In our study, this class plays a vital role as it acts as an interface between the users and the provider. Hence, this class has been extended by adding the algorithms used in the state-of-art to calculate the net profit and then adding the proposed optimization model for comparison purposes. Since a single cloud data center is used to obtain the results, the broker only performs the matchmaking decisions between the users and the hosts within a single cloud.

Host – This class instantiates the physical host with specific configurations required for the simulation. Several virtual machines can be created using the host to run an application. These virtual machines are allocated with the cloud resources present in the data center such as CPU, memory and storage. This class is modified in terms of storing and updating the history of each user request submitted through the user class. The history is further fed to the data center broker to be used in all the resource allocation policies which are added.

Virtual Machine (Vm) – This class creates a virtual machine with specified characteristics which runs on the host to process the cloudlets which are mapped by the broker. Each virtual machine can use the available resources in the host. In our case, for simplicity and without loss of generality, users only request for the CPUs. The virtual machine further accesses the number of CPUs estimated by the broker using various resource allocation policies to process the cloudlet.

RACE – This class has been added for calculating the net profit made by the cloud provider using various resource allocation policies. It is coded with different algorithms to calculate the income generated, the electricity expenses and the relinquishment loss. The difference between income and the total expenses gives the net profit. The RACE algorithms are then fed with outputs from different resource allocation strategies to calculate the net profit.

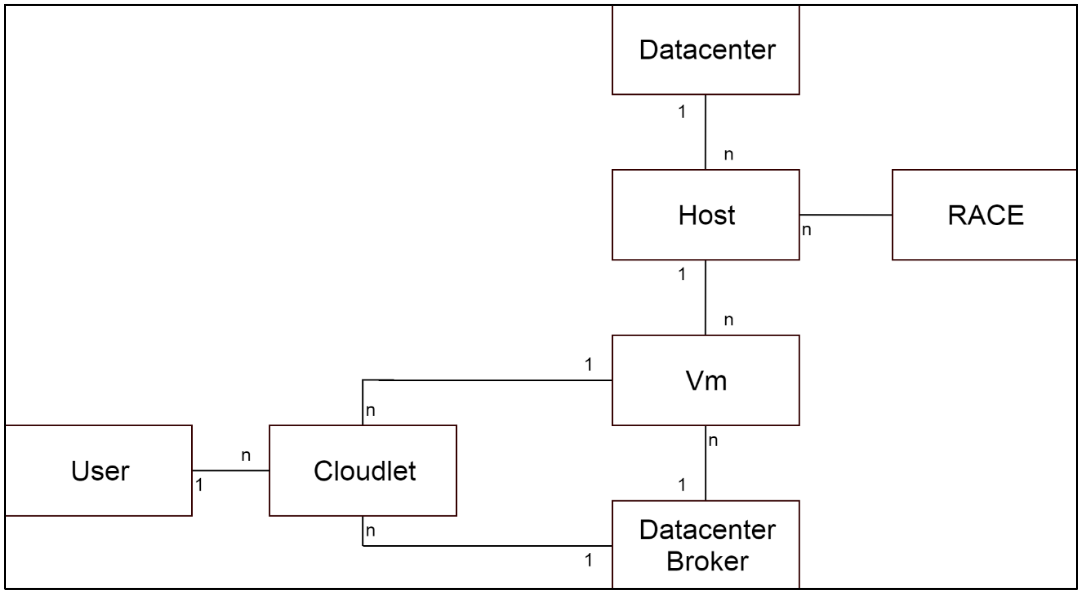


Figure 4.1: Relationship between the classes of CloudSim

Figure 4.1 shows the relationship between all the classes of CloudSim that are discussed above. As can be seen, a user request is processed using multiple cloudlets. Likely, the data center encapsulates multiple hosts which further provide the resources by creating multiple virtual machines. Broker finally maps the resources to the cloudlets.

4.1.2 Generating the workload

To investigate the effectiveness of the proposed models, we generate a synthetic workload. The major advantage of generating this workload over real workload is that it can be applied repeatedly in a controlled manner to carry out flexible simulations.

The most imporatant input parmaeter to generate the workload is user behavior. Figure 4.2 depicts the process of genrating and populating the history of the user. By definination, the relinquish probability is the ratio between the duration for which the user has not used the service and the total requested duration. For example, if a user requested 100 hours of service and relinquished after 40 hours, it means that 60% of the requested duartion was not used and therefore, the relinquish probability will be recorded as 0.6. Therefore, the value average relinquish probability is generated for each user to know the general behavior of user and the integer value of history length between $U\,(1,60)$ of each user is obtained randomly using built-in random integer generator in CloudSim.

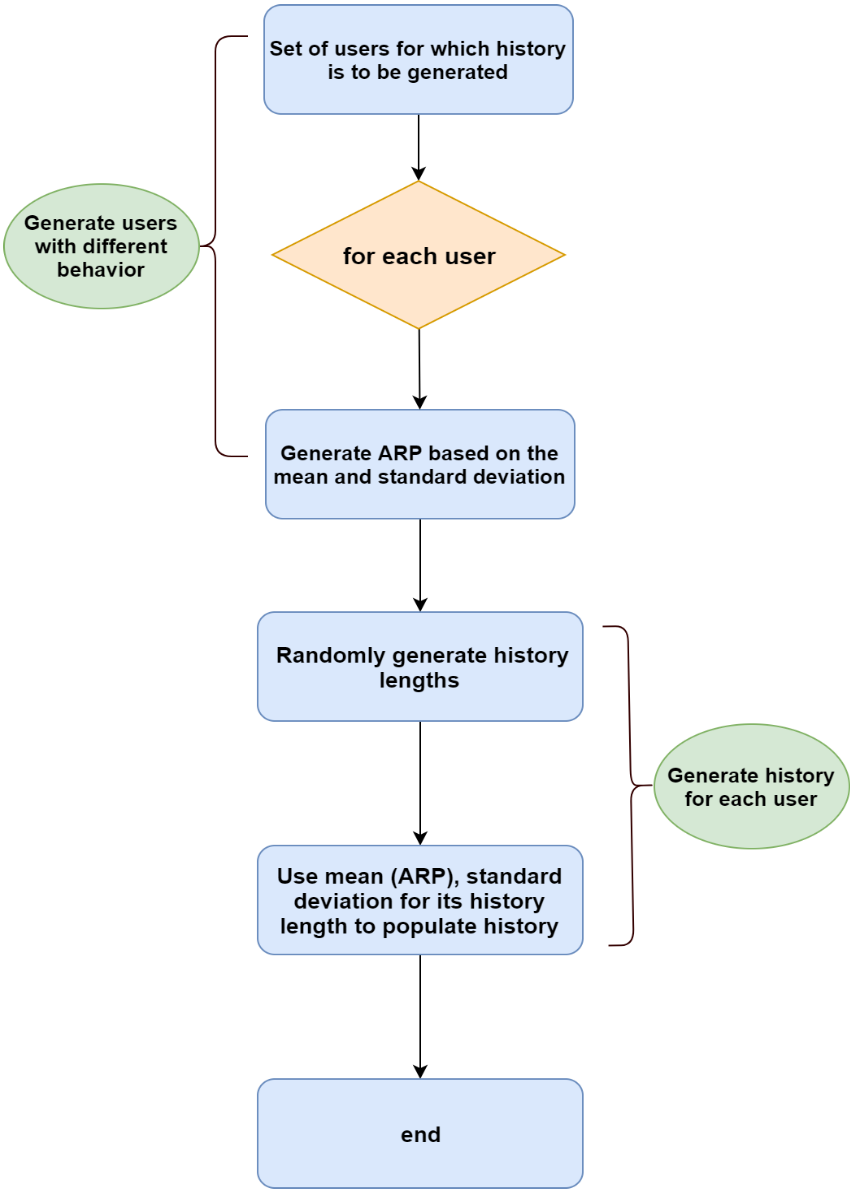

Set of users for which history is to be generated
for each user
Generate users with different behavior
Generate ARP based on the mean and standard deviation
Randomly generate history lengths
Generate history for each user
Use mean (ARP), standard deviation for its history length to populate history
end

Figure 4.2: Process to generate history of users

Generating the average relinquish probabilities (ARPs): The average relinquish probabilities are the values that are randomly generated to define the general behavior of a user. More specifically, the value of ARP gives an idea of what kind of user is requesting the resources based on how much, on average, it has used in the past. A request with an ARP ≥ 0.7 is considered as disloyal whereas a user with an ARPs ≤ 0.3 is given the loyal status. Users that fall between 0.3 and 0.7 are considered average users. Therefore, to obtain the data of realistic ARPs, the in-built gaussian distribution function in CloudSim with mean 0.5 and standard deviation (SD) 0.3 is used to generate the values. According to the definition of the gaussian distribution, 70% of the values generated will be between 0.5 ± 0.3, 95% will be between 0.5 ± 0.6 and 99% of the values will be between 0.5 ± 0.9. The value greater than one is considered as 1 and the value less than one is given the value 0.

Using ARPs to populate the history: A user can be a first-time user or could have used the service multiple times. A first-time user is treated as a loyal user. Whereas for returning users, based upon the history length of each user (the number of time the user is returning for), the corresponding value of the user's ARP is used as the mean for the gaussian distribution to populate the relinquishing probabilities for each time the user relinquished. For example, if a user is returning for 5th time and its ARP comes out to be 0.59, then using mean 0.59 and standard deviation 0.2, its usage history is populated for the last 4 times. Further, the other inputs for generating the workloads are used according to the requirement of the proposed techniques and would be discussed before presenting their respective results.

4.2 Assessment of relinquishment-aware cloud economic (RACE) model

As mentioned earlier, the goal of the RACE model is to provide an economic perspective on various resource assignment policies that can be used by the could providers. It is the tool which provides a cost-benefit analysis for the specific duration and uses the data traces as its input that is provided by the cloud provider for that duration. The novelty in the RACE model is that it can also calculate the loss incurred when users will leave the system preemptively. To that end, this section presents the results generated by running the RACE model on CloudSim for four different resource assignment scenarios.

Before discussing the scenarios, we discuss the input parameters to generate the workload (input parameters in place of real data traces) to carry out the simulations. To check the reliability, the model is tested with different arrival rates of users. In this study, we assume that users submit requests based on the M/M/1 queuing model. The set of arrival rates (λ) used to test the models is *{0.064, 0.032, 0.016, 0.008} req/sec*. The lowest arrival rate value used is λ=0.008 *req/sec*. The arrival rate values are doubled to add a new value to the set of arrival rated mentioned above. Hence, the set of interarrival times of the requests are generated randomly by the exponential distribution of mean *E[x] = {1/0.064, 1/0.032, 1/0.016, 1/0.008} secs*. We assume that the user requests a single service type, that is, vCPU with a minimum of 500 GHz and maximum of 900 GHz for a specific duration between the limit of 2-4 hours. Based upon the arrival rates, a set of total number of user enters (mentioned in Table 4.2) in the interval of 25 hours is used.

The input parameters used are user history (generated according to Section 4.1.2), interarrival time of user requests, start time of the users, end time of the users, amount of

Table 4.2: Input Parameters

Parameters	**Value**
Arrival rate of users (λ)	{0.064, 0.032, 0.016, 0.008}
Total resource pool	Fixed at 200k
Power of server	1 server of 525w
Total duration	25 hours
Requested duration by user	*U*(2, 4) hours
Requested resources by user	*U*(500, 900)
Requested service type by the user	*vCPU*
On-demand price of service	CAD 0.1/hour
Total no. of users arrived	{5500, 2700, 1450, 750}
Mean electricity price in Ontario for summers	$0.132/kwh
Schedule end time of user	Sum of start time and requested duration of respective user
Relinquish duration of user	Relinquish probability times the actual requested duration.

requested resources by the users, requested duration by users, used duration by the users. Table 4.2 summarizes all the input parameters used during the simulations. The start time of the user is the time when the user begins using the resources and start getting charged for the service. The difference between two start times is the interarrival duration generated by exponential distribution described earlier. The requested durations and the requested resources of the users are randomly generated within the limits stated in Table 4.2 using the built-in uniform random generator in CloudSim. The current relinquish probability is still generated by the gaussian distribution. The scheduled end time of a user is the sum of its start time and the requested duration. If the user relinquishes, the used duration of the user is the requested duration times the current relinquish probability. The relinquish time then becomes the sum of start time and the relinquish duration. Else, if the user does not relinquish, that is the relinquishing probability is zero, then the requested duration is considered as the duration used by the user.

4.2.1 Scenarios under which the model is tested

We used the RACE model to analyze four different scenarios as described below:

Scenario 1 - In this baseline scenario, users get assigned resources based on the service they requested. In other words, complex resource estimation is not required since users get exactly what they requested. Also, in this scenario, we assume that users do not relinquish and use the service until completion. Therefore, the income is calculated based on the allocated resources and the requested duration.

Scenario 2 – This scenario is similar to scenario 1, except that users can leave (relinquish) before the scheduled end time. Therefore, in this scenario, the profit depends upon both the income generated as well as the relinquishment loss.

Scenario 3 – In this scenario, the resources are assigned based on the BaaS method described in [13]. The authors calculate the amount of resources to be assigned using the history of the users in the form of relinquishing probability. The main idea is to assign lesser resources to users who have a disloyal behavior (i.e. users who tend to leave before the scheduled end time).

Scenario 4 – In this scenario, resources are assigned according model proposed in [14]. In this model, resources are allocated based on predicting user behavior using RP method and the current server utilization. The model uses different assignment schemes depending if the server is underutilized, moderately utilized, or over utilized.

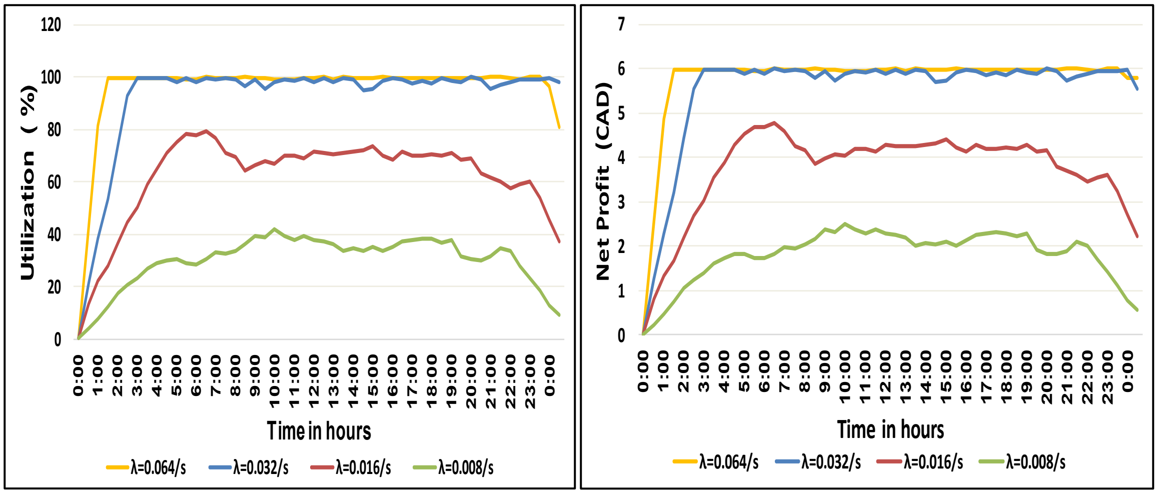
By simulating the four scenarios described above and using the proposed RACE model, a cost-benefit analysis is carried out to see which models are the most profitable and whether improving the overall server utilization would also increase profit or not. Further, scenarios 3 and 4 are used as the state-of-art with which we carry out the comparisons to check the effectiveness of the relinquishment aware resource optimization model proposed in Section 3.3.

4.2.2 Simulation results of the RACE model

The net profit and overall utilization of all scenarios described above are plotted for the different arrival rates mentioned in Table 4.2. All the simulations were performed for a duration of 25 hours and run 20 times to obtain reliable results. The average values of the results are presented. The confidence intervals were negligible and hence not plotted. Figure 4.3 depicts the overall utilization and the net profit at each time for all arrival rates for scenario 1.

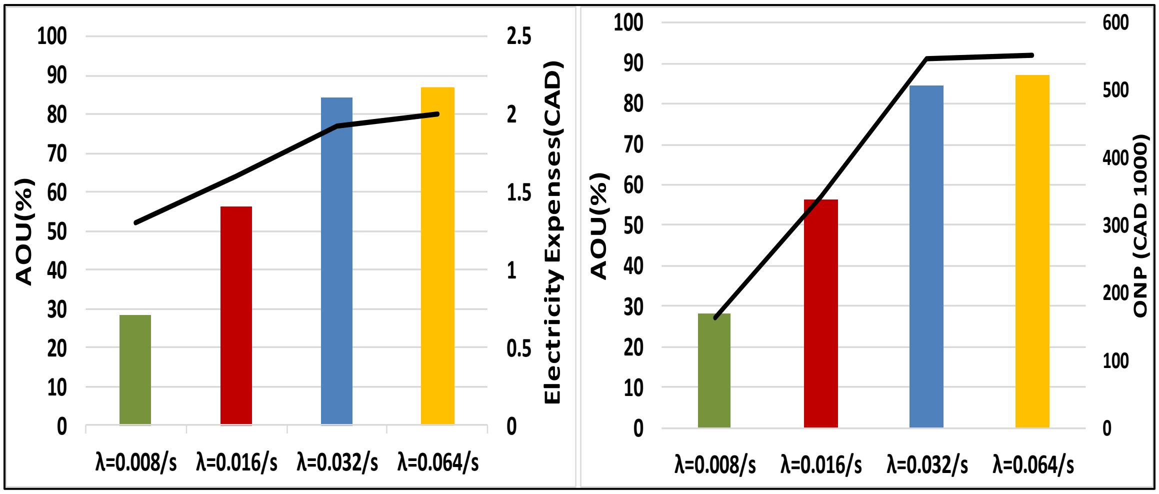
Figure 4.4 (a) shows more details on the relationship between the average overall utilization (AOU) of scenario 1 and the electricity expenses for all arrival rates whereas Figure 4.4 (b) shows the relationship between the average overall utilization and the overall net profit (ONP**).** As we can see, at *λ=0.064 req/sec,* scenario 1 reaches the maximum level of its utilization and net profit. Although not realistic as users never relinquish, this scenario provides an upper bound. Since users get assigned what they ask, the utilization goes up quickly and after some time, the provider is fully utilized. Scenario 1 has the highest average overall utilization (around 86%) and generates the maximum profit ($551k) at this arrival rate.

Further, at *λ=0.032 req/sec* with 2700 requests arrived in 25 hours, the system almost reaches its saturation point. So, if we compare the utilization and the net profits for *λ=0.032 req/sec* and *λ=0.064,* we see that the net profit increases slightly even if the arrival rate of users gets doubled. The reason being this scenario already runs at almost full capacity at *λ=0.032 req/sec*. This contrasts with the situation when the arrival rate of users is increased from *λ=0.016 req/sec to λ=0.032 req/sec* where the utilization and net profits always got doubled. This is because, with the given limit of requested resources, the available resources in the system are gradually utilized as comparatively less user requests are processed at low arrival rate of users. Noticeably, as no user relinquishes, the net profit



(a) (b)

Figure 4.3: (a) Utilization vs (b) Net profit for scenario 1 at different arrival rates



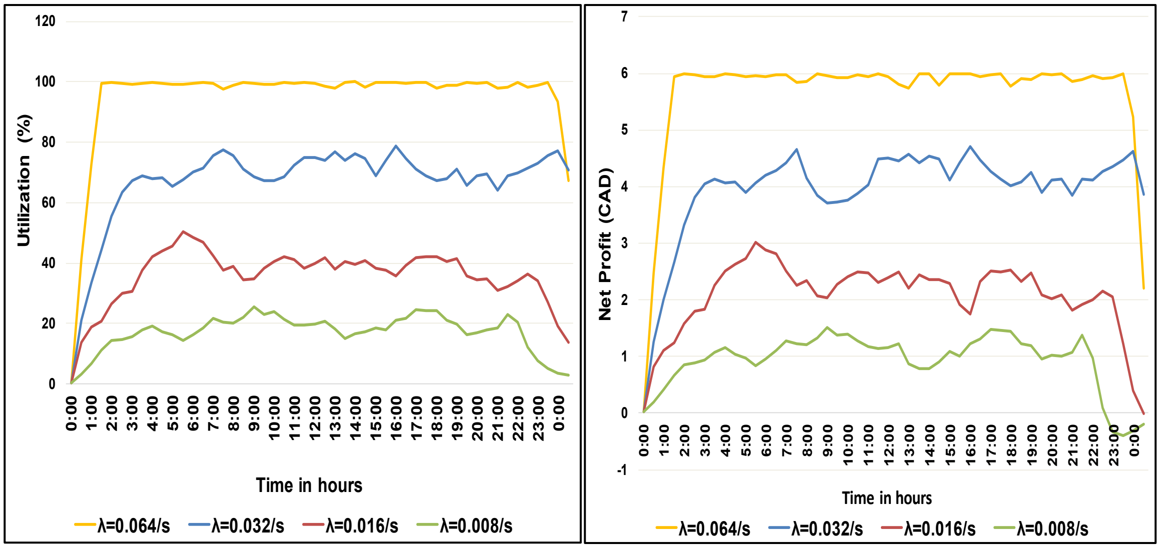
(a) (b)

Figure 4.4: Relationship between (a) Average overall utilization and electricity expenses (b) Average overall utilization and overall net profit for scenario 1

at each arrival rate is observed to be affected by the system utilization as the profit almost entirely depends upon the income generated by the resources allocated to the users.

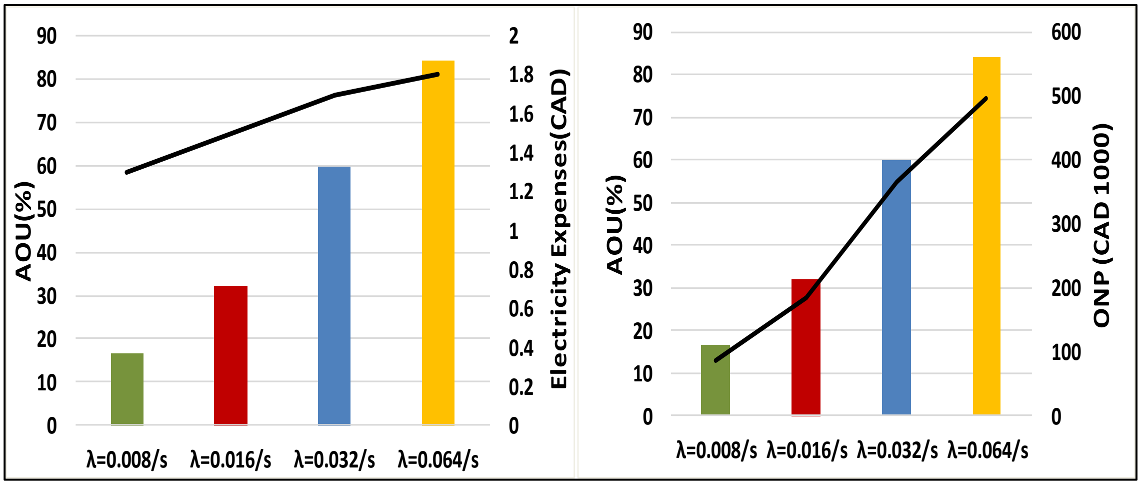
Although there are electricity expenses paid by the provider, those are very minimal as only a single server is used to provide the finite resource pool. Consequently, the electricity expenses do not make a significant difference in the net profit. It can also be seen that the electricity expenses in Figure 4.4 (a) are not directly dependent upon the utilization. The reason being that the electricity expenses (according to Equation 3.4) are not only related to the utilization, but also depend on a fixed part when the system is un-utilized/idle (due to less users in the systems or users relinquishing). For example, when the average utilization is 16.6% (at *λ=0.008 req/sec)*, the electricity expense for the period of 25 hours is $1.3. On the other side, when the system shows an average utilization of 84.3% (at *λ=0.064 req/sec*), the electricity expenses only increases to $1.8.

Apart from that, from Figure 4.5 (a) and (b), we can see that when users can relinquish their services (i.e. scenario 2), a small decrease in the utilization significantly



(a) (b)

Figure 4.5: (a) Utilization vs (b) Net profit for scenario 2 at different arrival rates

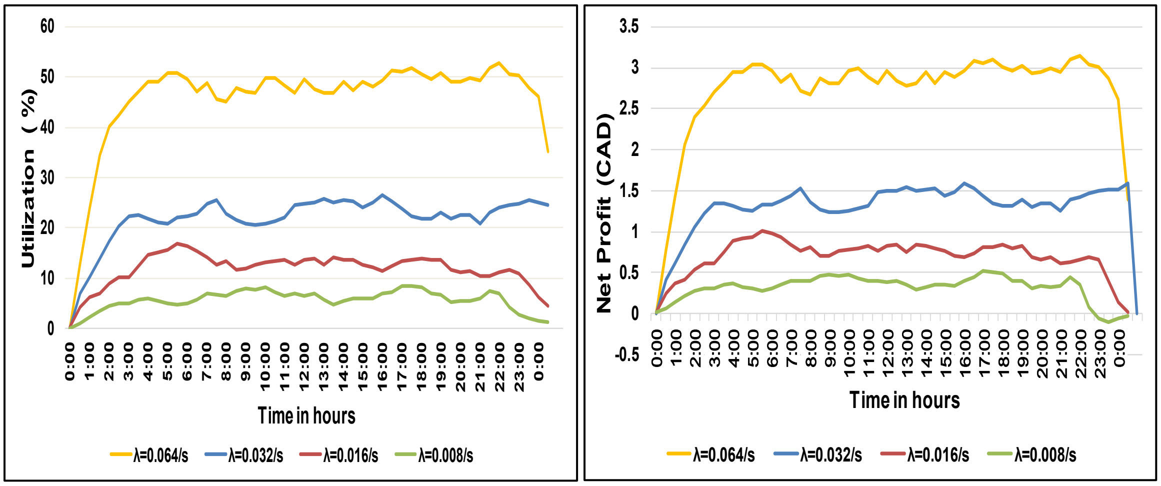


(a) (b)

Figure 4.6: Relationship between (a) Average overall utilization and electricity expenses (b) Average overall utilization and overall net profit for scenario 2

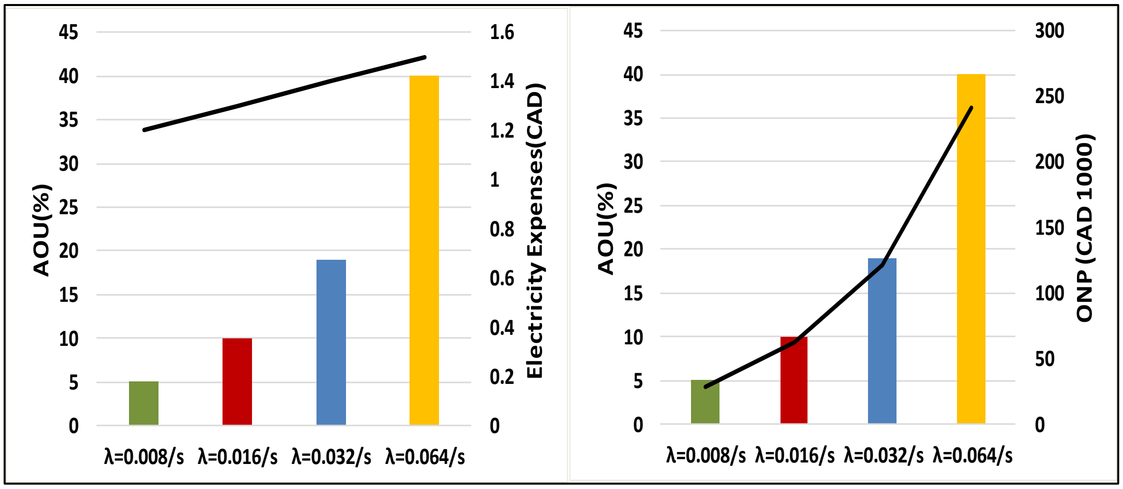
affect the profit. This is because users may leave the system before the scheduled end time. A similar trend is seen for all arrival rates in scenario 2. For instance, at time 10:00, net profit dips more as compared to the dip in utilization at the same time when the users are arriving at *λ=0.032 req/sec.* Similarly, at time 16:00, net profit line drops more than the corresponding utilization line. This reduction can be explained by the loss of income and the loss associated to the relinquishment (see Equation 3.5). Whereas, at *λ=0.064 req/sec,* the net profit goes one on one with the utilization. This is because the relinquishment loss is calculated till the next request arrives. So, at such a high arrival rate, the time between arrivals is very small and hence the calculated relinquishment loss is also reduced. Moreover, the impact of relinquishment on ONP of scenario 2 can be clearly seen in Figure 4.6 (b) where the ONP line does not grow linearly as compared to the ONP line of scenario 1 (Figure 4.5 (b)) where the ONP is majorly dependent upon only the income generated.

Moving further, in scenarios 3 and 4, two different resource assignment models are evaluated. From Figure 4.7 and Figure 4.9, it is observed that scenario 3 is assigning the least amount of resources as shown by the lower utilization percentage for each arrival rate. The net profit vs utilization trend of these two scenarios is similar to scenario 2 with a different set of values of utilizations and net profits for each arrival rate respectively. As depicted in Figure 4.8 (b), the average overall utilization of scenario 3 reaches a maximum



(a) (b)

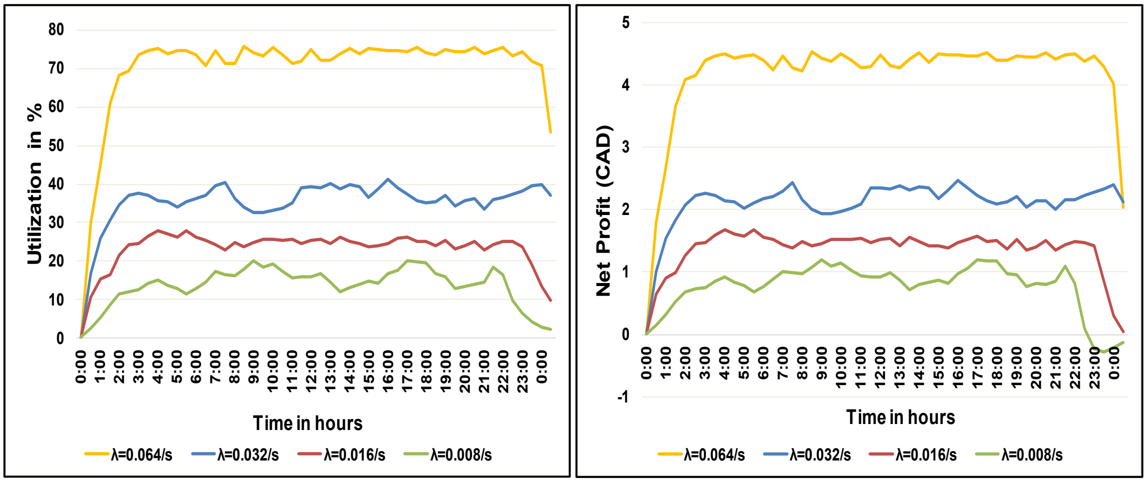
Figure 4.7: (a) Utilization vs (b) Net profit for scenario 3 at different arrival rates



(a) (b)

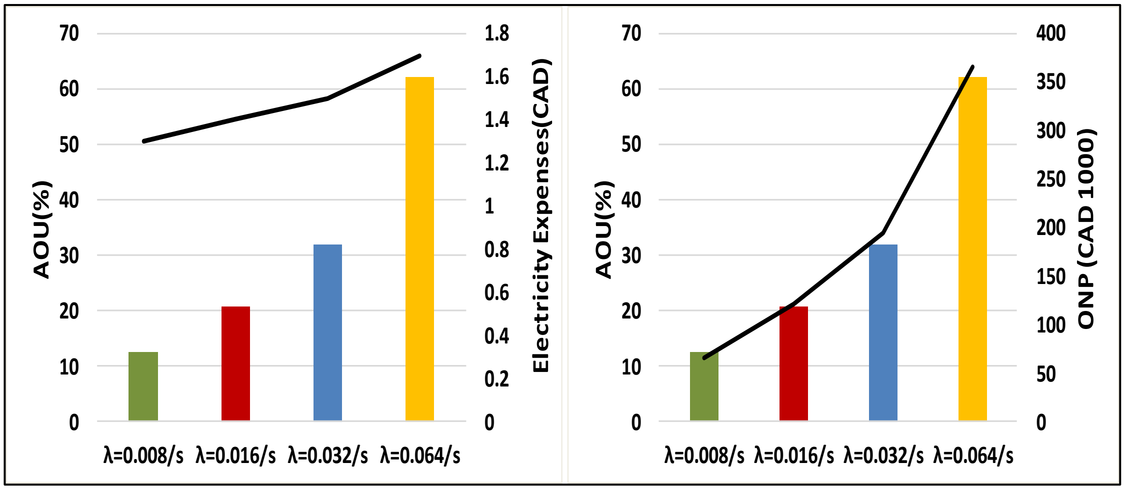
Figure 4.8: Relationship between (a) Average overall utilization and electricity expenses (b) Average overall utilization and overall net profit for scenario 3

of 40% when *λ=0.064 req/sec* whereas it is only 5% when the arrival rate of users is *λ=0.008 req/sec*. This shows that the assignment model in scenario 3 allocates the least amount of resources to prevent the relinquishment loss but made the least net profit. Scenario 4 on the other end earns more as compared to scenario 3 (Figure 4.8 (b)) as RP model improves its AOU at each arrival rate respectively.



(a) (b)

Figure 4.9: Utilization vs (b) Net profit for scenario 4 at different arrival rates



(a) (b)

Figure 4.10 : Relationship between (a) Average overall utilization and electricity expenses (b) Average overall utilization and overall net profit for scenario 4

Overall, the results of the cost-benefit analysis using the proposed RACE model show that assigning whatever the user demands do increase the profit due to increasing income. However, with finite capacity and increasing arrival rate (*λ ≥ 0.064 req/sec)*, the system will potentially remain highly utilized for scenario 2 but the net profit will be primarily affected by users who relinquish their services. Moreover, the latter two scenarios allocated resources based upon the prediction of the behaviour of the user in their own respective approaches but were not able to generate the income for the provider. Hence, they ended up having lower net profits for all the arrival rates. Therefore, to increase the profit in an environment where user can relinquish, better resource allocation schemes and better prediction techniques are required to optimally allocate the resources. The next section of this chapter puts light on which prediction model is better amongst the ones described in Chapter 3.

4.3 Comparison results of various methods to predict relinquish probability

This section presents the comparison results of the various prediction techniques used to predict the user behavior given his history. The outcomes of this section will notify us which method can make better predictions about the user behavior. The prediction will let the provider know how long the user is going to use the allocated resources based upon his history. The list of future work suggested by Hu in [14] highlights that instead of using average approximation algorithms, the quality of prediction could be improved by the means of machine learning algorithms. Therefore, in this thesis, we decided to look at the concept of linear regression to make predictions.

To evaluate the performance of the different methods, we used three user categories: loyal, disloyal and average. The users' history was generated the same way as explained Section 4.1.2. To perform the evaluation, Java and Octave platforms were used to generate the scripts, data sets and carry out the calculations respectively.

Table 4.3: An example of a user's data whose relinquishing probability is predicted using linear regression

History Length (*m=5*)	Relinquish Probability
1	0.35
2	0.56
3	0.28
4	0.67
5	0.45
6 (*m+1)th*	To be predicted by LR

Equations 3.8 and 3.9 state the different methods to carry out the predictions as proposed in [13] and [14] respectively. In this section, we use linear regression and compare the results with the two methods mentioned above. Table 4.3 provides an example of a single user whose history length is five. The data in Table 4.3 also shows the usage history of the user for the last five times. Similar kind of data was used to carry out a separate univariate regression analysis for a single user returning with various history lengths.

To simulate loyal, disloyal, and an average user, different ARPs (0.3, 0.7 and 0.5 respectively) were used and the history was populated randomly using the gaussian distribution using these ARPs as mean with standard deviation 0.2 (as mentioned in Section 4.1.2). To get reliable results and avoid randomness, the process of generating history and making prediction for the single user was repeated 50 times. Then, the mean of the predicted values is taken and plotted along with the 95% confidence interval in the figures below. The history length is denoted as *m* and the $(m+1^{th})$ value is plotted for each

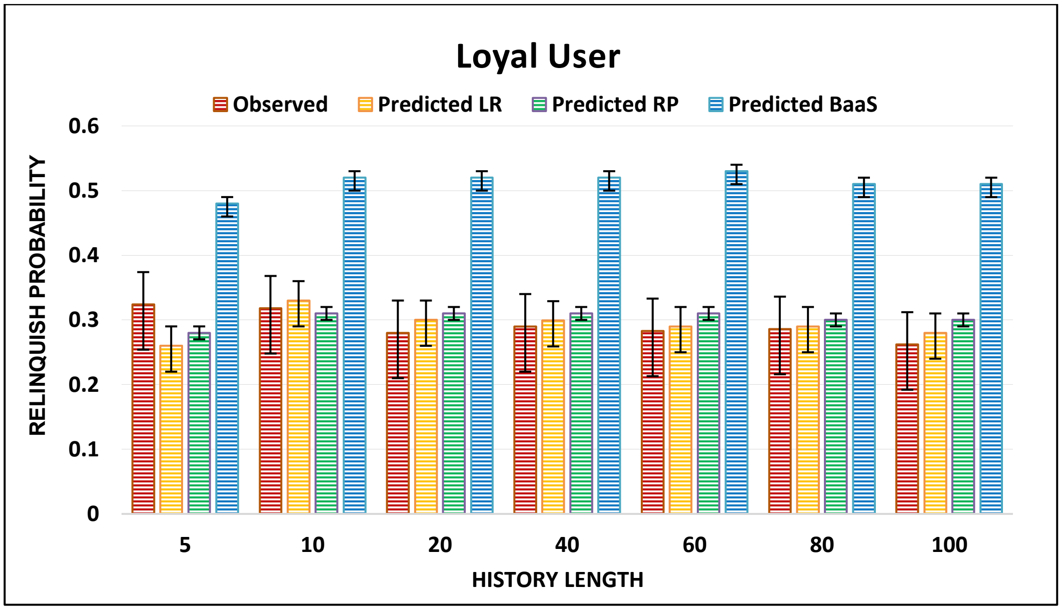


Figure 4.11: Prediction results for a loyal user

prediction approach. For instance, if history length is *h=5*, it states that the user is returning for the *m+1*th *(i.e. 6*th*)* time. The relinquish probability for the (*m+1*th*)* time is plotted. This notation is used throughout this section to explain the results.

Figures 4.11 to 4.13 depict the prediction results of a loyal, average and disloyal users respectively with variable history lengths. As seen from the graphs, predicted values from linear regression (LR), BaaS [13] and RP [14] are compared with the observed relinquish probability for (*m+1)th* time. For each status, seven different users were simulated with different history lengths respectively. As illustrated in Figure 4.11, seven different loyal users request a given service. Each user is having ARP = 0.3 with different history lengths. The users are returning for the 6th, 11th, 21st, 41st, 61st, 81st and 101st time respectively. A similar trend is followed for the other two types of users (average and disloyal). To evaluate the prediction accuracy of the three methods, we compare the predicted value with the observed value from the user.

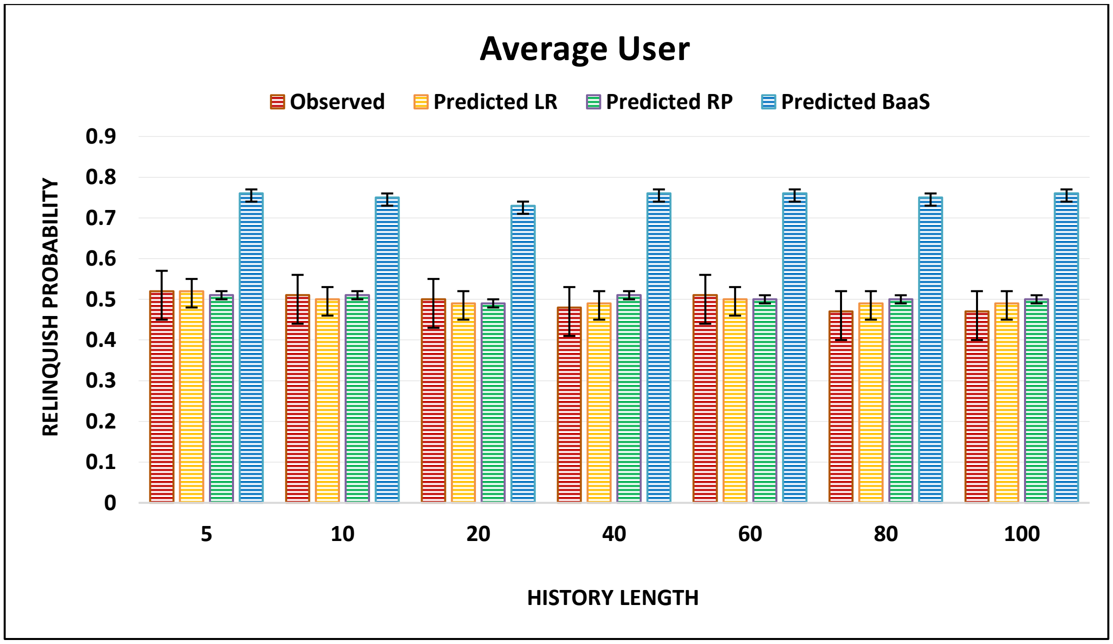


Figure 4.12: Prediction results for an average user

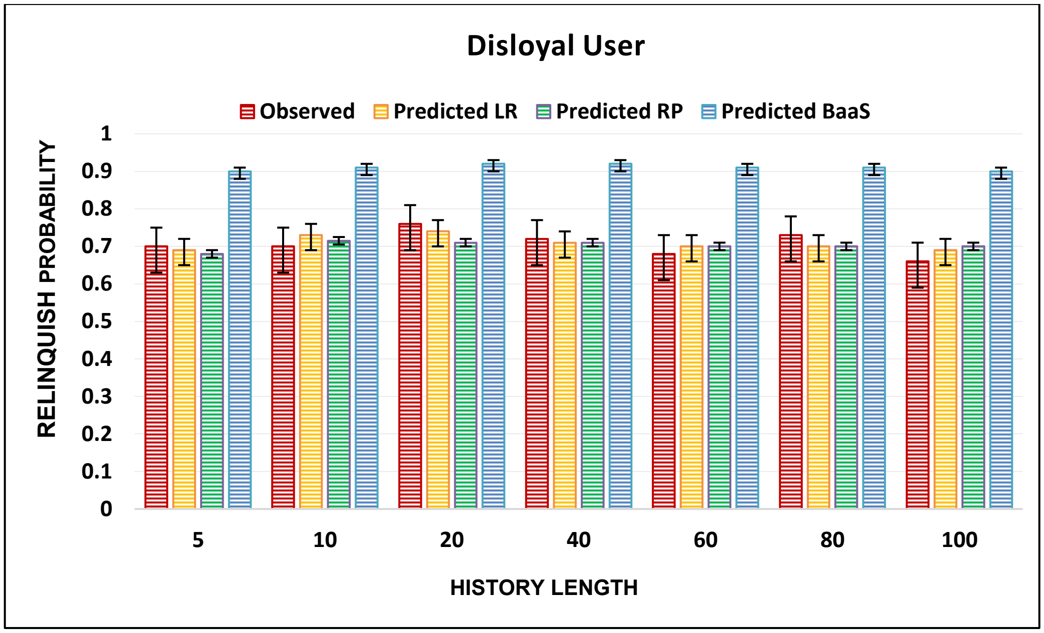


Figure 4.13: Prediction results for a disloyal user

It can be seen that BaaS shows the biggest difference (around 20%, 25% and 20%) between the observed value and the predicted value for loyal, average and disloyal users respectively. This is because the authors use SOP and AOP for predicting the user behavior. Further, to cope up with changes in the behavior of users each time they return, the authors use the variance as an additional parameter but this does not help in making the results more accurate. On the other hand, RP (which is the improved version of BaaS) is able to make better prediction with a difference of approximately 5%, 4% and 4% between the observed value and the predicted value for loyal, average and disloyal users respectively. Moreover, for small training sets (i.e. history lengths ≤ 20), the prediction method used in [14] provides better results (approximately 5% better) compared to linear regression. However, as the history length grows, the accuracy of linear regression outperforms the predictions made by the RP method by approximately 2%, 1.7%, 1% for loyal, average and disloyal users respectively. The major reason for the improved accuracy of the linear regression algorithm can be regarded as the minimized squared error function which provides the minimum value of the squared difference between the errors (refer to Section 3.2.3). In general, even with a single parameter (history length), linear regression proves to be the most effective tool amongst the three to make predictions about the user behavior. As a result, linear regression is used to make prediction as improving predictions can also improve the profit generated by the provider.

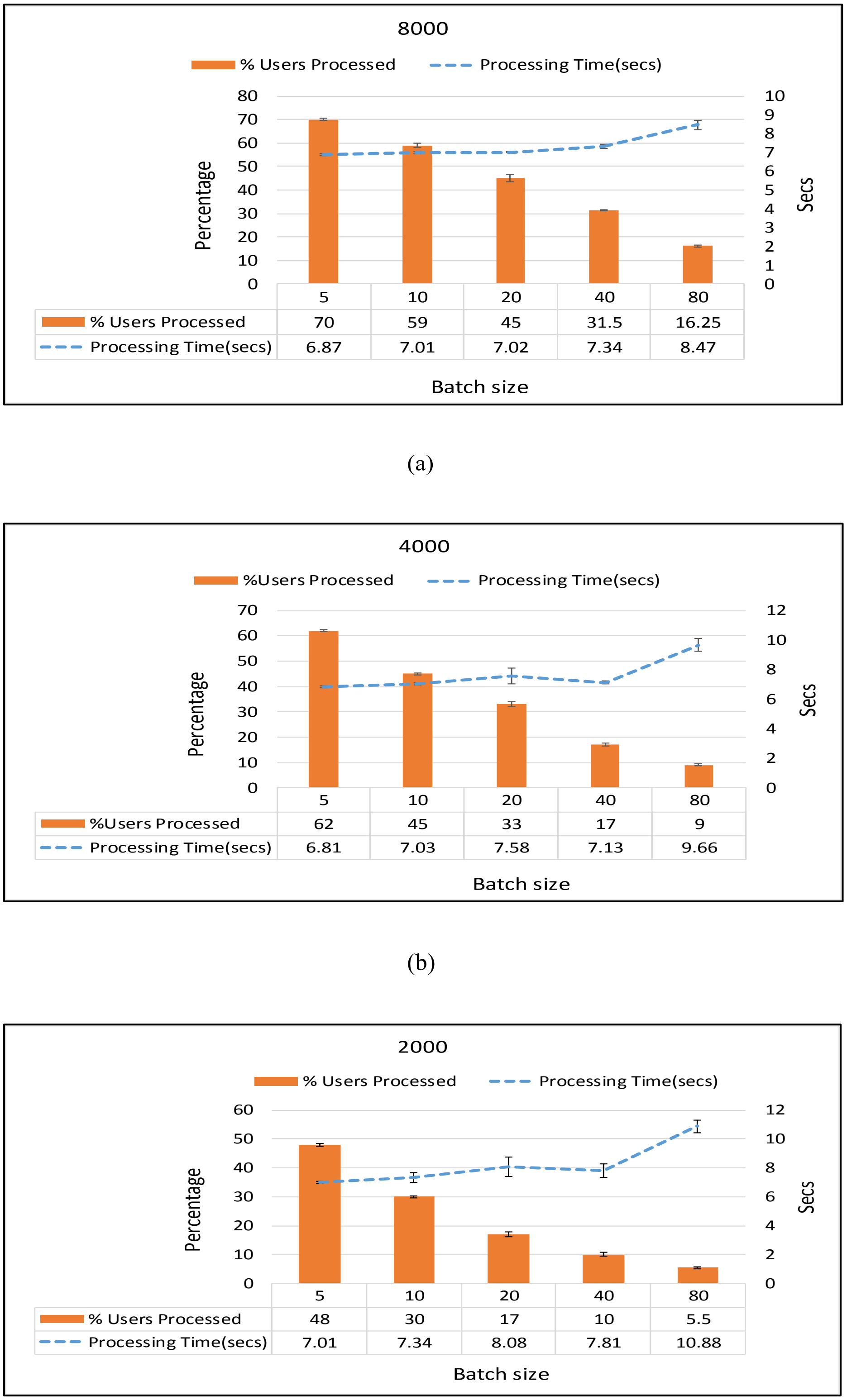
4.4 Performance evaluation of relinquishment aware resource optimization model

This section presents and discusses the experimental results of the relinquishment aware resource optimization model presented in Section 3.3. This model is solved with a commercial solver (IBM CPLEX [97]) on the same PC environment as described in Table 4.1**.** For a given batch size and resource capacity, the performance of the model is tested in terms of two parameters. The resultant values of the parameters are used to decide which batch size gives the best results in terms processing maximum user requests under the finite capacity. The parameters are:

- *Processing time of a workload batch* – The time required by the solver (CPLEX) to read the input data, generate the model, solve the model and produce the output in the form of decision variables.
- *Number of users processed in a batch* – This parameter gives us the percentage of users processed for the batch size under evaluation with the given capacity of resources

The batch sizes used in this study are based upon the workload batch sizes used by the other researchers. For example, based on the work in [98], a batch size of more than 10 user requests (or jobs) is regarded as a large workload. Similarly, the study that was done in [99] considers a batch of 5 requests to be of small size whereas a batch with 10 jobs is considered as medium. Therefore, to have a complete analysis, the set of batch sizes of {5, 10, 20, 40, 80, 100} requests are tested under each of the remaining capacities of 2000, 4000 and 8000 CPUs respectively. The input parameters of the model are predicted relinquish probability, requested duration, requested amount of resources, price of resource and electricity. These values are generated as described in Section 4.1.2. The experiment for each set of batch size with individual capacity value was run 20 times. The mean values of the percentage of users processed and the CPU processing times taken by solver are plotted along with the 95% confidence intervals. The built-in timer provided by CPLEX is used to measure the processing time.

Figure 4.14 (a) depicts the results when the model was tested for the remaining capacity of 8000. Based upon the predicted relinquish probability and the input parameters, the solver only allocates resources to users who would potentially generate more income for the provider and minimize the relinquish loss at the same time. Simultaneously, the solver also tends to maximize the number of users to be processed within that batch. As expected, the batch with the large workload of 80 user requests takes the longest time to process all the inputs and provides results in around 8 secs. The small and the medium sized batches are processed in almost the same time. Moreover, for small and medium workloads, around 70% of the users were processed with the least processing times. In Figure 4.14 (b), the remaining capacity was reduced by half to 4000 resources. Small and



(a)

(b)

(c)

Figure 4.14: Effect of remaining capacity (a) 8000 (b) 4000 (c) 2000 on the performance of various batch sizes

medium workloads performed the best in terms of percentage of users processed and time taken to process the batch. Whereas, the least percentage of users was processed in the large workload with a batch size of 80. This is because the system gets out of capacity for larger batch sizes as the sum of the requested resources by the entire batch greatly exceeds the available capacity. Reason being the set limits of the requested amount of resources for the user in Table 4.2. Hence, the percentage of users processed keeps decreasing with increase in batch size for a given finite capacity. To understand this phenomenon, the batch was finally tested with a remaining capacity of 2000. As shown in Figure 4.14 (c), when the batch size is increased, the percentage of users processed keeps decreasing while the processing time keeps increasing accordingly. As usual, the small batch size shows the best performance in terms of processing time and percentage of users processed.

To summarize, the experimental results demonstrate that a batch size with five user requests performs better than all the other batch sizes under various remaining capacities. The objective of using different batch sizes under varying capacity was to figure out the optimal batch size which could be used for the simulations where users enter the system in an open stream and the remaining capacity is constantly updated. Therefore, a batch size of five will be used in the next section to assess the performance of the proposed optimization model and carry out the comparison between state-of-art algorithms.

4.5 Comparison results of the proposed optimization model versus the state-of-art algorithms

In this section, we present the outcomes of the proposed relinquishment-aware resource optimization model discussed in Chapter 3. To perform the simulation, the proposed optimization model was integrated into DatacenterBroker class of along with all the scenarios mentioned in Section 4.1.1. by importing the Java libraries provided with the CPLEX package. More specifically, the Java libraries include the following two APIs: ILOG Concert Technology, ILOG OPL. These APIs are used to generate and solve the OPL model (discussed in Section 3.3) and invoke the CPLEX's optimizer to solve the linear programming model. In addition to that, the predictions were made by writing the linear regression algorithm in Java and integrating it into the DatacenterBroker class of CloudSim. Following that, the model collectively decides the amount of resources to be allocated to all the requests in the batch and simultaneously update the current capacity available for the next batch of requests. The outcomes are then compared with the related work.

For making it easy to explain the results, we term our proposed optimization model as Scenario 5. To compare scenario 5 with the former four scenarios, all models were run for a specified duration of 25 hours. It can be noted that 25 hours is not the actual clock time but the preconfigured simulation time on CloudSim for which the models are compared at different arrival rates. The workload was generated using the similar input parameters mentioned in Section 4.1.2 and each simulation was repeated 20 times to get 20 different sets of data for all the scenarios. The average values were then used to present the results. The confidence intervals were also calculated but since they were very small, they are not

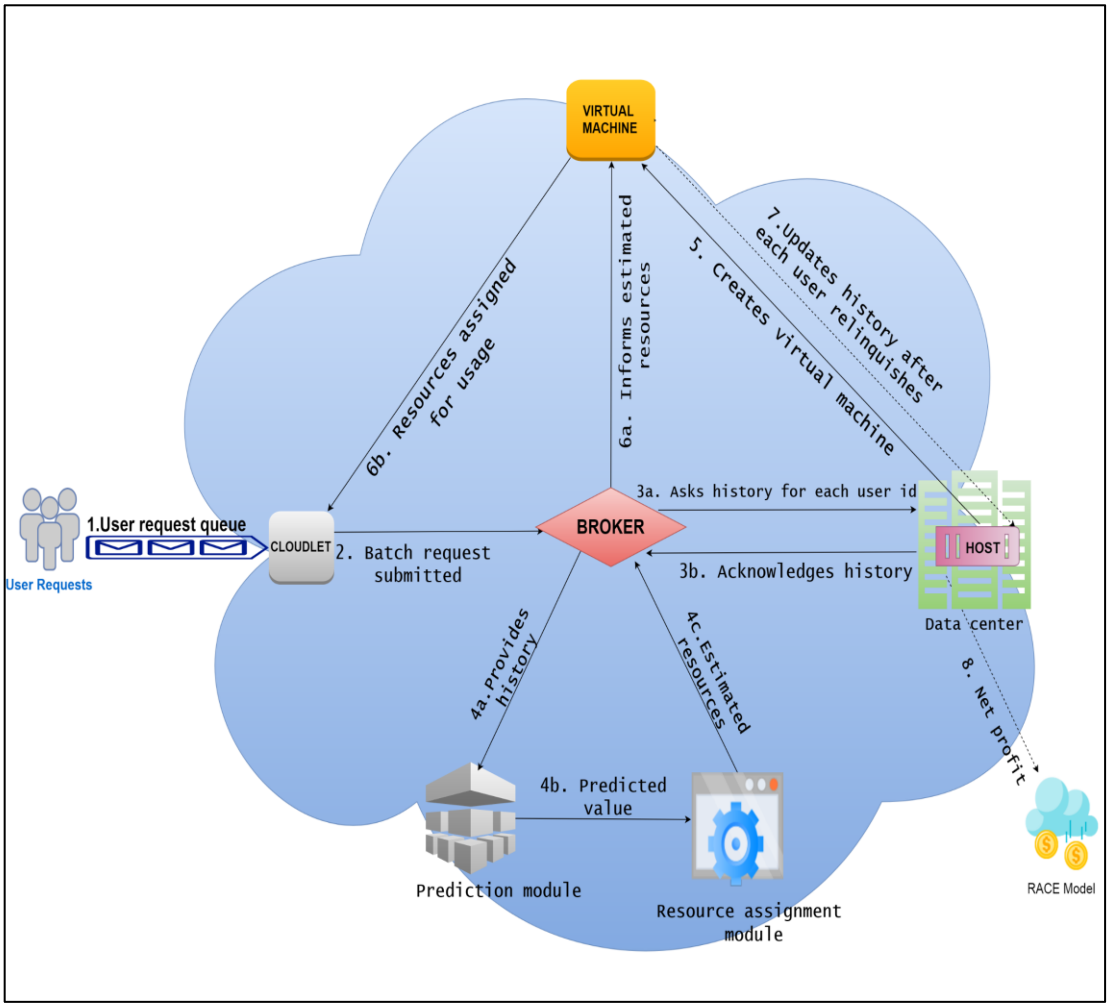

VIRTUAL MACHINE
6b. Resources assigned for usage
6a. Informs estimated resources
5. Creates virtual machine
7.Updates history after each user relinquishes
1.User request queue
User Requests
CLOUDLET
2. Batch request submitted
BROKER
3a. Asks history for each user id
3b. Acknowledges history
HOST
Data center
4a.Provides history
4c.Estimated resources
8. Net profit
4b. Predicted value
Prediction module
Resource assignment module
RACE Model

Figure 4.15: Overview of the simulation performed in CloudSim

shown in the graphs. The overview of the simulation for a batch is shown in Figure 4.15**.** When the requests start arriving (unlike scenarios 3 and 4 where the users were processed one by one), we assume that: 1) All the users in the incoming batch do not have a job completion deadline. This assumption is necessary since when user requests arrive in the system, they are queued in the network until the required batch size is formed. 2) The cloudlet submits the request for a single service type (vCPU) to the broker. 3a) The broker asks for the history of each user and 3b) gets the history of each user from the host. 4a) The history is fed to the prediction module where the three different prediction algorithms (depicted in Figure 4.16) are executed. Then, 4b) the prediction values are passed to the resource assignment module. The resource assignment schemes in this module (depicted in Figure 4.16) use the predicted value to estimate the amount of resources. 4c) This information is passed to the broker. 5) The host creates a virtual machine which further 6 a) processes the request from broker to 6 b) assign the resources to the users. It is important to note that before every batch is processed, the value of the current remaining capacity of the system is passed into the resource assignment module. It is also assumed that there is no processing delay experienced once the batch enters the system. The remaining capacity is then updated and users are blocked if the system is running at maximum capacity. Finally, 7) the history of each user is updated with a new entry once the user leaves the system. This new entry will be used by the prediction algorithm when the same user submits a new request. Similar to the approach followed in Section 4.3, the inputs for all the scenarios are kept the same for each run. The simulations were run for the same set of arrival rates outlined in Table 4.2**.** The graphs for the utilization and the corresponding net

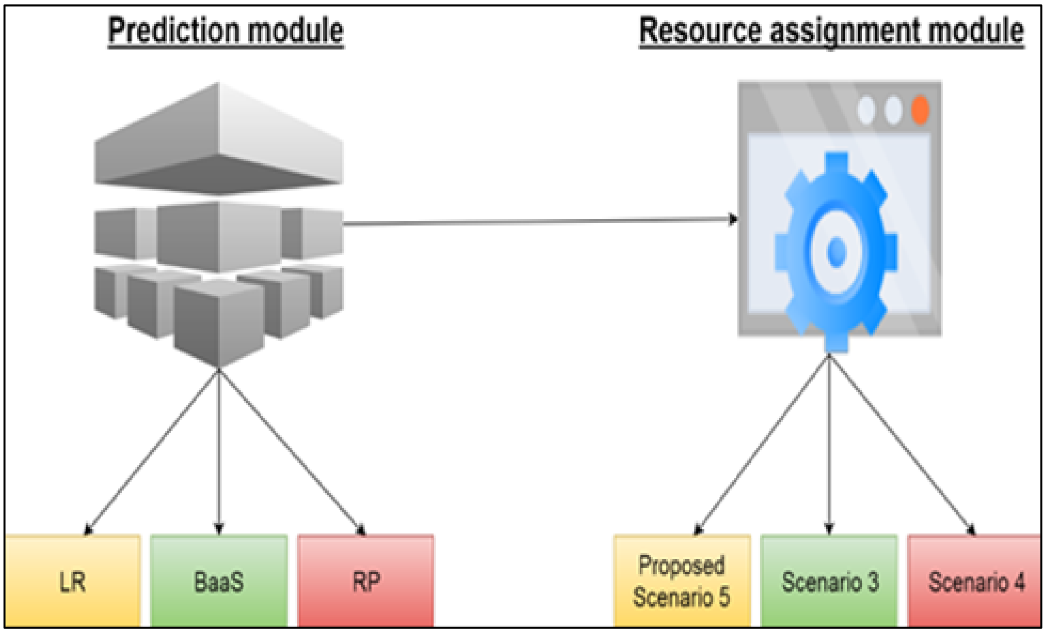
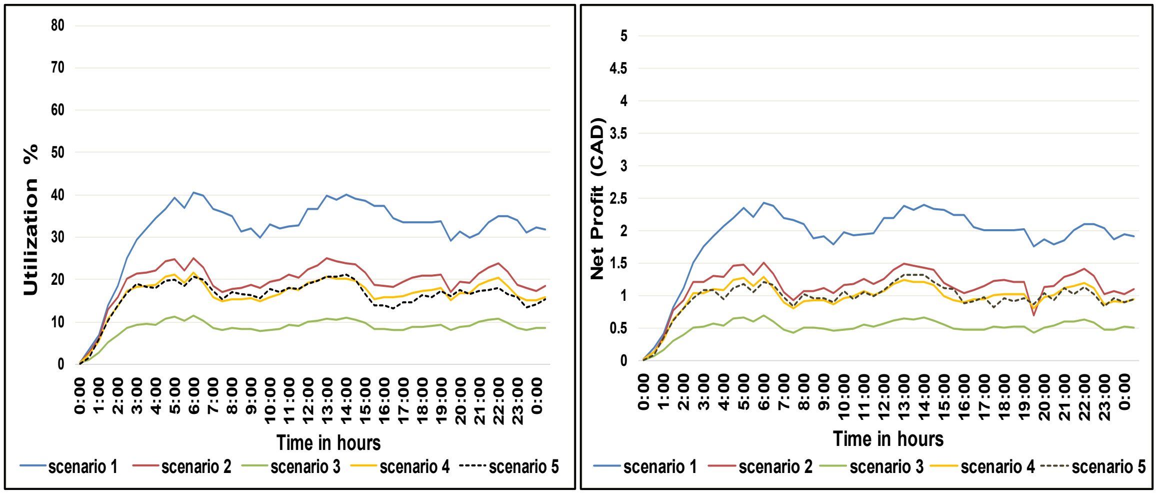


Figure 4.16: Components of different predictions modules used in the comparison

profit for all the scenarios are presented for each arrival rate. The major outcomes for all the comparisons are outlined in Table 4.4.

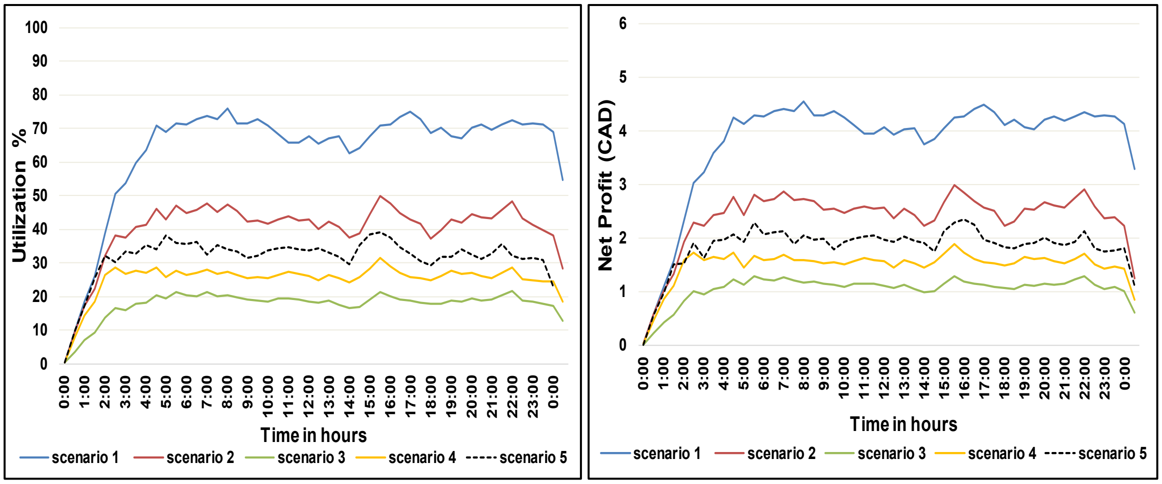
As demonstrated in Figure 4.17 (a and b), with a total of only around 750 users entering at λ=0.008 *req/sec*, the system is underutilized for all the scenarios. It can be seen that, from time 14:00 to 16:00 when utilization is dropping due to users relinquishing, the impact of relinquishment of users on net profit of scenario 5 is less as compared to other scenarios. This is because in scenario 5, if within a batch, users are predicted to relinquish, loyal users are given the greater share of the resources whereas disloyal users are given either 20% share or are declined depending upon how they are estimated to make profit. Therefore, at time 16:00, with the least utilization of scenario 5, the corresponding net profit is greater than scenario 4 and almost comparable to scenario 2.

It can also be noticed that for each scenario, the net profit at each time is generally reflected by the utilization but at some points, say from time 20:00 to 23:00, the net profit for



(a) (b)

Figure 4.17: (a) Utilization vs (b) Net Profit for all the scenarios at λ=0.008 *req/sec*

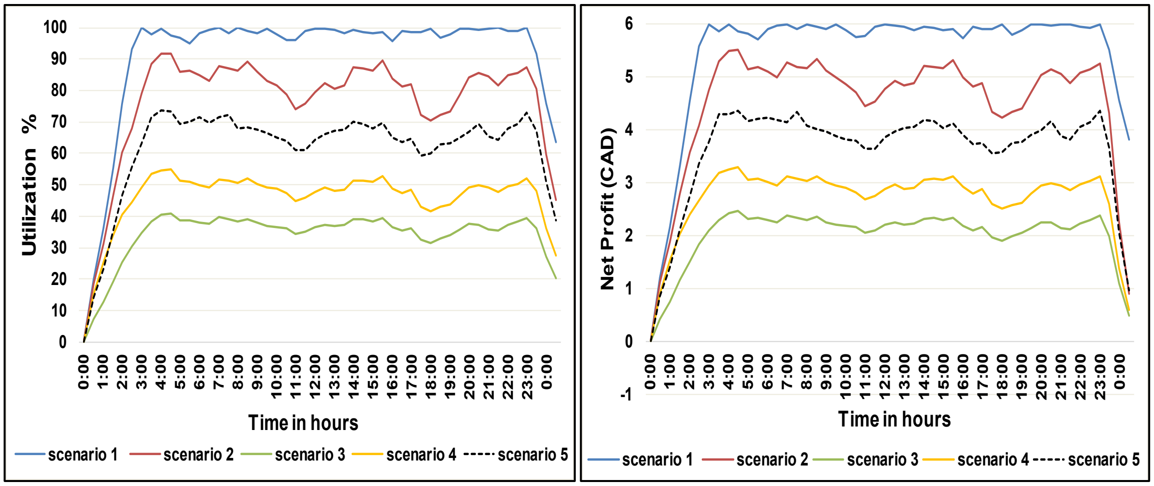


(a) (b)

Figure 4.18: (a) Utilization vs (b) Net Profit for all the scenarios at λ=0.016 *req/sec*

scenario 5 is not identical to the line of the corresponding utilization. This may likely be due to some instances where the linear regression estimates greater resources for some loyal users but they relinquish before their predicted duration of using the resources. This anomaly in prediction could be due to the fact that the history length of these users was too small for the linear regression algorithm to show its effectiveness. However, the improved performance of the prediction technique and scenario 5 was seen with the increase in arrival rates of the users.

In addition to the case discussed above, the arrival rate was doubled to λ=0.016 *req/sec* and 1450 users entered the system in this case. From Figure 4.18 (a and b), it is observed that, the net profit and the utilization curves for scenario 5 are higher than scenario 4. This is unlike the previous case when the arrival rate of users was smaller. As discussed earlier, this improvement can be attributed to the approach of scenario 5 to assign more resources to loyal users and tightening up for predicted disloyal users. Therefore, according to Equation 3.15, the difference between the income and the overall expenses is minimized and the overall net profit for scenario 5 (outlined in Table 4.4) is increased significantly as compared to scenario 4. Further, as seen in Figure 4.19 (a and b) when the arrival rate was

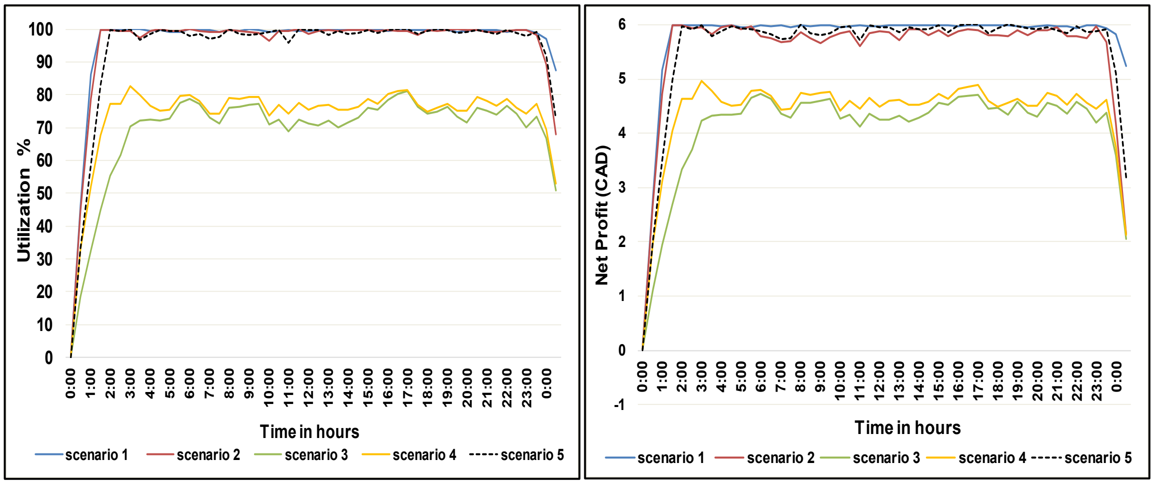


(a) (b)

Figure 4.19: (a) Utilization vs (b) Net Profit for all the scenarios at λ=0.032 *req/sec*

set to λ=0.032 *req/sec*, the performance of the proposed optimization model increases as the difference between the net profit of scenario 5 and scenario 2 decreases. The AOUs and ONPs in Table 4.4 of scenarios 2 and 5 show that with comparative profits for both scenarios, a provider using scenario 5 still can acquire more number of requests. Therefore, with scenario 1 running almost at full capacity and scenario 2 operating nearly full, it is most likely that scenario 5 can yield a substantial increase in profit

To see that, the simulations were finally run for λ=0.064 *req/sec*. As it is clearly seen in Figure 4.20 (a and b), for most of the times, the utilization of scenarios 2 and 5 remains stable as requests continuously keep on arriving but the corresponding net profit varies due to users relinquishing their service. Moreover, because of the high incoming stream of users (around 5500 in 25 hours), the available capacity remains contented most of the time as servers run at high utilization. Noticeably in Figure 4.20 (a and b), from time 4:00 till the end, scenario 5 generates slightly greater profit than scenario 2 which leads to a greater ONP for scenario 5. As discussed earlier, this is because the optimization model



(a) (b)

Figure 4.20: Utilization vs (b) Net profit for all the scenarios at λ=0.064 *req/sec*

used in scenario 5 uses linear regression to predict the behavior of users and intelligently assigns the resources so that the impact of relinquishment of users is minimized.

Thus, as per the deductions of Section 4.2, the objective of maximizing profit is achieved when resources in a finite resource pool get contented with users relinquishing their services. In addition to that, though the proposed model tends to process the maximum number of users, the predicted loyal users within the batch are given more priority in terms of allocating the share of remaining capacity to maximize profit. Consequently, some disloyal or average users were denied the service. Interestingly, as compared to its counterparts, AOU has also improved in all the cases for scenario 5.

On the other hand, when the resources are fully utilized at high arrival rates, many user requests were also blocked in some of the scenarios. Since the arrival of users does not depend upon the server capacity or total available resources, the blocking ratio varies and depends upon the arrival rates of the incoming requests. As this study concentrates on the economic aspects of cloud computing, therefore from a business point of view, it is important to know the blocking rate as denying the service leaves a negative impact on the users and deteriorates the market-value of the CSP. Therefore, the blocking probability gives another piece of information to the CSP on evaluating the performance of the resource allocation policy being used. The blocking probability (P_b) is calculated as:

$$P_b = \frac{Number\ of\ users\ blocked\ \ due\ to\ full\ capacity}{Total\ number\ of\ users\ arrived} \tag{4.1}$$

The blocking probabilities of all the scenarios are outlined in Table 4.4. It can be clearly seen that no users were blocked for lower arrival rates. Noticeably for *λ=0.064,* the blocking probability of scenario 5 is less than scenario 2 even with slightly higher AOU as scenario 5 makes better resource assignment decision during the contention of resources. Additionally, the proposed model also shows improved utilization at higher arrival rates as compared to the similar models proposed in the literature. In a nutshell, it can be said that rather than assigning whatever is requested or using approximate algorithms, the provider must assign the resources using the optimization techniques to maximize his net profit.

Table 4.4: Major outcomes

Arrival Rate	**λ=0.008**			**λ=0.016**			**λ=0.032**			**λ=0.064**		
	AOU (%)	ONP (CAD)	P_b	AOU (%)	ONP (CAD)	P_b	AOU (%)	ONP (CAD)	P_b	AOU (%)	ONP (CAD)	P_b
Scenario 1	28.7	189k	0	57.7	357k	0	84.7	537k	0.4	88.2	545k	0.54
Scenario 2	17.4	108k	0	35.3	209k	0	68.7	418k	0	84.4	505k	0.4
Scenario 3	7.8	47k	0	15.7	94k	0	30.8	189k	0	59.2	370k	0
Scenario 4	14.7	91k	0	22.6	139k	0	41.2	251k	0	66.2	403k	0
Scenario 5	14.4	93k	0	28.3	177k	0	55.8	349k	0	85.1	512k	0.32

Chapter 5: Conclusion and future work

This chapter initially provides a summary of our economics-inspired research work which proposes a framework to calculate and maximize the net profit of the cloud service provider in an environment where users can leave the system at any point in time. Then, we outlined the three contributions from our work and conclude this chapter with a list of future directions.

5.1 Summary of the thesis

Since its emergence, cloud computing has been the topic of attraction for both researchers and the corporates. The reason being the business it has provided to the IT enterprises as most of the users are migrating their in-house computing facilities to the centrally hosted cloud-based servers. Fascinatingly, users use the resources of the cloud providers and pay according to their timely usage. While this relocation is beneficial for the users as they save the upfront investments and additional hardware costs, cloud service providers, on the other hand, face many challenges in terms of maximizing their net profit. This idea set out the basis of this thesis for which the general goal was to propose a mechanism to maximize the net profit of cloud providers.

The goal was divided into three objectives as mentioned in Chapter 1. As users can relinquish according to their wish, providers lose money in the form of opportunity cost. Therefore, our first objective was to create a model which could provide a cost-benefit analysis based upon the relinquishment of users. The second objective of this work was to find a technique to improve the accuracy of the predictions based on the historical behavior of the users. The third and final objective was to propose a resource management framework to maximize the net profit of a CSP by optimally estimating the amount of resources to assign to each user requests.

5.1.1 Summary of contributions

This sub-section provides the summary of major findings of the thesis and their results. To fulfill the objectives presented in Chapter 1, the theory of the findings is presented in three phases in Chapter 3 whereas the results of the theory are outlined in Chapter 4. Collectively, the major contributions of this thesis are as follows:

- In the first phase, we focused on the techno-economical aspects of cloud service providers in a cloud federation environment. A Relinquishment-Aware Cloud Economics (RACE) model was proposed to evaluate the net profit of cloud service providers. The main contribution of the model is the consideration of income, electricity expenses and the relinquishment loss which is modeled to evaluate net profit. Various resource assignment schemes from the literature were evaluated using this method. Various scenarios were used accordingly to calculate the utilization and the net profit for different arrival rates of users. Several simulations were carried out using CloudSim to obtain the stated results in Section 4.2. Results show that with users relinquishing and a finite resource pool, blindly increasing the utilization is not economically beneficial for CSP. Although the net profit increases, it majorly got impacted by the relinquishment of users in a finite resource pool environment.

- Since users can relinquish their service at any point in time, it is essential to have ways in which cloud providers can predict the behavior of users. Different ways were proposed in the literature but they were mainly using some sort of average based on the historical behavior of users. Seeing this, the idea was to figure out a better prediction technique such that the CSP could make accurate decisions while estimating the amount of resources to be allocated. As a result, we developed a prediction algorithm based on the concept of linear regression. Users with different usage behaviors (loyal, average and disloyal) were evaluated for different history lengths. The results showed that the technique based on linear regression outperformed all the user behavior prediction techniques proposed in the literature.
- Lastly, a relinquishment aware resource optimization model was proposed to optimally estimate the amount of resources such that the net profit of the CSP can be maximized. The model uses the prediction results of the linear regression technique to make the optimal resource estimation based upon the user history. To carry out the experimental results, variable user batch sizes were tested for the processing times and the number of requests processed per batch. The performance analysis carried out in CPLEX depicts that a batch size of five user requests took the minimum processing time while processing the maximum percentage of requests per batch. Furthermore, the proposed optimization model is also compared against two algorithms from the literature. The simulations are implemented in CloudSim to carry out the comparison. The results of the simulations show that the optimization model generates the

maximum profit when resource contention occurs at high user arrival rate. In addition to that, the blocking probability of users is calculated as an additional parameter to check the performance of proposed model. Also, the percentage of users blocked by the proposed approach was comparatively less while it generated greater profit at almost comparable overall utilization of the server of other models.

5.2 Future work

To the best of our knowledge, since this study is unique and first of its own kind, there are many issues that can be addressed to extend this work. Here is a list of possible directions:

- The RACE model can be broadened by including additional cost parameters related to the reliability, security, communication redundancy and all the other energy costs of the cloud systems. This can be helpful in making the model more complete and let the CSP be aware of its complete budget outline.
- Linear regression was modeled using only a single feature: relinquish probability. We believe that including additional features such as profit generated from the user at each instance could potentially improve the performance of the linear regression algorithm.
- In the thesis, the resource price of Amazon EC2 instance is used. In the future, the resource price can also be predicted based upon the user history. This can further help CSP to increase the net profit and compensate the relinquishment

loss. Further, based upon the new pricing strategy, a reimbursement strategy can be added to have a complete pricing model.

- The simulations of the proposed optimization model were carried out for a single service type: CPU. As future work, all the service types could be included to make the model more realistic.
- Finally, real-time data traces could also be used to get actual habits and behavior of real cloud customers.

References

[1] M. Alhamad, T. Dillon and E. Chang, "Conceptual SLA framework for cloud computing," in *4th IEEE International Conference on Digital Ecosystems and Technologies*, Dubai, pp. 606-610, 2010.

[2] R. Buyya, C. Yeo, S. Venugopal, J. Broberg and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Computer Systems,* vol. 25, no. 6, pp. 599-616, June 2009.

[3] A. Qouneh, M. Liu and T. Li, "Optimization of Resource Allocation and Energy Efficiency in Heterogeneous Cloud Data Centers," in *2015 44th International Conference on Parallel Processing*, Beiging, pp. 1-10, 2015.

[4] M. Simjanoska, S. Ristov, G. Velkoski and M. Gusev, "Scaling the performance and cost while scaling the load and resources in the cloud," in *2013 36th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, Opatija, pp. 151-156, 2013.

[5] [Online]. Available: https://cloud.google.com/solutions/mobile/. [Accessed July 2017].

[6] M. Weinberger. [Online]. Available: http://www.businessinsider.com/google-cloud-platform-wins-snapchat-2017-2. [Accessed July 2017].

[7] [Online]. Available: https://cloud.google.com/docs/. [Accessed July 2017].

[8] R. Cohen, "Gartner Announces 2012 Magic Quadrant for Cloud Infrastructure as a Service”," [Online]. Available: http://www.forbes.com/sites/reuvencohen/2012/10/22/gartner-announces-2012magic-quadrant-for-cloud-infrastructure-as-a-service/ . [Accessed June 2017].

[9] I. S. Moreno and J. Xu, "Neural Network-Based Overallocation for Improved Energy-Efficiency in Real-Time Cloud Environments," in *2012 IEEE 15th International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing*, Guangdong, pp. 119-126, 2012.

[10] Netflix. [Online]. Available: https://www.netflix.com/signup/planform. [Accessed 19 September 2017].

[11] "Amazon EC2 Pricing," [Online]. Available: https://aws.amazon.com/ec2/pricing/on-demand. [Accessed 11 April 2017].

[12] S. Moulik, S. Misra and A. Gaurav, "Cost-Effective Mapping between Wireless Body Area Networks and Cloud Service Providers Based on Multi-Stage Bargaining," *IEEE Transactions on Mobile Computing,* vol. 16, no. 6, pp. 1573-1586, 2017.

[13] M. Aazam and E. N. Huh, "Broker as a Service (BaaS) Pricing and Resource Estimation Model," in *2014 IEEE 6th International Conference on Cloud Computing Technology and Science*, Singapore, pp. 463-468, 2014.

[14] Q. Hu, "Reactive Prediction Model for Cloud Resource Estimation (Maters Thesis)," Carleton University, Ottawa, Canada, 2016.

[15] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica and M. Zaharia, "A view of cloud computing," *Communications of the ACM,* vol. 53, no. 4, pp. 50-58, 2010.

[16] Q. Zhang, L. Cheng and B. R, "Cloud computing: state-of-the-art and research challenges," *Journal of Internet Services and Applications,* vol. 1, no. 1, pp. 7-18, 2010.

[17] S. Marston, Z. Li, S. Bandyopadhyay, J. Zhang and A. Ghalsasi, "Cloud computing — The business perspective," *Decision Support Systems,* vol. 51, no. 1, pp. 176-189, 2011.

[18] C. Xu, Z. Zhao, H. Wang, R. Shea and J. Liu, "Energy Efficiency of Cloud Virtual Machines: From Traffic Pattern and CPU Affinity Perspectives," *IEEE Systems Journal,* vol. 11, no. 2, pp. 835-845, 2017.

[19] A. R. Roy, S. R. Chowdhury, M. F. Bari, R. Ahmed and R. Boutaba, "Emulating an infrastructure with EASE," in *2016 12th International Conference on Network and Service Management (CNSM)*, Montreal, QC, pp. 167-173, 2016.

[20] L. L. Burch, P. U. Mukkara and D. G. Earl, "Techniques for sharing virtual machine (VM) resources". United States Patent US8831993 B2, 09 September 2014.

[21] M. F. Bari, R. Boutaba, R. Esteves, L. Z. Granville, M. Podlesny, M. G. Rabbani, Q. Zhang and M. F. Zhani, "Data Center Network Virtualization: A Survey," *IEEE Communications Surveys Tutorials,* vol. 15, no. 2, pp. 909-928, 2013.

[22] [Online]. Available: http://www.idc.com/home.jsp. [Accessed 08 August 2017].

[23] D. Hilley, "Cloud Computing: A Taxonomy of Platform and Infrastructure-level," 2009. [Online]. Available: https://smartech.gatech.edu/handle/1853/34402. [Accessed 18 June 2017].

[24] M. Nir, M. St-Hilaire and A. Matrawy, "Economic and Energy Considerations for Resource Augmentation in Mobile Cloud Computing," *IEEE Transactions on Cloud Computing,* vol. 99, pp. 1-1, 2015.

[25] M. Ali, "Green cloud on the horizon," in *Proceedings of the 1st International Conference on Cloud Computing (CloudCom)*, Manila, pp. 451–459, 2009.

[26] H. T. Dinh, C. Lee, D. Niyato and P. Wang, "A survey of mobile cloud computing: architecture, applications, and approaches," *Wireless Communications and Mobile Computing,* vol. 13, no. 18, pp. 1587-1611, 2013.

[27] A. Botta, W. D. Walter, V. Persico and A. Pescapé, "Integration of Cloud computing and Internet of Things: A survey," *Future Generation Computer Systems,* vol. 56, pp. 684-700, March 2016.

[28] L. Atzori, A. Iera and G. Morabito, "The Internet of Things: A survey," *Computer Networks,* vol. 54, no. 15, pp. 2787-2805, 2010.

[29] A. Bassi and G. Horn, "Internet of Things in 2020: A Roadmap for the Future," European Commission: Information Society and Media, 2008.

[30] S. C. B. Intelligence, "Disruptive Civil Technologies," Six Technologies with Potential Impacts on US Interests Out to 2025, April 2008.

[31] A. M. Kuo, "Opportunities and challenges of cloud computing to improve health care services.," *Journal of medical Internet research,* vol. 13, no. 3, 2011.

[32] B. P. Rao, P. Saluia, N. Sharma, A. Mittal and S. V. Sharma, "Cloud computing for Internet of Things & sensing based applications," in *IEEE 2012 Sixth International Conference on Sensing Technology (ICST)*, Kolkata, pp. 374-380, 2012.

[33] S. Dash, S. Mohapatra and P. K. Pattnaik, "A survey on application of wireless sensor network using Cloud computing," *International Journal of Computer Science & Emerging Technologies,* vol. 1, no. 4, pp. 50-55, 2010.

[34] A. Prati, R. Vezzani, M. Fornaciari and R. Cucchiara, "Intelligent video surveillance as a service," *Intelligent Multimedia Surveillance,* pp. 1-16, 2013.

[35] K. Lee, D. Murray, D. Hughes and W. Joosen, "Extending sensor networks into the Cloud using Amazon Web Services," in *2010 IEEE International Conference on Networked Embedded Systems for Enterprise Applications*, Suzhou, China, pp. 1-7, 2010.

[36] L. Mashayekhy, M. M. Nejad and D. Grosu, "Cloud federations in the sky: Formation game and mechanism," *IEEE Transactions on Cloud Computing,* vol. 3, no. 1, pp. 14-27, 2015.

[37] T. Kurze, M. Klems, D. Bermbach, A. Lenk, S. Tai and M. Kunze, "Cloud Federation," in *Proceedings of the 2nd International Conference on Cloud Computing*, pp. 32-38, 2011.

[38] S. G. Grivas, T. U. Kumar and H. Wache, "Cloud broker: Bringing intelligence into the cloud," in *2010 IEEE 3rd International Conference on Cloud Computing*, Miami, FL, USA, pp. 544-545, 2010.

[39] N. Sfondrini and G. Motta, "SLA-aware broker for Public Cloud," in *2017 IEEE/ACM 25th International Symposium on Quality of Service (IWQoS)*, Vilanova i la Geltru, Spain, pp. 1-5, 2017.

[40] D. Abramson, R. Buyya and J. Giddy, "A computational economy for grid computing and its implementation in the Nimrod-G resource broker," *Future Generation Computer Systems,* vol. 18, no. 8, pp. 1061-1074, 2002.

[41] A. Cuomo, G. Di Modica, S. Distefano, A. Puliafito, M. Rak, O. Tomarchio, S. Venticinque and U. Villano, "An SLA-based broker for cloud infrastructures," *Journal of grid computing,* vol. 11, no. 1, pp. 1-25, 2013.

[42] G. Anastasi, E. Carlini, M. Coppola and P. Dazzi, "Qbrokage: A genetic approach for qos cloud brokering," in *2014 IEEE 7th International Conference on Cloud Computing*, Anchorage, AK, USA, pp. 304-311, 2014.

[43] F. Jrad, J. Tao and A. Streit, "Simulation-based evaluation of an intercloud service broker," in *The Third International Conference on Cloud Computing, GRIDs, and Virtualization*, Nice, France, pp. 140–145, 2012.

[44] J. Emeras, S. Varrette, V. Plugaru and P. Bouvry, "Amazon Elastic Compute Cloud (EC2) vs. in-House HPC Platform: a Cost Analysis," *IEEE Transactions on Cloud Computing,* vol. 99, pp. 1-14, 2017.

[45] M. Macıas and J. Guitart, "Maximising revenue in cloud computing markets by means of economically enhanced SLA management," Computer Architecture Department Universitat Politecnica de Catalunya, 2010.

[46] M. Mazzucco, Dyachuk, D and R. Deters, "Maximizing Cloud Providers' Revenues via Energy Aware Allocation Policies," in *3rd International Conference on Cloud Computing*, Miami, pp. 131-138, 2010.

[47] H. Xu and B. Li, "Maximizing revenue with dynamic cloud pricing: The infinite horizon case," in *IEEE International Conference on Communications (ICC)*, Ottawa, pp. 2929-2933, 2012.

[48] X. Cui, B. Mills, T. Znati and R. Melhem, "Shadows on the Cloud: An Energy-aware, Profit Maximizing Resilience Framework for Cloud Computing," in *4th International Conference on Cloud Computing and Services Science(CLOSER 2014)*, Barcelona, 2014.

[49] P. Babu and K. R. R. Babu, "Cloud Revenue Maximization using Competition and Cooperation," in *2016 International Conference on Micro-Electronics and Telecommunication Engineering (ICMETE)*, Ghaziabad, pp. 240-244, 2016.

[50] J. O. Melendez, A. Biswas, S. Majumdar, B. Nandy, M. Zaman, P. Srivastava and N. Goel, "A Framework for Automatic Resource Provisioning for Private Cloud," in *13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing(CCGrid 2013)*, Delft, 2013.

[51] M. Hadji and D. Zeghlache, "Mathematical Programming Approach for Revenue Maximization in Cloud Federations," *IEEE Transactions On Cloud Computing,* vol. 5, no. 1, pp. 99-111, 2017.

[52] U. Lampe, M. Siebenhaar, A. Papageorgiou, D. Schuller and R. Steinmetz, "Maximizing Cloud Provider Profit from Equilibrium Price Auctions," in *2012*

IEEE Fifth International Conference on Cloud Computing, Honolulu, HI, pp. 83-90, 2012.

[53] M. Mitchell T, Machine Learning, McGraw Hill, 1997.

[54] S. Wang and R. M. Summers, "Machine learning and radiology," *Medical Image Analysis,* vol. 16, no. 5, pp. 933-951, 2012.

[55] S. Russell and P. Norvig, Artificial intelligence: a modern approach (3rd edition), New Jersey: Prentice Hall, Pearson Education, Inc, 2010.

[56] P. Kulkarni, Reinforcement and Systematic Machine Learning For Decision, Wiley-IEEE Press, 2012.

[57] "Unsupervised Learning," MathWorks, [Online]. Available: https://www.mathworks.com/discovery/unsupervised-learning.html. [Accessed 14 August 2017].

[58] "Linear Regression," MathWorks, [Online]. Available: https://www.mathworks.com/discovery/linear-regression.html. [Accessed 14 August 2017].

[59] T. Salman, D. Bhamare, A. Erbad, R. Jain and S. M, "Machine Learning for Anomaly Detection and Categorization in Multi-Cloud Environments," in *2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud)*, New York, pp. 97-103, .

[60] C. L. M. Belusso, S. Sawicki, V. Basto-Fernandes, R. Z. Frantz and F. Roos-Frantz, "Price modeling of laaS providers using multiple regression," in *2017 12th*

Iberian Conference on Information Systems and Technologies (CISTI), Lisbon, pp. 1-6, 2017.

[61] F. Farahnakian, P. Liljeberg and J. Plosila, "LiRCUP: Linear Regression Based CPU Usage Prediction Algorithm for Live Migration of Virtual Machines in Data Center," in *2013 39th Euromicro Conference on Software Engineering and Advanced Applications*, Santander, Spain, pp. 357-364, 2013.

[62] I. S. Moreno, P. Garraghan, P. Townend and J. Xu, "An Approach for Characterizing Workloads in Google Cloud to Derive Realistic Resource Utilization Models," in *2013 IEEE 7th International Symposium on Service Oriented System Engineering (SOSE)*, San Francisco, pp. 49-60, 2013.

[63] W. Fang, Z. Lu, J. Wu and C. Z, "RPPS: A Novel Resource Prediction and Provisioning Scheme in Cloud Data Center," in *IEEE Ninth International Conference on Services Computing (SCC)*, Washington, pp. 609 -616, 2012.

[64] I. Sadeka, J. Keung, K. Lee and A. Liu, "Empirical prediction models for adaptive resource provisioning in the cloud," *Future Generation Computer Systems,* vol. 28, no. 1, pp. 155-165, 2012.

[65] T. Truong-Huu and G. M, "Handling Uncertainty and Diversity in Cloud," in *2015 International Conference on Cloud Computing Research and Innovation*, Singapore, pp. 65-72, 2015.

[66] X. Yi, Y. Liu, Z. Li and H. Jin, "Flexible Instance: Meeting Deadlines of Delay Tolerant Jobs in The Cloud with Dynamic Pricing," in *2016 IEEE 36th*

International Conference on Distributed Computing Systems, Nara, Japan, pp. 415-424, 2016.

[67] J. Li, Z. Bao and Z. Li, "Modeling Demand Response Capability by Internet Data Centers Processing Batch Computing Jobs," *IEEE Transactions On Smart Grid,* vol. 6, no. 2, pp. 737-747, 2015.

[68] I. Menache, O. Shamir and N. Jain, "On-demand, Spot, or Both: Dynamic Resource Allocation for Executing Batch Jobs in the Cloud," in *11th International Conference on Autonomic Computing (ICAC'14)*, Philadelphia, PA, 2014.

[69] A. Chakraborti, D. Challet, A. Chatterjee, M. Marsili, Y. Zhang and B. K. Charkrabarti, "Statistical mechanics of competitive resource allocation using agent-based models," *Physics Reports,* vol. 552, pp. 1-25, 2015.

[70] M. A. Salehi, A. N. Toosi and R. Buyya, "Contention management in federated virtualized distributed systems: implementation and evaluation," *Journal of Software: Practice and Experience,* vol. 44, no. 3, pp. 353-368, 2014.

[71] S. Tang, B.-S. Lee and B. He, "Towards Economic Fairness for Big Data Processing in Pay-as-You-Go Cloud Computing," in *6th International Conference on Cloud Computing Technology and Science*, Singapore, pp 638-643, December 15-18, 2014.

[72] M. Mashaly and P. J. Kuehn, "Modeling and Analysis of Virtualized Multi-Service Cloud Data Centers with Automatic Server Consolidation and Prescribed Service Level Agreements," in *2016 IEEE 41st Conference on Local Computer Networks Workshops (LCN Workshops)*, Dubai, pp 9-16, 7-10 Nov. 2016.

[73] L. A. Sjaastad, "The Costs and Returns of Human Migration," *Journal of Political Economy,* vol. 70, no. 5, pp. 80-93, 1962.

[74] J. Cao, K. Hwang, K. Li and A. Y. Zomaya, "Optimal Multiserver Configuration for Profit Maximization in Cloud Computing," *IEEE Transactions on Parallel and Distributed Systems,* vol. 24, no. 6, pp. 1087-1096, 2013.

[75] M. Alghamdi, B. Tang and Y. Chen, "Profit-based file replication in data intensive cloud data centers," in *2017 IEEE International Conference on Communications (ICC)*, Paris, 21-25 May 2017.

[76] X. Li, Y. Li, T. Liu, J. Qiu and F. Wang, "The Method and Tool of Cost Analysis for Cloud Computing," in *IEEE International Conference on Cloud Computing*, Bangalore, 21-25 September 2009.

[77] K. Jungck and S. Rahman, "Cloud Computing Avoids Downfall of Application Service Providers," *International Journal of Information Technology Convergence and Services (IJITCS),* vol. 1, no. 3, pp. 1-20, 2011.

[78] T. Thanakornworakij, R. Nassar, C. Leangsuksun and M. Paun, "An Economic model for maximizing profit of a Cloud Service Provider," in *7th International Conference on Availability, Reliability and Security (ARES 2012)* , Prague, pp. 274 – 279, 20-24 Aug 2012.

[79] M. A. Khoshkholghi, M. N. Derahman, A. Abdullah, S. Subramaniam and M. Othman, "Energy-Efficient Algorithms for Dynamic Virtual Machine Consolidation in Cloud Data Centers," *IEEE Access,* vol. 5, pp. 10709-10722, 2017.

[80] F. Ryckbosch, S. Polfliet and L. Eeckhout, "Trends in Server Energy Proportionality," *Computer,* vol. 44, no. 9, pp. 69-72, 2011.

[81] C. D. Patel and A. J. Shah, "Cost Model for Planning, Development and Operation of a Data Center," Internet Systems and Storage Laboratory, HP Laboratories, Palo Alto , 2005.

[82] D. Paul, W.-D. Zhong and S. K. Bose, " Energy efficiency aware load distribution and electricity cost volatility control for cloud service providers," *Journal of Network and Computer Applications,* vol. 59, pp. 185-197, 2016.

[83] Y. Jin, Y. Wen and Q. Chen, "Energy efficiency and server virtualization in data centers: An empirical investigation," in *2012 Proceedings IEEE INFOCOM Workshops*, Orlando, FL, pp. 133-138, 2012.

[84] J. W. Payne and J. R. Bettman, "When Time is Money:Decision Behaviour under Opportunity-Cost Time Pressure," *Organizational Behavior and Human Decision Processes,* vol. 66, no. 2, pp. 131-152, 1996.

[85] K. Sonkar S and M. U. Kharat, "A review on resource allocation and VM scheduling techniques and a model for efficient resource management in cloud computing environment," in *2016 International Conference on ICT in Business Industry & Government (ICTBIG)*, Indore, pp. 1-7, 2016.

[86] J. Chen, Y. Qin, Y. Ye and Z. Tang, "A Live Migration Algorithm for Virtual Machine in a Cloud Computing Environment," in *2015 IEEE 12th Intl Conf on Ubiquitous Intelligence and Computing and 2015 IEEE 12th Intl Conf on Autonomic and Trusted Computing and 2015 IEEE 15th Intl Conf on Scalable*

Computing and Communications and Its Associated Workshops (UIC-ATC-ScalCom), Beijing, pp. 1319-1326, 2015.

[87] S. Kaur and S. Bawa, "A review on energy aware VM placement and consolidation techniques,," in *2016 International Conference on Inventive Computation Technologies (ICICT)*, Coimbatore, pp. 1-7, 2016.

[88] L. Wang, J. Xu, H. A. Duran-Limon and M. Zhao, "QoS-Driven Cloud Resource Management through Fuzzy Model Predictive Control," in *2015 IEEE International Conference on Autonomic Computing*, Grenoble, pp. 81-90, 2015.

[89] R. N. Calheiros and R. Buyya, "Cost-Effective provisioning and scheduling of deadline-constrained applications in hybrid clouds," in *Web Information Systems Engineering* - *Wise* , Heidelberg , Springer, 2012, p. 171–184.

[90] O. A. Abdul-Rahman and K. Aida, "Towards Understanding the Usage Behavior of Google Cloud Users: The Mice and Elephants Phenomenon," in *2014 IEEE 6th International Conference on Cloud Computing Technology and Science*, Singapore, pp. 272-277, 2014.

[91] M. Kudinova, A. Melekhova and A. Verinov, "CPU utilization prediction methods overview," in *Proceedings of the 11th Central & Eastern European Software Engineering Conference in Russia (CEE-SECR '15)*, Moscow, 2015.

[92] J. Yang, C. Liu, Y. Shang, Z. Mao and J. Chen, "Workload Predicting-Based Automatic Scaling in Service Clouds," in *2013 IEEE Sixth International Conference on Cloud Computing*, Santa Clara, CA, pp. 810-815, 2013.

[93] A. Ng, "Machine Learning," Coursera, [Online]. Available: https://www.coursera.org/learn/machine-learning/home/welcome. [Accessed 06 August 2017].

[94] F. F. Lubis, Y. Rosmansyah and S. H. Supangkat, "Gradient descent and normal equations on cost function minimization for online predictive using linear regression with multiple variables," in *2014 International Conference on ICT For Smart Society (ICISS)*, Bandung,, pp. 202-205, 2014.

[95] R. Calheiros, R. Ranjan, A. Beloglazov, C. D. Rose and R. Buyya, "Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and Experience,* vol. 41, no. 1, pp. 23-50, 2011.

[96] O. E. Board, "Electricity Rates," [Online]. Available: https://www.oeb.ca/rates-and-your-bill/electricity-rates. [Accessed 02 August 2017].

[97] "IBM ILOG CPLEX Optimization Studio V12.5.1," IBM, [Online]. Available: http://pic.dhe.ibm.com/infocenter/cosinfoc/v12r5/index.jsp. [Accessed 01 08 2017].

[98] H. Chang, M. Kodialam, R. R. Kompella, T. V. Lakshman, M. M. Lee and S. Mukherjee, "Scheduling in mapreduce-like systems for fast completion time," in *IEEE INFOCOM 2011*, Shanghai, China, pp. 3074-3082, 2011.

[99] N. Lim, S. Majumdar and P. Ashwood-Smith, " Engineering resource management middleware for optimizing the performance of clouds processing mapreduce jobs

with deadlines," in *Proceedings of the 5th ACM/SPEC international conference on Performance engineering (ICPE '14)*, NY, USA, pp. 161-172, 2014.