Cloud Resource Prediction for Healthcare as a Scientific Application

Thesis submitted in partial fulfilment of the requirements for the award of degree of

Master of Engineering
in
Software Engineering

Submitted By
Rashmeet Toor
(Roll No. 801431019)

Under the supervision of:
Dr. Inderveer Chana
Professor, CSED

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004

June 2016

CERTIFICATE

I hereby certify that the work which is being presented in the thesis titled, "*Cloud Resource Prediction for Healthcare as a Scientific Application*", in partial fulfilment of the requirements for the award of degree of Master of Engineering in *Software Engineering* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of Dr. Inderveer Chana and refers other researchers work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for the award of any other degree of this or any other University.

Rashmeet
(Rashmeet Toor)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

23/06/16
(Dr. Inderveer Chana)
Professor,
Computer Science and Engineering Department,
Thapar University, Patiala

Countersigned by

(Dr. Maninder Singh)
Head,
Computer Science and Engineering Department,
Thapar University, Patiala

(Dr S. S. Bhatia)
Dean (Academic Affairs),
Thapar University,
Patiala

Acknowledgement

I am thankful to the almighty for his blessings with which I could reach so far. I wish to express my deep gratitude and regard to **Dr. Inderveer Chana,** Professor, Computer Science and Engineering Department, Thapar University, Patiala, for her help and guidance. She has always motivated me to keep working with her simulating suggestions. It was a pleasure to do this work under her supervision.

I am also thankful to **Dr. Maninder Singh**, Head of Computer Science and Engineering Department and **Dr. Rupali Bhardwaj**, P.G. Coordinator, for their help and encouragement that triggered me for this research work. I would also like to thank the staff members and faculty of Computer Science and Engineering Department for providing all the help, seminars and inspiration which was required for the completion of my research work.

I would like to express my sincere thanks to CSIR (Council of Scientific and Industrial Research), Government of India to carry out my research work under the project titled "Autonomic Resource Prediction and Scheduling for Cloud-based Scientific Applications" with Scheme no.22 (0693)/15/EMR-II.

Last but not the least, I am grateful to all my family members and friends for the inspiration and moral support which enables me to pursue my work.

Rashmeet

Rashmeet Toor

(801431019)

Abstract

Scientific applications demand high performance computing in reasonable time which can be satisfied by using Cloud capabilities. The Scientists can scale up or scale down the infrastructure as per their requirement. Moreover, they need not plan the resources beforehand and can use them on the fly. They can customize the application environment as per requirements. Hence, Cloud is a cost effective, flexible and efficient solution for Scientific applications.

Cloud Computing serves multiple consumers on demand by pooling the resources and charge them as per usage. This makes managing the resources easier. Gradually, advancements were done for resource management by incorporating efficient resource scheduling and prediction techniques. Resource prediction helps in proper resource management not only by attaining QoS requirements for the end users but also by satisfying provider by cutting costs of resources. A lot of research work has been done for Cloud resource prediction, but the existing techniques are not specific to Scientific applications. Moreover, these techniques either satisfy end user or provider.

This thesis explores the gaps in the existing techniques and analyses them to propose a Resource Prediction Model in Cloud environment. The model analyses the application details, monitors the CPU Utilization and execution time, evaluates the optimal number of virtual machines for maximizing CPU Utilization and minimizing execution time, trains a model using the collected dataset and finally perform predictions for a similar application. In order to implement the Resource Prediction Model, a data mining healthcare application, named PharmacoGen was developed and proposed as a case study. The different phases of the proposed model were implemented using the case study to validate the model. The predicted VMs were validated to be having maximum ratio of CPU Utilization and execution time by simulating the application in CloudSim. The proposed model aims to eliminate the research gaps by providing the optimal number of resources for any kind of Scientific application by considering its parameters. It satisfies both the end user and provider as it selects VMs which maximize the CPU utilization and minimize the execution time.

TABLE OF CONTENTS

Certificate	…….	i
Acknowledgement	…….	ii
Abstract	.....	iii
Table of contents	…….	iv
List of Figures	.....	vi
List of Tables	…….	viii
Chapter 1	**Introduction……...**	**1**
1.1	Cloud Computing: Services and Deployment Models……	1
1.2	Evolution of Cloud Computing……..	3
1.3	Cloud Characteristics.……	5
1.4	Applications of Cloud…...……..	6
1.5	Issues in Scientific Applications and Cloud Solutions.….	7
1.6	Research Motivation……...	8
1.7	Organization of Thesis……	8
Chapter 2	**Literature Review……**	**10**
2.1	Cloud Applications in Healthcare……..…….....	10
2.2	Resource Management System……...	12
2.3	Research Questions...…….....	13
2.4	Resource Scheduling and Provisioning…….....	14
2.5	Resource Prediction……	16
2.6	Conclusion……...……	21
Chapter 3	**Problem Statement.…….**	**22**
3.1	Research Gaps…...…….....	22
3.2	Problem Formulation..…….	22
3.3	Objectives……...…….....	22
3.4	Methodology……	23
3.5	Conclusion...……	23

Chapter 4	**Proposed Resource Prediction Model…….……..…....**	**24**
4.1	Model Description……..…….……	24
4.2	Conclusion……..…….….....	28
Chapter 5	**Case Studies on Healthcare..……..……...……...**	**29**
5.1	PharmacoGen: Proposed Healthcare case study..……	29
5.2	Medical Expert System: A case study…….	35
5.3	Conclusion……..…….	37
Chapter 6	**Experimental Results…..……...…….....**	**38**
6.1	Tools for setting Cloud Environment…..……	38
6.2	Implementation of Proposed Model..…..……	41
6.3	Simulation Results……..……	49
6.4	Conclusion……...……	52
Chapter 7	**Conclusions and Future Scope…..……...…….....**	**53**
7.1	Conclusion…….…….…….	53
7.2	Thesis Contributions……..…….…….	53
7.3	Future Work……...…….…….	54
	References…….....	55
	List of Publications & Video Link…….....	61
A.	**Installation of CloudSim 3.0.3 toolkit…….....	**62**

List of Figures

Figure No.	Description	Page No.
1.1	The Cloud Computing Reference Model.…….....	2
1.2	Major Cloud Deployment Models…….....	3
1.3	Convergence of technologies for Cloud Computing.….....	5
4.1	Architecture of Proposed Model....…….....	24
4.2	Proposed Optimal VM Evaluator…..…….....	26
4.3	Steps of Proposed Prediction Model….…….....	28
5.1	Use Case Diagram of PharmacoGen…….…….....	30
5.2	Formatted dataset for analysis……...……....	32
5.3	Generated CSV file by application…….……	32
5.4	Files imported in application……..…….	33
5.5	Functionality of CSV reader/ writer in application..…….....	33
5.6	Analysis using Weka API……...……	34
5.7	Login Screen of PharmacoGen……...	34
5.8	Select Genes Screen of PharmacoGen..……..	35
5.9	Display Results Screen of PharmacoGen…….....	35
5.10	Use Case Diagram of Medical Expert System...…….	36
5.11	Screenshot of query of Medical Expert System……..	36
5.12	Screenshot of results of Medical Expert System ……	37
6.1	Layered architecture of CloudSim……...……	38
6.2	Working of CloudSim....……	40
6.3	Execution of CPUBENCH….……....	41
6.4	Code for getting the current time……..…….....	42
6.5	Displaying start and end time for calculation……..	42
6.6	Length of Cloudlets for PharmacoGen…….....	43
6.7	Length of Cloudlets for Medical Expert System…...……..	43
6.8	Code for creating Resource Monitor in CloudSim..……	43
6.9	Code for VMs Configuration…..……..	44
6.10	Code for Cloudlets Configuration according to PharmacoGen..	45

6.11	Code for Cloudlets Configuration according to Medical Expert System…….	45
6.12	Code of New Broker to get CPU Utilization……..…….....	46
6.13	Code for CPU Utilization evaluation……..	46
6.14	CPU Utilization of single VM in PharmacoGen……..……...	47
6.15	Execution time for single VM in PharmacoGen…….	47
6.16	Execution time for 3 VMs with 6 Cloudlets in PharmacoGen...	47
6.17	Ratio Comparison for PharmacoGen……..…….....	48
6.18	Ratio Comparison for Medical Expert System…….....	48
6.19	Generated Dataset…….....	48
6.20	Classification using Weka…….……..	49
6.21	Test set values for prediction…….…….	50
6.22	Results of prediction in Weka……...……..	50
6.23	Predicted Optimal number of VMs…….	50
6.24	Ratio comparison for application with 20 Cloudlets…….....	51
6.25	CPU Utilization for an application…….…….	51
6.26	Execution time for an application……...…….	51
	A.1……..…….	62

List of Tables

Table 2.1 Cloud Application in Healthcare……..12
Table 2.2 Search Keywords……..13
Table 2.3 Comparison of Resource Scheduling Techniques……15
Table 2.4 Current Resource Prediction Techniques and Algorithms…..…….20
Table 2.5 Comparison of Techniques for different parameters……...21

Chapter 1 Introduction

Cloud Computing heralds an era of on-demand utility computing. It encompasses the provisioning of various configurable resources like storage, applications, services and servers on-demand and pay-per-use basis [1]. Cloud computing is the best solution to the high performance demanding Scientific applications. This chapter introduces Cloud Computing, its architecture, evolution, characteristics, Cloud applications and enumerates issues in Scientific applications and their Cloud solutions.

1.1 Cloud Computing: Services and Deployment models

Cloud Computing has different meanings for different users. Start up companies can translate their ideas into business without much upfront costs by using Cloud services. System developers can utilize Cloud platform for developing software and not worry about infrastructure management. End users can access documents over the Cloud from anywhere and from any device via Internet. Cloud service rents resources to large enterprises, for which they pay only for what they use.

According to NIST, Cloud Computing is a paradigm that enables on-demand access to computing resources that can be shared and configured according to need. The provisioning requires minimal service provider interaction or management effort [2]. Cloud Computing supports the notion of everything as a service, in which different components of a system like hardware, development platforms, storage etc. are delivered and priced as a service. The services can be divided into three classes as shown in Figure 1.1. These three classes form a layered architecture, in which services from lower layers can form services in upper layers. In Section 1.1.4, the various deployment models for Cloud are discussed.

1.1.1 Infrastructure as a Service

The bottom layer of Cloud Computing services is Infrastructure as a Service (IaaS). In this service, infrastructure is provisioned on demand like in AWS (Amazon Web Services), Virtual Machines (VM) are provided with different operating system options and software stack as per users need.

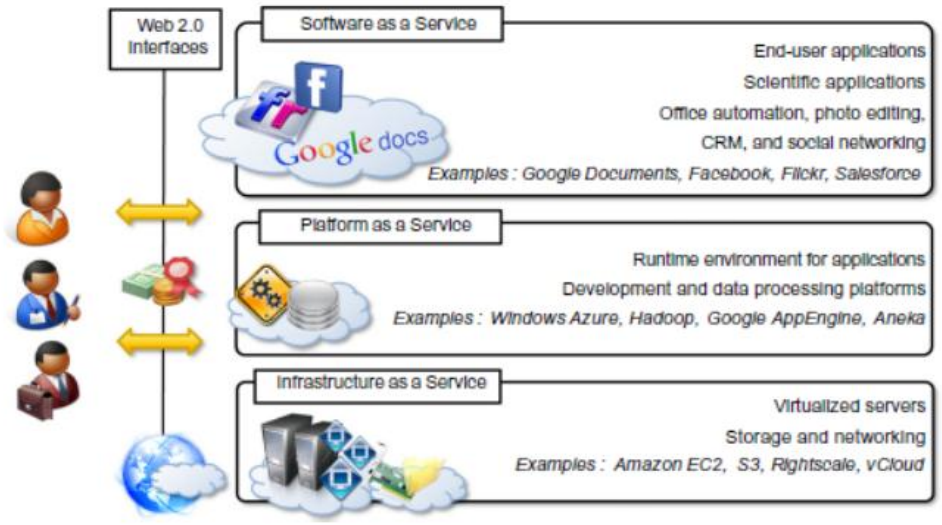


Figure 1.1 The Cloud Computing Reference Model [3]

1.1.2 Platform as a Service

New applications can be created and deployed over the Cloud by using Platform as a Service. It provides the environment for this purpose and the user need not know the memory or processors required by the application. Google AppEngine is one of the examples of such service.

1.1.3 Software as a Service

Instead of residing on local desktops, applications are provided through web portals in SaaS. This eliminates the maintenance part of customers and enhances development and testing by providers. An example of this is Salesforce.com which offers business productivity applications to customers on demand.

1.1.4 Deployment Models

There are different types of models for deploying Cloud environments, each with its own pros and cons. The Service Provider decides the model according to their requirements. The major models are as discussed below [4]:

i) Public Cloud: In this model, the resources are open to general public on a subscription basis. The application and data are deployed on the Cloud provider's premises. The main advantage of Public Cloud is that the user does not need initial investment on infrastructure, but it lacks control over data, security and network settings.

ii) Private Cloud: In order to have full control over data and network settings, an organization can use the Private Cloud. The Cloud is used exclusively by the organization which eliminates security and reliability issues, but it is not beneficial in case minimal up-front costs are required.

iii) Hybrid Cloud: This model is an amalgamation of public and private Cloud components. So it is more flexible and offers good control over the settings. It covers the limitations of both the above models.

The deployment models are as shown in Figure 1.2.

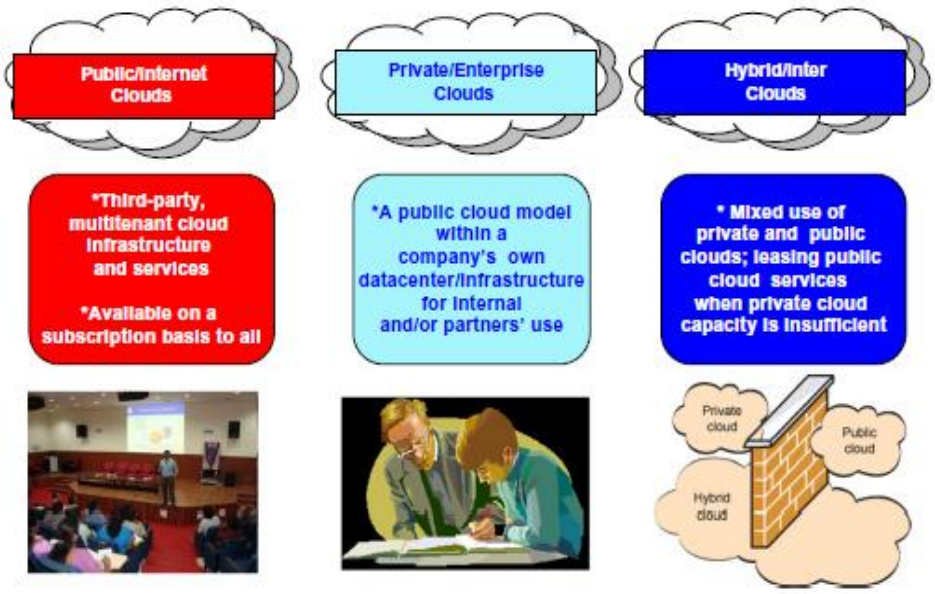


Figure 1.2 Major Cloud Deployment Models [3]

1.2 Evolution of Cloud Computing

The idea of Cloud computing came from evolution of different technologies over the past many years. It dates back to early 1970s when companies offering common processing tasks used mainframe as a utility to serve dozens of applications. Gradually, the different technologies refined and culminated into realization of Cloud. The core technologies behind this realization are as described below [2] [3]:

(i) Distributed Systems: A Distributed system is composed of independent computers which are perceived by users as a single entity. The same concept applies to Cloud

where resources are shared and hence their utilization is improved. The three main paradigms which led to Cloud computing are:

- Mainframes- Mainframes were large, powerful computers used for bulk data processing. Supercomputers and mainframes provided large computational power by using multiple processors but they were expensive.
- Clusters- In 1980s, a low-cost alternative to mainframes was started in which commodity machines were used for high performance computing.
- Grids- With the advent of Internet, geographically dispersed clusters could be merged to form the idea of Grid computing in early 1990s. This led to the use of resources as utility.

Cloud Computing has retained the features of high performance as well as utility computing from these systems.

(ii) Virtualization: Virtualization is a technology for creation of varied computing environments. This technology boosted the idea of Cloud Computing by facilitating VMs utilized by data centers to host different applications and handle numerous users.

(iii) Web 2.0 and Service Oriented Computing: Web 2.0 has led to the access of desktop functions over the internet by integration of various technologies such as XML (eXtensible Markup Language), JavaScript, AJAX (Asynchronous JavaScript and XML), web services etc. Service-oriented computing supports the notion of loosely-coupled, protocol independent and reusable services as the building blocks of an application. Cloud Computing utilizes these two technologies for building applications in the Software as a Service domain.

(iv) Utility Computing: Utility computing implies that resources will be provisioned on a pay-per-use basis which means users will pay for only what they use. Cloud Computing has adopted this idea, which makes it more cost effective.

The convergence of different core technologies for Cloud computing is depicted in Figure 1.3.

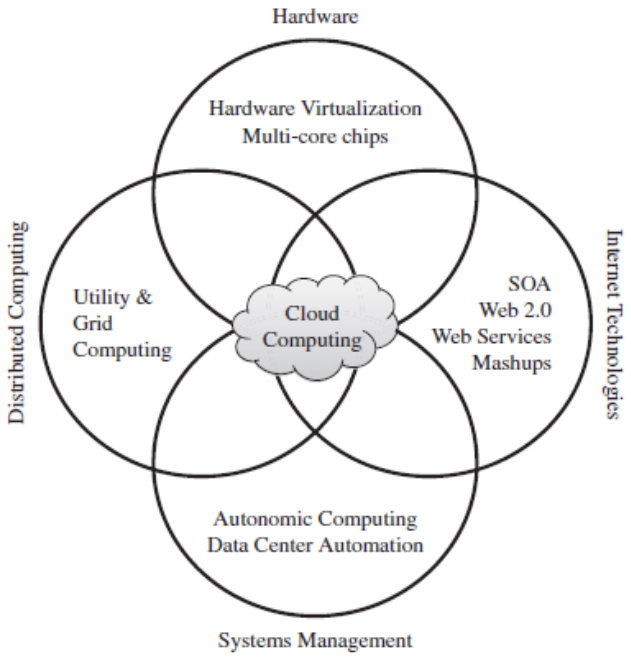


Figure 1.3 Convergence of technologies for Cloud Computing [2]

1.3 Cloud Characteristics

The characteristics of Cloud Computing are as discussed below [3] [4]:

(i) Utility-based pricing: Users are charged for the services on a pay-per-use basis. This feature lets the organization reduce its maintenance, operational and upfront costs and increase the net gain. Due to this reason, many start-up companies and small scale enterprises do not need a large investment to start a new business.

(ii) Resource Pooling: The resources are pooled to serve multiple consumers on demand. The various physical and virtual resources are assigned dynamically. So service providers can manage resource consumption as per their needs.

(iii) Elasticity: In case of rapid changes in demand, the capabilities can be scaled up and down accordingly. So organizations can extend and enhance their capabilities very easily using the feature of elasticity in Cloud.

(iv) Broad network access: Users can access their data and Cloud capabilities from anywhere, and at any time via any device like phone, laptop, tablet or workstation.

(v) Dynamic Resource Provisioning: Instead of provisioning the resources according to peak demand, Cloud provisions resources based on the current demand. This leads to better utilization of resources as well as reduction of operating costs.

(vi) Multi-tenancy: The IT infrastructure and services held by multiple providers are concentrated into large datacenters, leading to better resource utilization and energy efficiency.

(vii) Service-oriented: Every provider needs to take care of service management by offering services to customer based on the negotiated SLA (Service Level Agreement).

1.4 Applications of Cloud

There are numerous domains where Cloud Computing is becoming an essential part. Infact, some existing applications have been shifted over the Cloud for better performance, scalability and efficiency. The various domains which are being affected by Cloud Computing are discussed as below [5]:

(i) Due to technological advances, data can be collected from various devices like Body Area Networks, Wireless Networks, Bluetooth connections and other sensors. Data analytics can be performed on such massive data by integration with the Cloud. It supports batch processing tasks which are completed in shorter period of time than the traditional systems.

(ii) Mobile applications are integrated with the Cloud so that data can be easily communicated across remote areas. Users can access their data even from remote locations. Cloud supports efficient storage and better availability of data.

(iii) Scientific applications require high performance computing in reasonable time. They require ever-increasing resources for growing problem sizes. Cloud is the perfect platform for such applications as it provisions capabilities on-demand. Mathematics software packages like Matlab and Mathematica can use Cloud Computing to perform otherwise expensive evaluations. Healthcare applications like gene expression dataset analysis or MRI (Magnetic Resonance Imaging) brain

imaging workflow are examples of Scientific applications, which require high performance computing.

(iv) One of the domains where Cloud Computing has done magnificent work is Healthcare. Applications using a combination of sensors, medical records and analytics have proved beneficial for management, analysis and sharing of data leading to better health solutions.

1.5 Issues in Scientific Applications and Cloud Solutions

Scientific applications involve construction of numerical solution techniques and mathematical models for solving scientific, engineering or medical problems. These applications require large amount of computing resources in order to perform large scale experiments. Initially clusters were used to satisfy the high performance computing requirements and then Grid computing came into picture. Although Grid computing has been used in many Scientific applications, yet few issues [6] are prevalent as discussed below:

i) Since Grids are shared worldwide, research groups need to submit a proposal describing their research work in order to gain resources. Minor research projects are not able to get resources as other projects compete for it too.

ii) Grids are not much flexible in case of Scientific applications. Grid computing offer a pre packaged environment for execution of applications. Limited options are available in tools, APIs, operating systems or services. Some Scientific experiments cannot be performed under such constraints and have to be redesigned.

Cloud computing, the current trend in IT (Information Technology) offers solution to Scientific applications. Some of them are discussed below:

i) Cloud computing offers a variety of services by means of virtualization technologies. These services cover the entire computing stack and charge the customer on a pay per use basis. The scientists need not plan the required resources and rather use them on the fly.

ii) Scientists can scale up or scale down the computing infrastructure as per application requirements and budget of users.

iii) Scientists can customize the execution environment for performing their experiments. This makes a lot of options available to scientists for constructing their specific Scientific solutions.

Cloud computing is gradually being used in computational science, which is evident from the fact that Yahoo Inc. has extended its partnership with top universities in USA for the advancement of Cloud computing research and applications in the computational science and engineering domain.

1.6 Research Motivation

Effective resource management has been an issue in executing Scientific applications in a Cloud environment. A traditional resource management system handles scheduling, mapping and provisioning of resources on a best fit basis. This approach is not effective in most of the cases. Resource under-provisioning harms system performance and may cause SLA violations. On the other hand, resource over-provisioning increases idle resources and cost wastage. An effective resource management system should automatically adjust resources to minimize cost while satisfying agreed SLAs. This can be done by designing a Cloud resource prediction model. It will also improve resource utilization and balance the load in the datacenter. This brings profit to Cloud service providers and satisfaction to end users. However, building an effective management system which includes all these aspects is a challenging task. This research work is based on the need to create a resource prediction model for efficient resource management in Scientific applications. Healthcare domain comprises various kinds of Scientific applications, so this work is oriented towards enhancing the performance of such healthcare related Scientific applications.

1.7 Organization of Thesis

After the introduction to thesis as discussed in this Chapter,the rest of the thesis is organized in the following order:

- **Chapter 2 Literature Review -** It surveys the various techniques used for resource management and hence resource prediction in Cloud. Since this study is focussed on

healthcare related Scientific applications, it analyses the types of healthcare scenarios in Cloud environments. Further, it compares the various prediction and scheduling techniques.

- **Chapter 3 Problem Statement -** It describes the research gaps and formulates the problem statement. It provides an overview of objectives of the problem statement and also discusses the methodology to solve it.
- **Chapter 4 Proposed Resource Prediction Model-** Resource prediction model for Cloud environment is explained step by step in this chapter. It presents the necessary details of the model and algorithm used.
- **Chapter 5 Case Studies on Healthcare –** It elaborates the healthcare applications used as case study for resource prediction. It provides technical information about the development of the application and describes its implementation with results.
- **Chapter 6 Implementation Details -** This chapter includes the details of implementation and the results as investigated for the case study.
- **Chapter 7 Conclusion and Future Scope -** This chapter concludes the research work and provides insights of the contributions of thesis and the future work.

Chapter 2 Literature Review

Recent developments in Cloud computing has touched many different domains bringing with it high availability, high reliability, low costs, scalability and better performance. Enterprises like Google and Amazon have migrated their massive data into the Cloud for high performance computing. Further, individuals access various components of computing stack on pay as you use basis by renting it from Amazon's EC2. This proves to be an economical and flexible solution, which is the main reason users opt for Cloud Services. This innovative wave of Cloud has affected healthcare domain also to a large extent. A wide variety of computational techniques, storage techniques, softwares and tools are already shaping the future of healthcare. A huge number of healthcare applications are Scientific in nature. It is very important to understand and manipulate the resources required for efficient working of such applications. This chapter first presents the Cloud applications in healthcare domain and then discusses the current techniques in resource provisioning. A detailed discussion of resource prediction is presented followed by analysis of state of the art techniques.

2.1 Cloud Applications in Healthcare

Technical innovations in the healthcare domain have led to solutions that are safe, cost effective, high-quality and easily accessible. Cloud Computing is one of the technologies, which is helping to realize better healthcare applications. Due to technological advances like various sensors, Body Area Networks, wireless networks and bluetooth connections, healthcare data can be easily collected and communicated across remote areas. Cloud provides a platform for efficient storage, better availability and analysis of such massive medical data. In order to comprehend the resource management for any application, the first and foremost step is to understand the various applications in the domain. Some major applications or frameworks studied in this review are:

2.1.1 Cloud framework for Healthcare Monitoring System

A better healthcare monitoring system is provided in [8] by utilizing the Cloud capabilities. The data from various remote patients is stored on a Cloud based medical repository using Aneka. Healthcare could reach global level than only personal level

because of the advent of such type of Cloud framework. The remote patients as well as concerned doctors can communicate among themselves on this platform.

2.1.2 Digital Home based Self care

Applications have been made where the patients can manage their own medical data at home. Microsoft HealthVault is an example of such web-based system where one can store his/her fitness information as well as of his family. A Cloud based solution [9] was proposed which enables patients of chronic diseases to share the health information for different purposes. They proposed a hybrid Cloud using OpenStack and Amazon Web Services (AWS).

2.1.3 Elderly person Reminder App, Care giver Assistant App

An application is developed for smart homes on android platform for tracking of daily life activities of users (mainly elderly people). This application [10] assists health care professionals to keep track of the elderly people. The family members can also keep an eye on their daily activities from remote location or the elderly person himself can be helped through this app so that they can perform multiple activities. The application communicates via web services on the Cloud. Sensors are deployed in the smart home to sense the environment continuously.

2.1.4 Patient Health Management System

Monitoring of a patient is possible at home as well as multiple patients in hospitals or at public health care units [11]. In the proposed architecture, Cloud is used to store medical history of each patient so that the data is easily accessible and can be processed anytime. The unique part of this architecture is the provision of Real Time Health Advice and Action (ReTiHA) service, which is invoked when the Emergency medical response system fails.

2.1.5 Cloud Emergency Medical Service

Provision of Electronic Medical Records in Cloud environment can prove to be successful for Emergency Medical Services (EMS). The proposed service is accessible on android platform and utilizes Google Cloud Messaging for communication. Hence, Cloud provides an efficient, fast and secure platform for Emergency cases [12].

The explosion of genetic data led researchers to not just only accurately analyze such data but also there was a need to find out optimal performance solutions. In [13], approaches to parallelize such massive data are proposed. The 1000 Human Genome Project has already produced 200TB of genomic data which is publicly available on Amazon Cloud Storage2. Cloud technologies can be used for such high performance solutions. The summary of Cloud applications is presented in Table 2.1.

Table 2.1 Cloud Applications in Healthcare

Author	Year	Findings	Technique used
S. Mukherjee, K. Dolui and S. K. Datta [11]	2014	Patient health management system	Cloud storage, ReTiHA for emergency cases
M. Fahim, et al. [10]	2012	Elderly person reminder app, care giver assistant app	Cloud, Android platform
V. Koufi, et al. [12]	2012	Cloud emergency medical service	NefaliEMS
B. E. Reddy, TV Suresh Kumar and G. Ramu [8]	2012	Cloud framework for healthcare monitoring system	Aneka, Xen Server
J. Ma, C. Peng and Q. Chen [9]	2014	Digital home based self-care	OpenStack, AWS

2.2 Resource Management System

Customers submit their jobs along with QoS requirements to service provider and request computing resources from the Cloud. The Cloud service provider checks the condition of resources in the Cloud datacenter at first. If requested resources are available, it will allocate resources according to a scheduling policy. The resource manager takes care of this process. An efficient resource management system can satisfy customers by accomplishing their jobs with required performance and in a budget which is lower than expected. Not only this, it also improves resource utilization and balances the load in datacenter, which in turn satisfies the Cloud service providers by bringing them incremental profit. However, building an efficient resource management system is an arduous task. The Cloud environment is constructed by a wide variety of heterogeneous machines. Different generations of machines with varied specifications are scaled dynamically to ensure cost benefit and high performance. Additionally, Cloud environment deals with heterogeneous workloads as well. Multiple customers have

varying computing objectives and hence, there is diversity in resource demand. This heterogeneity in machines and workloads complicates the method of allocating resources in order to meet time, performance and resource utilization constraints. The second reason, due to which efficient resource management cannot be easily performed, is dynamicity of the Cloud [14]. Again, machines and workloads are both dynamic in Cloud environment. Customers rent virtual machines in order to complete their jobs, each of which is a mixture of various tasks and having different arrival patterns. Due to this dynamic resource usage, traditional methods can hardly predict the future resource usage. The computing resources in Cloud are also dynamically allocated according to the requirement. The dynamicity motivates techniques for dynamic allocation policies for resources which is another challenge to be solved.

2.3 Research Questions

The Literature Review described in this chapter, aims to summarize various management methodologies in Cloud applications and the techniques used for prediction of resources. Various research questions were proposed in order to plan the survey as described below:

1. What is Cloud Computing?
2. What are the various Scientific applications over the Cloud in Healthcare domain?
3. What are the various methods to predict optimal resources for a Scientific application?
4. What is resource provisioning and how are the resources managed in Cloud?
5. What are the essential parameters to be considered when predicting resources for Cloud applications?

After defining the research questions, search strings were defined based on these questions. Search keywords and similar words used in the study are as described in Table 2.2.

Table 2.2 Search Keywords

Search keywords	**Synonyms**
Cloud Computing for E-health	Cloud based Medical Services
Resource Prediction algorithms in Cloud	Resource Management in Cloud Computing
Efficient resource management techniques	Resource provisioning in Cloud

Dynamic Resource Allocation in Cloud	Efficient resource scheduling in Cloud environment
Scientific applications over Cloud	Resource management in Scientific applications

Based on the search questions and keywords, filtering of papers was done to eliminate irrelevant papers. Based on the extracted relevant papers, the resource management for Cloud applications is broadly classified into two tasks:

1) Resource Scheduling and Provisioning
2) Resource Prediction

2.4 Resource Scheduling and Provisioning

In order to maximize resource utilization and efficiently manage the resources, many studies have focused on improving the scheduling and allocation strategies. In Cloud computing, every application runs on the provisioned virtual machines. The user sends the request to process their jobs and the resource manager maps the requests to available resources. This mapping algorithm is enhanced in various studies so that the resources are better utilized leading to reduction in costs. Various approaches include:

(i) Berger model [15] is one of the numerous job scheduling models in a Cloud environment. In this model**,** an upgradation is given by including machine learning approach in the traditional model. This eliminates the disadvantage of incompletion of tasks when task-resources do not match. In this model, the user tasks are classified based on parameters like memory, size, bandwidth and completion time. Different tasks with deadline are assigned priority by a priority assigner. If the jobs do not match with available resources, then a neural network re-classifies those tasks. This method aims to improve completion time of tasks and bandwidth utilization.

(ii) Another study [16] also focuses on improved task scheduling but it is more oriented towards reducing the cost. It calculates the costs of each resource and assigns the priority to tasks according to it. Using this cost profit analysis, it improves the costs and processing times.

(iii) The proposed approach in [17], [18] focuses on two QoS requirements, namely budget and deadline to optimize resource scheduling. It is specific to workflow applications, considering them to be Directed Acyclic Graphs. The user specifies the budget and deadline constraints, and mapping of tasks to optimal service is done using Genetic Algorithm (GA). The results show that Genetic algorithm when combined with non-GA heuristics according to cost or budget constraints, produce better results.

(iv) Another approach for minimizing cost and meeting the deadline is proposed in [19]. This approach was proposed for grid utilities. It combines optimization of scheduling based on QoS constraints as well as prediction of execution times for advance planning.

(v) Markov chain based prediction method is used in [20]**,** which considers the rate of CPU usage, resource failure rate and level of network load to forecast future resource state for scheduling. An evaluation measurement is designed to quantify the prediction result for scheduling.

(vi) An efficient resource scheduling approach for scientific workflows has been designed in [55] which affects the performance of concurrent systems and applications in Cloud. An autonomic fault tolerant scheduling approach has been designed using hybrid heuristics and virtual machine migration. This approach improves total mean execution time, makespan and standard deviation time.

The various approaches discussed above have different objectives. Table 2.3 compares and contrasts the different approaches:

Table 2.3 Comparison of Resource Scheduling Techniques

Author	**Year**	**Method**	**Technique used**	**Improved Parameters**	**Application Specific**	**Stake Holder**
A. Bala and I. Chana	2015	Hybrid heuristics	Virtual Machine Migration	Execution time, Standard deviation time and makespan	Yes, Scientific workflows	Service provider and End user

K. Dinesh et al [15]	2012	Machine learning	Berger Scheduling model + Neural Network	Completion time, Bandwidth utilization	No	Service provider and End user
S. Selvarani et al. [16]	2010	Cost based scheduling	Priority assigned to tasks according to tasks	Cost, Processing Time	No	Service provider and End user
S. Lilli et al. [20]	2009	Prediction	Markov chain based	CPU usage, network load, resource failure	No	Service provider and end user
J. Yu and R. Buyya [17]	2006	Machine learning	Genetic Algorithm based scheduling	Budget, Deadline	Yes, Scientific grid application	End user
J. Yu and R. Buyya [18]	2006	Machine learning	Genetic Algorithm based scheduling	Budget, Deadline	Yes, Scientific Cloud application	End user
J. Yu and R. Buyya [19]	2005	Optimization algorithm	Divide and conquer with Markov Decision	Budget, Deadline	Yes, Grid workflow application	End user

2.5 Resource Prediction

In order to dynamically provision and scale the resources, there are three approaches [21]:

(i) The first one is a reactive approach, which refer to pre-defined threshold-based rules given by application provider. It does not anticipate future needs, instead decisions of scaling are made according to the last values of variables which are monitored. Amazon AutoScaling and some other brokers offer this rule-based auto-scaling mechanism. For example, user can specify a rule of running 3 instances from 11:00-18:00 every day and 5 instances at some other time. Or they can provide a threshold value of CPU utilization to be monitored, and if the threshold value is exceeded or not reached, they can specify specific actions for instances. In [22], threshold based dynamic allocations are performed. For providing such rules, users need to properly understand their requirements. It is evident that if the resource scaling system is reactive, it might not be able to

scale efficiently in case of sudden traffic surge due to some special offer or market campaign. This critically impacts application performance, leading to delay in response time and in some cases, application might be unavailable.

(ii) The second mechanism is a predictive-based approach (also known as proactive), in which historical data of resource usage is analysed and a mathematical model is constructed to anticipate future resource demands. For the prediction model, various techniques can be used like machine learning, linear regression and statistical methodologies. This approach enhances the scaling decision and eliminates the disadvantages of the first approach.

(iii) The last approach is a hybrid method. In a recent study, a hybrid technique is proposed that follows reactive rules for scaling up and regression-based mechanism for scaling down. Such approach is beneficial if appropriate parameters for scaling are known.

It is evident in research that resource prediction techniques improve the costs, execution times, resource utilization along with meeting the specified SLAs.

2.5.1 Need of Resource Prediction

Resource prediction is the need of current Scientific applications because of various reasons:

(i) Service Provider's Perspective: Cloud infrastructures are open to users with different IT backgrounds. These users negotiate a Service Level Agreement (SLA) with the service provider specifying their required conditions. They usually do not know how many resources they might require, but they are aware of the time required for their tasks to be completed. So they tend to express the required QoS in terms of service-level metrics like job deadline instead of resource-level metrics like CPU specification. Usually, providers provision whole resources in order to prevent SLA violations. But it is not a cost-effective option. The providers deal with service-level metrics and they need to map it to resource-level metrics in order to fulfil SLAs and minimize cost. For this purpose, resource requirement is predicted using various techniques so that efficient mapping and scheduling of resources is possible [23].

(ii) End user's perspective: Customers using Cloud services are becoming more and more concerned about the cost of the resources. In Cloud environments, the requests are often served on a best effort basis on the available resources. These requests are referred to as On Demand requests (OD). However, customers can also gain resources using Advance Reservations (AR). In an AR, customers can specify the deadline by which the request must be completed. This guarantees the Quality of Service because accepting an AR request implies the provider enters into a Service Level Agreement with the customer. Moreover, the cost of resources gained by advance reservation plan is cheaper than on-demand. For reserving resources in advance, the users must know their resource requirement. Resource prediction aids to this purpose [24] [25].

(iii) Better Resource Provisioning: Resource prediction also eliminates the chances of over provisioning or under provisioning of resources. Over provisioning does not affect the customer as long as their concern for fulfilment of SLAs is taken care of. But it is an issue for service providers as it increases the cost. Under provisioning might decrease the cost, but it may not fulfil the SLA requirements. If the required resources are known in advance, then efficient provisioning is possible. Hence, resource prediction is an important step for managing resources.

(iv) Better Resource Scheduling and Auto-scaling: The main problem in dynamic provisioning is that new resources are not acquired instantaneously, rather they incur a start-up time which is not negligible. In such scenario, the provider either delays handling client requests until the resources become available or decline the requests. The quality of service is deteriorated in both the cases. Resource prediction enables an insight into future usage, which in turn supports making scaling decisions in advance and compensates for the start-up time [26].

(v) Better Energy efficiency: Due to increasing interest in Cloud services, the datacenter is now larger in scale. The Cloud broker makes full use of hardware to earn more profits from the datacenter, which leads to immense energy consumption [7]. Utilization analysis and predictions can help in reducing the massive amount of energy utilized in Cloud data centers and optimizes the system performance.

Examples of Cloud Resource Management systems include Haizea lease manager which is a middleware that uses leases to manage resources and schedule requests. Amazon CloudWatch is another example of Cloud resource management system which monitors the metrics of deployed applications and allows metric based scaling.

2.5.2 Resource Prediction Techniques

The various resource prediction techniques are discussed below:

(i) When an application needs to scale out, it takes about ten minutes to provision a virtual machine. This latency degrades the notion of on-the-fly provisioning. In [27], a self-adaptive system for resource demand provisioning is proposed. The system predicts the VMs required in advance using temporal mining of historical data. The main highlight of this study is the use of prediction ensemble, where different base prediction methods are given weights and final prediction is close to the best predictor. The selection of best model and training is based on Cloud prediction costs.

(ii) Analysis and then prediction of resource utilization log helps in dynamic allocation of resources as suggested in [28]. This study provides two improvements to the tradional M/M/1 Queuing Theory Predicting Method (MMQMPM). However, the main focus of the study is to control energy consumption.

(iii) Another approach for dynamic resource demand prediction is defined in [29]**.** It is different from other approaches because of two reasons. It focuses on the nature of service tenants i.e. it classifies the tenants as per increase or decrease in their resource requirements and then prioritizes the prediction. Secondly, the allocation of virtual machines to host is enhanced by using best-fit heuristic approach.

(iv) In order to reduce the latency which is evident when a new virtual machine is initialized, a prediction based resource measurement and provisioning is proposed. The historical data is used for time-series analysis using machine learning approaches- Neural Network and Linear Regression along with sliding window technique [31].

(v) Dynamic resource scaling is performed on the basis of prediction of resource workload from past workload using machine learning like SVM (Support Vector Machine) and linear regression. The decision of scaling is based on the current as well predicted state of resources. The underlying parameters considered for it also include various costs and service time forming a broker profit function [25].

(vi) R. N. Calheiros and R. Buyya [30] have designed a complete autonomic prediction system with application provisioner, workload analyzer and load predictor in order to fulfil QoS, reduce costs and energy consumption and improve performance. The future demands are predicted first and then analytical model is generated to decide on the optimal VMs.

(vii) Due to the self-similar nature of web traffic, a pattern based prediction technique is devised which is based on identifying similar past occurences of current workload history. In [26], the focus is on predicting resource usage using String matching concept.

(viii) The resource prediction can also be performed by using statistical prediction methods like double exponential smoothing in [24]. Current and historical records are used for time-series analysis in this study.

(ix) The web requests demand is predicted by performing time series analysis using linear regression. Relationship between web request volume, cost and latency is generated using queuing theory to find optimal number of VMs [21]**.**

The techniques and their respective algorithms are enumerated in Table 2.4.

Table 2.4 Current Resource Prediction Techniques and Algorithms

S. No.	Author	Year	Techniques	Algorithm
1	M. Verma et al. [29]	2016	Time Series	Trend seasonality, exponential moving average
2	A. Biswas et al. [25]	2014	Time Series	SVM, Linear Regression
3	J. Jiang et al. [21]	2013	Time Series	Linear regression, queuing theory
4	J. Huang et al. [24]	2012	Time Series	Double exponential smoothing

5	S. Islam et al. [31]	2012	Time Series	Neural Network/ Linear Regression with sliding window
6	R.N. Calheiros and R. Buyya [30]	2011	Queuing Theory	M/M/1/k queue
7	Y. Jiang et al. [27]	2011	Time series	Best of Moving Average, Auto Regression, ANN, SVM
8	Y. Shi et al. [28]	2011	Utilization Log Analysis	Improved MMQMPM method
9	E. Caron et al. [26]	2010	Machine learning	Pattern matching

2.5.3 Analysis of Prediction Techniques

There are 6 major parameters which are considered by different applications. These parameters are set in order to satisfy either the end user or the provider of Cloud applications. Moreover, very few techniques are application specific. There is no technique specifically proposed for Scientific applications. The comparison of techniques discussed in Table 2.4 for different parameters is presented in Table 2.5.

Table 2.5 Comparison of Techniques for different parameters

S. No.	Optimal VM	Energy Consumption	Budget	Schedule	Utilization	Application Specific
1	✓	×	×	×	×	×
2	✓	✓	✓	✓	×	×
3	✓	×	✓	✓	×	×
4	×	×	×	×	✓	×
5	×	×	×	✓	×	✓
6	✓	×	×	✓	✓	×
7	×	×	✓	×	×	×
8	×	✓	×	×	✓	×
9	✓	×	×	✓	✓	×

From the analysis, it can be inferred that most of the techniques are focussed to improve budget and schedule. These techniques are mainly from the point of view of only the provider. Thus, a better technique is required which covers the end user and provider's view as well as satisfies most of the above parameters.

2.6 Conclusion

This Chapter compared resource provisioning and prediction techniques and analysed the comparison. The next Chapter aims to formulate the objectives to eliminate the research gaps realized from the above analysis.

Chapter 3 Problem Statement

From the analysis of the existing resource prediction techniques, it is realized that there is a need to improve the methodology. This Chapter discusses the research gaps and provides the objectives for the formulated problem.

3.1 Research Gaps

From the literature survey, it is evident that different techniques for resource prediction consider different parameters. There is a research gap due to the following factors:

- Both the end user and the provider are not satisfied simultaneously.
- None of the existing techniques is specific to Scientific applications.
- Instead of generating a methodology according to application domain, only a single technique is applied for all kinds of applications in the existing work.

3.2 Problem Formulation

Resource prediction has been an issue in Cloud based environments due to numerous factors. In some cases, the end users are not satisfied and in other the provider. There is a need to propose a methodology of efficient resource prediction for Scientific applications over the Cloud. This will enhance the QoS and satisfaction of users.

From the research gaps analysis, the problem is to formulate a Cloud resource prediction model for efficient working of Scientific applications. In such a framework, details of the application are provided and the system should generate resource requirements as output. The research issues should be eliminated by incorporating parameters for both end user and provider and by using a phase based methodology. In this research, healthcare applications are taken as case study, but the same approach can be applied to other kind of Scientific applications. It aims to maximize CPU utilization and minimize execution time, satisfying both the end user and provider.

3.3 Objectives

The main objectives of the thesis are:

- To explore the various techniques which can be implemented for resource prediction in a Cloud environment.
- To develop a data mining healthcare application to be used as a case study.

- To propose a Cloud resource prediction model for efficient working of Scientific applications.
- To implement the proposed approach using the developed case study in Cloud environment.

3.4 Methodology

The above stated objectives would be met by the following methodology:

- Understand the various applications in Cloud and the resource prediction techniques for them.
- Get acquainted with the various data mining algorithms and how to work upon them using tools.
- Develop the data mining healthcare application in java. Understand the use of weka API in java for this purpose.
- Setup the Cloud environment for monitoring of applications using CloudSim toolkit in java.
- Propose the model of resource prediction using data mining algorithm.
- Test the approach for a Scientific application and compare the parameters.

3.5 Conclusion

This Chapter discussed the research gaps and enumerated the objectives for the formulated problem. The next Chapter discusses the proposed Resource Prediction Model to eliminate the research gaps.

Chapter 4 Proposed Resource Prediction Model

This chapter elaborates the model generated for resource prediction in Scientific applications of healthcare domain over the Cloud. For the purpose of prediction, two healthcare applications are taken. One of the case studies is the PharmacoGen application, which outputs drug combinations or drug replacements for specific persons according to their genes. It is basically a data mining application developed as a case study for this work. The other one is a medical application, in which the user selects the symptoms and the application responds with possible disease, its cause and treatment. It is a query based application which uses the database to infer results.

4.1 Model Description

The resource prediction model can be generated for any kind of Scientific application. In this study, a model for data mining applications is proposed by considering two applications. A data mining application used in business is a kind of Scientific application. It will have both data mining and query operations, due to which both the applications are taken into consideration. For any other kind of application, the procedure remains the same. The different phases of the model are described in this section.

4.1.1 Architecture of the Proposed Model

The proposed model contains different phases like collection of data and analysis, training of data and prediction for new application. The architecture of the model with different phases and their components is as shown in Figure 4.1. The prediction model is mainly divided into five components, including Analyser, Resource Monitor, Optimal VM Evaluator, Model Trainer and Predictor. The first three components are parts of the

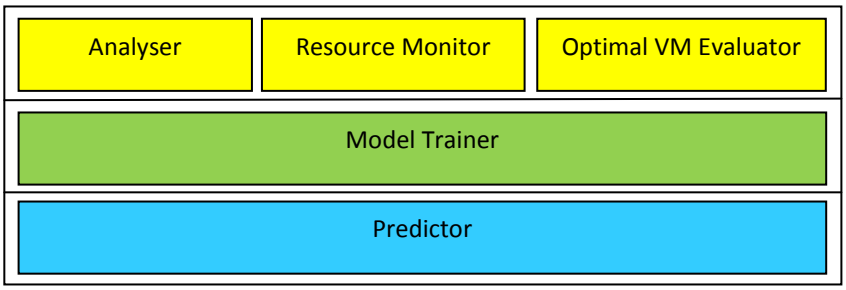


Figure 4.1 Architecture of Proposed Model

first phase which is data collection and analysis. The second layer depicts the training phase and next is the prediction phase. The steps of the model are discussed in detail in the next section.

4.1.2 Steps of Resource Prediction Model

Scientific applications can be of various kinds including compute intensive, memory intensive, data mining applications and many others. For applications of a new domain, training phase is necessary so that similar applications can be predicted in future. In this study, data mining application in healthcare is being focussed. The steps of the model for an application depicted in Figure 4.3 are as follows:

i. Analyze Application Domain: The first and foremost step is to check whether the application is under a known domain, for which the model has already been created or an unknown domain. If it is the former, then the parameters of application i.e. Cloudlet length and number of Cloudlets is given to already trained model, to predict optimal number of VMs for Cloudlet modules. In case of latter, data collection and training is performed to create the model for application.

ii. Analyze Application details: The use case diagram of the application provides the information of main modules of the application. These modules become the tasks or Cloudlets for the model. Hence, they are referred to as Cloudlet modules in this study. The length of Cloudlets can be calculated using MIPS (Million Instructions Per Second) of computer and execution time of Cloudlets on that computer. The Cloudlet length of applications in this study are calculated using the formula below:

$$\text{Cloudlet Length} = \text{MIPS} * \text{Execution time}$$

The MIPS can be known by executing a benchmark like CPUBENCH [52].

iii. Resource Monitor: The end user expects less execution time over the Cloud and the provider requires most utilization of CPU and memory so that lesser number of virtual machines is required which in consequence leads to less cost. The main purpose of this study is to extract a trade-off between the two. When an application is deployed over the Cloud, a resource monitor is necessary to examine its various parameters. In AWS services, Amazon Cloud Watch [47] serves the purpose of monitoring the various parameters. The Resource Monitor

keeps track of the CPU utilization and execution time of the Cloudlets of application for different VMs. In this study, CloudSim has been used for creating a Cloud environment as well as a Resource Monitor for monitoring the application in that environment.

iv. Optimal VM Evaluator: The data from Resource Monitor is evaluated by the Optimal VM Evaluator to discover optimal number of VMs required for the processing of different Cloudlets. Since the requirement is to maximize CPU utilization and minimize execution time, the ratio of both parameters is taken and the maximum ratio provides the optimal number. The pseudo code of algorithm for optimal VM Evaluator is as shown in Figure 4.2.

```
OptimalVMEvaluator()
main()
{
init();
r=0;
m→ optimal VMs
t→ CPU threshold
while(Cloudlet modules c_i arrive)
        {
  list= getResourceList();// from resource monitor
 n= list.size();
      for(i=0;i<n;i++)
                {
                   if(C_util>t)
                   {
                          ratio= C_util/E_time;
                          if(ratio>r)
                          {
                                  r=ratio;
                                  m=n;
                          }
                    }
                  else
                  continue;
                }
        }
print m as optimal VMs
}
```

Figure 4.2 Proposed OptimalVMEvaluator

In the pseudo code, m depicts the optimal number of VMs for Cloudlet modules of an application and t depicts the threshold value of CPU. For every Cloudlet module c_i, the evaluator gets the list of resources monitored by the Resource Monitor. Before calculating the ratio, it checks if the CPU utilization is less than t. If it is true then it continues the loop for next value. This ensures that utilization is greater than 20 percent as taken in this study. The evaluator iterates through the list to calculate the ratio of CPU Utilization and Execution time. The number of VMs with the maximum ratio is taken to be optimal. C_util and E_time depict the normalized CPU utilization and Execution time monitored by Resource Monitor.

Not only for training, but this code can be used to optimize the number of VMs in application while running. Once the optimal number of VMs is known, it can compare it with current number of resources. The code can be incorporated after the above algorithm. It would be:

```
if(m>v)
create (m-v) VMs;
else
destroy (m-v) VMs;
    where m is optimal number of VM and v is current number of resources
```

v. Train data: Data is trained using K-star classification algorithm [48] to generate a model for future predictions. Training phase is required when resources are to be predicted for a new domain application like it is done for data mining in this study.

vi. Perform predictions: Prediction of resources for any data mining application can be performed using the model generated by the previous phase in this study. The training phase is not required for predicting every new application of data mining, rather the model created using this method is saved and used for future predictions. The user only needs to provide Cloudlet length and number of Cloudlets in the application.

The steps of collecting the data, training the model and prediction in the proposed model are described as a flowchart in Figure 4.3. The data collection and analysis are depicted with the same colour in the flowchart, then the training phase and finally the prediction phase which outputs the optimal number of virtual machines are shown. The task of data

collection involves three steps: analyse application details, monitor the parameters and find out optimal number of virtual machine for the Scientific application.

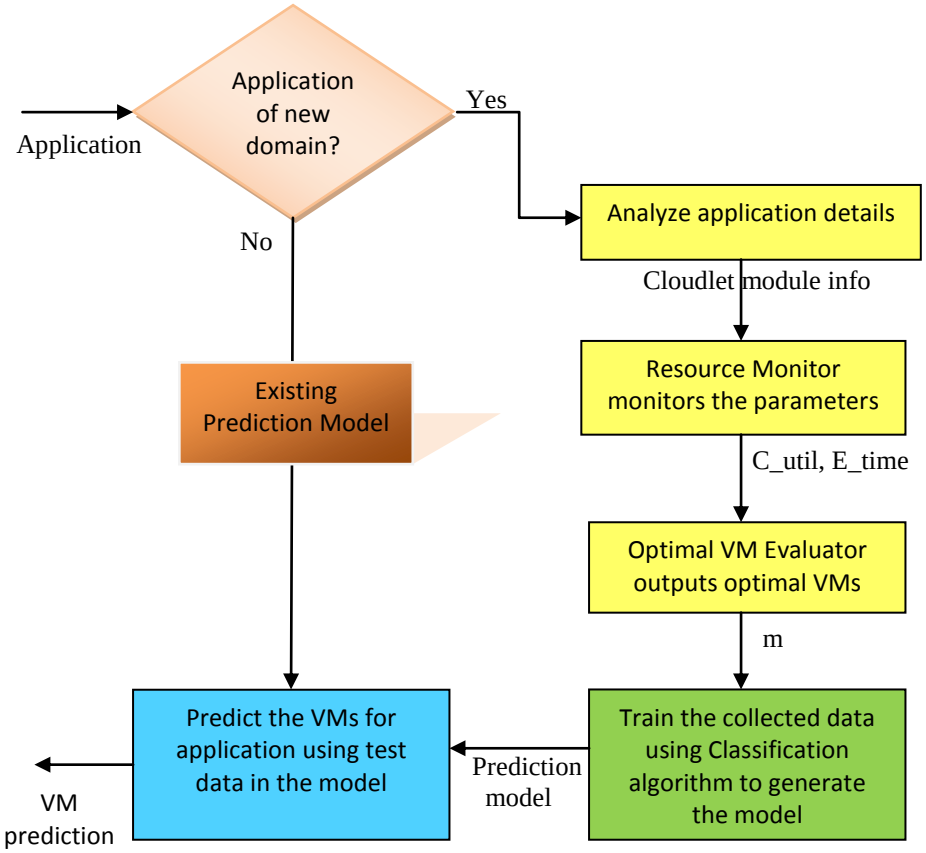


Figure 4.3 Steps of Proposed Prediction Model

4.1 **Conclusion**

The prediction model is mainly divided into three phases, including data collection and analysis, the training phase and the prediction phase. The phases mainly contain five components as discussed in its architecture. These components communicate to perform resource predictions for scientific applications of different domains. Hence, this phase based model is specifically designed for Scientific applications and is validated for healthcare data mining application as discussed in the next chapter.

Chapter 5 Case Studies on Healthcare

The advent of Internet services has led to massive explosion of healthcare data including images, videos and records. The developments of IT based innovative solutions have immensely impacted healthcare industry as well. Various techniques like text mining, data mining, network analysis and similarity based measures and Cloud computing have enhanced the development of healthcare solutions. The analysis, storage, extraction and retrieval of data have become easier than ever. This chapter discusses two case studies which are medical applications. One is an application involving simple query based results, while another application was developed for the purpose of prediction of resources. The tasks of these two case studies are taken as cloudlets of a data mining application. This chapter elaborates the development and use of the case studies.

5.1 PharmacoGen: Proposed Healthcare Case study

Data mining is used to extract new information or pattern from data. It is extensively used in bioinformatics domain for extracting unknown gene, protein and disease associations. Clustering of genes according to gene expression data has led to discovery of unknown genes and diseases. Examining differentially expressed genes in normal and diseased state results in discovering genetic profile or signature for the particular disease [32] [33] [34] [35]. PharmacoGen is also a data mining application to study the relationship of different drugs according to genes of a person. It is used as a case study for resource prediction over the Cloud. The application was proposed and developed specifically for patients of Type 2 Diabetes Melitus (T2DM) to serve as case study for this thesis. This section elaborates the concept of pharmacogenetics, its working, use case diagram and technical details.

5.1.1 Pharmacogenetics

Mining of gene-gene, gene-disease and gene-protein associations has been a prodigious source of knowledge extraction about genes and diseases [36] [37]. Similarly, gene and drug associations can be mined to extract relationships between drugs for specific individuals. This domain is known as pharmacogenetic as it involves use of correlation of drugs and genes to extract knowledge. This knowledge in consequence, leads to the idea of personalized medicine because drugs have been suggested based on personal genomic profile.

5.1.2 Design of PharmacoGen

The IT driven healthcare advancements have made genetic tests possible. Various existent services like 23andMe, Promethease, FamilyTreeDNA, Ancestory.com just need the DNA sample and they respond with a report of SNPs (Single Nucleotide Polymorphism) present in the individual. The variation of an organic molecule at same locus of two different individuals of same species is known as SNP [38]. If SNPs for an individual are known, then gene patterns for a particular disease can be inferred. Also, relationships between genotypes and phenotypes can be known.

In PharmacoGen, a druggist enters the SNPs/genes present in a T2DM patient and the application responds with possible drug combinations for that patient. These relations can also help in case of drug replacement or repositioning when allergies of the individual to specific chemicals are known. The use case diagram [44] for this application is shown in Figure 5.1.

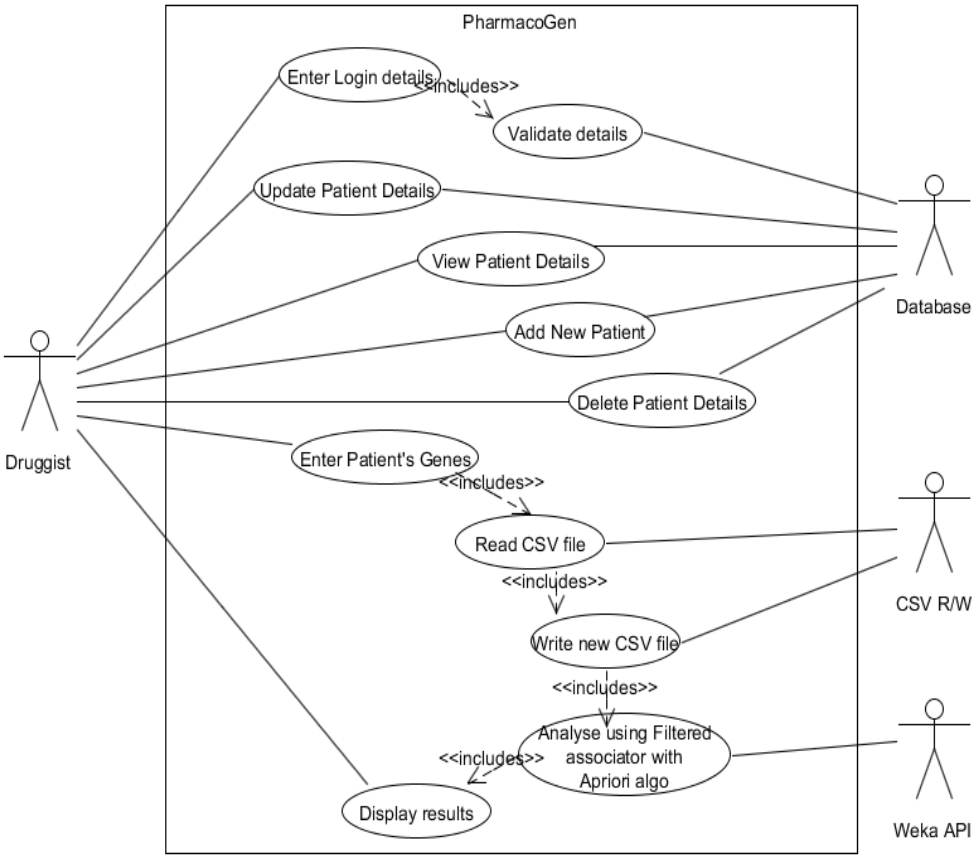


Figure 5.1 Use Case Diagram of PharmacoGen

The other functionalities of the application include storing, updating, adding and deleting the records of patients. The actors in Use Case Diagram involve a druggist, a database, Weka API and CSV reader/writer. The database is used for authentication and management of records. CSV reader/writer reads the CSV file of known genes of diabetes and selects the genes as required by user to write a new CSV file, whereas the Weka API performs the tasks of analysis of data of the new file. The analysis of Use Case Diagram helps to extract the different tasks in cloudlets, which are required for the proposed prediction model.

5.1.3 Processing of data

The application mines the data of gene-chemical associations to generate feasible drug results for diabetic patients. The Comparative Taxicogenomics Database (CTD) [39] provides curated data describing relationships between genes-proteins, chemicals-drugs, diseases, phenotypes and others. The database is maintained by the departments at North Carolina State University [56] and MDI Biological Laboratory [57]. Association analysis is done using filtered Apriori algorithm. The following steps were performed for analysis:

i. Data extraction: Database was downloaded from CTD for T2DM where data is in the form of records containing drugs/chemicals, associated genes and their interaction types. The interaction specifies that effect of a drug on a Gene either increases its expression or decreases it. The genes which do not affect the expression have been ignored.

ii. Data pre-processing: The genes/SNPs curated from CTD were confirmed to be associated with T2DM from various other sources [40] [41] including SNPedia [38]. 60 genes out of 3068 were confirmed to be relevant and associated with type 2 diabetes. In order to use apriori algorithm, the data format was changed. Increase of the expression of a gene means greater chances of occurrence of T2DM. The drugs which decreased the expression of gene, were given a value of True (T) as they decrease the risk of diabetes. Those which increase the expression i.e. having false value, were instead kept unknown as they are not required in the results. The extracted data is stored as a CSV file. Figure 5.2 depicts the formatted data.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	gene	bispheno	aflatoxin	benzo(a)p	butyralde	potassiur	quercetin	rotenone	thiram	arsenic	diethylnit	tetrachlo	cisplatin	irinoteca	pentanal	resveratr	rosiglitaz	troglita
2	CDKAL1	?	T	T	T	T	T	T	T	?	?	?	?	?	?	?	?	?
3	CDKN2B	?	?	?	?	?	?	?	?	T	T	T	?	?	?	?	?	?
4	CDKN2A	?	?	?	?	?	?	?	?	T	T	T	?	?	?	T	?	?
5	FTO	T	?	T	T	?	?	?	?	?	?	?	T	T	T	?	?	?
6	HHEX	T	?	T	?	?	T	?	?	?	?	T	?	?	?	?	?	?
7	IGFBP2	?	?	?	?	?	?	?	?	?	?	T	?	?	?	T	T	T
8	KCNJ11	T	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?
9	PPARG	T	?	T	?	?	?	T	?	T	T	T	T	?	?	T	T	T
10	TCF7L2	T	?	?	?	?	T	?	?	?	?	T	T	T	?	T	?	?
11	PDE4B	?	?	?	?	?	?	?	?	T	?	T	?	?	?	?	?	?
12	RBMS1	?	?	?	?	?	?	?	?	?	?	?	?	?	?	T	?	?
13	SREBF2	?	?	?	?	?	?	?	?	?	?	T	?	?	?	?	?	?
14	SLC12A3	T	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?
15	ELMO1	T	?	?	?	?	?	?	?	?	?	T	?	?	?	?	?	?
16	CCDC33	T	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?
17	ZFAND6	?	?	?	?	?	?	?	?	T	?	T	?	?	?	?	?	?
18	SREBF1	T	?	?	?	?	?	?	?	?	T	T	?	?	?	T	T	T
19	TRIB3	?	?	?	?	?	?	?	?	?	?	T	?	?	?	?	T	T
20	SLC30A8	?	?	T	?	?	?	?	?	?	?	?	?	?	?	?	?	?
21	INS1	T	?	?	?	?	?	?	?	?	?	?	?	?	?	?	T	?
22	NOS2	?	?	?	?	?	T	?	?	T	?	?	?	?	?	T	T	T
23	TGFB1	T	?	T	?	?	T	?	?	T	?	?	?	?	?	T	T	T
24	SOD1	?	T	?	?	T	T	T	?	T	T	?	T	?	?	?	?	?
25	TNF	?	?	?	?	?	T	?	?	T	?	?	?	?	?	T	T	T
26	IL6	?	?	?	?	?	T	?	?	?	?	T	?	?	?	T	T	T
27	RELA	?	?	?	?	?	?	?	?	?	?	?	?	?	?	T	T	?

Figure 5.2 Formatted dataset for analysis

iii. Association analysis: When the druggist selects the genes for his query, a new CSV file is created which contains only the selected genes in the same format as the original file as shown in Figure 5.3. Then the Weka API, which is a java API for Weka data mining tool, is used to perform association analysis on this newly created file [42]. A filtered associator was applied to pre-processed data, where Apriori algorithm [43] was used as the associator and a filter of removeType was used. The filter was applied to ignore those chemicals/drugs which increase the expression of all selected genes. The support is by default 0.5 for this application and confidence is 0.9.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	gene	bisphenol	aflatoxin	benzo(a)p	butyralde	potassiur	quercetin	rotenone	thiram	arsenic	diethylnit	tetrachlor	cisplatin	irinotecar	pentanal	resveratrc	rosiglitazc	troglitazo	Atrazine	Propylthic	Estrad
2	CDKN2B	?	?	?	?	?	?	?	?	T	T	T	?	?	?	?	?	?	?	?	?
3	FTO	T	?	T	T	?	?	?	?	?	?	?	T	T	T	?	?	?	?	?	?
4	HHEX	T	?	T	?	?	T	?	?	?	?	T	?	?	?	?	?	?	?	?	?
5	KCNJ11	T	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	T	T	?
6	PPARG	T	?	T	?	?	?	T	?	T	T	T	T	?	?	T	T	T	?	?	T
7																					
8																					

Figure 5.3 Generated CSV file by application

4.2.5 Technical Details

This web application was developed using Netbeans IDE 7.4 [45]. The database used for management of data is MySql. Weka API was included in the project using Weka jar file. Similarly, the CSV reader and writer were included using the opencsv-3.7 jar file [46]. Figure 5.4 represents the imported files.

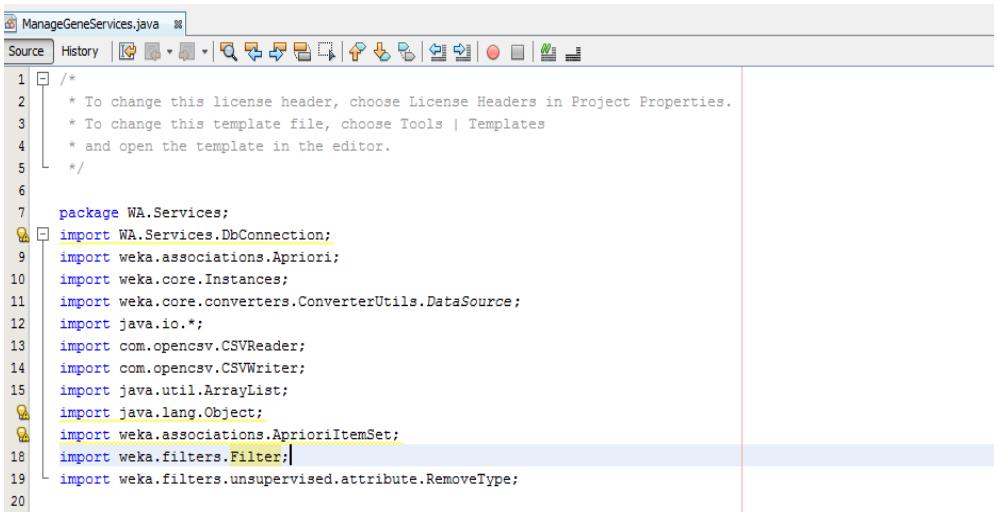


Figure 5.4 Files imported in application

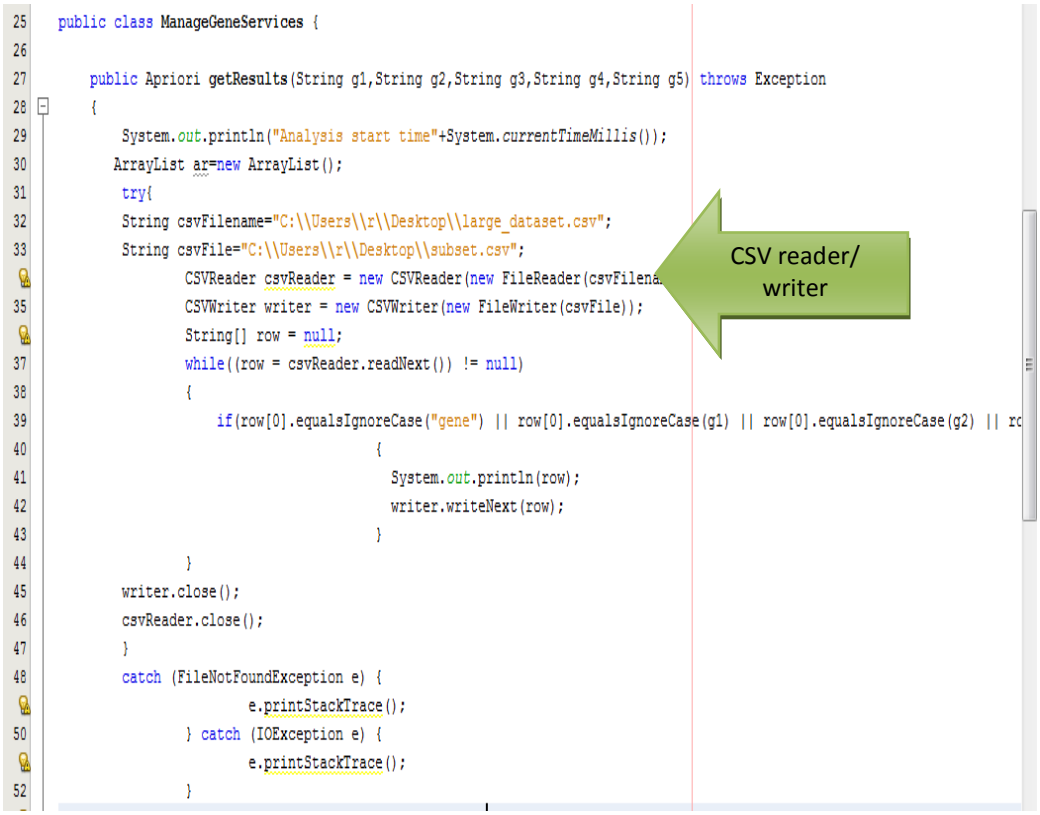


Figure 5.5 Functionality of CSV reader/ writer in application

The functionalities of CSV reader and writer are displayed in Figure 5.5.

It reads the file of associations of genes-drugs related to T2DM and writes a new file with genes selected by the druggist.

The Weka API analyzes this newly created file. The code for it is as shown in Figure 5.6.

```
53              String dataset="C:\\Users\\r\\Desktop\\subset.csv";
54          DataSource source=new DataSource(dataset);
55          Instances data=source.getDataSet();
56          String[] options = new String[2];
57          options[0] = "-T";                                  // "range"
58          options[1] = "numeric";                                 // first attribute
59          RemoveType remove = new RemoveType();                     // new instance of filter
60          remove.setOptions(options);                         // set options
61          remove.setInputFormat(data);                        // inform filter about dataset **AFTER** setting options
62          Instances newData = Filter.useFilter(data, remove);
63          Apriori model=new Apriori();
64          model.buildAssociations(newData);
65          System.out.println("Analysis end time"+System.currentTimeMillis());
66          return model;
67
68      }
69  }
```

Apriori algo with filter

Figure 5.6 Analysis using Weka API

4.2.2 Working of PharmacoGen

The screenshots of Login page (Figure 5.7) and Selecting genes (Figure 5.8) are presented in this section. For convenience, 5 genes are assumed for selection by the druggist. Figure 5.9 displays the result for the selection.

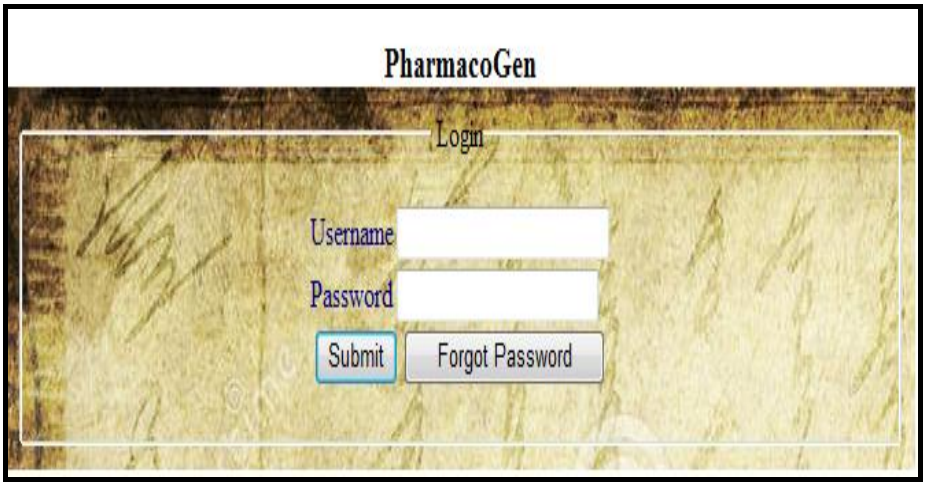


Figure 5.7 Login Screen of PharmacoGen

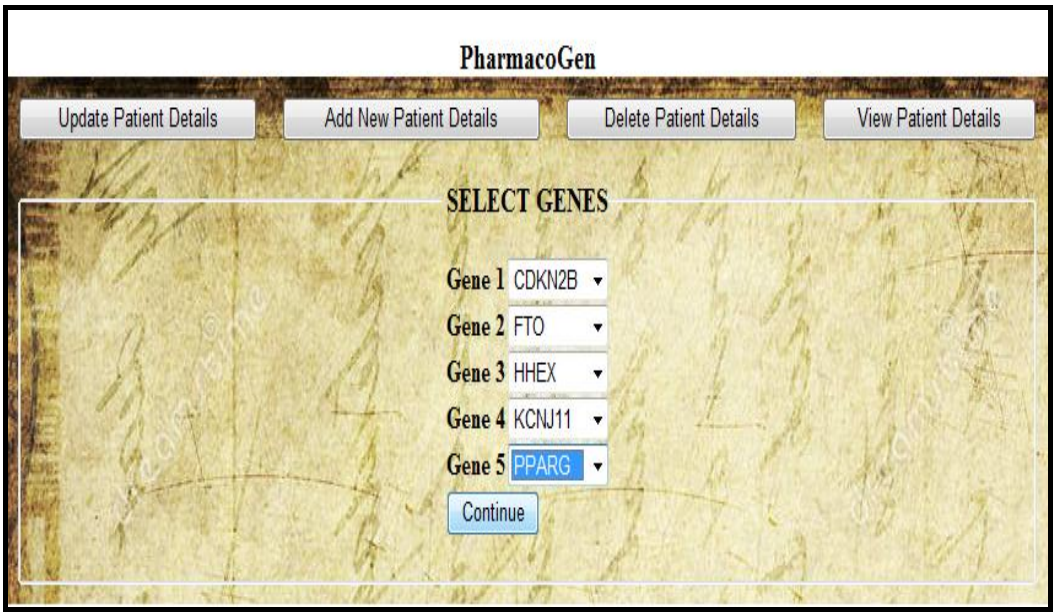


Figure 5.8 Select Genes Screen of PharmacoGen

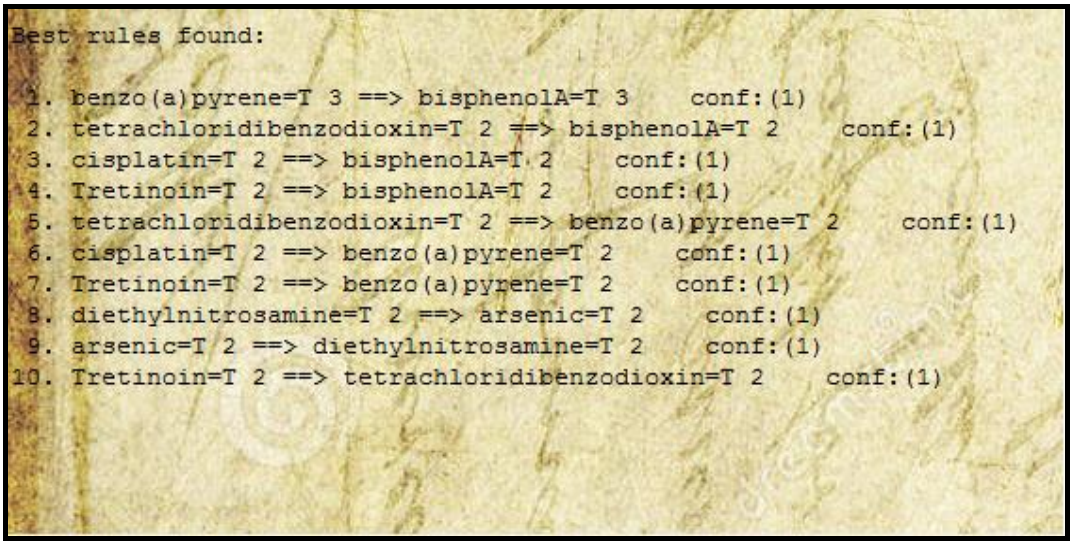


Figure 5.9 Display Results Screen of PharmacoGen

The results represent relation between various drugs. These relations can be used to replace one drug with another in case a patient has allergy to one of them or they can be used together to minimize the effect of genetic expression on the disease.

5.2 Medical Expert System: A Case Study

The other case study is a Medical Expert System. In this application, user enters the symptoms and the application responds with possible disease, its causes and treatment solution. The disease, its causes and treatment solutions are stored in a database in MS

Access for study purposes that can be later stored in a Cloud Storage. The use case diagram of this application is depicted in Figure 5.10. The major tasks of this application are adding the symptom, deleting the symptom from the list or displaying the results. The screenshots of the application is shown in Figure 5.11 and Figure 5.12.

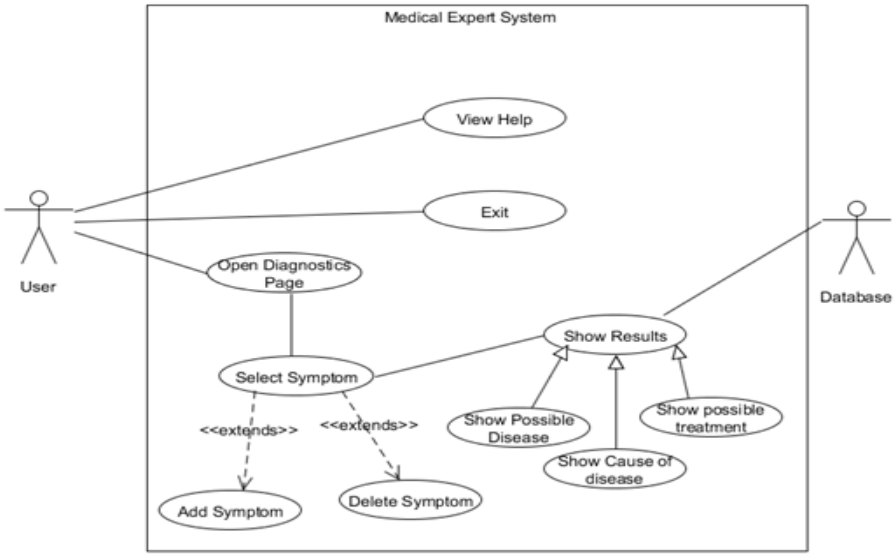
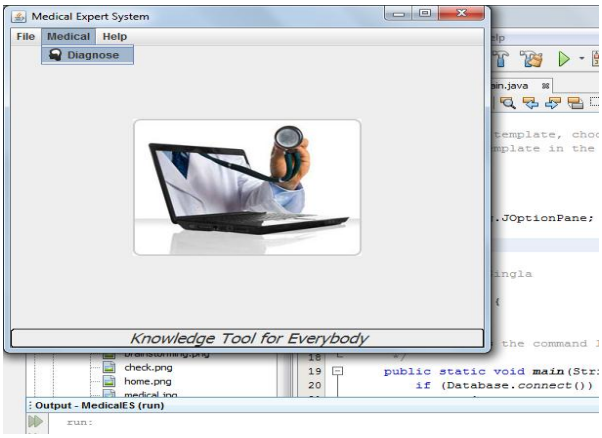


Figure 5.10 Use case diagram of Medical Expert System



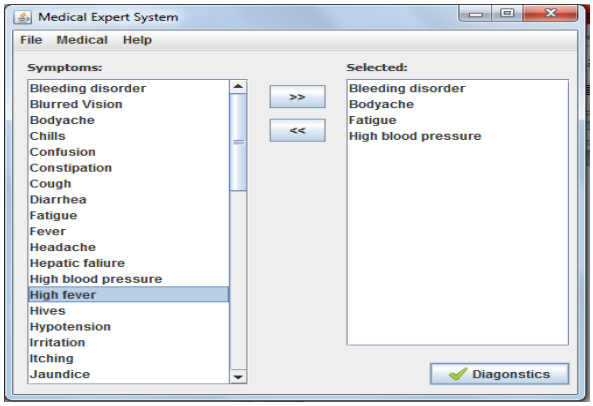


Figure 5.11 Screenshot of query of Medical Expert System

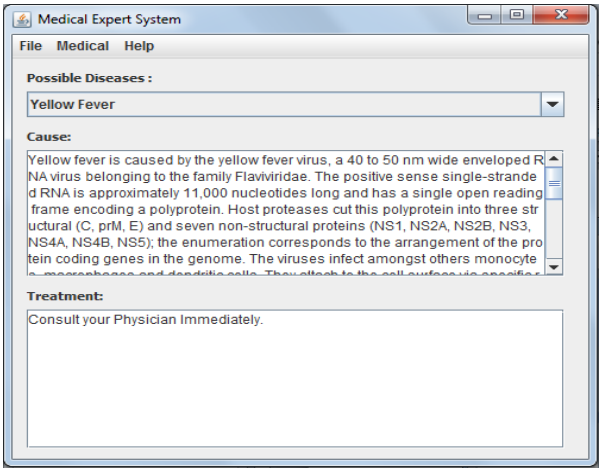


Figure 5.12 Screenshot of results of Medical Expert System

A data mining application will have tasks of mining as well as of queries. This case study satisfies the query part of a data mining application used in business. The combination of the above discussed two applications is used as case study for the proposed Cloud Resource Prediction model. The application tasks depict cloudlets of a data mining application. The use case diagrams of these two case studies helps to extract information of main modules and hence the tasks for the application are known. As can be seen from the above discussed Use Case Diagrams, there are mainly 6 tasks of Pharmacogen and 3 tasks of Medical Expert System.

5.3 Conclusion

A data mining application will have tasks of mining as well as of queries, which are fulfilled by the two case studies of healthcare domain discussed above. This Chapter elaborated the development and working of the proposed application Pharmacogen. The next Chapter provides the experimental results performed on the two applications.

Chapter 6 Experimental Results

This chapter discusses about the tools to setup Cloud environment, implementation of every phase of the proposed model and the results obtained from it.

6.1 Tools for Setting Cloud Environment

Tools and softwares used for setting up the proposed model in Cloud environment are discussed below.

6.1.1 CloudSim Toolkit

Cloud computing allows applications and infrastructure to be provided as a service on a pay-as-you use basis. The provisioning, configuration, scheduling and deployment are dynamic and complex in nature. Due to such conditions, it is difficult to evaluate the performance of scheduling policies, resource performance models and workload models under varying configurations. CloudSim is a simulation toolkit which overcomes this challenge by providing simulation environment for Cloud computing systems. The kit simulates components of Cloud including data centers, VMs, hosts and it also supports application of configured resource provisioning and scheduling policies [49] [50]. The layered architecture of CloudSim is depicted in Figure 6.1.

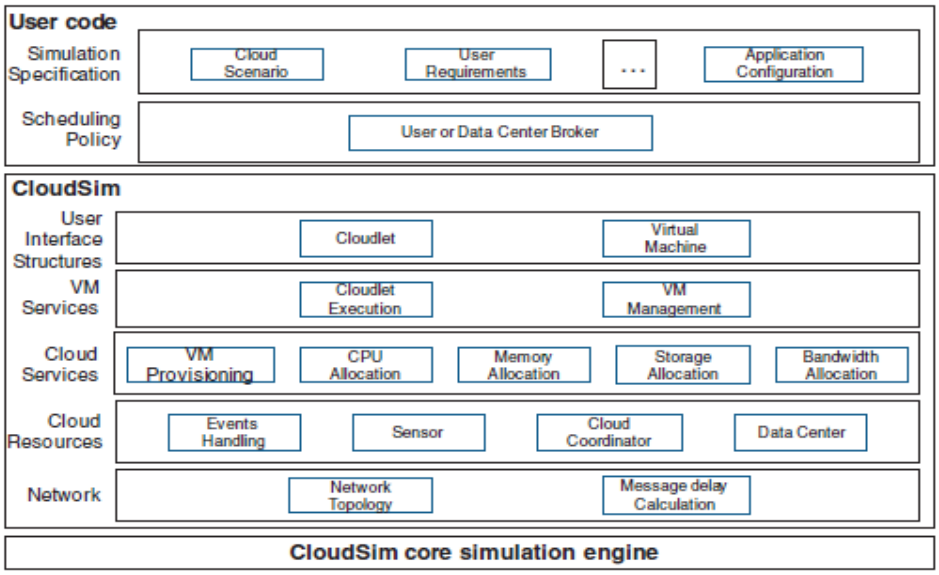


Figure 6.1 Layered architecture of CloudSim [49]

The various components of CloudSim which are essential for the Cloud environment and for the proposed system include the following [49]:

(i) Datacenter: This class manages various hosts with same or different configurations of hardware. Datacenter instantiates an application provisioning component which implements policies to allocate memory, bandwidth and storage device to VMs and hosts.

(ii) Host: This class depicts a physical resource and contains information regarding type of processing cores and amount of memory and storage. It also contains information of allocation policy for provisioning bandwidth and memory to VMs and for sharing the processing amongst VMs.

(iii) VM: This class models a Virtual Machine. It is managed by a Cloud host and has the characteristics of processor, memory, storage size and a VM provisioning policy extended from the CloudletScheduler.

(iv) DatacenterBroker : This class is a broker which mediates QoS driven negotiations between Cloud service provider and SaaS. It queries the CIS (Cloud Information Service) to discover a suitable Cloud provider and undertakes negotiations for allocation of resources in order to meet QoS requirements.

(v) Cloudlet: This class depicts Cloud-based application services. It has various parameters like data transfer overhead and instruction length. It also supports modeling of performance metrics for different applications.

(vi) CloudletScheduler: This is an abstract class which is extended by implementation of policies for sharing processing power amongst Cloudlets in VM. It offers different provisioning policies like time-shared and space-shared.

(vii) VMScheduler: This is also an abstract class that models the policies for allocating processor cores to VMs. It is implemented by a host component. The class can be overridden to perform application-specific sharing policies.

(viii) VmAllocationPolicy: This is an abstract class representing a provisioning policy utilized for allocating VMs to hosts. It supports selection of available host to meet various requirements for deploying a VM.

The working of CloudSim including all these components is depicted in Figure 6.2. CIS is a registry which contains information about the data centers available on the Cloud. Every newly created data center is first registered with CIS. Every data center has hosts, which further have VMs. The data center broker retrieves the data center characteristics like processing elements, memory and bandwidth from CIS. Also the tasks or Cloudlets for application are submitted to the broker, which then assigns them to VMs in the data center's hosts. After completion of tasks, VMs are destroyed by the broker.

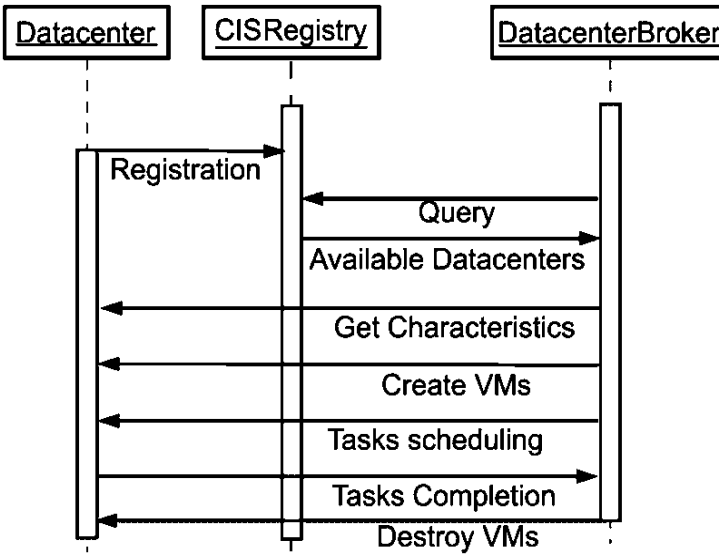


Figure 6.2 Working of CloudSim [49]

6.1.2 Weka Tool

Weka is a software containing various data mining and machine learning algorithms written in Java. It was developed at the University of Waikato, New Zealand. It offers data mining tasks by providing tools for data pre-processing, classification, clustering, regression, association rules mining and visualization. Weka is platform-independent and open source. Weka API is used for performing data mining operations in other applications [51].

6.1.3 Netbeans

Netbeans [45] is an application platform framework providing development environment for java applications. It is platform independent and supports various features like development of user interface, storage management, Netbeans library and wizard framework. Netbeans IDE 7.4 was used for loading CloudSim toolkit.

6.1.4 CPUBENCH

CPUBENCH [52] is a benchmark for calculating the speed of CPU. It performs 12 varied tests for this purpose including MFLOPS, MIPS, PI, Dhrystone and Whetstone. It has been used in this study to calculate MIPS.

6.2 Implementation of Proposed Model

For the implementation, firstly the data is collected and trained and then the testing phase is performed. This section describes the data collection and training phase for resource prediction.

6.2.1 Data Collection

i. Cloudlet Length Calculation

The tasks of the two case studies are the Cloudlets. The number of Cloudlets is known from the use case diagrams (Figure 5.1 and Figure 5.10) of the case studies, which is 7 and 4 respectively. For allocating VMs, the CloudSim must be provided with the length of Cloudlet, file size, output size and number of processing elements. For calculating the length of the Cloudlet, the following parameters are needed:

1. MIPS: Million instructions per second is a measure of performance of CPU. It can be calculated by using various benchmarks. CPUBENCH is used to calculate the MIPS of the computer, in which tasks of applications are run. Figure 6.3 depicts the use of CPUBENCH to calculate MIPS. The MIPS for Windows 7 64-bit PC with i3 Intel core and 3GB memory is found to be approximately 800 MIPS.

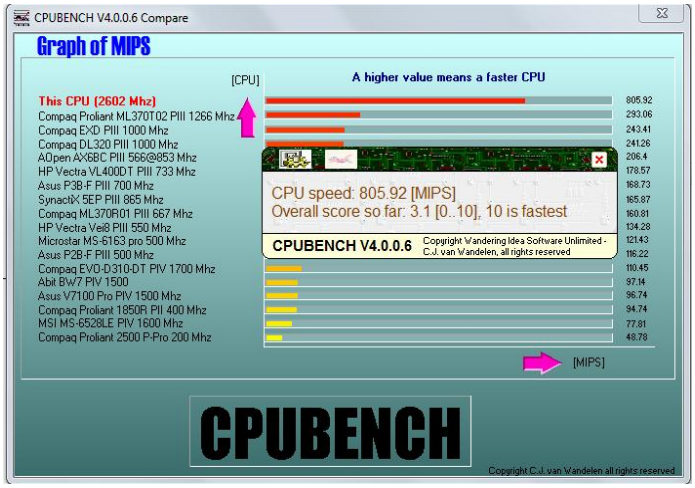


Figure 6.3 Execution of CPUBENCH

2. Time of execution: Time of execution of Cloudlets can be known by configuring the code to include current time at the start and end of execution of Cloudlet as shown in Figure 6.4. By subtracting the two, we get the execution time as in Figure 6.5. The time of execution of different tasks were calculated from this formula. The task of analysis takes 441 ms.

```
Apriori model=new Apriori();
model.buildAssociations(newData);
System.out.println("Analysis end time"+System.currentTimeMillis());
return model;
```

Figure 6.4 Code for getting the current time

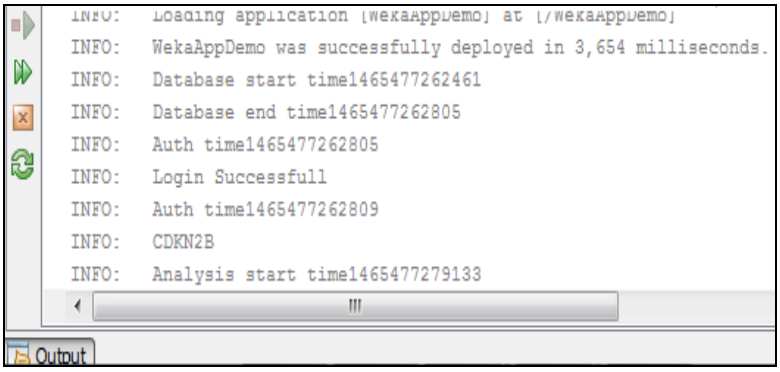


Figure 6.5 Displaying start and end time for calculation

The Cloudlet length is represented in MI i.e. Million Instructions. The formula for length of Cloudlet is as follows:

Length of Cloudlet = MIPS * Time of Execution in seconds

So, for the analysis task the length of Cloudlet = 800 * 0.441 = 352.8

The length of other Cloudlets for PharmacoGen is calculated as shown in Figure 6. 6. The length for Medical Expert System is shown in Figure 6.7.

	Task	Execution time	Cloudlet Length
2	Validate details	348	278.4
3	Update details	350	280
4	Add new patient	349	279.2
5	Delete details	349	279.2
6	View details	348	278.4
7	Analysis	441	352.8

Figure 6.6 Length of Cloudlets for PharmacoGen

	Task	Execution time	Cloudlet Length
2	Add symptom	480	384
3	Delete symptom	479	383.2
4	Show results	590	472

Figure 6.7 Length of Cloudlets for Medical Expert System

ii. Resource Monitor Implementation

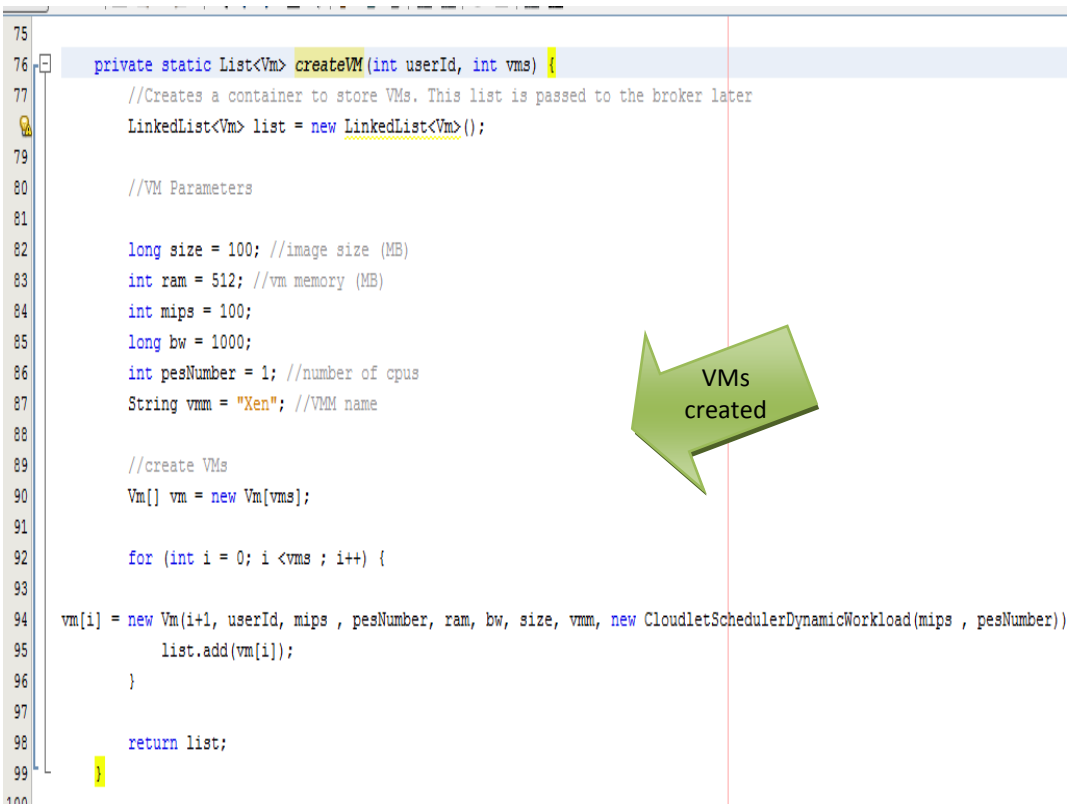
The Resource Monitor in the proposed method monitors the VMs and stores the details of execution time and CPU utilization of virtual machines used in healthcare application case studies. For monitoring such parameters, the code is as described below in Figure 6.8:

```
48  public class TestClass {
49
50      private static List<Cloudlet> cloudletList;
51      private static List<Vm> vmlist;
52
53      public static void main(String[] args) {
54          int num_user = 1;
55          Calendar calendar = Calendar.getInstance();
56          boolean trace_flag = false;
57          CloudSim.init(num_user, calendar, trace_flag);
58
59          Datacenter dc1 = createDatacenter("MyDatacenter", 1);
60
61          NewBroker broker = (NewBroker)createBroker("MyBroker");
62          broker.submitVmList(createVM(broker.getId(), 5));
63          broker.submitCloudletList(createCloudlet(broker.getId(), 10, 1));
64
65          CloudSim.startSimulation();
66
67          List<Cloudlet> newList = broker.getCloudletReceivedList();
68          List<Vm> List2 = broker.getVmList();
69          CloudSim.stopSimulation();
70
71          printCloudletList(newList, List2);
72
73          dc1.printDebts();
74      }
```


Figure 6.8 Code for creating Resource Monitor in CloudSim

CloudSim does not have any built-in feature to monitor the CPU utilization and execution time of each VM. So, the code was configured to monitor the required parameters. TestClass.java was created with the main method to create entities, start the simulation and print the results as shown above. Further, this class contains two methods for creating the VMs and Cloudlets.

The configurations of VMs are as shown in Figure 6.9. In this study, RAM of VM is taken as 512 MB, MIPS as 100 and a single processing element. The scheduler used is DynamicWorkload, assuming the requests to application to be dynamic and not following any fixed pattern. The values for Cloudlet length were given according to PharmacoGen as shown in Figure 6.10. The value of file size is 4KB, that is 4000 MB and output size is 810 MB. Utilization model for Cloudlets is a stochastic one rather than full. The Cloudlets for Medical Expert System are shown in Figure 6.11.




```
75
76      private static List<Vm> createVM(int userId, int vms) {
77          //Creates a container to store VMs. This list is passed to the broker later
            LinkedList<Vm> list = new LinkedList<Vm>();
79
80          //VM Parameters
81
82          long size = 100; //image size (MB)
83          int ram = 512; //vm memory (MB)
84          int mips = 100;
85          long bw = 1000;
86          int pesNumber = 1; //number of cpus
87          String vmm = "Xen"; //VMM name
88
89          //create VMs
90          Vm[] vm = new Vm[vms];
91
92          for (int i = 0; i <vms ; i++) {
93
94  vm[i] = new Vm(i+1, userId, mips , pesNumber, ram, bw, size, vmm, new CloudletSchedulerDynamicWorkload(mips , pesNumber))
95              list.add(vm[i]);
96          }
97
98          return list;
99      }
```

Figure 6.9 Code for VMs Configuration

```
100
101     private static List<Cloudlet> createCloudlet(int userId, int cloudlets, int idShift) {
102         // Creates a container to store Cloudlets
            LinkedList<Cloudlet> list = new LinkedList<Cloudlet>();
104
105         //cloudlet parameters
106         long fileSize = 4000;
107         long outputSize = 810;
108         int pesNumber = 1;
109         UtilizationModel utilizationModel = new UtilizationModelStochastic();
110
111         Cloudlet[] cloudlet = new Cloudlet[cloudlets];
112
113   cloudlet[0] = new Cloudlet(1, 278, pesNumber, fileSize, outputSize, utilizationModel, utiliza nModel, utilizationModel)
114   cloudlet[1] = new Cloudlet(2, 280, pesNumber, fileSize, outputSize, utilizationModel, utilizationModel, utilizationModel)
115   cloudlet[2] = new Cloudlet(3, 279, pesNumber, fileSize, outputSize, utilizationModel, utilizationModel, utilizationModel)
116   cloudlet[3] = new Cloudlet(4, 279, pesNumber, fileSize, outputSize, utilizationModel, utilizationModel, utilizationModel)
117   cloudlet[4] = new Cloudlet(5, 278, pesNumber, fileSize, outputSize, utilizationModel, utilizationModel, utilizationModel)
118   cloudlet[5] = new Cloudlet(6, 352, pesNumber, fileSize, outputSize, utilizationModel, utilizationModel, utilizationModel)
119
120
121         for (int i = 0; i < cloudlets; i++) {
122             // setting the owner of these Cloudlets
123             cloudlet[i].setUserId(userId);
124             list.add(cloudlet[i]);
125          }
126          return list;
127     }
```

Cloudlets created

Figure 6.10 Code for Cloudlets configuration according to PharmacoGen

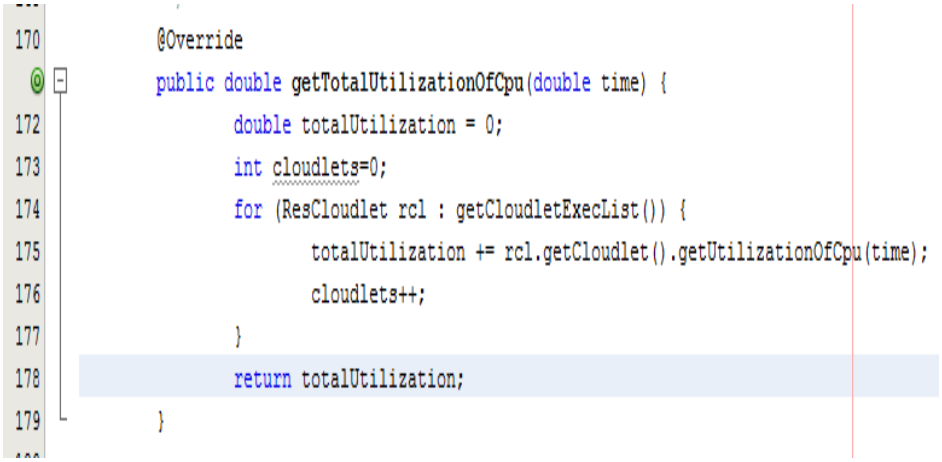
```
100
101     private static List<Cloudlet> createCloudlet(int userId, int cloudlets, int idShift) {
102         // Creates a container to store Cloudlets
            LinkedList<Cloudlet> list = new LinkedList<Cloudlet>();
104
105         //cloudlet parameters
106         long fileSize = 4000;
107         long outputSize = 810;
108         int pesNumber = 1;
109         UtilizationModel utilizationModel = new UtilizationModelStochastic();
110
111         Cloudlet[] cloudlet = new Cloudlet[cloudlets];
112
113       cloudlet[0] = new Cloudlet(1, 384, pesNumber, fileSize, outputSize, utilizationModel, utilizationModel, utilization
114       cloudlet[1] = new Cloudlet(2, 383, pesNumber, fileSize, outputSize, utilizationModel, utilizationModel, utilization
115       cloudlet[2] = new Cloudlet(3, 472, pesNumber, fileSize, outputSize, utilizationModel, utilizationModel, utilization
116
117         for (int i = 0; i < cloudlets; i++) {
118             // setting the owner of these Cloudlets
119             cloudlet[i].setUserId(userId);
120             list.add(cloudlet[i]);
121          }
122          return list;
123     }
```

Figure 6.11 Code for Cloudlets configuration according to Medical Expert System

The NewBroker.java class extends the broker class. After submitting the Cloudlets to the VM, it gets the current total value of CPU utilization for VM at that time and prints it. The code for it is as shown in Figure 6.12. The method getTotalUtilizationOfCpu of class VM returns the total utilization for every VM at a particular time. This method was overridden in the CloudletSchedulerDynamicWorkload class to display the values for all the Cloudlets. The code for it is depicted in Figure 6.13.

```
58              cloudletsSubmitted++;
59              vmIndex = (vmIndex + 1) % getVmsCreatedList().size();
60              getCloudletSubmittedList().add(cloudlet);
61              //Cloudlet was submitted...checking VM Status
62
63              if (vm!=null){
64                  vm.updateVmProcessing(CloudSim.clock(), null);
65                  double currentCPU = vm.getTotalUtilizationOfCpu(CloudSim.clock());
66  if(currentCPU == 0)
67      System.out.println("Cloudlet: " + cloudlet.getCloudletId() + " - VM: " + vm.getId()
68                          + " - Current CPU Usage Percent: 0" );
69  else
70                  System.out.println("Cloudlet: " + cloudlet.getCloudletId() + " - VM: " + vm.getId()
71                          + " - Current CPU Usage Percent: " + currentCPU*100);
72              }
73              this.pause(delay);
74              CloudSim.runClockTick();
75          }
76      }
77  }
```

Figure 6.12 Code of New Broker to get CPU utilization



```
170         @Override
171         public double getTotalUtilizationOfCpu(double time) {
172                 double totalUtilization = 0;
173                 int cloudlets=0;
174                 for (ResCloudlet rcl : getCloudletExecList()) {
175                         totalUtilization += rcl.getCloudlet().getUtilizationOfCpu(time);
176                         cloudlets++;
177                 }
178                 return totalUtilization;
179         }
```

Figure 6.13 Code for CPU utilization evaluation

The results from this simulation provides data for training in the next phase. The values of CPU utilization and execution time are stored by the resource monitor. Figure 6.14 depicts the values of CPU utilization for a single virtual machine in PharmacoGen application. The virtual machine has different utilizations for different Cloudlets, so overall utilization is the average of all the values. Execution time for this VM is depicted in Figure 6.15. Figure 6.16 shows the execution time of 6 Cloudlets of PharmacoGen on 3 VMs. The average execution time for a VM is calculated using the average of the three times. Hence, CPU utilization and execution time for PharmacoGen and Medical Expert System are monitored by the resource monitor.

```
Cloudlet: 2 - VM: 1 - Current CPU Usage Percent: 0
20.0: MyBroker: Sending cloudlet 3 to VM #1
Cloudlet: 3 - VM: 1 - Current CPU Usage Percent: 13.755232886683544
25.88866801945725: MyBroker: Sending cloudlet 4 to VM #1
Cloudlet: 4 - VM: 1 - Current CPU Usage Percent: 39.007591744244394
25.88866801945725: MyBroker: Sending cloudlet 5 to VM #1
Cloudlet: 5 - VM: 1 - Current CPU Usage Percent: 19.503795872122197
30.0: MyBroker: Sending cloudlet 6 to VM #1
Cloudlet: 6 - VM: 1 - Current CPU Usage Percent: 75.75324004493855
35.88866801945725: MyBroker: Cloudlet 1 received
35.88866801945725: MyBroker: Cloudlet 2 received
```

Figure 6.14 CPU utilization of single VM in PharmacoGen

```
========== OUTPUT ==========
VM ID       1
Cloudlet ID    STATUS    Data center ID       Time    Start Time    Finish Time
    1        SUCCESS        2            15.89            10            25.89
    2        SUCCESS        2            13.03            20            33.03
    3        SUCCESS        2            5.43             30           35.43
    5        SUCCESS        2            6.5             35.89           42.39
    4        SUCCESS        2            6.6             35.89           42.49
    6        SUCCESS        2            6           40           46
```

Figure 6.15 Execution time for single VM in PharmacoGen

Output - CloudSim (run)

```
VM ID       1
Cloudlet ID    STATUS    Data center ID       Time    Start Time    Finish Time
    1        SUCCESS        2            14.18            10            24.18
    4        SUCCESS        2            16              34.08            50.08
VM ID       2
Cloudlet ID    STATUS    Data center ID       Time    Start Time    Finish Time
    2        SUCCESS        2            10              20            30
    5        SUCCESS        2            15.8              34.18            49.98
VM ID       3
Cloudlet ID    STATUS    Data center ID       Time    Start Time    Finish Time
    3        SUCCESS        2            14.06            30            44.06
    6        SUCCESS        2            18.07            34.18            52.25
```

Figure 6.16 Execution time for 3 VMs with 6 Cloudlets in PharmacoGen

ii. Optimal VM Evaluator

The monitored data from resource monitor is used by the Optimal VM evaluator to find out the best possible number of VMs for specific application. The algorithm discussed in chapter 4 is used by the evaluator to calculate ratio value for PharmacoGen and Medical Expert System as shown in Figure 6.17 and Figure 6.18 respectively. The ratio for normalized data is maximum in case of 4 VMs in PharmacoGen and 2 in case of Medical

Expert System. Similarly, simulations were performed to collect data for 20 instances and the data collected is of the form as shown in Figure 6.19.

VM	CPU Utilization	Execution Time	Ratio
1	24.6	36	1.17
2	18.6	39.9	0.65
3	8.81	30.7	0.15
4	11.6	16.4	2.59

Figure 6.17 Ratio comparison for PharmacoGen

VM	CPU Utilization	Execution Time	Ratio
1	8.3	37.6	0.38
2	14.7	20.2	2.85
3	5.3	13.5	1

Figure 6.18 Ratio comparison for Medical Expert System

No of Cloudlet	Cloudlet length	VMs
6	291	4
3	413	2
10	1200	6
15	850	11
8	632	5
5	800	4
4	150	2
6	524	5
12	900	7
7	563	5
11	478	6
9	455	5

Figure 6.19 Generated Dataset

6.2.2 Training Phase

The values from the Optimal VM Evaluator are utilized by the Classification algorithm in Weka [54] to train a model for prediction. K-star algorithm is used to train the data for classification as shown in Figure 6.20.

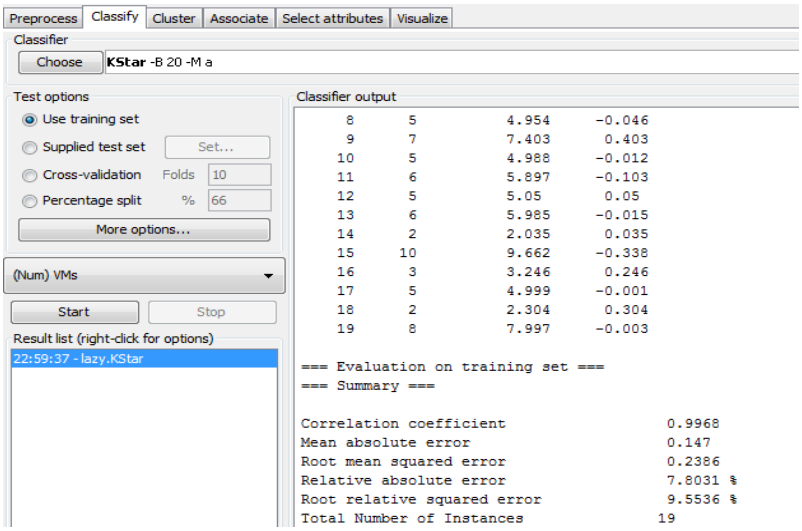


Figure 6.20 Classification using Weka

K star is a lazy classifier which is instance-based. The class of training instances similar to test instance become the base for the class prediction. This is determined by a similarity function which is an entropy based distance function. It is quite similar to KNN algorithm [48]. The K star function can be defined as [53]:

$$K^*(y_i, x) = -ln\ P^*(y_i,x)$$

where P* is probability of all transformational paths from x instance to y instance.

6.2.3 Testing/Prediction Phase

A model for classification was generated in training phase using the K-star classification algorithm. This model is saved and loaded in future to carry out the testing or prediction phase. The results obtained in this study are described in the next section.

6.3 Simulation Results

The test set is as shown in Figure 6.21. The unknown values of VM depict that they need to be predicted. The classification model generated using K-star algorithm outputs the optimal number of VMs for a given application as per the trained model. The results for the given test set are shown in Figure 6.22.

No of Clou	Cloudlet l	VMs
20	1200	?
7	450	?
3	260	?
13	150	?

Figure 6.21 Test set values for prediction

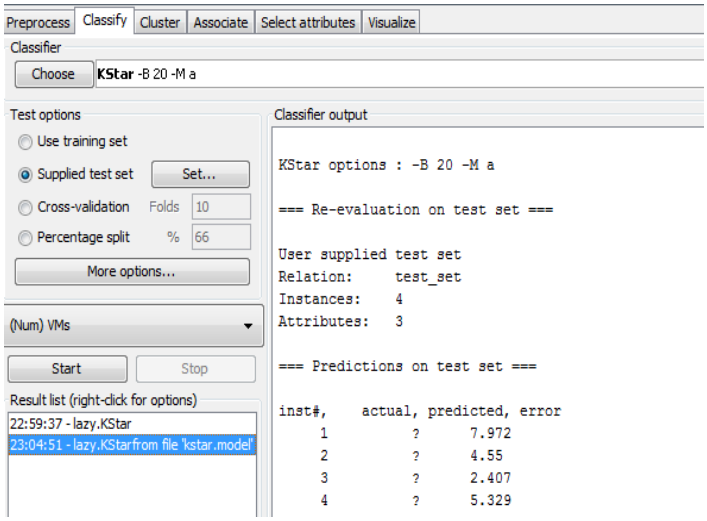


Figure 6.22 Results of prediction in Weka

The predicted values for test set are shown in Figure 6.23. The values are taken using the greatest integer function.

No of Cloudl	Cloudlet length	VMs
20	1200	7
7	450	4
3	260	2
13	150	5

Figure 6.23 Predicted Optimal number of VMs

The values for CPU utilization and execution time are compared for 20 Cloudlets of length 1200 with different number of VMs from the above figure. This was done to evaluate the performance of the framework. When more than 10 VMs are requested in this application, broker is unable to create more VMs as they fail by required MIPS, so the results show 10 VMs only. It can be noticed that using 7 VMs for this application

would have a considerable CPU utilization and less execution time. The utilization is also greater than the threshold. It has the maximum ratio as shown in Figure 6.24. The comparison of CPU utilization for different VMs is shown in Figure 6.25 and execution time is shown in Figure 6.26.

VM	CPU Utilization	Execution Time	Ratio
1	55.3	120.4	1
2	39.7	96.3	0.91
3	46.2	95.1	1.03
4	25.6	63.2	1.2
5	19.2	49.1	0.96
6	21	69.1	0.7
7	27.4	54.6	1.22
8	14.3	36.4	1.05
9	6.2	22.1	0.97
10	7.4	21.7	1.2

Figure 6.24 Ratio comparison for application with 20 Cloudlets

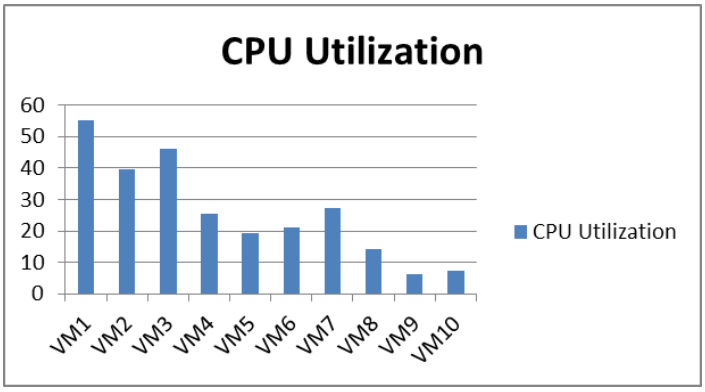


Figure 6.25 CPU utilization for an application

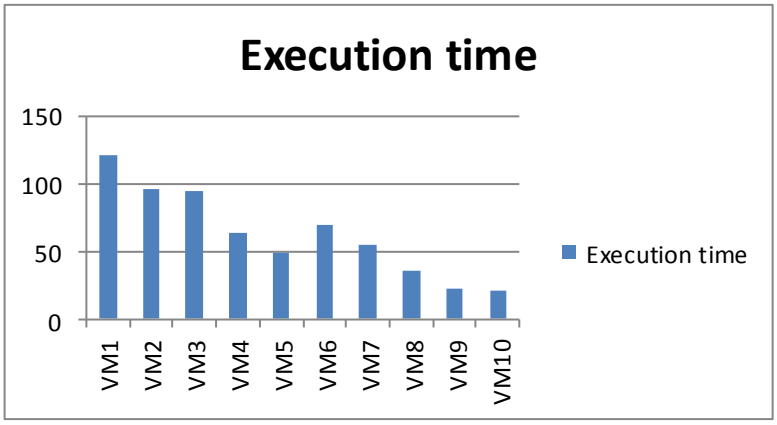


Figure 6.26 Execution time for an application

The performance of K-star algorithm can be evaluated using mean absolute error and root mean squared error. The value of mean absolute error is 0.147 and root mean squared error is observed to be 0.23 in this study. The values depict an efficient performance.

6.4 Conclusion

This Chapter elaborates the implementation details of each phase of the proposed model. In the results, it can be seen that an optimal number of VMs are first recognized and then a model is trained from this data. Prediction is done using lazy data mining algorithm and optimal VMs are predicted for another Scientific application. This model helps to attain the goals of a trade-off between CPU utilization and execution time of resources for applications.

Chapter 7 Conclusions and Future Scope

7.1 Conclusion

The aim of this work was to propose a Cloud Resource Prediction Model for Scientific applications in healthcare domain. The model satisfies the requirements of end users as well as the provider. Due to optimal use of resources, the end users' requests are processed in time and the cost of resources is justified for the provider. The results of the prediction were confirmed to be true by applying the model to medical case studies. It predicts optimal number of virtual machines with a trade-off between CPU utilization and execution time. Also, the model can be used for different domains, not just data mining in healthcare. Hence, it is an adaptive solution.

This work also presents a novel technique in the form of an application to find optimal drug combinations or replacements using the genetic data for a specific diabetic patient. The work involves knowledge of genetic data and data mining algorithms.

7.2 Thesis Contributions

The contributions of thesis work are presented as below:

- It provides an efficient resource prediction model in Cloud environment for healthcare Scientific applications by considering CPU utilization and execution time.
- It presents a new technique to find associations between drugs specific to individuals according to their genes.
- A case study application, named PharmacoGen has been proposed and designed that is used for implementing the proposed prediction model and validating the results.
- It predicts the optimal number of virtual machines which can balance the trade off between CPU Utilization and execution time. The results of predicted VMs were validated for another application by performing the provisioning using CloudSim.

7.3 Future Work

The model can be enhanced using the following future work:

- This work focussed on data mining healthcare application, but it can be implemented with others as well.
- Currently, CPU utilization and execution time has been monitored. But, other parameters like cost can also be assimilated.
- It is intended to perform real tests in future, instead of simulations.
- A fully functional autonomic system can be developed based on this model.

References

[1] P. Mell and T. Grance, "The NIST definition of Cloud computing", *NIST special publication 800- 145*, pp. 20-23, 2011

[2] W. Voorsluys, J. Broberg, & R. Buyya, "Introduction to Cloud computing", *Cloud computing: Principles and paradigms*, pp. 1-44, 2011.

[3] R. Buyya, C. Vecchiola and S. Selvi, Mastering Cloud computing: foundations and applications programming, Newnes, 2013.

[4] Q. Zhang, L. Cheng and R. Boutaba, "Cloud computing: state-of-the-art and research challenges", *Journal of internet services and applications*, vol. 1, no. 1, pp. 7-18, 2010.

[5] A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin and I. Stoica, "Above the Clouds: A Berkeley view of Cloud computing", *Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS*, vol. 28, no. 13, 2009.

[6] C. Vecchiola, S. Pandey and R. Buyya, "High-performance Cloud computing: A view of Scientific applications" in *10th IEEE International Symposium on Pervasive Systems, Algorithms, and Networks (ISPAN)*, pp. 4-16, 2009.

[7] McKinsey and Co., "Clearing the Air on Cloud Computing", Technical Report, 2009.

[8] B.E. Reddy, TV Suresh Kumar and G. Ramu, "An efficient Cloud framework for health care monitoring system," in IEEE *International Symposium on Cloud and Services Computing (ISCOS),*2012.

[9] M. Jianping, C. Peng and Q. Chen, "Health Information Exchange for Home-Based Chronic Disease Self-Management--A Hybrid Cloud Approach," in *5th IEEE International Conference Digital Home (ICDH),*2014.

[10] M. Fahim, I. Fatima, S. Lee and Y. K. Lee, "Daily life activity tracking application for smart homes using android smartphone." in *Proceedings of 14th IEEE International Conference on Advanced Communication Technology ,ICACT*, 2012.

[11] M. Sayan, K. Dolui, and SK Datta, "Patient health management system using e-health monitoring architecture," in *IEEE International Advance Computing Conference (IACC)*, 2014.

[12] V. Koufi, F. Malamateniou, G. Vassilacopoulos and A. Prentza, "An android-enabled mobile framework for ubiquitous access to Cloud emergency medical services," in IEEE *Second Symposium on Network Cloud Computing and Applications (NCCA),* 2012.

[13] M. Kutlu and Gagan Agrawal, "Cluster-based SNP calling on large-scale genome sequencing data," in *14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid),* 2014.

[14] L. Xu, L. and J. Li, "Building Efficient Resource Management Systems in the Cloud: Opportunities and Challenges", *International Journal of Grid and Distributed Computing*, vol. 9, no. 3, pp. 157-172, 2016.

[15] K. Dinesh, G. Poornima, and K. Kiruthika, "Efficient resources allocation for different jobs in Cloud." *International Journal of Computer Applications,* vol. 56, no. 10, 2012.

[16] S. Selvarani and G.S. Sadhasivam, "Improved cost-based algorithm for task scheduling in Cloud computing", in *IEEE international conference on Computational intelligence and computing research (ICCIC),* pp. 1-5, 2010.

[17] J. Yu and R. Buyya, "Scheduling Scientific workflow applications with deadline and budget constraints using genetic algorithms", *Scientific Programming*, vol. 14, no. 3-4, pp. 217-230, 2006.

[18] J. Yu and R. Buyya, "A budget constrained scheduling of workflow applications on utility grids using genetic algorithms", in *IEEE Workshop on Workflows in Support of Large-Scale Science, WORKS'06,* pp. 1-10, 2006.

[19] J. Yu, R. Buyya and C.K. Tham, "Cost-based scheduling of Scientific workflow applications on utility grids", in *First IEEE International Conference on e-Science and Grid Computing*, pp. 8, 2005.

[20] S. Lili, Y. Shoubao, G. Liangmin, and W. Bin, "A Markov chain based resource prediction in computational grid", in *Fourth IEEE International Conference on Frontier of Computer Science and Technology, FCST'09*, pp. 119-124, 2009.

[21] J. Jiang, J Lu, G. Zhang and G. Long, "Optimal Cloud resource auto-scaling for web applications", in *13th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid),* pp. 58-65, 2013.

[22] J.O. Melendez, A. Biswas, S. Majumdar, B. Nandy, M. H. Zaman, P. Srivastava and N. Goel, "A framework for automatic resource provisioning for private

Clouds", in *13th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, pp. 610-617, 2013.

[23] G. Reig, J. Alonso and J. Guitart, "Prediction of job resource requirements for deadline schedulers to manage high-level SLAs on the Cloud", in *9th IEEE International Symposium on Network Computing and Applications (NCA)*, pp. 162-167, 2010.

[24] J. Huang, C. Li, and J. Yu, "Resource prediction based on double exponential smoothing in Cloud computing", in *2nd IEEE International Conference on Consumer Electronics, Communications and Networks (CECNet)*, pp. 2056-2060, 2012.

[25] A. Biswas, S. Majumdar, B. Nandy, B and A. El-Haraki, "Automatic Resource Provisioning: A Machine Learning Based Proactive Approach", in *6th IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, pp. 168-173, 2014.

[26] E. Caron, F. Desprez and A. Muresan, "Forecasting for grid and Cloud computing on-demand resources based on pattern matching", in *Second IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, pp. 456-463, 2010.

[27] Y. Jiang, C. S. Perng, T. Li, and R. Chang, "Asap: A self-adaptive prediction system for instant Cloud resource demand provisioning", in *11th IEEE International Conference on Data Mining (ICDM)*, pp. 1104-1109, 2011.

[28] Y. Shi, X. Jiang and K. Ye, "An energy-efficient scheme for Cloud resource provisioning based on CloudSim", in *IEEE International Conference on Cluster Computing (CLUSTER)*, pp. 595-599, 2011.

[29] M. Verma, G. R. Gangadharan, N. C. Narendra, R Vadlamani, V. Inamdar, L. Ramachandran, R. N. Calheiros and Rajkumar Buyya. "Dynamic resource demand prediction and allocation in multi-tenant service Clouds", *Concurrency and Computation: Practice and Experience*, 2016.

[30] R. N. Calheiros and R. Buyya, "Virtual machine provisioning based on analytical performance and QoS in Cloud computing environments", in *IEEE International Conference on Parallel Processing (ICPP)*, pp. 295-304, 2011.

[31] S. Islam, 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. 155-162, 2012.

[32] J. Freudenberg and P. Propping, "A similarity-based method for genome-wide prediction of disease-relevant human genes", Bioinformatics, vol. 18, no. 2, pp. S110-S115, 2002.

[33] Y. C. Chen, W. C. Ke and H. W. Chiu, "Risk classification of cancer survival using ANN with gene expression data from multiple laboratories," *Computers in biology and medicine*, vol. 48, pp. 1-7, 2014.

[34] K. Jung, M. Grade, J. Gaedcke, P. Jo, L. Opitz, and H. Becker, "A new sensitivity-preferred strategy to build prediction rules for therapy response of cancer patients using gene expression data", *Computer methods and programs in biomedicine*, vol. 100, no. 2, pp. 132-139, 2010.

[35] K. Ahmed, A. A. E. Abdullah-Al-Emran, T. Jesmin, R.F. Mukti, M. Rahman and F. Ahmed, "Early detection of lung cancer risk using data mining", *Asian Pacific Journal of Cancer Prevention*, vol. 14, no. 1, pp. 595-598, 2013.

[36] J. Sheng, F. Li and S.T.C. Wong, "Optimal drug prediction from personal genomics profiles," *IEEE Journal of Biomedical and Health Informatics, vol.* 19, no. 4, pp. 1264-127, 2015.

[37] M.B. Carson, and H. Lu, "Network-based prediction and knowledge mining of disease genes," *BMC medical genomics*, vol. 8, Suppl. 2, pp. S9, 2015.

[38] "SNPedia", [Online]. Available: http://www.snpedia.com/index.php/SNPedia. [Accessed 11 Dec 2015].

[39] "Comparative Taxicogenomic Database", [Online]. Available: http://ctdbase.org/ . [Accessed 21 Jan 2016].

[40] P. R. Burton, D. G. Clayton, L. R. Cardon, N. Craddock, P. Deloukas, A. Duncanson and J. A. Todd, "Genome-wide association study of 14,000 cases of seven common diseases and 3,000 shared controls", *Nature*, vol. 447, no. 7145, pp. 661-678, 2007.

[41] E. Zeggini, M. N. Weedon, C. M. Lindgren, T. M. Frayling, K. S. Elliott, H. Lango and J. C. Barrett, "Multiple type 2 diabetes susceptibility genes following genome-wide association scan in UK samples", *Science (New York, NY)*, vol. 316, no. 5829, pp. 1336, 2007.

[42] "Weka in Java", [Online]. Available: https://weka.wikispaces.com/Use+WEKA+in+your+Java+code. [Accessed 3 March, 2016]

[43] J. Hipp, U. Güntzer and G. Nakhaeizadeh, “Algorithms for association rule mining—a general survey and comparison”, *ACM sigkdd explorations newsletter*, vol 2, no. 1, pp. 58-64, 2000.

[44] G. Booch, J. Rambaugh and I. Jacobson, The Unified Modeling Language User Guide, Addison Wesley Longman Publishing Co., Inc., 2005.

[45] “Netbeans”, [Online]. Available: https://netbeans.org/ . [Accessed 19 Feb, 2016]

[46] “CSV reader and writer”, [Online]. Available: http://opencsv.sourceforge.net/ . [Accessed 7 March, 2016]

[47] “Amazon Web Services”, [Online]. Available: https://aws.amazon.com/Cloudwatch/ . [Accessed 29 Jan, 2016]

[48] “K-star algorithm”, [Online]. Available: http://wiki.pentaho.com/display/DATAMINING/KStar. [Accessed 25 April, 2016]

[49] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De 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.

[50] R. Buyya, R. Ranjan and R. N. Calheiros, “Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities”, In *International Conference on High Performance Computing & Simulation, HPCS'09*, pp. 1-11, IEEE 2009.

[51] S. Singhal and M. Jena, “A study on WEKA tool for data preprocessing, classification and clustering”, in *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, vol. *2*, no. 6, pp. 250-253, 2013.

[52] “CPUBENCH Benchmark”, [Online]. Available: http://www.softpedia.com/get/System/Benchmarks/CPUBENCH.shtml . [Accessed 15 April, 2016]

[53] S. Vijayarani and M. Muthulakshmi, "Comparative analysis of bayes and lazy classification algorithms", in *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 2, no. 8 , pp. 3118-3124, 2013.

[54] I. H. Witten, E. Frank, L. Trigg, M. Hall, G. Holmes and S. J. Cunningham, "Weka: Practical machine learning tools and techniques with Java implementations", pp. 81-17, 1999.

[55] A. Bala and I. Chana, "Autonomic fault tolerant scheduling approach for scientific workflows in Cloud computing" *Concurrent Engineering*, vol. *23*, no. 1, pp. 27-39, 2015.

[56] "North Carolina State University", [Online]. Available: https://www.ncsu.edu/ . [Accessed 13 May, 2016]

[57] "MDI Biological Laboratory", [Online]. Available: https://mdibl.org/ . [Accessed 13 May, 2016]

List of Publications and Video Link

Publications

Accepted

R. Toor and I. Chana, *"*A proposed pharmacogenetic solution for IT based Healthcare*",* in *Springer Second International Conference on ICT for Sustainable Development, (ICT4SD 2016)*, Goa, 1-2 July, 2016.

Communicated

R. Toor and I. Chana, "Application of I.T. in Healthcare: A Systematic Review", in *ACM, SigBio Record Newsletter*, 2016.

Video Link

https://youtu.be/K6hIeJLaGdw

Appendix A: Installation of CloudSim 3.0.3 toolkit

CloudSim 3.0.3 toolkit is available at http://www.Cloudbus.org/CloudSim/ . In order to create a Cloud environment, the following steps should be followed:

1. Install Netbeans.
2. After downloading the Clousim toolkit, extract it in a folder named CloudSim.
3. In Netbeans, click on the file tab and open the extracted file by browsing its location.
4. It already contains the libraries required, example code and necessary classes as shown in Figure A.1. The code can now be configured according to use.

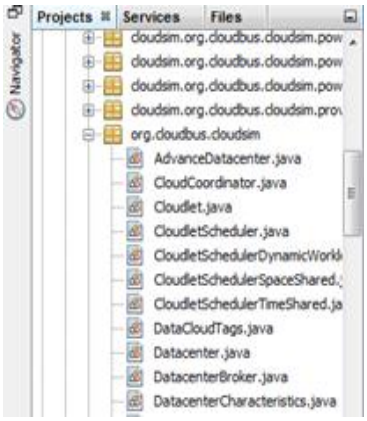


Figure A.1

rashmeet_801431019_thesis.pdf

ORIGINALITY REPORT

12%	7%	6%	6%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Thapar University, Patiala Student Paper	3%
2	www.sersc.org Internet Source	1%
3	dspace.thapar.edu:8080 Internet Source	1%
4	Jing Jiang, , Jie Lu, Guangquan Zhang, and Guodong Long. "Optimal Cloud Resource Auto-Scaling for Web Applications", 2013 13th IEEE/ACM International Symposium on Cluster Cloud and Grid Computing, 2013. Publication	1%
5	gdeepak.com Internet Source	1%
6	jarrett.cs.mu.oz.au Internet Source	<1%
7	dhsprogram.com Internet Source	<1%

Ramezani, Fahimeh, Jie Lu, and Farookh