Efficient Resource Prediction and Scheduling Approach for Scientific Applications in Cloud Environment

A Thesis

submitted in partial fulfillment of the requirements for the award of degree of
DOCTOR of PHILOSOPHY

by

Gurleen Kaur
(901503030)

under the guidance of

Dr. Anju Bala
Associate Professor
Computer Science and Engineering Department
Thapar Institute of Engineering and Technology, Patiala -147004

Computer Science and Engineering Department
Thapar Institute of Engineering and Technology,
Patiala - 147004, INDIA

January 2020

Candidate Declaration

I hereby certify that the work, which is being presented in the thesis, entitled **Efficient Resource Prediction and Scheduling Approach for Scientific Applications in Cloud Environment**, in partial fulfillment of the requirements for the award of the degree of **Doctor of Philosophy** and submitted to the institution is an authentic record of my own work carried out during the period **January 2016** to **August 2019** under the supervision of **Dr. Anju Bala**. I have also cited the reference about the text(s)/figure(s)/table(s) from where they have been taken.

The matter presented in this thesis has not been submitted elsewhere for the award of any other degree or diploma from any institution.

Gurleen Kaur

(**Gurleen Kaur**)
Regn. No. 901503030

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

Anju Bala 30/8/19

(**Dr.Anju Bala**)
Associate Professor
Department of Computer Science & Engineering
Thapar Institute of Engineering and Technology, Patiala, 147004
Punjab, INDIA

i

.....dedicated to all young generation researchers

Abstract

Scientific Computing uses the state-of-the-art of high performance computing capabilities to solve the complex problems in various scientific domains such as weather forecasting, earthquake, sub-atomic particle behavior, turbulent flows and manufacturing processes etc. As the demand of resource requirements for solving the scientific problems is dynamic, so there is a need for a flexible platform which can handle the above-mentioned challenges in scientific applications concerning data storage and computation.

Cloud computing provides a dynamic environment for deploying scientific applications by offering services such as infrastructure, platform and software. Various other features such as on-demand service, resource pooling, pay-as-per-use, elasticity, *etc* has attracted the scientists to deploy scientific applications on cloud. For effective utilization of virtualized resources in cloud, there is a need for efficient prediction based scheduling of tasks inorder to maximize performance and minimize execution time. Therefore, it is essential to first predict the resource requirements for scientific applications and then schedule them appropriately to meet the Quality of Service (QoS) requirements of the scientific users by taking SLA violations into consideration.

To achieve the set objectives, an extensive literature survey of existing scientific applications has been done. Furthermore, state-of-the-art prediction techniques and scheduling approaches have been surveyed. From the literature, it can be inferred that prediction based scheduling is a challenging issue which needs to be handled carefully. To address these problems, firstly a Regressive Ensemble Approach for Predicting (REAP) resource usage has been proposed and based on the predicted set of resources a scheduling approach (RPS) has been devised.

The proposed approaches have been validated using two scientific applications "Cybershake" and "Floodplain". The former has been executed on simulator (WorkflowSim

and CloudSim) and the latter has been used on actual cloud platform (Microsoft Azure). The proposed REAP framework has been divided into two modules. In the first one, scientific application has been deployed and resource usage dataset has been generated for further experimentation. The second one comprises feature selection and resource usage prediction using machine learning models. A meta-heuristic technique, Genetic Algorithm (GA), has been employed to select the relevant features which have significant role in predicting the desired combination of genes or traits. A total 19 input features and two target features were found; nonetheless, after deploying GA, the count of input features was reduced to eight. Additionally, the selected machine learning models were combined using proposed ensemble algorithm which utilizes the capabilities of these models for predicting the CPU and memory usage of the scientific application. Different evaluation measures such as Root Mean Square Error (RMSE), Accuracy (Acc) and Execution Time for prediction (ET) were evaluated and compared. Also, the predicted set of resources, namely CPU and memory usage, were stored in the database for scheduling the application tasks.

In RPS approach, firstly the predicted CPU & memory usages have been taken as input and the application execution requirements have been compared against the availability of resources on VMs. After mapping the application to suitable VMs, the tasks of mapped application were scheduled using a combination of swarm intelligence and Technique for Order of Preference by Similarity to Ideal Solution(TOPSIS). In this research work, TOPSIS, a multi-criteria decision making algorithm, has been used to compute the fitness value of the tasks in swarm algorithm to optimize the scheduling of resources. The values calculated by TOPSIS have been assigned as local best (pbest) values of tasks. Subsequently, all the tasks have been compared, the one with highest fitness value has been considered as global best (gbest) and it has been assigned to VM for execution. This process has been repeated until all the tasks were scheduled. The main objectives of the proposed approach have been the reduction of execution time, cost and SLA violation rate. The proposed approach has been validated by comparing the results with

existing scheduling heuristics. The main objectives of the proposed approach have been the reduction of execution time, cost and SLA violation rate.

The results clearly state that the proposed ensemble approach (REAP) is better than the existing ones as it improves the overall accuracy, reduces the prediction time and yields the lowest error rate. Moreover, when compared with the existing scheduling heuristics of DataAware, FCFS, MaxMin, MinMin and MCT, the proposed approach schedules the tasks in the least execution time and maintains the cost and SLA violation rate at minimal levels.

Acknowledgements

Almighty has bestowed me with the wisdom and perseverance to accomplish this milestone. I am forever grateful to GOD for HIS everlasting blessings.

At the outset, I would like to offer my deep gratitude to my supervisor Dr. Anju Bala for her invaluable advice and encouragement at every step of my PhD program. Without her unfailing support and belief in me, this thesis would not have been possible. Her constant encouragement, immense knowledge and scientific acumen lightened up the way in my darkest times. I am indebted for her insightful discussions and enlightening suggestions that assisted me to shape the direction of this research. The time and energy spent on careful and patient proof readings of this manuscript helped me to present this thesis with the desired quality. And for this, I am truly grateful.

I am highly thankful to Dr. Maninder Singh, Head, Computer Science and Engineering Department for his constant motivation and encouragement. I would like to express my sincere thanks to Dr. Inderveer Chana, Associate Head, Computer Science and Engineering Department for generously sharing her time and knowledge for carrying out this research in an efficient manner.

I also wish to thank Dr. Rajesh Kumar and Dr. Amanpreet Kaur for being my Doctoral committee members and advising me at initial stages of my research studies. I would also like to thanks Dr. Rafat Siddique, Dean of Research and Sponsored Projects for academic support. My sincere thanks to Dr. Parkash Gopalan, Director, Thapar Institute of Engineering and Technology, for providing all the facilities and resources that have been instrumental in the creation robust research environment in the University.

I wish to further extend my thanks to Dr. Prashant Singh Rana, Assistant Professor, Computer Science and Engineering Department for teaching me various new technologies that helped me to achieve my objective. I would like to thank administrative and technical

staff members of Computer Science and Engineering Department who have been kind enough to advise and help in their respective roles. I would also like to thank Ms. Shveta Verma, my research mate for all the great times that we have shared and making this research experience enjoyable and memorable.

This thesis would never have been completed without the motivation and support of my family. My hardworking family deserve a special mention here. My mother, Ms. Jatinder Kaur infused moral, ethical and religious values in me. She made me smile and realize their faith in me during the rough road of this journey. She has passed onto me a wonderful humanitarian lineage and a good foundation to face the challenges of life. I gratefully acknowledge the patience and love of my brother, Gursimran Singh who supported the family in my absence and let me pursue my dream. A special mention to my father, (Late) S. Darshan Singh who was always a great support and inspired me to dream big.

My warmest thanks to my dear husband S. Gurpreet Singh whose unconditional support during all these years is highly appreciated. He instilled confidence in me and inspired me in all dimensions of life for successful completion for this journey. I will always be indebted to you for your everlasting love and commitment that carries me through always. I look forward for a life full of happiness and togetherness.

Last but not the least, I would extend my deep sense of gratitude to my father-in-law, S. Gurdeep Singh, mother-in-law, Ms. Jaginder Kaur and brother-in-law, Gursharan Singh who extended their cooperation and encouragement to me. I am indebted to them and this thesis would not have been complete without their everlasting support.

Gurleen Kaur

Table of Contents

List of Figures

List of Tables

List of Abbreviations

SKA	Square Kilometer Array
QoS	Quality of Service
SLA	Service Level Agreement
BLAST	Basic Local Alignment Search Tool
LINC	Laboratory Instrument Computer
IBM	International Business Machine
ENIAC	Electronic Numerical Integrator And Computer
FORTRAN	Formula Translation
ARPANET	Advanced Research Projects Agency Network
CT	Computerised Tomography
GIMS	Great Internet Mersenne prime Search
DNA	Deoxyribonucleic Acid
BIRN	Biomedical Informatics Research Network
CPU	Central Processing Unit
PC	Personal Computer
GA	Genetic Algorithm
LSST	Large Synoptic Survey Telescope
LHC	Large Hadron Collider
NCCS	NASA Center for Climate Simulation
LIGO	Laser Interferometer Gravitational Wave Observatory
SDDS	Sloan Digital Sky Survey
ESCAT	Electron Scattering Code
NIST	National Institute of Standards
NIA	Nature Inspired Algorithm
IaaS	Infrastructure as a Service

PaaS	Platform as a Service
SaaS	Software as a Service
IT	Information Tool
ARPA	Advanced Research Projects Agency
REAP	Regressive Ensemble Approach for Predicting resource usage
RPS	Resource Prediction based Scheduling
NP	Nondeterministic Polynomial time
PSO	Particle Swarm Optimization
TOPSIS	Technique for Order of Preference by Similarity to Ideal Solution
GROMACS	GROningen MAchine for Chemical Simulations
LUX	Large Underground Xenon
FEM	Finite Element Method
RDBMS	Relational Database Management System
GCRM	Global Cloud Resolving Model
RMSE	Root Mean Square Error
I/O	Input/Output
HEP	High Energy Physics
QoI	Quality of Information
SGT	Strain Green Tensor
VMs	Virtual Machines
PMs	Physical Machines
LWFS	Light Weight File System
LA	Learning Automata
SW	Single Window
MW	Multiple Window
HPC	High Performance Computing
NCSA	National Center for Super computing
EC2	Elastic Cloud Computing

SOAP	Simple Object Access Protocol
DES	Double Exponential Smoothing
LR	Linear Regression
NN	Neural Network
SVM	Support Vector Machine
ANN	Artificial Neural Network
ACO	Ant Colony Optimization
DT	Decision Tree
RF	Random Forest
ABC	Artificial Bee Colony
DAG	Directed Acyclic Graph
CSPs	Cloud Service Providers
BRR	Bayesian Ridge Regression
BRNN	Bayesian Regularized Neural Network
ELM	Extreme Learning Machine
ET	Execution Time
TS	Task Scheduling
EC_t	Total Execution Cost
SLAV	Service Level Agreement Violation
OPSA	Optimized Prediction based Scheduling Algorithm
ACU	Average CPU Utilization
AMU	Average Memory Utilization
RCS	Relative Closeness Score
FV	Fitness Value

Fellowship Awarded

Awarded Maulana Azad National Fellowship under Registration Number F1-17.1/2017-18/MANF-2017-18-PUN-75779 from April 2017 to August 2019. This fellowship is awarded to merit holders in Master's program to pursue Doctorate of Philosophy in Sciences and Engineering. This fellowship helped me in the successful completion of this thesis.

Chapter 1

Introduction

Scientific computing is a multidisciplinary field which is growing at a fast pace. Problem domains for scientific computing include applications such as molecular mechanics, weather forecasting, scientific plotting, earthquake, flood prediction, etc *which have dynamic resource demands. Hence, there is a need of high performance computing platform for executing such applications. In past several years, various distributed computing platforms such as cluster, grid and cloud have emerged for deploying scientific applications. Cloud Computing has emerged as a latest paradigm that offers a dynamic and extensible infrastructure for computing and storage. It offers huge variety of resources on-demand to cater the computational obligations of the scientific applications. But, there are certain issues which need to be addressed inorder to make efficient utilization of this technology.*

This chapter exhibits an overall glimpse of thesis and presents scientific computing, several scientific domains, their evolution and paradigms, in conjunction with the various characteristics. The section also discusses how cloud computing serves as a viable platform for executing scientific applications. Further, it provides an overall view of the fundamentals of cloud computing along with the motivation to propose a scheduling approach based on prediction of resource usage for scientific applications. The chapter ends with thesis organization and its pertinent contributions.

1.1 Scientific Computing: An Overview

Scientific computing is a multidisciplinary field [1] which is growing at a fast pace. It is built on the basis of three different elements namely (a) algorithms, modeling & simulation software, (b) computer & information science and (c) computing infrastructure as shown in Figure 1.1. Scientific computing is the key to settle the issues in numerous domains and to give breakthroughs in new learning by consolidating the lessons and methodologies from different large scale distributed computing areas such as high-performance computing, high-throughput computing, etc.

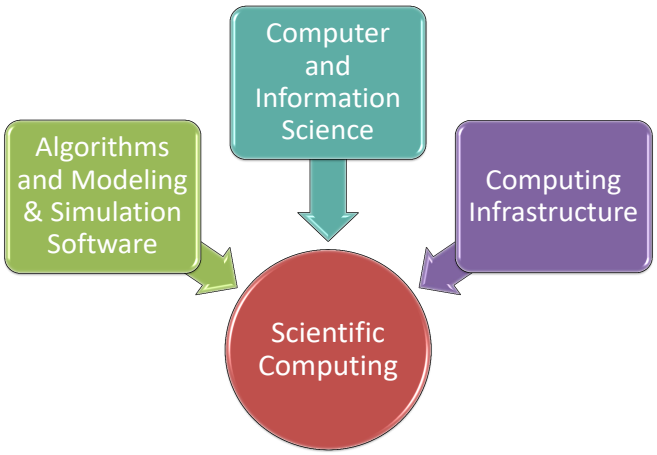

Figure 1.1: Elements of Scientific Computing

Scientific computing uses advanced computing capabilities to deal with complex problems in various scientific domains [2] (e.g., biology, physics, chemistry and mathematics) as shown in Figure 1.2. Problem domains for scientific computing include numerical simulations (e.g., earthquake, sub-atomic particle behavior, turbulent flows, biomolecules etc.), model fitting and data analysis (geophysics, linguistics and biological systems) and computational optimization (machine learning, technical and manufacturing processes).

1.1.1 Evolution of Scientific Computing

In 1940s, the first electronic digital computer namely ENIAC [3, 4] was designed to compute ballistics. At this stage, the researchers of different prestigious organizations started working towards achieving new milestones [5, 6, 7] in scientific computing as shown in Figure 1.3. The first ever machine to mimic the network of neurons was built by Marvin

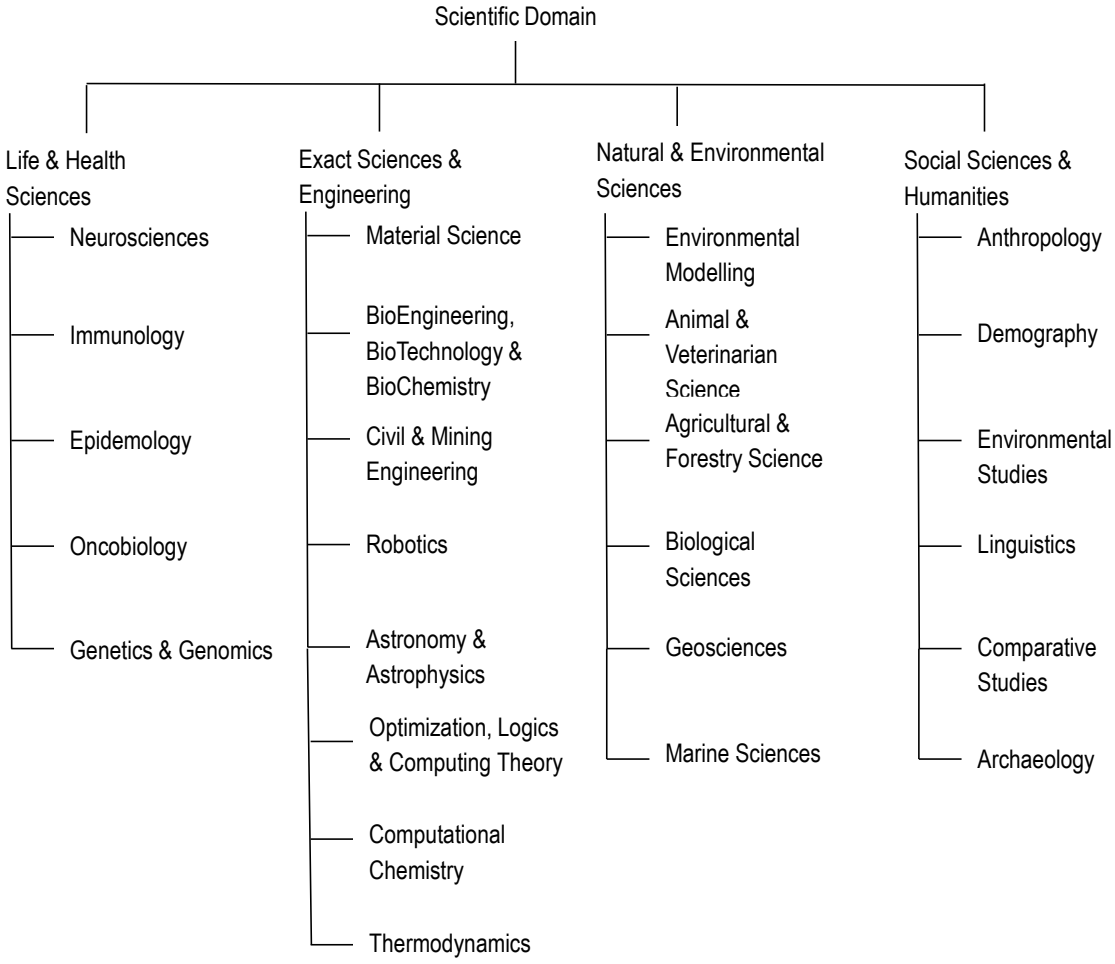

Figure 1.2: Taxonomy of Scientific Domains

Minsky in 1951. In 1954, John Backus developed first scientific programing language FORTRAN at IBM. Then John Kendrew in 1959 built a computerized atomic model of myoglobin with the help of crystallography data. To process the data in real time, the first lab-based computer was designed by Charles Molnar and Wesley Clark in 1962 at MITs Lincoln Laboratory. They designed Laboratory Instrument Computer (LINC) for aiding the research in health sector. In 1963, the first artificial robot arm RANCHO ARM was developed in California which was fully controlled by computer. The United States Department of Defense, started a project namely Advanced Research Projects Agency Network (ARPANET) in 1967 which was funded by Advanced Research Projects Agency (ARPA) for research networking. In 1969, the ocean atmosphere circulation model gave way to climatic simulations which proved to be a powerful research tool

for global warming. Then in 1970s, the first Computerised Tomography (CT) scanner was developed, protein data bank was established, first scientific calculator namely H-35 was released and first Cray supercomputer was installed at Los Alamos which processed large amount of data at fast speed. In 1983, the concept of parallel processing was developed which proved to be a beneficial feature for artificial intelligence and fluid-flow simulations.

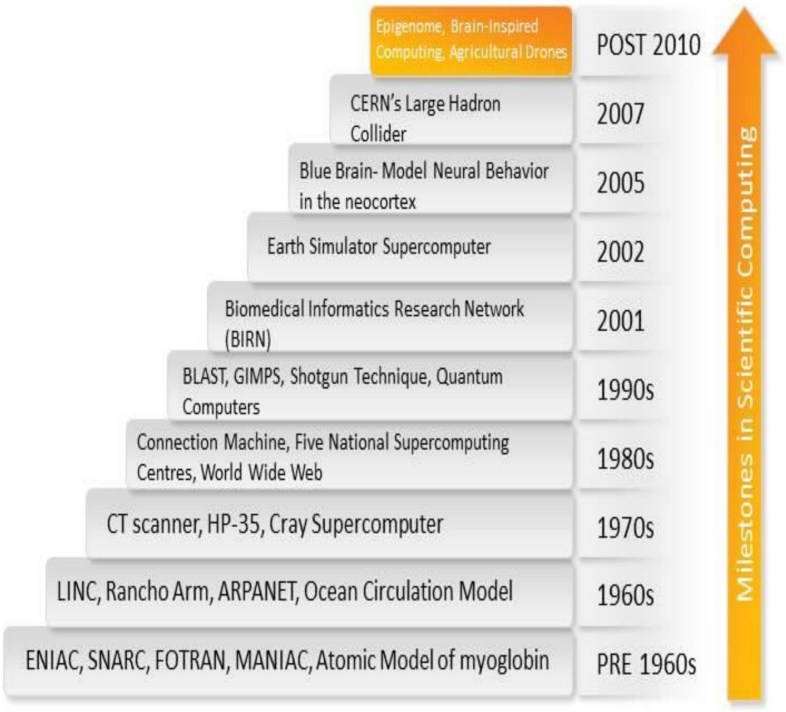

Figure 1.3: Milestones in Scientific Computing

Further, the US National Science Foundation established five National SuperComputing centres inorder to provide sufficient high-computing resources to academicians. In 1989, World Wide Web was invented by an English engineer, Berners-Lee, for collaborating research on a common platform. In 1990s, Basic Local Alignment Search Tool (BLAST) [8] was developed, Great Internet Mersenne Prime Search (GIMPS) was launched, and shotgun technique was developed to combine large fragments of human DNA code. With the beginning of 21st century, Biomedical Informatics Research Network (BIRN) was launched by US National Institute of Health, earth simulator was developed, data mining methods were developed for processing huge astronomical datasets, neural behavior of neocortex was modeled which gave way to brain-inspired computing.

1.1.2 Scientific Paradigms

Scientific research refers to a specific method or technique for obtaining information about natural phantasm. This method has four dimensions or we can say paradigms [9, 10] as shown in Figure 1.4, namely experimental, theoretical, computational and data-intensive. Thousands of years back the first paradigm came into existence that is Experimental Science which is based on the observation and experimentation. This dimension formed the ground of science which further supported the next superstructure on it that is Theoretical Science. This paradigm formed on the basis of mathematics (Newtons Laws, Maxwells Equations, etc.). Then with the beginning of 20th century, a new era of information technology came which boosted the simulation of complex phenomena. This gave birth to third paradigm of science that is Computational Science which involves lots of CPU utilization [11]. And today (21st century), scientists are overwhelmed with data sets from many different sources, which have given way to the fourth paradigm Data-Intensive Science.

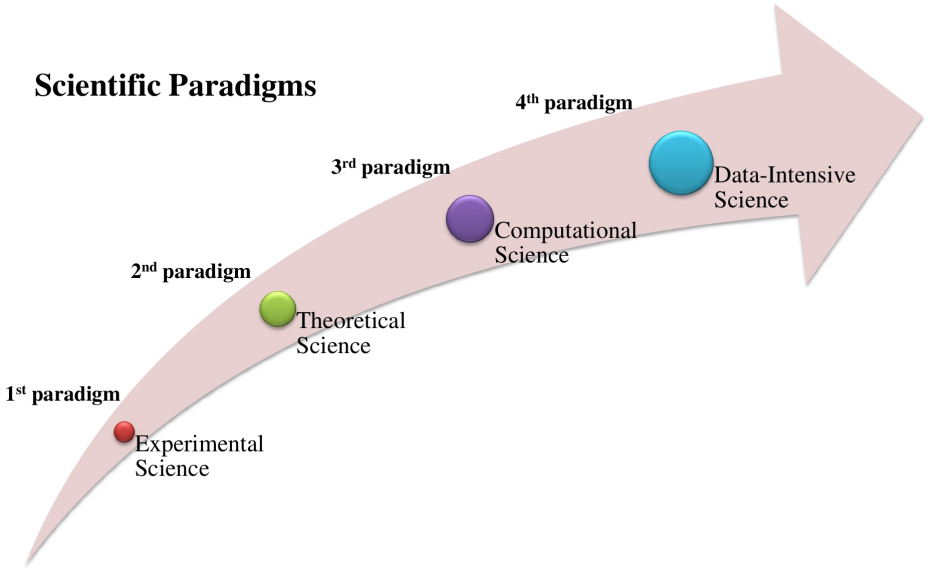

Figure 1.4: Taxonomy of Scientific Paradigms

- **Experimental Science**

  The foremost dimension of scientific research is observation and experimentation which is also termed as Experimental science. This is the foundation stone or we can say the 1st paradigm of science. This paradigm came into existence thousands of years ago. Taking an example of "Traditional Astronomy" from the history- In 3000 BCE, wanderers (planets) were observed in the sky by Babylonians. But later in 100 CE, a model was created by Plotemy which explained these wanderings in

the universe. In this model, earth was fixed in the center with sun, moon and other planets revolving around earth. Then Nicolaus Copernicus in 1500s proposed a new model of universe in which sun was fixed in center with earth at the third position orbiting it. But in early 1600s, Johannes Kepler suggested some modification in the model; orbits were changed from circles to ellipses. Further, Galileo supported Kepler's model by adding experimental evidence using new invent of information technology called telescope.

- **Theoretical Science**

  The second paradigm of science is Theoretical science which can be broadly termed as description and explanation. This dimension involves the making of models which describe the natural phenomena in detail. It is involved with more profound elucidation of the experimental results, making predictions and making new speculations. Around 300 BCE, Aristotle started the work of grouping diverse living things in view of their similitudes and contrasts. A couple of years after the fact, Gregor Johann Mendel's trials with peas proposed a system for legacy: genes, which were transporters of inheritable attributes. In 1953, Francis Crick and James Watson distinguished the twofold helix structure of the synthetic DNA in the core of cells and set up that it contained the qualities of the creature. This disclosure was trailed by another that clarified how cells utilize qualities to make proteins, which are responsible for shaping all the parts of living beings.

- **Computational Science**

  Newton's laws aroused the utilization of numerical models for logical hypothesis. Subsequently, numerical models were not generally pragmatic independent from anyone else. This issue started to be unraveled by the utilization of electronic data innovation to scientific research: to be specific, the computational reproduction of numerical models. For instance, the earth-moon-sun gravitational issue could be reenacted by beginning with starting positions and speeds for those three bodies and figuring new positions and speeds "one time step later" by computational utilization of Newton's laws, then rehashing this procedure for some large number of steps. Since computational researchers dependably need to enhance the exactness of their outcomes and handle more concerning issues, they have a voracious craving for greater PCs that can perform monstrous quantities of these estimations in a sensible time. A noteworthy stride in the utilization of computational science was taken when it was understood that along with mathematical models non-mathematical models could be reenacted. For instance, traffic simulations can anticipate where bottlenecks and different issues may happen.

- **Data-Intensive Science**

  As the speed of data development surpasses Law of Moore towards the beginning of this new centenary, exorbitant information is making incredible inconveniences to individuals. Despite, there are so much potential and exceptionally helpful values covered up in the tremendous volume of information. This has paved path towards the beginning of fourth model of science namely data-intensive science. A considerable number of fields and sections, extending from financial and business exercises to open organization, from national security agencies to large scale scientific researches in different areas, are facing issues of big data. Numerous scientific fields have turned out to be profoundly information driven with the advancement of computer science technologies. There are numerous data intensive scientific domains for example selenology, aerology, and computational biology which generates enormous data. For instance, the Large Synoptic Survey Telescope (LSST) records 30 trillion bytes of picture information in a solitary day. The span of the information equals to two whole Sloan Digital Sky Surveys daily [12]. Stargazers will use propelled examination strategies to this information to explore the roots of the universe. The Large Hadron Collider (LHC) [12] is a particle accelerator that creates 60 terabytes of data for every day. The pattern sets in those data can give us a phenomenal comprehension the way of the universe. 32 petabytes of data generated from climatic observations and simulations were saved on the large scale computing cluster in the NASA Center for Climate Simulation (NCCS). The amount of human genome information is also so vast that interpreting them initially took a decade for processing it. One familiar point amongst these domains is that they produce tremendous amount of data which calls for automated analysis of those data sets. Moreover, to facilitate individual research groups located remotely with the data sets so generated; a centralized repository is required.

1.1.3 Characteristics of Scientific Applications

Scientific applications involve huge computations [13, 14, 15] that utilizes the assets of whatsoever machines are accessible. Complex algorithms and data structures ought to be used so that the computation should be possible with worthy effectiveness, or even so that it should be possible by any stretch of the imagination. The planned calculation is frequently not well characterized and fulfillment of the particular should be recommended by practice, relationship to different heuristics rules. Data types may incorporate not simply structures of content and numbers, but the entire multimedia range, from sound to syn-

chronous time arrangement, from still pictures to video. Scientific applications normally require profound knowledge of scientific domain, and rely on unobtrusive interchange of various approximations. They frequently actualize complex science. As a rule, numerical calculations should be precisely organized not simply to maintain a strategic distance from wrong outcomes, but rather to avoid processs instability. Sensitivity of input data from external sources can be an issue, and lacking information or inadequate input data is normal. Sagacious show of processed amounts might be fundamental for breaking down and deciphering comes about or for intelligent control to direct the calculation. Scientific data processing are such experiments which need to be self-restrained, documented, and classified so that they can be contrasted with other test perceptions.

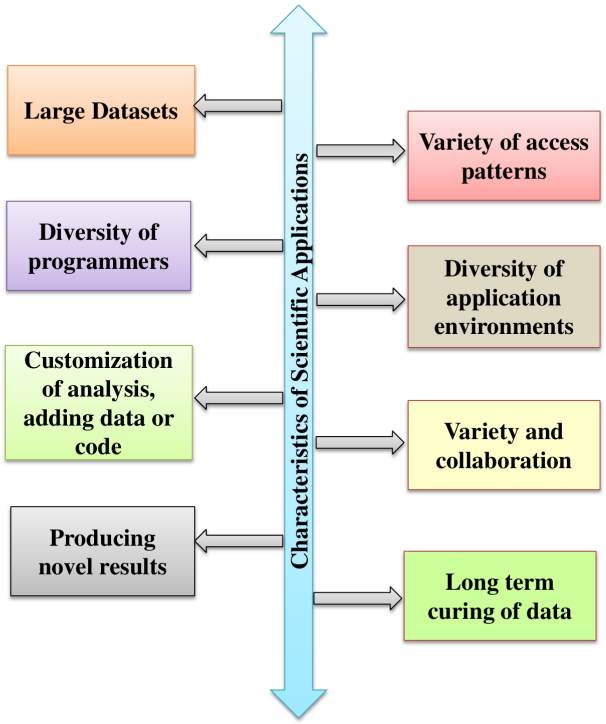

Figure 1.5: Characteristics of Scientific Applications

Scientific applications intend to solve scientific, mathematical or medical problems by constructing mathematical models and numerical solution techniques. Figure 1.5 reflects the characteristics [16] of scientific applications which makes theie implementation a challenging task. Applications such as ESCAT, Inspiral, Epigenome, CERN's Large Hadron Collider (LHC), CyberShake, PRISM, Montage, Laser Interferometer Gravitational Wave Observatory (LIGO), Sloan Digital Sky Survey (SDSS) etc require large amount of computing as well as storage resources to perform large scale experiments.To fulfill the feature requirements of scientific applications [17], cloud computing has come up as feasible platform. It promises low cost, on-demand resource availability, elastic scalability, high reliability and robust performance [18]. Due to all these facilities of-

fered by cloud, scientific community is migrating their applications onto cloud in-order to process them using large distributed computing framework. The next section discusses about cloud computing as a viable platform for deploying scientific applications.

1.2 Cloud Computing- A Viable Platform for Scientific Applications

Cloud Computing has emerged as a new paradigm for large-scale high performance computing [19]. As per National Institute of Standards (NIST) [20], "Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction".

Cloud Computing model is comprised of various important features [21] such as on-demand services, scalability, reliability, wide network access *etc* as depicted in the Figure1.6.

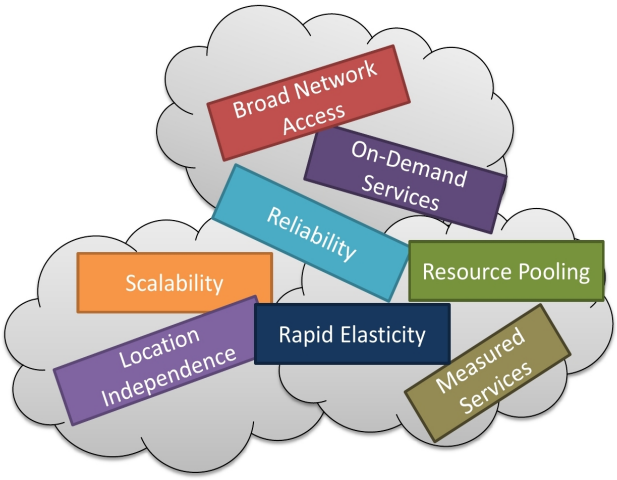

Figure 1.6: Characteristics of Cloud Computing

These characteristics can be implemented by using services and deployment models offered by Cloud.

1.2.1 Service Layers of Cloud

Cloud Computing offers services such as Software as a Service, Platform as a Service and Infrastructure as a Service which are classified according to the abstraction level and service model. The classification according to service type is illustrated through Figure 1.7.

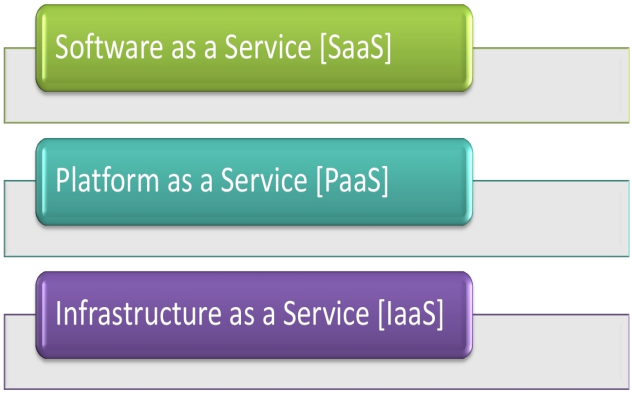

Figure 1.7: Service Model of Cloud Computing

- *Software-as-a-Service(SaaS)* : In SaaS, software functionality is delivered as a service. SaaS provides benefits to service consumers; no initial cost to purchase software, free of cost maintenance, accessible through the internet- anytime, anywhere, high availability and pay as per use pricing. SaaS should preserve a comparatively greater level of its quality than conventional system. The most renowned vendors of SaaS are Gmail and Salesforce.
  - *Example*: Google docs, Facebook, Salesforce etc.
- *Platform-as-a-Service(PaaS)*: PaaS enables the deployment and scalability of users application- a user can deploy applications on PaaS. Google and Microsoft are PaaS providers. Users can execute their applications on PaaS with little modification and probably execute already existing applications.
  - *Example*: Microsoft Azure and Google App Engine.
- *Infrastructure-as-a-Service(IaaS)* : Through IaaS the delivery of resources to the cloud consumers such as servers, storage and associated tools essential over the internet, permitting enterprises to develop an application environment from scratch based on requirement is very easy and inexpensive. Billing is based on the usage of service and can get complicated with tiered on-demand valuing. Amazon is a

popular IaaS provider.

  - *Example*: Amazon EC2, Eucalyptus and Nimbus.

To utilize the above mentioned services, applications need to be deployed on the Cloud. Therefore, for the deployment of various applications in Cloud, deployment models are required.

1.2.2 Types of Clouds

Although, cloud computing has emerged primarily from the appearance of public computing utilities, but some other deployment models are also adopted based on the variation in physical location and distribution. Therefore, a Cloud can be classified as private, community, public and hybrid based on model of deployment as shown in Figure1.8.

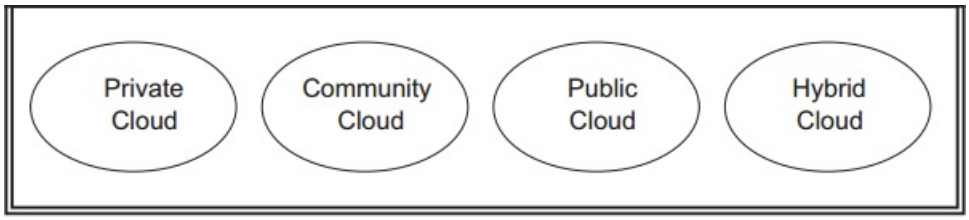

Figure 1.8: Types of Cloud based on deployment models

- *Private Clouds*: A private cloud can be executed internally or by a cloud service provider, benefits of which cannot be fully exploited, and the degree of customization probably might be inadequate.
  - *Examples* : Branches of business company, Educational Institutes (Thapar University).
- *Community Cloud*: For exclusive use of specific community of users from organization sharing some security requirements, project, mission and compliance considerations. One or more organizations specific community can own, manage and operate, such clouds or even third party services can be used. Community clouds can exist in premises or off premises of organization community.
  - *Examples*: Small and middle term business and startups.
- *Public Cloud*: The service is offered to the public and in general delivered by a single cloud provider in which scalability and resource pooling can be completely exploited.

  - *Examples*: Google AppEngine, Amazon Elastic Compute Cloud (EC2), IBMs Blue Cloud.

- *Hybrid Cloud* : Hybrid clouds provide a grouping of many organization procedures, combining their respective benefits and drawbacks. For example: data that need to be secure can reside in a private cloud, whereas public data or applications could be executed in the public cloud.
  - *Examples*: Cloud bursting.

Extensive number of scientific applications and research groups utilize computerized process controls, typically as workflows. Alongside generic Cloud based workflow frameworks, numerous explicit Cloud based workflow frameworks have been developed for executing scientific applications like space sciences and life sciences. Although, cloud is an economical, efficient and flexible solution for scientific applications but still there are various challenges discussed in next section which need to be addressed.

1.2.3 Cloud Computing Adoption Challenges

To access cloud computing services, Scientists can rent computing resources from different cloud service provider. But, there are certain issues which are to be resolved in order to make an efficient utilization of this technology. Various challenges faced by scientific community while deploying applications on cloud computing are described below:

- **Resource Prediction:** In the cloud, various computing resources such as storage, memory, processor, virtual machines and bandwidth are offered to different clients via multi-tenant cloud model. Numerous resources (physical as well as virtual) are assigned dynamically as per requirement of the customer. Nevertheless, general clients, particularly non-IT clients, have no clue to make productive utilization of resources, which will adversely affect the resource utilization leading to excessive monetary cost. Subsequently, an accurate resource demand prediction is significant for effective resource scheduling that will enhance the utilization of resources in cloud environment and spare pointless financial cost for clients.

- **Resource Scheduling:** To enhance the resource usage in the cloud, resource scheduling is one of the main points. Scheduling can be defined as a procedure of mapping the tasks of the application to computing resources for execution. Efficient scheduling of resources can lead to huge improvements in performance of the systems to manage workloads. Amid this procedure, different strategies are made to schedule the resources within budget in order to meet the QoS prerequisites of jobs.

- **Availability and Reliability of Service:** It is important that cloud service providers should offer reliable services to the users. But, certain failure prone conditions lead to unavailability of services thus causing losses. Service Level Agreement (SLA) is a good alternative which can be used to avoid violations and reduce service anavailability costs.
- **Security:** Data stored by the users on the cloud should be accessed with proper authorization inorder to ensure its security and privacy.
- **Cost:** The data-intensive scientific applications demand high computation and large storage capacity. The huge amount of information which needs to be processed and stored on cloud is a huge bottleneck. The cloud service providers must employ such algorithms which can help to select relevant set of features without affecting the results.

This sections briefs certain obstacles which hinders the adoption of cloud computing. The following section discusses the research motivation for this thesis.

1.3 Research Motivation

In order to execute high performing scientific applications, parallel and distributed environment such as cloud is needed. Cloud computing offers tremendous amount of on-demand resources for catering to the computational requirements of such applications [22, 23]. As opposed to Grid and HPC structures, cloud systems engage with a wide userbase where each customer connection takes place at a specific time with distinct resource requirements. Now, these varying interactions exert some challenges for cloud resource usage such as mapping of tasks and resources [24]. To schedule the tasks based on the resource requirements of the application is a challenging task. Also, with the decline in disk storage cost, the dimensions of databases have grown exponentially. To process such large datasets, it is consequential to minimize the number of features [25, 26] because each feature used adds to the manufacturing cost as well as the running time of a system. Thus, a prediction based scheduling approach is required which can select relevant features, predict the resource usage requirements of application tasks and schedule them efficiently. The motive behind this research work is to enhance the performance [27, 28], minimize execution time, cost and SLA violations [29, 30]. The following section describes the pertinent contributions of the thesis.

1.4 Thesis Contributions

The pertinent contributions of the thesis are mentioned below:

- A detailed survey has been conducted to explore various scientific applications, such as healthcare, genomics, physical sciences, drug discovery, flood prediction, etc., which are currently being executed in the cloud environment.
- A comprehensive investigation has been conducted to study the existing machine learning approaches for cloud resource usage prediction.
- Scheduling techniques, including bio-inspired, nature inspired and other optimization techniques have been examined for resource scheduling.
- A resource prediction approach for cloud resources based on the resource usage patterns (CPU and memory usage) of different applications applying machine learning techniques has been proposed and designed. This technique automatically predicts the requisite set of resources for the scientific applications that are being executed in the cloud environment.
- A resource scheduling technique has been developed that takes the predicted set of resources as input and schedules these scientific applications by the efficient utilization of resources while simultaneously reducing the SLA violations at runtime, thereby achieving cost-effectiveness and desired performance.
- Finally, the experimental results have been compared for two scientific applications, namely "Cybershake" and "Floodplain", on the simulator and cloud environments. The outcomes clearly validate the higher performance of the proposed approach in terms of accuracy, execution time, error rate, cost and SLA violations when compared with the existing ones.

1.5 Thesis Organization

The introduction to this thesis is presented in ***Chapter 1***. The remainder of the thesis is structured as follows:

Chapter 2 Literature Survey:

Chapter 2 presents a detailed survey of the existing scientific applications which can be deployed on cloud. These applications are further classified based on their computational requirements and task dependencies. Also, the metrics to fulfill the QoS requirements are

briefly discussed. Besides, the existing resource prediction techniques are surveyed and compared on the basis of different parameters such as time, cost, power, SLA violation, scalabilty, CPU load, memory usage and accuracy. Additionally, an extensive survey of the existing scheduling approaches is presented along with the problems solved and the platforms used. Based on the gaps prevailing in the existing literature, the problem formulation and objectives of this thesis are sketched. This chapter is derived from:

- **Gurleen Kaur** and Anju Bala, *"A survey of prediction-based resource scheduling techniques for physics-based scientific applications"*, Modern Physics Letters B, World Scientific Publications, 32(25), 1-26, 2018. [**SCI Indexed, Impact Factor - 0.929**]

Chapter 3 REAP: Proposed Resource Usage Prediction Technique:

Chapter 3 describes a regressive ensemble approach for predicting the resource usage of scientific applications such as "Cybershake" & "Floodplain" and the objectives that the method aspires to achieve. This section also elaborates the key features and QoS requirements of the proposed approach. Moreover, the design methodology and the constituents of the proposed framework are explained in detail. This discussion includes a description of one of the scientific applications "Cybershake" and its structure, working process and key characteristics, feature selection process using meta-heuristic technique, machine learning prediction models and their methods, package requirements and tuning parameters; and the proposed ensemble algorithm. The time complexity of the proposed algorithm is also presented. Furthermore, an overview of the evaluation metrics to measure the performance of the proposed approach is given. The method selects the important features from the dataset for processing and predicts the resource usage of the scientific applications by ensembling the machine learning prediction models. The performance of the proposed approach is validated on the basis of accuracy and total prediction time. The results clearly state that the method is superior to the existing approaches as it improves the overall accuracy by 2% and reduces the prediction execution time by 16.2%. Further, the results of comparison on the basis of error rate clearly indicate that the proposed approach has higher performance than the existing learning automata ensemble approach. This chapter is partially derived from:

- **Gurleen Kaur**, Anju Bala and Inderveer Chana, *"An intelligent regressive ensemble approach for predicting resource usage in cloud computing"*, Journal of Parallel and Distributed Computing, Elsevier Publications, 123, 1-12, 2018. [**SCI Indexed, Impact Factor - 1.819**]

Chapter 4 RPS: Proposed Prediction Based Scheduling Approach:

Chapter 4 states the problem and the fitness value formulation for the resource prediction based scheduling approach. The section analyzes the need for prediction based scheduling and proposes a framework for executing it. The components include a deployment module in which the scientific applications are applied in the cloud environment, a prediction module which forecasts the resource usage requirements of the application and a scheduler module which arranges the tasks of the application based on multiple criteria. The chapter also details an optimization-based scheduling algorithm which combines the features of an optimization technique with TOPSIS, a multi-criteria decision making algorithm. Firstly, the algorithm checks the resource requirements of the applications against the availability, computes the relative closeness score using TOPSIS, and allocates the task with the highest score to the VM for execution. Finally, the results of the proposed prediction based scheduling approach are validated for one of the scientific applications "Cybershake" along with the existing scheduling heuristics. The proposed approach outperforms the existing ones in terms of execution time, cost and SLA violation rate. The execution rate is curtailed by 35.59%, the cost is approximately 33% lower than the existing ones, and the SLA violation is minimal at 0.91%. This chapter acquired the content from:

- **Gurleen Kaur** and Anju Bala, *"An efficient resource predictionbased scheduling technique for scientific applications in cloud environment"*, Concurrent Engineering: Research and Applications, SAGE Publications, 27(2), 112-125, 2019. [**SCI Indexed, Impact Factor - 1.127**]
- **Gurleen Kaur** and Anju Bala, "OPSA: An Optimized Prediction based Scheduling Approach for Scientific Applications in Cloud Environment", *Cluster Computing*, 2019 [**SCI Indexed, Impact Factor - 1.851**]. [Communicated]

Chapter 5 Verification and Validation of the Proposed REAP and RPS Approaches

Chapter 5 begins with the case study of another scientific application "Floodplain" considered for validating the proposed approach in the actual cloud environment. The section elaborates the characteristics chosen through the feature selection approach and discusses a cloud test bed that was set up for testing and validating the proposed approach using the Microsoft Azure cloud platform. The chapter also provides the results attained after implementing the proposed approaches. Firstly, the outcomes of Regressive Ensemble Approach for Predicting resource usage are shown. The performance of the approach is validated on the basis of RMSE, accuracy and total execution time. The

results clearly state that the proposed ensemble approach is better than the existing ones as it has the lowest RMSE (0.325%), highest accuracy (93.56%) and minimum prediction time (0.26 ms). Finally, the results of the proposed prediction based scheduling approach along with the existing heuristics are validated on the Azure scheduler by creating jobs for Floodplain scientific application in the scheduler job collection directory. The proposed approach outperforms the existing ones in terms of execution time, cost and SLA violation rate. The content of this chapter is derived from:

1. **Gurleen Kaur** and Anju Bala, "Prediction based Task Scheduling Approach for Floodplain Application in Cloud Environment", *Engineering with Computers*, 2019 [**SCI Indexed, Impact Factor - 3.551**]. [Communicated]

Chapter 6 Conclusion and Future Directions:

This chapter summarizes the conclusions drawn from the thesis along with the possible future directions.

Chapter 2

Literature Survey

Scientific applications simulate real world objects into mathematical models and their actions by applying formulas. These applications are more complex in nature and spend a lot of waiting times for computing their jobs. Therefore, the scientific community requires an adaptable platform that can deal with computational challenges like data volume, platform heterogeneity, complexity of application, etc.

Cloud computing has emerged as a potential platform for deploying scientific applications as it offers a tremendous variety of resources on-demand to scientific users. As the demand for cloud services is increasing, it has become essential to schedule the resources efficiently based on application's resource usage requirement. Henceforth, an extensive analysis of scientific applications is done in this chapter to understand the Quality of Service (QoS) requirements. Also, the existing research has been studied to carry out resource prediction and scheduling of tasks in a proficient way.

This chapter investigates the various scientific applications which can be deployed on the cloud and discusses the existing techniques which can predict the resource requirements of applications in general. The state-of-the-art approaches for cloud resource scheduling are also discussed. In the end, the chapter recapitulates the gaps found in existing literature and lists the objectives of the thesis.

2.1 Scientific Applications

Scientific applications simulate real world objects into mathematical models and their actions by applying formulas. OPNET, NS2 or MATLAB is used to simulate a network; GROMACS (GROningen MAchine for Chemical Simulations) is used for chemical simulations and similarly numerous applications exist that solve real life problems. As scientific applications are more complex in nature and spend a lot of waiting times for computing their jobs. Hence, the research community has to experience peak demand bottlenecks. To achieve the best performance of scientific applications; low latency, high bandwidth interconnections, parallel file systems and high performance computing are required [13]. It is important for the users to be well aware about the type of application as it affects the performance and competence of the servers in the Cloud data center. So, to efficiently utilize the resources, it is necessary to understand the resource preferences through classification of various scientific applications.

2.1.1 Classification of Scientific Applications

Based on the resource requirements in Cloud Computing environment, scientific applications can be classified as CPU intensive and Memory intensive as shown in Figure 2.1. Existing CPU and memory intensive applications being deployed in a Cloud environment have oscillating resource demands. These demands contribute to complex behavior in resource usages leading to changes in their intensity and composition with respect to time.

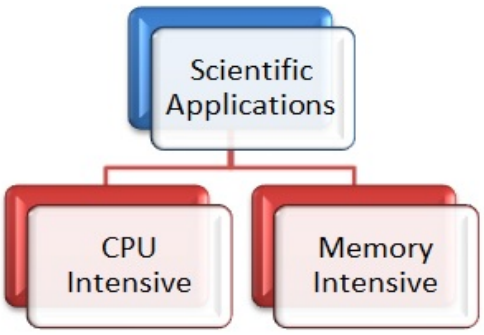

Figure 2.1: Classification of Scientific Applications Based on Resource Requirements

(*i*) CPU Intensive Scientific Applications

Different scientific applications in Cloud have different resource preferences; some may perform excellent on disk and may have poor performance on CPU or vice-versa. CPU intensive scientific applications can be defined as the applications with high CPU usage, for example Epigenome (maps DNA segments generated through the gene sequencing machine). CPU bound applications demand a high amount of CPU resources and low amount of memory resources. While executing CPU intensive scientific applications, the count of context switches per second is minute as it requires a maximum time slice for maintaining the effectiveness of the cache. BLAST searches [8], Monte Carlo simulations [11], *etc* are few examples of CPU-intensive scientific applications.

(*ii*) Memory Intensive Scientific Applications

Scientific applications with high memory usage are memory limited or memory intensive applications. Broadband [31] (generates and compares seismograms from earthquake simulation codes) requires 1GB of physical memory and more than 75% of its runtime is consumed by a task, so, it is considered as a memory intensive application. Large Underground Xenon (LUX) detector [32] is another example of memory bound scientific application which gathers the evidence about the interaction between dark matter and ordinary matter on earth. Weather resource forecasting [33] and computational fluid dynamics are also counted as memory intensive scientific applications. Table 3.1 summarizes the relative resource usage of various scientific applications. Most of the applications are memory intensive and very few applications like Epigenome, Blast Searches and Monte Carlo Simulations are CPU intensive.

Further, an overview of categories of scientific applications is given that describes which application belongs to a particular discipline.

2.1.2 Categories of Scientific Applications

The branches of science can be broadly categorized as natural sciences, formal sciences, social sciences and applied sciences. Natural sciences can be divided into life sciences which include ecology, agriculture, anatomy *etc* and physical sciences which comprise earth science, chemistry, astronomy *etc*. Applied sciences use the principles and theories

Table 2.1: Summary of Resource Usage by the Scientific Applications

Scientific Applications	CPU Intensive	Memory Intensive
Epigenome [31]	Yes	No
Weather Resource Forecasting [33]	No	Yes
Broadband [31]	No	Yes
Blast Searches [8]	Yes	No
LUX Detector [32]	No	Yes
Monte Carlo Simulations [11]	Yes	No
PlasmaPhysics [8]	No	Yes
Turbulence [8]	No	Yes
AstroPhysics [8]	No	Yes
Titan [34]	No	Yes
LU Decomposition [34]	No	Yes
Sparse Cholesky [34]	No	Yes
Rendering [34]	No	Yes
Ab-initio quantum chemistry (Hartree-Fock) [34]	No	Yes
Electron scattering [34]	No	Yes
Synthetic Aperture Radar [34]	No	Yes

of the natural sciences. Mathematics form a base for a wide range of scientific disciplines as it studies quantity, change and structure, spatial relations *etc.* Figure2.2 summarizes the four main disciplines of scientific applications.

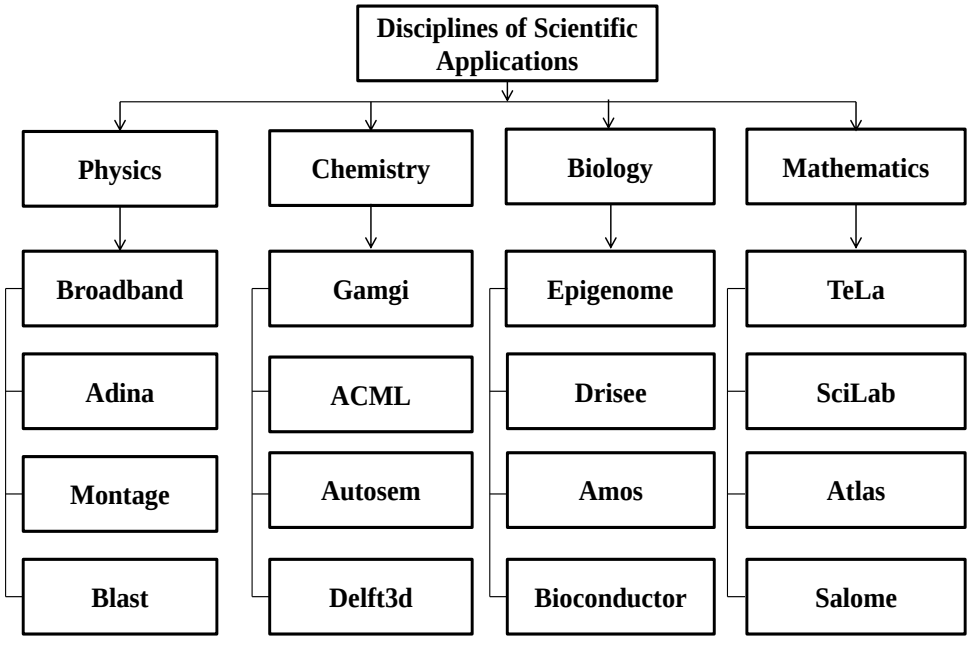

Figure 2.2: Taxonomy of Scientific Applications

2.1.3 Existing Scientific Applications

Scientific applications intend to solve scientific, mathematical or medical problems by constructing mathematical models and numerical solution techniques. Applications such as ESCAT [35], Inspiral [36] , Epigenome [37] , Large Hadron Collider (LHC) at CERN [12], CyberShake [36] , PRISM [35], Montage [31] [38], Laser Interferometer Gravitational Wave Observatory (LIGO)[12], Sloan Digital Sky Survey (SDSS) [12] *etc* require large amount of computing as well as storage resources to perform large scale experiments. An analysis of various scientific applications being deployed in Cloud environment is discussed below:

Heber Gerd *et al.* [39] argued that computational science community is adopting new style of computing namely Post-cluster computing. This study aims to develop a framework to deploy scientific applications as web services and a finite-element method (FEM) analysis system that uses commercial relational database management systems (RDBMSs) as storage opposed to flat files. The systems so developed allows user to write applications that are more distributed, flexible, and easier to implement.

Cao Junwei and Li Junwei [12] proposed a novel data-driven framework to proess large-scale datasets. The main aim of the study was to perform data reduction of large scale scientific datasets. Laser Interferometer Gravitational Wave Observatory (LIGO) has been used a case study to evaluate the performance of the proposed framework.

Zhao Dongfang *et al.* [40] have made an attempt to improve the I/O throughput for scientific applications by incorporating effective compression techniques. Climate data was acquired from the high-resolution Global Cloud-Resolving Model (GCRM) that amounts to 244.25GB, then a real application MMAT [40] was executed on the GCRM dataset that computed the maximum, minimum and average readings of temperatures. I/O throughput was significanty improved as the compression layer was implemented at the filesystem level.

Mastelic Toni *et al.* [41] presented a procedure for comparing different computing infrastructures for a real world scientific application for computing Weibel instabilities within quark-gluon plasma during particle collisions, for example, those in the Large Hardon Collider at CERN. To compare the performance tradeoff of the infrastructures three dimensions are combined namely performance, energy and resource consumption overheads.

The proposed approach is suitable for supporting researchers in their decision-making procedure to choose an appropriate infrastructure. In future, to enhance their complexity model authors will add a learning curve parameter, as well as task parallelization, and utilize it for making top to bottom examination of various cloud setups.

T. Fahringer, N. Mazzocca *et al.* [42] introduced a high-level approach for surveying the execution conduct of complex scientific applications running on a high-performance system via simulation. To analyze the performance of the proposed approach, it was applied on a real-world scientific application (LAPW0), running on a group of SMP nodes.

Cohen Jeremy *et al.* [14] formed Robust Application Porting for HPC in the Cloud (RAPPORT)for running high performance scientific applications in the cloud. The author focused on two domains namely High Energy Physics (HEP) and Bioinformatics (Phylogenetics and Gene structure prediction). Various benefits of cloud platforms for executing scientific applications are discussed in this paper.

Kestor Gokcen *et al.* [43] evaluated the energy cost of data movement in scientific applications. A set of well-engineered micro-benchmarks was developed and an incremental-step approach was used for estimating the cost to move data from one level to next level in memory hierarchy. The main aim of the study was to recognize the cause of energy waste and develop measures to minimize it.

Zou Hongbo *et al.* [44] have deployed 5bytes-lossy, bzip2, and gzip compression algorithms to compress the large datasets being generated by scientific applications. Also the benefits of quality-aware data management system are demonstrated that effectively reduces the data movement cost and improves the Quality of Service (QoS) while meeting the Quality of Information (QoI) constraint stated by users.

Ogrizovi, Dario *et al.* [17] have discussed various economical and technical advantages of executing scientific applications in Cloud environment. Various challenges such as interconnection network, noise, security, data transfer, procurement, licensing, portability and interoperability have been discussed. The author has also mentioned the benefits of deploying scientific applications on Cloud such as cost, instant access, elasticity, configurability, sharing and collaboration.

Son, Seung Woo *et al.* [45] presented a compiler oriented method for management of disk power in server environment that implement array based scientific applications. The method used is proactive as it applies compiler analyses to discover the active and idle times of disk which facilitates the selection of suitable power-mode for the given disk. The author has also demonstrated loop distribution, and loop tiling which can significantly increase the advantages of disk-power-management.

Kirk, W. *et al.* [46] proposed PowerPack framework which enables the management and measurement of power at component level and cluster level for scientific applications. This framework identifies how to conserve power in scientific applications without affecting the performance. The objective of the work done was to reduce the power consumption without any effect on the performance of scientific applications, thereby reducing the cost and improving the reliability.

Liu, Hua, and Manish Parashar [47] , presented a programming system which enhances the Common Component Architecture (CCA) to empower the self-management and self-healing of component based scientific applications. Two scientific applications namely, a self-managing hydrodynamics shock simulation and a self-managing CH4 ignition simulation, are used to demonstrate the system.

Oldfield, Ron A. *et al.* [48] has discussed the use of Light Weight File System (LWFS). The author stated that for massively parallel processing machines efficient I/O is an important challenge which cannot be solved using general purpose file systems. LWFS facilitates the development of light weight I/O libraries which extends the I/O functionality while providing scalable and secure platform to scientific applications.

Paul G. Brown [49] surveyed few scientific applications using SciDB; 1000 Genomes (informatics), Large Underground Xenon Detector (experimental physics) and Remote Sensing (geoinformatics image data analysis). Ober, Ileana, *et al.* [50] defined a method to support the usage of abstraction principles related to software development techniques based on model. To check the viability of proposed approach, the author has applied it to a HPC application simulating defraction of a plane electro-magnetic wave by mono-material 3D objects.

Juve, Gideon *et al.* [31] have examined the performance of Clouds in terms of cost by

executing three scientific applications Montage (an astronomy application), Broadband (a seismology application) and Epigenomics (a bioinformatics application). The experiments are conducted on Amazon EC2 and NCSAs (National Center for Supercomputing Applications) Abe cluster. The final output shows that clouds are feasible option for executing scientific applications in terms of cost and performance if compared with traditional HPC platform.

Berriman, G. Bruce *et al.* [37] have described how Cloud Computing is applicable to scientific applications. The authors also stated that Amazon EC2 is the best available Cloud platform to deploy scientific applications but Magellan, and FutureGrid are also under development process to make them available for science community and that too free of cost. The objective of the study was to understand the performance of three scientific applications namely Epigenome, montage and broadband. Finally, on the basis of performance and cost of these applications on the Cloud a comparison is done.

Prodan, Radu *et al.* [11] have explored the Google App Engine for deploying scientific applications. A novel master-slave framework was proposed to enable the implementation as well as integration of newly developed algorithms. This framework dynamically schedules, executes and monitors the scientific applications on Google infrastructure, taking into consideration the fault tolerant parameter. Parallel rank sort algorithm is implemented in the proposed framework and results are generated by deploying Monte Carlo simulations on Google Cloud environment.

Li, Tonglin *et al.* [51] proposed and implemented a state-management system for for executing high computing scientific applications (Image Comparison and Event Processing) in Cloud environments. Implementation is done on FRIEDA (Flexible Robust Intelligent Elastic Data Management), a data-management and execution framework for Cloud environment. The experiments are performed on FutureGrid and Amazon, the results show that the proposed framework has a nominal overhead and is feasible for state-management of scientific applications in Cloud environment.

Zhao, Dongfang and Ioan Raicu [52] presented a novel storage architecture addressing the I/O problem of traditional high performance computing architecture. The authors have described the procedure to design and execute a new system named FusionFS. Also the steps to migrate FusionFS from traditional HPC systems to Cloud are explained.

Li Liu, Miao Zhang *et al.* [36] summarized that Cloud computing provides a compatible environment for executing large scale scientific applications (Epigenomics, Montage, Inspiral and Cybershake). Yet, it gives birth to new challenges to efficiently schedule the resources for scientific workflow application while fulfilling the QoS requirements of the consumers. The author has proposed adaptive penalty function in co-evolutionary genetic algorithm and compared it with various other optimization algorithms such as Random Forest, Particle Swarm Optimization, and Heterogeneous Earliest Finish Time using WorkflowSim. The objective of the author was to minimize the execution cost and optimize the makespan considering the deadline constraint. Load balancing and task failure optimization objectives are mentioned to be considered as future work.

Blesson Varghese, Ozgur Akgun *et al.* [53] proposed a six step benchmark methodology to maximize the performance of Scientific Applications. The hypothesis is validated on three scientific applications: a) financial risk simulation b) molecular dynamics simulation and c) mathematical solver, by incorporating the comparative validation. The objective of the work done was to maximize the performance of the scientific applications on Cloud by selecting an appropriate virtual machine. The results are generated by deploying the applications on Amazon Elastic Cloud Compute (EC2). At the end, it is shown that the benchmarking method of choosing amongst VMs is successfully improving the performance of the scientific applications but at the expense of execution cost and execution time.

Peng, Junjie *et al.* [54] analyzed various cpu-intensive applications and proposed an execution time prediction model and a scheduling method to improve the efficiency of cpu-intensive applications by reducing the execution time. The author has used cloudstack version 5.0, a Cloud computing platform for deploying the applications and performing the experiments. The model so proposed is able to predict the execution efficiently but scheduling method could be improved further for effective and efficient scheduling of the resources.

Zhao, Dongfang *et al.* [8] have explored various data-intensive scientific applications. The author has proposed a distributed layer of storage on computing nodes to handle the I-O bottleneck of real world scientific applications. FusionFS file system has been built to support data-intensive operations. Based on ROSS and CODES simulation tools, FusionSim is built to simulate FusionFS behavior. The proposed system is efficiently performing the scaling but is not able to balance the load and support the operation of parallel write.

Alexandru Iosup *et al.* [55] analyzed the efficiency of Cloud services for deploying scientific applications. The author stated that the Cloud has a great potential for scientific community as it is more reliable, scalable and cost effective alternative as compared to supercomputers, clusters and grids. They have deployed various cpu-intensive scientific applications on four Cloud platforms namely GoGrid, AmazonEC2, Mosso and Elastic Hosts and also presented a comparison between these platforms. Based on the experimental results, the author concluded that handling large data sets, fault-tolerance and monitoring are the quality of service parameters that need to be explored.

Moise, Diana [56] performed big data analysis on HPC system, using a scientific application based on Hadoop to analyze molecular dynamics simulations data. The author has used Hadoop MapReduce to execute the scientific application and discussed various challenges such as data formats, storage issue, inputting data into Hadoop, optimization issues *etc* to be handled while performing such computations on scientific data.

Vckler, Jens-Snke *et al.* [57] has executed a scientific application on Cloud environment to process the astronomy data for exploring planets similar to earth revolving around other stars. The author deployed this application on multiple clouds (FutureGrid, Amazon EC2 and NERSCs Magellan Cloud) with the help of Pegasus and Condor. A comparison is shown amongst these three clouds where amazon is giving lower CPU speed, and poor WAN performance. Further a number of issues are mentioned to be addressed namely optimal utilization of resources , distinct clustering strategies, evaluating the tradeoffs between various system parameters and load on the host , substitute data staging techniques, review novel protocols and data storage solutions.

Pelle Jakovits and Satish Narayana Srirama [13] discussed that how distributed computing framework can be used to adopt the scientific applications in the Cloud environment. They have used Hadoop Mapreduce distributed framework for solving complex scientific computing problems.

Jaliya Ekanayake *et al.* [58] have used MapReduce for executing data intensive scientific applications. They discussed their experience with implementing two typical scientific analyzes: High Energy Physics data analyzes and Kmeans clustering The author has shared their experience of using MapReduce for two distinct scientific data-analysis namely High

Energy Physics data-analysis and K-means Clustering. Utilization of iterative algorithms by scientific applications limits the applicability of Hadoop MapReduce, thus they have proposed a streaming-based MapReduce implementation named CGL-MapReduce. Finally, a comparison is shown between Hadoop and CGL-MapReduce where the overall performance of Hadoop has significantly lowered due to lack of support for iterative MapReduce tasks of scientific applications.

Kaur *et al.* [59] also proposed resource elasticity framework for QoS aware execution of health care applications. Multiple applications can be deployed on the proposed framework but fault predictions have not been done.

Sindrilaru *et al.* [60] also implemented a technique named Simple Object Access Protocol (SOAP) messages to detect faults for actual workflow engine inorder to improve response times. Benchmark based method can be used in the future.

To detect replication failures, Zhao *et al.* [61] have implemented heartbeat message protocol and identical concept has been presented by Jhawar *et al.* [62] which uses heartbeat message protocol to detect crash failures among virtual machine instances in Cloud environment but these are not dedicated towards scientific applications.

Guan *et al.* [63] utilized bayesian models and decision trees to develop a mechanism for predicting faults inorder constructing reliable Cloud Systems. Real life applications can be deployed using proposed fault prediction system.

Bala *et al.* [64] proposed an intelligent failure prediction model for scientific workflow applications that are used to handle task failure proactively. Dataset is generated after executing Montage, Cybershake, Inspiral and Sipht workflow applications on WorkflowSim. Experimental results are evaluated using machine learning approaches Naive Bayes, Random Forest, LR and ANN. The proposed prediction model can be useful for resource scheduling for multiple scientific applications by predicting resources taking time and cost parameters into consideration.

Table 2.2 summarizes the various existing scientific applications, their type, platform on which they are deployed and the challenges faced.

Table 2.2: Existing Scientific Applications

Application	**Type**	**Platform**	**Challenges**
Laser Interferometer Gravitational Wave Observatory (LIGO) [12]	Memory Intensive	Grid	To Reduce data volume.
MMAT [40]	CPU Intensive	IBM BlueGene/P supercomputer	To improve I/O throughput.
Large Hardon Collider [41]	CPU Intensive	Grid, Cluster and Cloud	Compare performance tradeoff of the infrastructures.
Linearized Augmented Plane Wave (LAPW0) [42]	CPU Intensive	Heterogeneous System (HeSSE) Simulator	To assess the performance behaviour of application.
BioInformatics (Phylogenetics and GeneStructure Prediction) [14]	CPU Intensive	Cloud (Eucalyptus)	To identify potential benefits of deploying scientific applications on Cloud.
NEKBone, LULESH, GTC [43]	CPU Intensive	Dual Socket AMD Opteron 6272	To estimate total energy cost of data movement.
wupwise, swim, mgrid, applu, mesa, galgel [45]	Array based	Simulator	Disk-Power management.
Fourier Transform [46]	CPU Intensive	PowerPack Framework	Conserve energy without impacting performance.
Self-managing hydrodynamics shock simulation and self-managing CH4 ignition simulation [47]	CPU Intensive	Beowulf Cluster	To empower self-management and self-healing of component based scientific applications.
Epigenome [31]	CPU Intensive	Cloud (Amazon EC2)	Deliver required performance at reasonable price.
Broadband [31]	Memory Intensive	Cloud (Amazon EC2)	Deliver required performance at reasonable price.
Monte Carlo Simulations [11]	CPU Intensive	Cloud (Google App Engine)	Schedule, execute and monitor scientific application considering fault tolerance parameter.
Image Comparison and Event Processing [51]	Memory Intensive	Cloud (FutureGrid and Amazon)	to track the applications entire lifetime.
Financial Risk Simulation, Molecular Dynamics Simulation and Mathematical Solver [53]	CPU Intensive	Cloud (Amazon EC2)	To select an appropriate virtual machine for enhancing performance of the scientific applications.

to be cont'd on next page

Table 2.2: Existing Scientific Applications (cont.)

Application	**Type**	**Platform**	**Challenges**
PlasmaPhysics, Turbulence, AstroPhysics and Basic Local Alignment Search Tool(BLAST) [8]	Memory Intensive	Simulator (FusionSim)	To tackle the HPC I/O bottleneck of scientific applications.
Molecular Dynamics Simulation [56]	Memory Intensive	Hadoop MapReduce	Optimizing the de- ployment and the application.
Astronomy [57]	Memory Intensive	Cloud (FutureGrid, Amazon EC2 and NERSC's Magellan Cloud)	Evaluation tradeoffs between various system parameters.
High Energy Physics and Kmeans Clustering [58]	Memory Intensive	Cloud (Hadoop MapReduce and CGL-MapReduce)	Compare performance of Hadoop MapReduce and CGL-MapReduce.
Montage, Cybershake, Inspiral and Sipht [64]	CPU and Memory Intensive	Cloud (Pegasus and Amazon EC2)	To minimize the workflow tasks failure effect.

Various scientific applications namely LIGO, Epigenome, Broadband, Cybershake, LAPW0, Molecular Dynamics Simulation *etc* have been surveyed. Based on literature survey, various Quality of Service (QoS) requirements of scientific applications have been listed as discussed further subsection.

2.1.4 QoS requirements of Scientific Applications

Quality of Service (QoS) is required to monitor the efficient execution of applications and measure the services delivered. In order to optimize QoS parameters and to assist efficient performance of workloads and applications, the below mentioned characteristics [17] and [19] must be addressed effectively.

- *Execution Time*: A key challenge while executing scientific applications is ensuring reduction of overall execution time. Execution time can be defined as the overall time taken while running the application. Fulfilling this parameter will reduce the execution cost and optimize the makespan and CPU time.
- *Cost*: While implementing scientific applications, cost is also a very important parameter which should be taken into consideration. HPC applications require a large amount of resources for execution, which incurs a huge cost. There is a need for such a platform which facilitates cost-effective resource usage.

- *Latency*: The time taken from input of data into a system to the desired outcome is known as latency. Scientific applications demand low latency and high throughput as it will improve the scalability. Low-latency will enable interactive handling of large datasets than has ever been possible in the past.
- *High Bandwidth*: For scientific applications, high bandwidth is the most critical requirement as it allows them to achieve good performance for collective communication operations. There are a number of areas such as remote graphical visualization, movement of high rate sensor data from space, military command and control, *etc* which require high bandwidth networking in order to support the scientific research community.
- *Interoperability*: Interoperability is an important issue that needs to be addressed for high performance scientific applications. For executing scientific applications it is important to have correlation between different systems and protocols, in terms of compliance to scientific standards, or deliberately in terms of the ability to share and re-use data gathered in different contexts.
- *Response Time*: Communication delays can widely affect the execution of a scopic class of high performance distributed applications, such as scientific simulations. Due to the sharing of resources among a large number of tenants, stable and homogeneous network connections between physical machines doesnt exist which leads to poor response time.
- *Handling Streaming Data*: Scientific applications such as astronomical survey projects, meteorological and geodetic measurements generate high volume data. However, the increase in data volumes and increasing demand for real time data analysis has made it necessary to handle the streaming data in an efficient manner.

Issues such as handling streaming data, reducing execution time, cost, and enabling result sharing across collaborations are scrutinized using workflow systems. Literature survey shows that although scientific applications are deployed on supercomputers, grids as well as clusters but Cloud has become a new trend for executing large scale high computing applications. It can be concluded that performance of scientific applications can be enhanced on Cloud environment if proper scheduling of resources is done. But prior to scheduling it is important to predict the resources so that they can be scheduled efficiently. Therefore, in next section, various existing resource prediction techniques are discussed.

2.2 State-of-the-art: Resource Prediction Techniques

Resource prediction assists in effective resource management not only by fulfilling the QoS requirements of the end users but also by pacifying Cloud service providers by rationalizing the cost of resources. Lots of research work has been done to predict the resources in Cloud environment, but the existing prediction techniques are not specifically dedicated to scientific applications. In addition, these existing techniques either satisfy end users or Cloud resource providers. The various existing resource prediction techniques are discussed below:

James Oly and Daniel A. Reed [65] evaluated Markov Model for predicting I/O requests for scientific applications. They also proposed three algorithms for I/O prefetching namely Greedy Prediction, Path Prediction and Amortized Prediction. A suite of six scientific applications have been used to evaluate the performance of markov models namely Cactus, CONTINUUM, Dyna3D, Hartree-Fock, HYDRO and SAR.The results show that markov models provide a balance between implementation complexity and predictive power resulting in reduced execution times of scientific applications.

Mastelic Toni *et al.* [66] proposed a novel resource allocation technique which predicts the process execution and the required resources then allocates the cloud resources efficiently and finally optimizes the cost. The proposed approach helps companies to estimate the migration cost of their business to cloud. Other Cloud attributes such as QoS will be considered as future work. Y. Jiang *et al.* [67] proposed a self-adaptive system (ASAP) for resource demand provisioning. The system uses temporal mining of historical data to predict the virtual machines (VMs) required aforetime. The author has used ensembling of prediction methods where weights were assigned to different base prediction methods and the ultimate prediction was close to best predictor. Cloud prediction costs form the base for selection of best model.

Y. Shi *et al.* [68] suggested that analysis and prediction of resource utilization logs aids in dynamic allocation of resources. The main objective of the study was to reduce the consumption of power and improve the response time. The author has used Linear Predicting Method and Flat Period Reservation Reduced Method to acquire information about resource usage from resource utilization log and enhanced the working of M/M/1 queuing prediction method in terms of response time and energy consumption.

M. Verma *et al.* [69] presented a novel framework for dynamic resource demand prediction and allocation in Cloud environment. This framework classifies the service users on the basis of their resource requirements. If the users demand for the resource is going to increase then that user is given priority inorder to reduce the prediction time. This approach also uses best fit heuristic approach to add the service users to corresponding matching virtual machines and allocates the VMs to physical host machines. R. N. Calheiros and R. Buyya [70] designed a full fledge autonomic prediction system which includes application provisioner, workload analyser and load predictor. The objective of the work was to fulfill QoS requirements, minimize costs, reduce power consumption and enhance application performance. The system first predicts the upcoming demands and then generates an analytical model to decide on the optimal Virtual Machines (VMs).

E. Caron *et al.* [71] focused on predicting resource usage using string matching approach. String matching is a pattern based prediction technique which identifies similar occurrences of current workload in past. Current and historical usage records are utilized for time-series analysis in [73]. A statistical prediction method namely Double Exponential Smoothing (DES) is used in this study to predict the resources. J. Jiang *et al.* [72] predicted demand of web requests using Linear Regression (LR) which performs time series analysis. Queuing theory is used to generate the relation between web requests, cost and latency in order to find optimal number of Virtual Machines (VMs).

Fu *et al.* [73] have constructed a Neural Network (NN) to estimate the number of component failures in a given time interval and another work of Fu et al. [20] predicted component failures by applying failure proactive prediction framework based on the concept of temporal and spatial correlations which generates an average accuracy of 74% with online prediction. But other failures such as resource, task or application can be handled to increase the availability and reliability of various applications. S. Islam *et al.* [74] incorporated Neural Network (NN) and Linear Regression (LR) to develop a prediction based proactive measurement and provisioning of resources. Machine learning and sliding window techniques have been used by the prediction models. Historical usage and current usage data is used to predict the future utilization of resources. Amazon EC2 is used to perform the experiments and generate results.

A. Biswas *et al.* [75] performed scaling of resources based on the prediction of resource workload using previous workload logs by applying machine learning prediction technique

namely Support Vector Machine (SVM) and Linear Regression (LR). Parameters considered includes various costs and service time which forms a broker-point function. Zhang *et al.* [76] also presented a failure prediction framework for autonomic management of network system applications. But they have not used the proposed autonomic framework in Cloud environment for deploying various scientific applications. Further, enhancement of machine learning techniques are to be made in order to optimize the performance of these approaches.

Kousiouris *et al.* [77] have used ANN to estimate resource provisioning in Cloud and also verified the enhanced prediction accuracy using ANN. Further the work of Islam *et al.* [14] presented a framework that utilized neural networks and linear regression models along with sliding window technique for on demand resource allocation in Cloud. But it can be enhanced for scheduling various resources after the estimation of resource usage. Guo *et al.* [78] have combined reinforcement learning and machine learning models which is further utilized to determine optimal resource allocation for a multitier application. It can be extended for resource scheduling for multiple scientific applications.

The prediction techniques and their respective proposed methodology along with parameters and platform are enumerated in Table 2.3.

Table 2.3: Existing Resource Prediction Techniques

Author	**Technique**	**Proposed Methodology**	**Parameters**	**Platform**	**Application**
M. Verma *et al.* [69]	Time Series	Trend seasonality, exponential moving average	Resource requirement, prediction time, cost	Cloud architecture	Multi-Tenant SaaS application
A.Biswas *et al.* [75]	Time Series	SVM, Linear Regression	No. of predictions, cost, execution time	Weka	Not mentioned
J. Jiang *et al.* [72]	Time Series	Linear regression, queuing theory	Cost-latency trade off	Amazon	Web applications
J. Huang *et al.* [79]	Time Series	Double exponential smoothing	Prediction Accuracy, resource idle rate	CloudSim	Not mentioned
S. Islam *et al.* [74]	Time Series	Neural Network/ Linear Regression with sliding window	Prediction accuracy, performance, cost	Amazon EC2	E-Commerce applications

to be cont'd on next page

Table 2.3: Existing Resource Prediction Techniques (cont.)

Author	**Technique**	**Proposed Methodology**	**Parameters**	**Platform**	**Application**
R.N. Calheiros and R. Buyya [70]	Queuing Theory	M/M/1/k queue	Resource demand, arrival pattern, estimation error, deadline, budget	CloudSim	Scientific applications
Y. Jiang *et al.* [67]	Time Series	Best of Moving Average, Auto Regression, ANN, SVM	Latency, cost, VM Type and Request Start/End Time	IBM Smart Cloud Enterprise (SCE)	Workload from IBM SCE
Y. Shi *et al.* [68]	Utilization Log Analysis	Improved MMQMPM method	SLA violation rate, power consumption, response time	CloudSim	Web applications
E. Caron *et al.* [71]	Machine Learning	Pattern Matching	Scalability, cost	Animoto, Large Hadron Collider, Computing Grid, NorduGrid, SHARCNET	Cloud and grid client applications
Eddy Caron *et al.* [80]	Time Series	String matching based scaling algorithm Knuth-Morris-Pratt (KMP) algorithm	Total no of CPU, no. of serviced requests, cost	Amazon EC2, IBM cloud platform	IBM Cloud applications
Z Gong *et al.* [81]	Time Series	PRedictive Elastic reSource Scaling (PRESS)	CPU load, memory, I/O, network, response time	Xen platform, RUBiS, Google Cluster	Workload of Google application
Zhiming Shen *et al.* [82]	Time Series	CloudScale system architecture, Online adaptive padding scheme, Prediction error correction	CPU load, memory usage, response time, job progress	Xen platform, RUBiS, Hadoop MapReduce system, IBM system	CPU and memory intensive applications of Hadoop, RubiS,IBM
J Huang *et al.* [79]	Time Series	Double exponential Smoothing, Prediction model and scheduling algorithm	CPU load, memory usage, prediction accuracy	CloudSim	Not mentioned

to be cont'd on next page

Table 2.3: Existing Resource Prediction Techniques (cont.)

Author	**Technique**	**Proposed Methodology**	**Parameters**	**Platform**	**Application**
H Mi *et al.* [83]	Time Series	Genetic algorithm based approach (GABA), Self-reconfiguration architecture, Browns quadratic exponential smoothing	No of requests per VM, VM load Time interval	TPC-W, Httperf tool	Web applications
Nilabja Roy *et al.* [84]	Time Series, Control Theory	Look-ahead controller algorithm, Second order autoregressive moving average method (ARMA)	No. of users in the system, response time, VM cost, application reconfiguration cost	Custom testbed	1998 Soccer world cup website
Waheed Iqbal *et al.* [85]	Time series, Threshold based rules	Prototype system for resource provisioning	CPU load, no. of requests, no. of VMs, Response time	Eucalyptus based testbed, RUBiS, httperf	Multi-tier web application
Abhishek Chandra *et al.* [86]	Time series, Queuing theory	Dynamic resource allocation architecture, Time-domain queuing model	Request rate, service demand, response time, arrival rate, queue length	Matlab simulator,NetSim library, DASSF simulation package	Web applications
Michael Cardosa and Abhishek Chandra [87]	Time series	Resource usage histograms, Clustering based resource aggregation, Adaptive algorithm for parameter selection	CPU load, total no of nodes, data transfer overhead, queue latency	PlanetLab traces by CoMon, Matlab	PlanetLab trace
Fang, W. *et al.* [88]	Time series	RPPS (Resource Prediction and Provisioning scheme) architecture, ARIMA prediction model	No. of requests, CPU load, prediction accuracy, total no of nodes	Xen platform, KVM	workload traces collected using typical CPU intensive applications

to be cont'd on next page

Table 2.3: Existing Resource Prediction Techniques (cont.)

Author	**Technique**	**Proposed Methodology**	**Parameters**	**Platform**	**Application**
Khatua, S. *et al.* [89]	Time series, Threshold based rules, Feedback control	Monitoring and Optimizing Virtual Resources (MOVR) architecture	No of requests, queue length, time in the queue, provider time (controller waiting time)	Amazon EC2, Xen platform, Eucalyptus, Php-Collab application, Zabbix	Php-Collab application
Prodan, R. *et al.* [90]	Time Series, Neural Networks	Massively Multiplayer Online Games (MMOG) architecture, Neural network based prediction	No. of entities, CPU time, prediction accuracy	RuneScape traces from MMORPG game simulator, FPS game simulator	Online games
Dutta, S. *et al.* [91]	Time Series	SmartScale architecture, Optimal scaling algorithm	No of requests, response time, cost of VM, cost for Application reconfiguration	KVM hypervisor, Olio benchmark	Olio- Cloud benchmark

Previous studies have only focused on time series, queuing theory techniques of machine learning that has been discussed in Table 2.3. However, the research does not take into account the techniques specifically proposed for scientific applications. Hence, there is a need for efficient prediction techniques that can predict resource usage for scientific applications. Also, there are eight major parameters which are considered by different applications. The comparison of prediction techniques for different parameters is summarized in Table 2.4.

Table 2.4: Comparison of Prediction Techniques for different parameters

Author	**Time**	**Cost**	**Accuracy**	**CPU Load**	**Memory Usage**	**Scalability**	**Power Consumption**	**SLA Violation**
M. Verma et al. [69]	✓	✓	✗	✗	✗	✗	✗	✗
A.Biswas et al. [75]	✓	✓	✗	✗	✗	✗	✗	✗
J. Jiang et al. [72]	✗	✓	✗	✗	✗	✗	✗	✗
J. Huang et al. [79]	✗	✗	✓	✗	✗	✗	✗	✗
S. Islam et al. [74]	✗	✓	✓	✗	✗	✗	✗	✗
R.N. Calheiros and R. Buyya [70]	✓	✓	✗	✗	✗	✗	✗	✗
Y. Jiang, et al. [67]	✓	✓	✗	✗	✗	✗	✗	✗
Y. Shi, et al. [68]	✓	✗	✗	✗	✗	✗	✓	✓
E. Caron, et al. [71]	✗	✓	✗	✗	✗	✓	✗	✗
E. Caron, et al. [80]	✗	✓	✗	✓	✗	✗	✗	✗
Z Gong, et al. [81]	✓	✗	✗	✓	✓	✗	✗	✗
Zhiming Shen, et al. [82]	✓	✗	✗	✓	✓	✗	✗	✗
H Mi, et al. [83]	✓	✗	✗	✓	✗	✗	✗	✗

to be cont'd on next page

Table 2.4: Comparison of Prediction Techniques for different parameters (cont.)

Author	**Time**	**Cost**	**Accuracy**	**CPU Load**	**Memory Usage**	**Scalability**	**Power Consumption**	**SLA Violation**
G Chen, et al. [92]	✓	✗	✗	✓	✗	✗	✗	✗
Nilabja Roy, et al. [84]	✓	✓	✗	✗	✗	✗	✗	✗
Waheed Iqbal, et al. [85]	✓	✗	✗	✓	✗	✗	✗	✗
Abhishek Chandra, et al. [86]	✓	✗	✗	✓	✗	✗	✗	✗
Michael Cardosa and Abhishek Chandra [87]	✓	✗	✗	✓	✗	✗	✗	✗
Fang, W. et al. [88]	✗	✗	✓	✓	✗	✗	✗	✗
Khatua, S. et al. [89]	✓	✗	✗	✗	✗	✗	✗	✗
Prodan, R. et al. [11]	✓	✗	✓	✗	✗	✗	✗	✗
Dutta, S. et al. [91]	✓	✓	✗	✗	✗	✗	✗	✗

Data from several studies have identified that the existing techniques are primarily focused on execution time and cost. However, the research does not take into account various important parameters. For example: very few researchers have taken accuracy, CPU load, memory usage and SLA violations into consideration. Therefore, there is a need for a better prediction technique which can improve the accuracy, efficiently manage the CPU and memory usage thus minimizing the SLA violations. Moreover, resource scheduling has been suggested by various authors to be enhanced as a future work. Therefore, in next section, existing resource scheduling approaches are discussed.

2.3 State-of-the-art: Resource Scheduling Approaches

Cloud Computing offers resources on-demand assuring reliable and assured services in pay as-per-usage. Variety of Cloud services can be demanded concurrently by a number of Cloud consumers. Therefore, to fulfill the requirements of the Cloud consumers, it is necessary to provide all the requested resources in a well-organized manner. Various alternative methods of resource allocation in cloud environment determined from the literature are given below:

Scheduling can be defined as a procedure of mapping the tasks of the application to computing resources for execution. Efficient scheduling of resources can have a great impact on the performance of the system in terms of workload management. Several scheduling algorithms are used [93, 94] for appropriate scheduling of Scientific applications.

Ge Rong *et al.* [95] proposed distributed performance-directed DVS (Dynamic Voltage Scaling) scheduling strategies for scientific applications. The main aim of the study was to minimize the energy consumption without incrementing the execution time. The proposed approach is largely manual so it will be fully automated in future while addressing the limitations of the various scheduling techniques. The energy savings will also be improved in future while maintaining performance through better prediction methods.

Latiff, Muhammad Shafie Abd *et al.* [96] proposed an optimization technique named Global League Championship Algorithm (GBLCA) to schedule the scientific applications in the cloud environment. CloudSim simulator is used to perform the experiments

and generate results. The outcome shows that there is a remarkable improvement in makespan; applications are securely scheduled in a reasonable time. GBLCA technique provides better scheduling of scientific applications in comparison to Min-Min, Min-Max, Ant Colony Optimization and Genetic Algorithm.

Suraj Pandey *et al.* [93] used Particle Swarm Optimization (PSO), a heuristic approach, to schedule tasks on cloud resources in accordance to applications. The approach is applied to Scientific applications to reduce the processing and transfer costs. Final outcomes proved that PSO can reduce the amount of cost while effectively sharing the workload onto resources.

Ioana Banicescu *et al.* [97] evaluated the performance of two scientific applications: computational field simulation on unstructured grids, and N-Body simulations, by integrating Adaptive Weighted Factoring with dynamic loop scheduling technique. The main objective of the study was to balance the load for improving the performance of scientific applications.

The performance of support vector machine (SVM), neural networks (NN) and Linear Regression (LR) have been evaluated by Bankole *et al.* [98] for the TPC-W benchmark web application. Response time and throughput have been considered as the SLA metric in their prediction model for providing decision on robust scaling. Further, it would be useful for implementing resource allocation and scheduling approaches in actual cloud testbed.

J. Li *et al.* [99] developed a co-ordinated scheduling algorithm for scheduling interactive workloads and batch jobs. Firstly, a priority function is used to measure the requests urgency and request migration is performed to combine the residual capacities for batch jobs. Secondly, prediction of workload demand is done and a dynamic resource planning scheme is devised. Finally, cost-cutting spot instances are considered if residual capacities are not sufficient for batch jobs. For interactive workloads, the data is created from four real applications,namely NASA, DAS-2, Grid5000 and LCG. For batch jobs, the data is created from five real applications, which are SIPHT (biology), LIGO (gravitational physics), Epigenomics (biology), CyberShake (earthquake science), and Montage (astronomy).

A failure aware workflow scheduling approach have been proposed by Yu *et al.* [100] for predicting resource failures online but integration with workflow schedulers is not taken into consideration. A fault tolerant workflow scheduling approach has been proposed by Poola *et al.* [101] which reduced the cost rate by 70%. The author suggested that to further reduce the usage cost for scientific workflows, failure prediction approaches can be applied. Ardagna *et al.* [102] have also proposed energy aware autonomic resource allocation policies for multitier web applications in virtualized environments but they have not employed machine learning for resource allocation which can further improves the performance of various scientific applications.

To compute the mean processing time of each workload Topcuoglu *et al.* [103] presented the HEFT algorithm. This algorithm can also be used to calculate the average communication time among the resources of two workloads. Firstly, the workloads in the application are well-ordered on a rank function and then higher priority is given to the workload with higher rank value. Once the scheduling of workloads is done based on priorities then each one is mapped to the resource which completes the execution at earliest time.

Mazandarani, Ayda and Momeni, Hossein [104] proposed a QoS-aware Scientific Application Scheduling Algorithm (QSASA)based on HEFT algorithm. The main aim of the study is to schedule scientific workflow on target cloud resources based on users preferences using User preference Fitness Function (UPFF). This approach minimized the total execution cost of scientific application workflows.

Zhangjun Wu *et al.* [105] proposed a hierarchical scheduling approach which combined service and application level scheduling. Service level scheduling mapped the tasks to services, while workload level scheduling allocated the tasks to VMs in cloud data centers.

To address the power consumption problem for high performance systems [106] presented a framework for direct, automatic profiling of power consumption for non-interactive, parallel scientific applications on high-performance distributed systems. The study is conducted using NAS parallel benchmarks on a 32-node Beowulf cluster.The finding suggests smart schedulers could be used to optimize for energy while maintaining performance.

A cost based scheduling algorithm was proposed by Jia Yu *et al.* [107] which reduce the

execution cost while meeting the deadline constraint for delivering results. By rescheduling unexecuted workloads this algorithm can adjust the delays of service accomplishments.

Meng Xu *et al.* [108] used various loosely coupled scientific applications and several QoS. An approach to manage multiple workflows with different QoS requirements was proposed and executed. This approach improved the access speed for scheduling, decreased the makespan and cost of executing scientific applications.

Rizos Sakellariou *et al.* [109] proposed a straightforward Directed Acyclic Graph (DAG) model for scientific applications that schedules the nodes of DAG onto resources. The objective of the work was to optimize the overall time while taking into consideration the budget constraint. To conclude the above literature survey, most of the work done in resource allocation area of Cloud is static and that too for homogenous compute nodes. Table 2.3 summarizes the existing scheduling approaches in cloud environment.

Table 2.5: Existing Scheduling Approaches

Approach	**Scheduling Criteria**	**Scheduling Aspects**	**Problem Discussed**	**Tool**
Task Scheduling algorithm for improving cost[110]	Cost, Performance	Unscheduled group of tasks	Measures both resource cost and computation performance, Improves the computation ratio	CloudSim
Scheduling Transaction-Intensive Cost-Constrained Cloud Workflows [111]	Execution cost and time	Workflows with large number of instances	To minimize the cost under certain user,designated deadlines,Enables the compromises of execution cost and time	SwinDe W-C
A Compromised Time-Cost Scheduling Algorithm [112]	Cost and Time	An array of workflow instances	It is used to reduce the cost and time	SwinDe W-C

to be cont'd on next page

Table 2.5: Existing Scheduling Approaches (cont.)

Approach	**Scheduling Criteria**	**Scheduling Aspects**	**Problem Discussed**	**Tool**
A Particle Swarm Optimization-based Heuristic for Scheduling [93]	Resource Utilization, Time	Group of tasks	It is used for saving cost to three times as compared to existing, It is used for good distribution of workload onto resources	CloudSim
SHEFT Workflow Scheduling Algorithm [113]	Execution time, Scalability	Group of tasks	It is used for optimizing workflow execution time, It also enables resources to scale elastically during during workflow execution	CloudSim
Market-oriented Hierarchical Scheduling Strategy [105]	Makespan, Cost, CPU time	Service level scheduling, Task level scheduling	The overall running cost of Cloud workflow systems will be minimized, It can be used to optimize both makespan and cost simultaneously	SwinDe W-C
Multiple QoS Constrained Scheduling Strategy of Multi-Workflows [108]	Scheduling success rate, cost and makespan	Multiple workflows	It is used to schedule the workflow dynamically, It is used to minimize the execution time and cost	CloudSim
Optimal Workflow-Based Scheduling Algorithm [114]	CPU Utilization and Execution time	Multiple workflows	It is used to find solution that meets all user preferred QoS constraints, It is used to improve CPU utilization	CloudSim
RASA Workflow Scheduling [109]	Makespan	Grouped Tasks	It is used to reduce makespan	GridSim

The literature review shows that tremendous work has been done for scheduling the resources but still there is a huge scope to schedule the resources for scientific applications

as the existing approaches are not dedicated towards scientific applications. Moreover, most of the approaches have not taken prediction as a base for scheduling. Efficient scheduling can reduce the execution time, cost and SLA violations. So there is a need to develop new solutions to handle the scheduling problems.

2.4 Gap Analysis

Based on the literature survey following gaps have been identified:

- Due to the large size datasets of scientific applications, there is need for large computing capabilities [50].
- Non functional information such as resource consumption information would lead to more rapid (reduced execution time) or less energy consuming (save cost) execution of scientific applications [50].
- Scientific applications have fluctuating resource demands which can be handled effectively when deployed on cloud [115]. Cloud Computing provide on demand service to users which guarantees convenience and effectiveness.
- Effective utilization of resources in cloud is very complex [116]. A uniform scheduling method is needed for scientific applications to solve resource utilization issue [117] . Efficient prediction of resources can lead to effective scheduling. Hence, there is a need to design an efficient resource prediction technique which can predict a requisite set of resources for future and optimize resource deployment [66].
- To enhance scheduling efficiency for scientific applications of different size, it is important to have prior information of the resource requirements for executing the applications [64].
- Scheduling [93] can be described as the mapping and execution of the workload of cloud users based on the resource prediction outcomes of the application. Considering the large execution time and cost of resources required by scientific applications, resource scheduling has become a research challenge in cloud [116]. Efficient scheduling [64] can reduce the execution time, cost and power consumption taking into consideration QoS factors. So there is a need to develop new solutions to handle the scheduling problems.

- Effective scheduling can help in improving the application efficiency, reduce the cost, increase the resource utilization and minimize the SLA violations [116] [117] .
- Hybridization of existing scheduling algorithm with other metaheuristic optimization techniques should be explored. This can enhance the scheduling effectiveness for scientific applications [96].

2.5 Problem Statement

Cloud computing has emerged as a conducive technology for the scientists to run complex applications as the research community is able to access on-demand computational resources within a short span of time instead of experiencing peak demand bottlenecks. Executing complex scientific experiments necessitates considerable number of high performance computing resources, huge data storage space, *etc.*, which involve enormous capital investment. Moreover, the increasing requirements of scientific applications demand systems which can handle the fast computation of voluminous datasets while optimizing the available resources. Hence, there is a need to develop an approach which can predict the future resource requirements of the applications and schedule them efficiently to meet the QoS requirements of the scientific users by taking the SLA violations into consideration.

In this work, a resource prediction and scheduling approach would be proposed and developed for predicting and scheduling the resources in the cloud as per the requirements of the deployed scientific application. The aim of the work is to optimally utilize the cloud resources and reduce the overall cost, execution time and SLA violations of the scientific applications in the cloud environment.

The broad objectives of this research work are:

1. To study and analyze existing resource prediction and scheduling techniques for scientific applications.
2. To design and implement an efficient resource prediction technique for optimal utilization of resources.
3. To develop a scheduling approach based on the proposed prediction technique to reduce cost, execution time and SLA violations of scientific applications.
4. To validate the proposed approach in Cloud environment.

2.6 Conclusion

This chapter has presented the prevailing scientific applications and their associated QoS requirements. It has also focused on exploring the existing resource prediction and scheduling techniques for cloud-based scientific applications. From the literature survey, the gaps have been analyzed and the research problem has been formulated.
The next chapter proposes a solution to the research problem by proposing a regressive ensemble approach for predicting resource usage in cloud computing. The proposed technique addresses the gaps identified in the literature survey and fulfills the objectives of the research work.

Chapter 3

REAP: Proposed Resource Usage Prediction Technique

Cloud computing has become a prime infrastructure for scientists to deploy their scientific applications as it offers a parallel and distributed environment for largescale computations. During deployment, prediction of resource usage is essential to achieve optimal scheduling of scientific applications. The existing resource prediction models fail to provide reasonable accuracy owing to the high variances in cloud metrics. Therefore, to handle the changing cloud resource demands, it is necessary to accurately predict the future requirements for automatically provisioning the resources. In this chapter, a Regressive Ensemble Approach for Prediction (REAP) is proposed, which integrates the feature selection and resource usage prediction techniques to achieve high performance. The effectiveness of the proposed approach is evaluated in the cloud environment by conducting a series of experiments.

Initially, a scientific application "Cybershake" is deployed and a dataset is generated by performing an extensive number of experiments. Further, feature selection is performed using a meta-heuristic approach to reduce the size of the dataset without affecting the output. Finally, a machine learning based ensemble approach is applied to predict the resource usage. The proposed approach outperforms the existing models by significantly improving the accuracy while reducing the prediction time and error rate.

3.1 Objectives of the Proposed Technique

The cloud computing environment offers a customizable infrastructure in which applications can provision the required resources prior to execution. The method's elasticity and pay-as-you-go pricing model can reduce the cost for client's [118, 119]. The innumerable services offered by the cloud providers and the extravagent developments in the domain have attracted many scientists to deploy their applications on cloud. The change in number of tasks involved in the scientific application directly affects the demand of cloud resources. Hence, to handle the fluctuating demand, there is a need to manage the resources in an efficient manner.

Efficient management [120] can improve the utilization of resources in cloud environment, enhance the application's performance and reduce the resource usage cost. For effective resource management, it is significant to predict the usage of cloud resources such as CPU and memory. A considerable amount of literature has been published on resource prediction as discussed in Section 2.2 of Chapter 2 but numerous challenges still exist.

a) To execute high performance scientific applications, a parallel and distributed environment such as cloud is needed. Cloud computing offers tremendous amount of on-demand resources to meet the computational requirements of the scientific applications[22].

b) Unlike Grid and HPC structures, cloud systems engage a variety of users. Each customer connection happens at a specific time with distinct resource requirements and time lengths. These varying interactions exert particular challenges for cloud useful resource usage, such as the mapping of tasks and resources[24].

c) Allocation of several requests to a VM leads to resource under-provisioning, which degrades the performance of an application allocated to it as the requested resources are more than the available ones. Hence, a prediction model performing on the basis of resource usage history can notify the resource manager that the VM may experience an under-provisioning situation shortly[64].

d) With a decline in the disk storage cost, the dimensions of the databases have escalated. To process large datasets, there is a need to minimize the number of features because each one adds to the manufacturing cost as well as the running time of a system. Feature selection reduces the size of the dataset by opting for the most relevant ones in the database[26].

e) The existing prediction models do not provide satisfactory accuracy because of high variances in the cloud metrics. Therefore, an ensemble approach that combines the different prediction models can help in the efficient management of cloud resources[27, 28].

To address the above-mentioned challenges, an ensemble approach is proposed in this chapter, which predicts the resource usage for real-world scientific applications such as "Cybershake" and "Floodplain" in the cloud environment. The primary objectives of the proposed prediction technique are summarized below:

a) The proposed ensemble framework predicts the resource usage for scientific applications in the cloud environment.

b) Owing to dataset unavailability, an intensive experimentation is being conducted by simulating the scientific applications on WorkflowSim to generate resource usage dataset.

c) A meta-heuristic genetic algorithm based on greedy approach is used for selecting the relevant features since it reduces the size of the data by eliminating the unimportant and redundant ones.

d) The proposed REAP of resource usage for scientific applications is evaluated under the cloud environment, which improves the accuracy rate and curtails the prediction time and error rate.

The next section illustrates the key traits of the proposed prediction technique.

3.1.1 Key Traits of REAP

The proposed Regressive Ensemble Approach for Prediction (REAP) tends to perform the prediction of resource usage for scientific applications in cloud environment. The key traits of REAP are mentioned below:

- The proposed prediction framework takes into consideration the varying cloud resource demands.
- It facilitates the cloud users to get an estimation of the resources for executing tasks which results in understanding the workload needs of an application.
- The proposed framework predicts the usage of resources inorder to improve the prediction accuracy, reduce the prediction time and error rate.

After illustrating the key features of REAP, the following section presents the QoS requirements taken into consideration by the proposed prediction technique.

3.1.2 QoS Requirements of the Proposed Prediction Technique

The end users states the QoS requirements to the Cloud Service Providers (CSPs) in the form of Service Level Agreements (SLAs). The SLA clearly defines the limitations for the given QoS requirements and the associated penalities for CSPs incase of SLA violations. It is the responsibility of the CSPs to make sure that an appropriate amount of resources are allocated to an application inorder to fulfill the users demands and minimize the SLA violations. This will help the CSPs to retain their customers and increase the revenue. The QoS parameters considered in this research work are given below:

- **Accuracy:** Accuracy can be defined as the degree to which the results of a specific calculation or measurement are clustered around the correct value.
- **Prediction Time:** It is the total time taken by a model for prediction of usage of resources.
- **Error Rate:** The error rate is calculated by measuring the difference between actual and predicted average CPU and memory usage.

3.2 Proposed Resource Usage Prediction Technique

In this work, the proposed solution has been designed to predict the resource usage (CPU and memory), for effective scheduling of the resources. The proposed ensemble approach takes into account scientific applications like "Cybershake" and "Floodplain" for predicting usage of resources in a cloud environment. Initially, scientific applications are executed and resource usage dataset is collected. Further, data cleansing is done by removing the missing values and relevant features are selected by deploying Genetic Algorithm. Next, the machine learning regression models are combined using wrapper method to predict the usage of resources.

After a brief review of the proposed ensemble approach, the following section presents the design methodology that has been procured to effectuate this approach.

3.2.1 Design Methodology

The motive of proposed approach is to design an ensemble approach to predict the usage of resources for a scientific application. This section details the entire execution flow of the proposed approach. The entire process is divided into seven phases as shown in Figure 3.1.

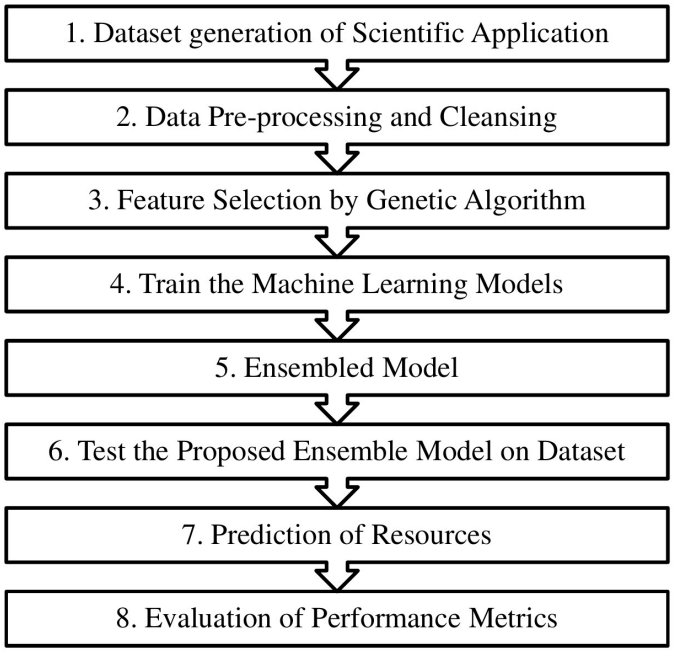

Figure 3.1: Flowchart of the Proposed Approach

Intially, cybershake is deployed on WorkflowSim and after performing a series of experimentation a regression based dataset of resource usage dataset is generated. Further, the generated dataset is pre-processed and cleaned inorder to remove all the null values. Also, the alpha-numeric values are converted to numeric values so that machine learning could be applied easily. Next, genetic algorithm is applied on the dataset to select the relevant set of features. This reduces the overall size of dataset by eliminating the redundant features. Then, the machine learning regression models are trained on the finalized dataset. The entire dataset is divided into two equal halves so that proper training and testing can be performed. Further, the models are grouped as per the proposed algorithm discussed in Section 3.2.2. The proposed model is trained on the train dataset and tested on the test dataset, so that its performance could be recorded for final evaluation. Finally, the performance of resource prediction models and the proposed ensemble model is evaluated on cloud environment. The performance is analysed on the basis of Root Mean Square Error(RMSE), Accuracy(%) and Total Time(ms).

3.2.2 Proposed Framework

The proposed ensemble framework is depicted in Figure 3.2 which is divided into two modules. First module includes a scientific application, Cybershake, which is taken as one of the case study for this research work. In this module, the application is deployed on WorkflowSim and a resource usage dataset is generated by conducting an extensive amount of experimentation. The second module comprises of feature selection and resource usage prediction using machine learning models.

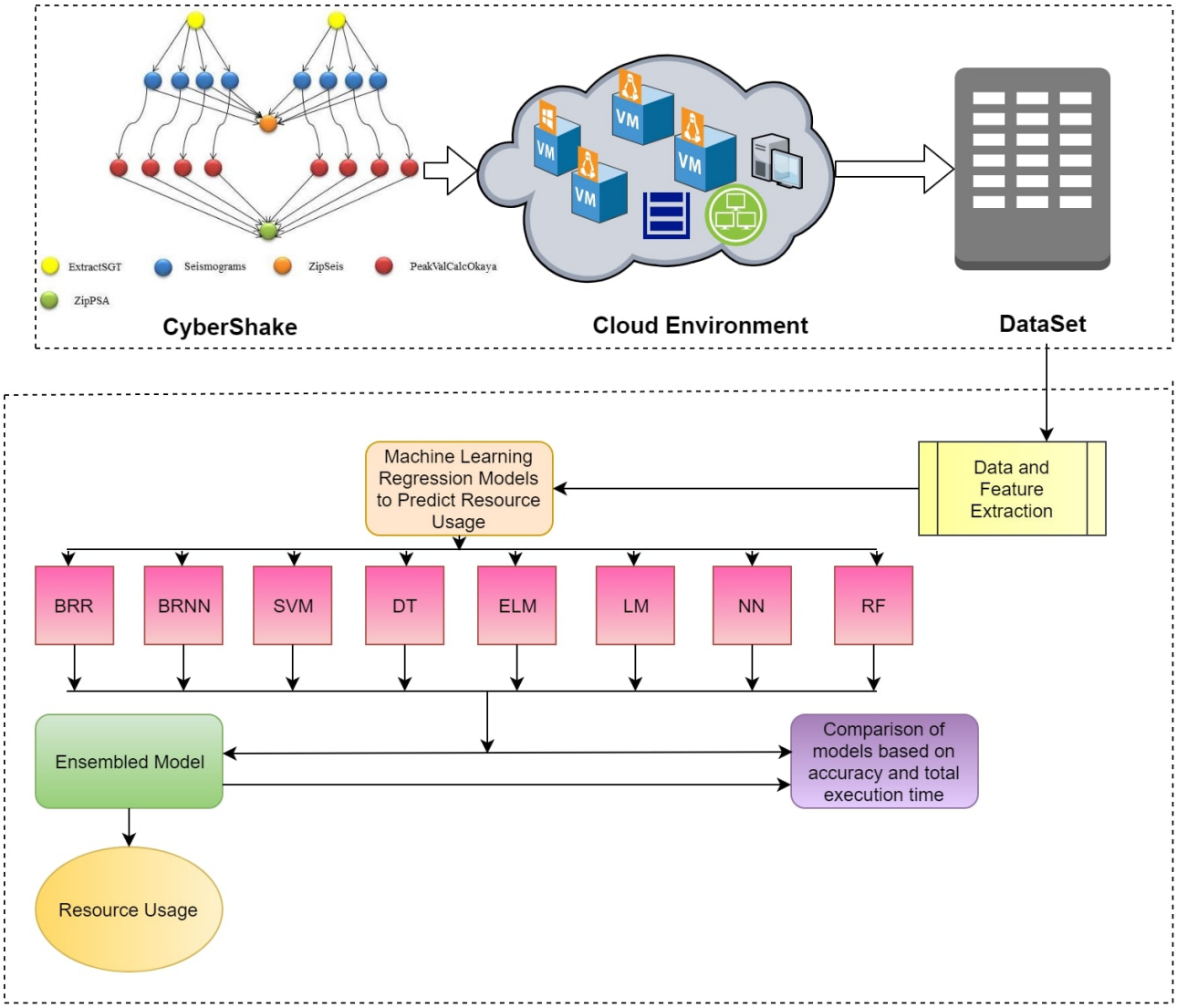

Figure 3.2: Proposed Ensemble Framework

- **Scientific Application "Cybershake" as a Case Study**
  CyberShake is a physics-based computational approach to probabilistic seismic hazard analysis (PSHA) [121]. The CyberShake approach uses full 3D wave propa-

gation simulations to forecast ground motions that will be produced by specific ruptures which is expected to produced significantly more accurate estimates for many sites than commonly used empirical-based ground motion decay attenuation relationships. The structure of cybershake is illustrated through Figure 3.3.

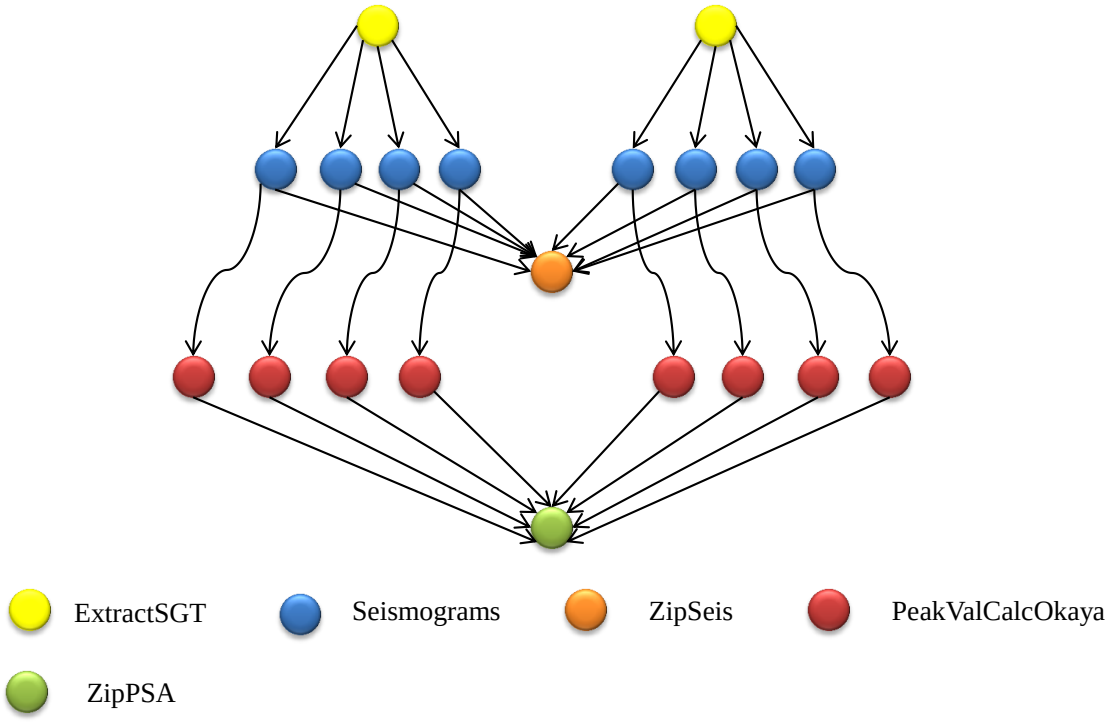

Figure 3.3: Structure of CyberShake

To calculate the probability of future earthquake shaking at geographical sites, Cybershake ground-motion model incorporates simulations conditional on fault rupture. CyberShake represents the aleatory variability in wave excitation through conditional hypocenter distributions and conditional slip distributions. It characterizes the epistemic uncertainty in the wavefield calculations in terms of alternative 3D seismic-velocity models [122]. Cybershake hazard model which involves computation of nealry 240 million synthtic seismograms is deployed in Los Angeles region. These calcualtions are made feasible by developing clever algorithms based on seismic reciprocity and highly optimized anelastic wave propagation codes, but they still strain the capabilities of worlds fastest supercomputers, which are currently operating at petascale. Figure 3.4 depicts the working process of CyberShake.

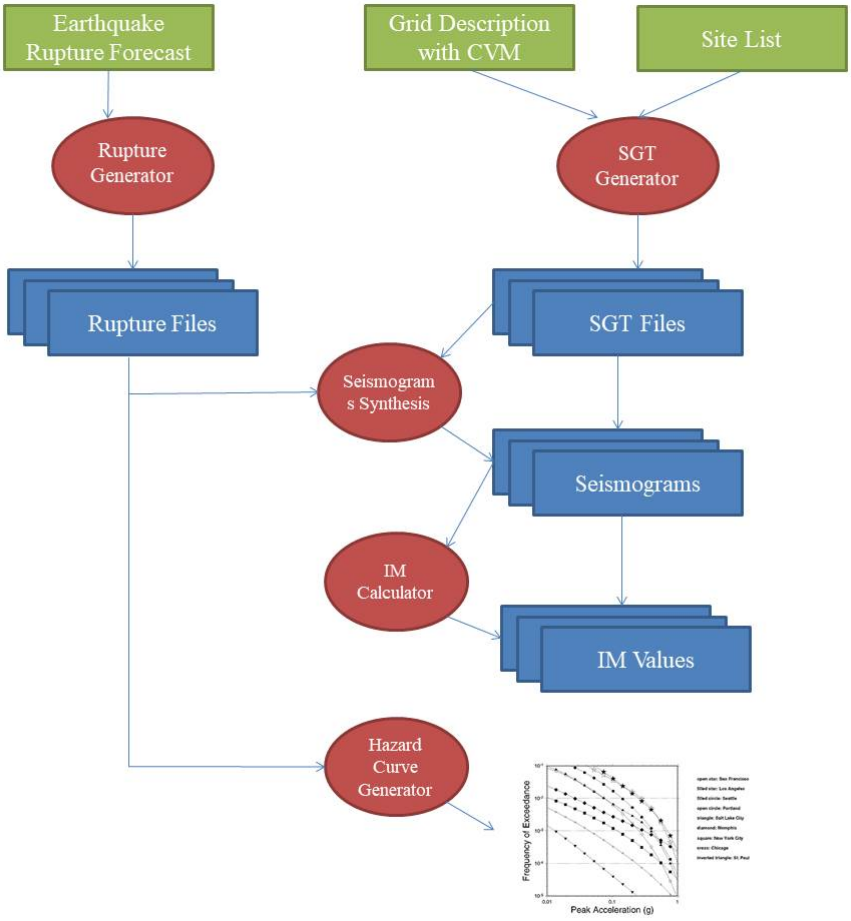

Figure 3.4: Cybershake Working Process

The key features of CyberShake application are:

a) Utilizes 3D simulations and finite fault rupture description

b) Intense Computational demands

c) Require parallel algorithms

d) Require high throughput

The CyberShake application was obtained from workflow gallery- Pegasus WMS [123]. The total size of the dataset was 16GB with estimated number of tasks equivalent to 1,93,000. Various jobs included in the application are Strain Green Tensor (SGT) Files, Seismograms, ZipSeis, PeakSA and ZipPeakSA. ExtractSGT stands for Extract Strain Green Tensors which means to generate tensors from the simulations being performed. The value of Seismograms is calculated from the ExtractSGT data for every predicted rupture and further these values are combined to one value namely ZipSeis. Every Seisomgram task has one PeakValCalcOkaya

job, the values of PeakValCalcOkaya are stored as ZipPeakSA. The description of Cybershake jobs along with their average CPU and memory usage requirements is illustrated using Table 3.1.

Table 3.1: Resource Usage Requirements of Cybershake

Job Name	**Description**	**Mean CPU Usage (%)**	**Mean Memory Usage(MB)**
ExtractSGT	Transformation of a body from a reference configuration to a current configuration.	65.82	20.64
Seismogram	It is a graph output by a seismograph. It is a ground motion at a measuring station as function of time. Seismograms typically record motions in three Cartesian axes (x,y and z), with the z axis perpendicular to the earths surface and the x- and y- axes parallel to the surface.	72.01	57.19
ZipSeis	Compresses the seismogram files in zip form.	6.83	6.25
Peak Intensity	It is the intensity at the highest point of the peak for a peptite. High peak intensity indicates higher readings and low peak intensity indicates lower readings.	16.89	3.11
ZipPeakSA	Compresses peak intensity readings with respect to seismograms in zip form.	2.89	6.16

The application is deployed on workflowsim, a cloud workflow simulator, for generating the resource usage dataset. The generated dataset comprised of nineteen input features namely version, count, index, name, jobCount, fileCount, childCount, id, namespace, name2, version3, runtime, file, link, register, transfer, optional, type and size. But, some of the features were having redundant values which had no relevance for computing the resource usage of that particular job. Hence, there is a need to eliminate unimportant features as each feature used adds to the processing cost.

Further, a meta-heuristic feature selection technique was applied on the dataset to eliminate the unimportant features.

- **Feature Selection**

  Feature selection has become an important task to be accomplished in various high computing applications. There is a huge variety of techniques already available in machine learning but Genetic Algorithm (GA) solves this task in a natural way [124]. Genetic Algorithm is a greedy algorithm whose fundamental goal is to select the best set of inputs to produce the best output as shown in Figure 3.5.

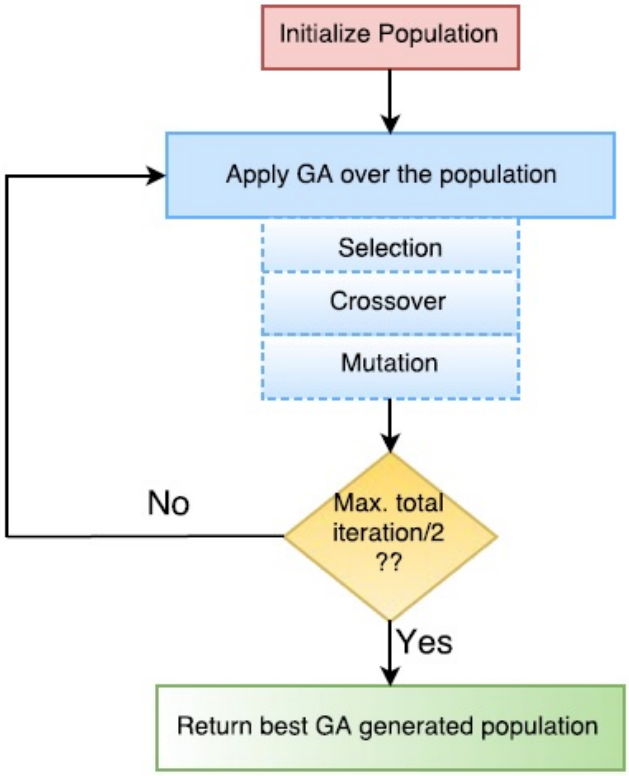

Figure 3.5: Feature Selection Using Genetic Algorithm

Feature selection in GA allows us to extract only those constraints or inputs which have significant role in predicting a desired combination of genes or traits. The procedure of selecting relevant features using GA is explained through Algorithm 3.1 Firstly, the genes of individuals are initialized randomly. Next, the fitness is assigned to each individual using Rank based method as shown in Eq 3.1.

$$\varphi(j) = p.R(j) \qquad j = 1, ..., N \tag{3.1}$$

Here p is a constant variable known as selective pressure, and its value is fixed somewhere in the range of 1 and 2. Greater selective pressure values will make the fittest individuals to have greater likelihood of recombination. The parameter

Algorithm 3.1 Feature Selection Using Genetic Algorithm

Input CyberShake Application features
Output Best set of features
Begin

Initialize Population p randomly
Determine fitness value of Population p
Repeat
Select parents from Population p
Perform Crossover (C_i) on parents generating Population $p+1$
Perform Mutation (M_i) on Population $p+1$
Evaluate the fitness value of Population $p+1$
Until next generation is good than the previous generation
Return best generation
End

$R(j)$ is the rank of individual j. After the assignment of fitness value, selection operator selects the individuals based on their level of fitness to recombine for the next generation. The number of selected individuals is $\frac{N}{2}$, being N the population size. Next, the selected individuals are recombined using crossover operator inorder to generate new population as shown in Figure 3.6.

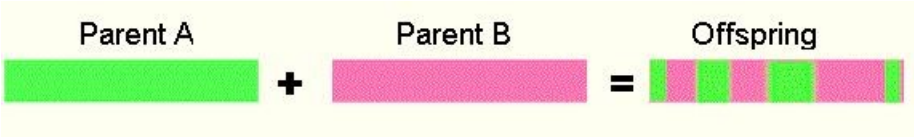

Figure 3.6: Crossover Process of Genetic Algorithm

Here, uniform crossover method is used which selects two individuals randomly and their features are combined to generate four offsprings to form new population. This process is carried on until the size of newly generated population is same as the old one. There is a possibility that offsprings generated through crossover operator are quiet similar to their parents. Therefore, to have diversity in the new population, mutation operator is used. The mutation operator randomly changes the value of some features in the new population as shown in Figure 3.7. In order

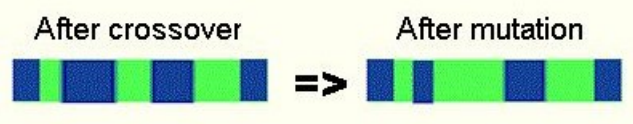

Figure 3.7: Mutation Process of Genetic Algorithm

to decide if a feature will be mutated, a random number is generated between 0

and 1. If this number is lower than mutation rate, that variable is flipped. The mutation rate is usually chosen to be $\frac{1}{m}$; m being number of features. With that value for the mutation rate, one feature of each individual will be mutated. The whole fitness assignment, selection, recombination and mutation process is repeated until a stopping criterion is satisfied.

Table 3.2: Illustration of the Features

Feature	**Description**
Id	The unique id number of the job being executed.
jobCount	Total number of jobs being executed in background.
ChildCount	Total number of child task associated with the task in execution.
name2	This attribute contains 5 names- ZipPSA, ZipSeis, ExtractSGT, SeismogramSynthesis, PeakValCalcOkaya. For processing purpose these names are given a numeric values 1,2,3,4,5 respectively.
Runtime	It is the execution time taken by a particular jobID. The units used to denote runtime is microseconds(ms).
File	Name of the files being executed. Includes 9 different file names- Cybershake_PSA.zip, Cybershake_Seismograms.zip, FFI_0_1_fx.sgt, FFI_0_1_fy.sgt, FFI_0_1_subfx.sgt, FFI_0_1_subfy.sgt, FFI_0_1_txt.variations04327-h00000,Seismogram_FFI_0_1_ID00003.grm, PeakVals_FFI_0_1_ID00003.bsa
Link	It indicates whether the file is an input file or an output file.
Size	Defines the size of the job. The unit used to represent size is bytes.
Memory Usage	Total bytes of memory used by each job during execution.
CPU Usage	The total percentage of CPU being utilized while executing a particular job.

There were total nineteen input features namely version, count, index, name, jobCount, fileCount, childCount, id, namespace, name2, version3, runtime, file, link, register, transfer, optional, type, size and two target features namely CPU Usage and Memory Usage. As per the results, the features which were selected for further processing are Id, jobCount, fileCount, name2, runtime, file, link, size, memory usage and CPU Usage. The description of the selected features is given in Table 3.2. Further, the machine learning models used to predict the usage of resources are discussed below.

- **Machine Learning Models**

  Machine learning is a process of modeling and analyzing learning processes that enhance the system efficiency and improve the performance of an application. Ma-

chine learning can be categorized as supervised, unsupervised and reinforcement learning. Numerous applications of machine learning involve only those tasks that can be employed as supervised. It enables the systems to learn from data, identify the hidden patterns and make decisions with least human interference.

a) **Bayesian Ridge Regression:** Bayesian ridge regression (BRR) [125] model comprises of special cases like t-test and anova. This model was intended to fit parametric regression models utilizing distinctive kinds of shrinkage techniques. BRR is formulated as depicted in Eq 3.2:

$$Y = XB + e \tag{3.2}$$

Here, Y is the dependent variable, X is independent Variable, B is the regression coefficent, e is Error. B is calculated as: $B_{OLS} = (X^T X)^{-}1X^T Y$ [OLS: Ordinary Least Squares] where $X^T X = R$ and R is Correlation Matrix.

b) **Bayesian Regularized Neural Network:** The need for lengthy cross-validation is eliminated or reduced by bayesian regularized neural networks (BRNN) [120, 126]as they are better than standard back-propagation nets. In this mathematical method of Bayesian regularisation, non-linear regression is converted to a well-posed statistical problem in manner of a ridge regression. The formula to compute BRNN is shown in Eq 3.3:

$$Y_i = g(X_i) + e_i = \Sigma_{k=1}^{s} w_k g_k(b_k + \Sigma_{j=1}^{p} X_{ij}\beta_j^{[k]}) + e_i; \qquad i = 1, \cdots, n \tag{3.3}$$

Here, $e_i \sim N(0, \sigma_e^2)$, s is number of neurons, w_k is weight of the kth neuron, k= 1,.,s , b_k is a bias for the kth neuron, k= 1,.,s , $\beta_j^{[k]}$ is the weight of the jth input to the net, j=1,.,p , $g_k(.)$ is the activation function , $g_k(x) = (exp(2x) - 1)/(exp(2x) + 1)$, The software will minimize $F = \beta E_D + \alpha E_w$ where $E_D = \Sigma_{i=1}^{n}(y_i - \hat{y}_i)^2$ Error sum of squares, E_w is the sum of squares of network parameters (weights and biases), $\beta = 1/(2\sigma_e^2)$
$\alpha = 1/(2\sigma_\theta^2), \sigma_\theta^2$ is a dispersion parameter for weights and biases.

c) **Neural Network:** Neural networks (NN) are distinguished under various types such as artificial neural network, recurrent neural network, recursive neural network and so on. These are statistical learning models which deal with neurons similar to biological neural networks[120]. These neurons are interconnected to each other which send message to each other and the values are calculated using supervised or unsupervised learning. The connections within the network can be systematically adjusted based on the inputs and

outputs and we can process them using various propagation techniques. NN is formulated as shown in Eq 3.4:

$$P_j(t) = \Sigma_i^n O_i(t) w_{ij} \tag{3.4}$$

Here, $P_j(t)$ is input to the neuron j, $O_i(t)$ is output of the predecessor neurons (used to calculate P_j), i is the number of neurons in a level, w is the assigned weight and t is the value of the neuron.

d) **Support Vector Machine:** Support Vector machine (SVM) [74] is a machine learning algorithm which is used for both regression and classification problems [127, 128]. The main principle used is optimal separation. The one which is a good classifier is the one with the maximum distance between data points of different classes. The output of the algorithm is a hyperplane which is used for categorising new data. SVM is formulated as depicted in Eq 3.5:

$$min_{\alpha,\alpha^*} \frac{1}{2}(\alpha - \alpha^*)^T Q(\alpha - \alpha^*) + Z^T(\alpha_i - \alpha_i^*) \tag{3.5}$$

Subject to $0 \leq \alpha_i, \alpha_i^* C$;
i=1,$\cdots$,l
$e^T(\alpha - \alpha^*) = 0$
$e^T(\alpha + \alpha^*) = C_v$
Here, e is the unity vector, C is the upper bound, Q is l by l positive semidefinite matrix, $Q_{ij} = y_i y_j K(x_i, x_j), Kx_i, x_j = \theta(x_i)^T \theta(x_j)$

e) **Decision Tree (Regression Tree):** Decision tree (DT) [129, 130] classifies instances by starting at the root of the tree and moving through it till a leaf node. We will calculate the probability of occurrence of all the events at each level using ID3 algorithm. This consists of a decision node which specifies each attribute, edge which splits one attribute into many. We also have a leaf node which tells us about the target attribute and its probability of occurrence. We also have a path which specifies the attributes to make a final decision.
For the given data : $y \in R^n, x \in R^{n*p}$; each observation $(y_i, x_i) \in R_{p+1}$; i = 1,$\cdots$, n. Suppose we have partition of R_p into m regions $R_1, \cdots, R_m$. We predict the response using a constant on each R_i. The formulation for DT is

shown in Eq 3.6:

$$f(x) = \Sigma_{i=1}^{m} C_i.1_(x \in R_i) \tag{3.6}$$

Inorder to minimize $\Sigma_{i=1}^{n}(y_i - f(x_i))^2$one needs to choose: $(\hat{C_i}) = ave(y_j : x_j \in R_i)$. Consider splitting variable $j \in 1, , p$ and splitting point $S \in R$. Define two half planes: $R_1(j, s) = \{x \in R^p : x_j \leq s\}$ and $R_2(j, s) = \{x \in R^p : x_j > s\}$

f) **Extreme Learning Machine:** This model[131] is a learning algorithm for the single hidden layer neural networks used in classification and regression[132]. The ELM used for single hidden layer feedforward neural network training can adaptively set the hidden layer node number and it can randomly assign the input weights so that output layer weights obtained by the least square method, the whole learning process completed with very little error(minimum number of error). The training speed compared with the traditional BP algorithm based on experiments is improved. For N arbitrary distinct samples $(x_i, t_j) \in R^n * R^m$
Standard SLFNs with L hidden nodes and activation function g(x) are mathematically modeled as $H\beta = T$ which is equivalent to Eq 3.7,

$$\Sigma_{i=1}^{L} \beta_i G(a_i, b_i, x_j) = t_j; \qquad j = 1, \cdots, N \tag{3.7}$$

Here, a_i is the input weight vector connecting the ith hidden node and the input nodes, β_i is the weight vector connecting the ith hidden node and the output node, b_i is the threshold or impact factor of the ith hidden node and H is hidden layer output matrix.

g) **Linear Regression Model:** Linear Regression (LR) is the most commonly used category of predictive analysis [125]. It is used to show relationship between two or more variables where there are two types of variables one is dependent and the other is explanatory. LR is formulated as depicted in Eq 3.8:

$$Y = \beta X + A + e \tag{3.8}$$

Here, β is slope of the line (predicted increase or decrease for Y scores for each unit increasing X), X is the independent variable, A is Y-intercept (level of Y when X is 0) and e is the random error term.

h) **Random Forest:** Random forest (RF) [22]is a learning method that works by developing a huge number of choice trees at time of training and outputs the mean forecast (regression) of the individual trees. The formula for computing

RF is shown in Eq 3.9:

$$h_k(X) = h(X|\theta_k); \qquad k = 1, .., n \tag{3.9}$$

Here, θ_k: are independent identically distributed random vectors, X is input variable, n is the number of trees and $h = \{h_1(X), .., h_k(X)\}$ ensemble of classifiers.

The methods are available in *R* open source software [133] which is licensed under GNU GPL. To obtain better results, parameters of the models need to be tuned. The brief detail of the methods with the required packages and their tuning parameters is described in Table 3.3.

Table 3.3: Machine Learning Regression Models

Model Name	**Method Used**	**Package Required**	**Tuning Parameters**
BRR	bridge	monomvn	T = 1000, lambda2 = 1
BRNN	brnn	Brnn	neurons=2, mu=0.005, mu_dec=0.1, mu_inc=10, mu_max=1e10,min_grad=1e-10
SVM	ksvm	kernlab	kernel="rbfdot", type="nu-svr"
DT	rpart	None	usesurrogate=0, maxsurrogate=0
ELM	elm	elmNN	nhid=10, actfun="sig"
LM	lm	None	None
NN	nnet	Nnet	maxit=100, MaxNWts=10000
RF	randomForest	randomForest	ntree=500, mtry=2

The selected machine learning models are further assembled using the proposed ensemble algorithm which is discussed in the following section.

- **Proposed Ensemble Algorithm**
  Ensembling is the process of stacking multiple machine learning models and improve the prediction accuracy or decrease variance, by combining the capabilities of mod-

els. The machine learning regression models are applied to generated dataset for predicting resource usage. These models are further grouped based on the proposed ensemble algorithm 3.2.

Algorithm 3.2 Proposed Ensemble Model Algorithm

Start

Set BestAcc = 0
Set BestMSet= NULL
Set ModelList=$[m_1, m_2, m_3, m_4 \ldots m_n]$
Set pd= PredictedDataSet
Set Actual= pd[1]
for each i in 1,...,n do
$Setx \leftarrow rand(m_2 : m_n)$
$Sets \leftarrow sample((m_1, m_n), x)$
$e \leftarrow ensemble$(s)
$acc \leftarrow mean(e == pd[, 1]) * 100$
if $acc > BestAcc$ then
$BestAcc \leftarrow acc$
$BestMSet \leftarrow s$
end if
end for each
return BestAcc
return BestMSet

Stop

In the proposed algorithm, a variety of combinations are formed for different models and mean accuracy (*acc*) is calculated for each combination. The computed accuracy rate is further compared with the best accuracy (*BestAcc*) already generated. If the calculated *acc* is better than *BestAcc* then *BestAcc* is replaced with the calculated *acc* and the provided combination of models is returned as the best model set to ensemble. The primary focus of proposed algorithm is to generate a best set of models which can be assembled to enhance the performance of regression models for predicting the usage of resources. The proposed framework aims at prediction of cloud resources for scientific applications which results in improved accuracy and reduced prediction time. Further, the time complexity of the proposed algorithm is discussed and compared with existing algorithms.

3.2.3 Time Complexity

Time complexity describes the overall time required for the execution of an algorithm. This section describes the time complexity of proposed ensemble algorithm 3.2. The first step (line 2) initializes variable BestAcc (best accuracy) with value 0 and it requires O(1) time. Similarly, first six statements (lines 2-7) are for initialization of variables thus time required is O(1). In next statement (line 8), for loop is initialized for n times, hence, time taken is O(1)*n. The next four statements(line 9-12) in for loop are functions along with assignments, time for each statement with n iterations is O(1+1)*n. Further statements (lines 13-15) are performing comparison and assignments, thus, time required is O(1)*n. Therefore, the computational complexities for all the steps amount to O(n).

The complexity of the proposed algorithm is compared with SaDE, and BPNN [134] for generations (g). The complexity of BPNN is O(g x m x n2) and SaDE is O(g x m x n2 x N) whereas the complexity of proposed algorithm is O(g x n). This proves that the proposed algorithm has less complexity than the already existing algorithms by factor of m, N. Further, the evaluation metrics considered to optimize the performance of the proposed prediction approach is discussed in next section.

3.2.4 Evaluation Metrics

Various parameters such as RMSE, accuracy and prediction time are calculated to evaluate the performance of proposed ensemble model.

- **Root Mean Squared Error(RMSE)**

  The average of the distance between the predicted value and the ground truth is generally referred as root mean squared error. In the current dataset we predict certain output values and the average variance is treated as Root Mean Squared error.

  $$RMSE = \sqrt{\frac{\Sigma_{i=1}^{n}(X_{obs,i} - X_{model,i})^2}{n}} \tag{3.10}$$

  where X_{obs} is observed values or the known results and X_{model} is expected values or the unknown results at time/place i and n is the sample size or number of observations.

- **Accuracy**

  Accuracy can be defined as the degree to which the results of a specific calculation or measurement are clustered around the correct value. It can be calculated as

deviation of predicted resource usage from the actual resource usage as shown in eq3.11.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} * 100 \tag{3.11}$$

Here, TP, TN, FP, FN represent the number of true positives, true negatives, false positives and false negatives, respectively.

- **Prediction Time (ET)**

  Prediction Time (ET) can be defined as the total time taken by the model for prediction. ET can be calculated in terms of total time taken by a query to execute for predicting resource usage. It is formulated as indicated in the equation given below.

  $$ET = T_e - T_s \tag{3.12}$$

  Here, T_e indicates end time for executing data and T_s starting time when execution actually begins.

3.2.5 Experimental Results

The tools used to set up testbed for experiments include RStudio 1.0.143, Netbeans IDE 8.2, CloudSim 3.0, WorkflowSim 1.0, Java SDK 8. R tool is applied for implementing machine learning models using scientific application datasets. WorkflowSim extends the features of CloudSim that facilitates to simulate cloud environment by creating datacentres, hosts, VMs, cloudlets, etc. It has been used to collect the resource usage requirements of scientific applications. Further, four heterogeneous virtual machines are created for parallel execution of scientific application. The performance of proposed resource prediction model has been validated on cloud simulator environment.

Case I: Performance evaluation of machine learning models

An intensive experimentation has been conducted for carefully evaluating the effectiveness of proposed ensemble model. Firstly, the performance of machine learning regression models is measured and then the ensemble model is assessed. The experimental results are shown in Table 3.4. Amongst twenty five machine learning regression models, these eight models were selected as the best model to form an ensemble model based on the proposed algorithm 3.2.

Table 3.4: Performance Evaluation of Machine Learning Models

Model Name	**RMSE**	**Accuracy(%)**	**Total Time(ms)**
BRR	4.64	47.44	1.52
BRNN	4.7	47.44	26.07
SVM	0.4	91.21	1.64
DT	0.53	80.43	0.52
ELM	0.75	93.62	0.43
LM	0.3	89.13	0.44
NN	0.43	84.78	0.51
RF	0.46	78.26	0.52
Proposed	**0.25**	**95.45**	**0.36**

- Root Mean Square Error (RMSE)

  The proposed prediction ensemble model has the least rmse (0.25%) in comparison to existing machine learning models as shown in Figure 3.8. On the other hand, BRR and BRNN with a slight variation has the highest rmse rate (4.64%) and (4.7%). Amongst existing models, LM has the lowest rmse (0.3%) closely followed by SVM (0.4%), NN (0.43%), RF (0.46%), DT (0.53%) and ELM (0.75%).

- Comparison of Actual and Predicted Resource Usage

  To evaluate the performance of proposed ensemble approach, various regression based machine learning models have been used for predicting the resource usage of Cybershake application with different sizes. Figure 3.9 illustrates the comparison of actual CPU usage and predicted CPU usage for Cybershake with 30, 50, 100 and 1000 jobs. The proposed approach REAP predicts the CPU usage more accurately (95.45%) in comparison to existing prediction approaches.

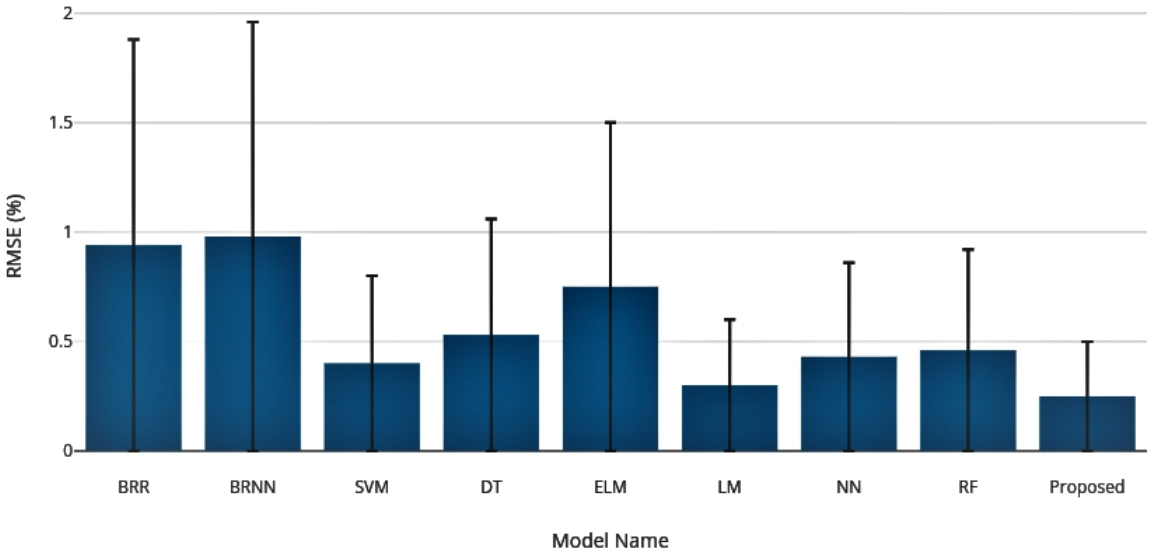

Figure 3.8: Root Mean Square Error

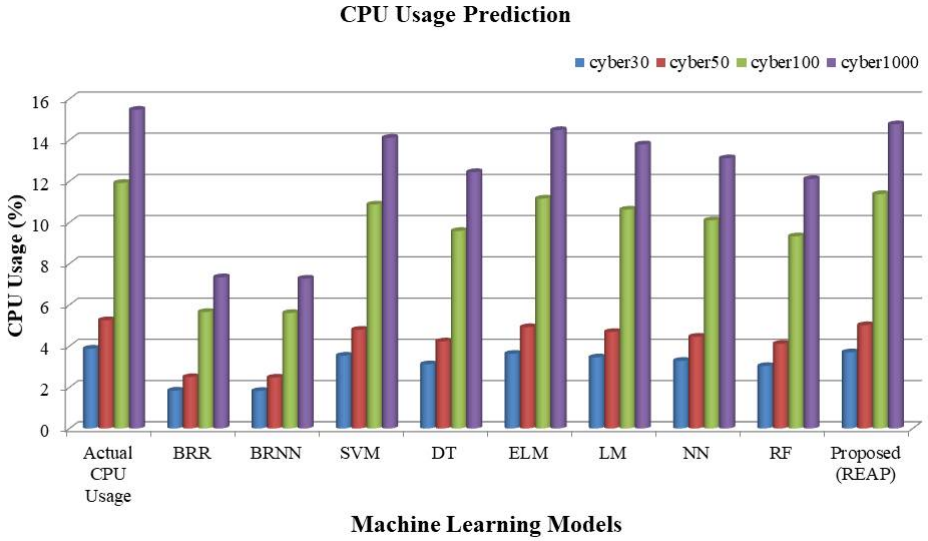

Figure 3.9: Comparison of Actual and Predicted CPU Usage for Cybershake

Similarly, Figure 3.10 compares the actual memory usage and predicted memory usage for Cybershake with 30, 50, 100 and 1000 jobs. The memory usage results of proposed prediction approach REAP are more close to the actual usage results as compared to rest of the machine learning prediction approaches.

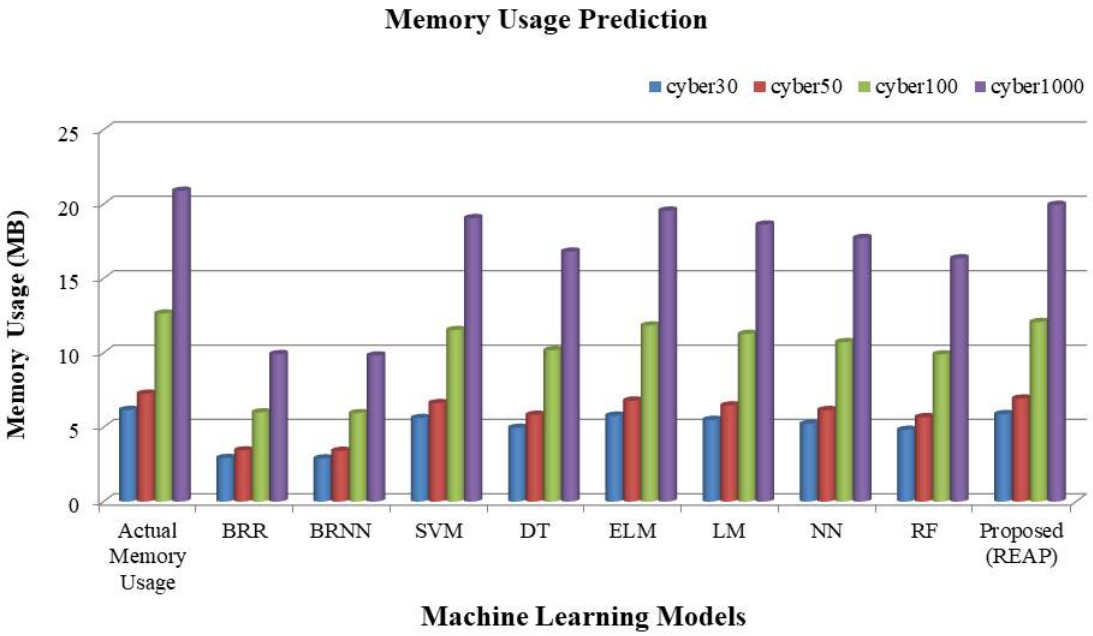

Figure 3.10: Comparison of Actual and Predicted Memory Usage

- Accuracy

  Accuracy is computed as the deviation of predicted resource usage from the actual resource usage. Figure 3.11 illustrates that ELM has the highest accuracy rate (93.62%) whereas BRR and BRNN has the lowest accuracy rate (47.44%) amongst existing machine learning models. Also, SVM has accuracy rate of (91.21%) which is better than LM (89.13%), NN (84.78%), DT (80.43%) and RF (78.26%). After ensembling these models on the basis of proposed ensemble algorithm 3.2, the accuracy rate is incremented to (95.45%) which is an enhancement of approximately 2%.

- Prediction Time

  It is evident from Figure 3.12, the total time taken by the proposed approach to predict the resources is least (0.36 ms) while BRNN has the maximum prediction time (2.607 ms). Further, a substantial decrement is shown by SVM (1.64 ms) and BRR (1.52 ms), followed by DT and RF (0.52 ms). NN predicts the resource usage in (0.51 ms) whereas LM and ELM has almost similar prediction time of (0.44 ms)and (0.43 ms), respectively.

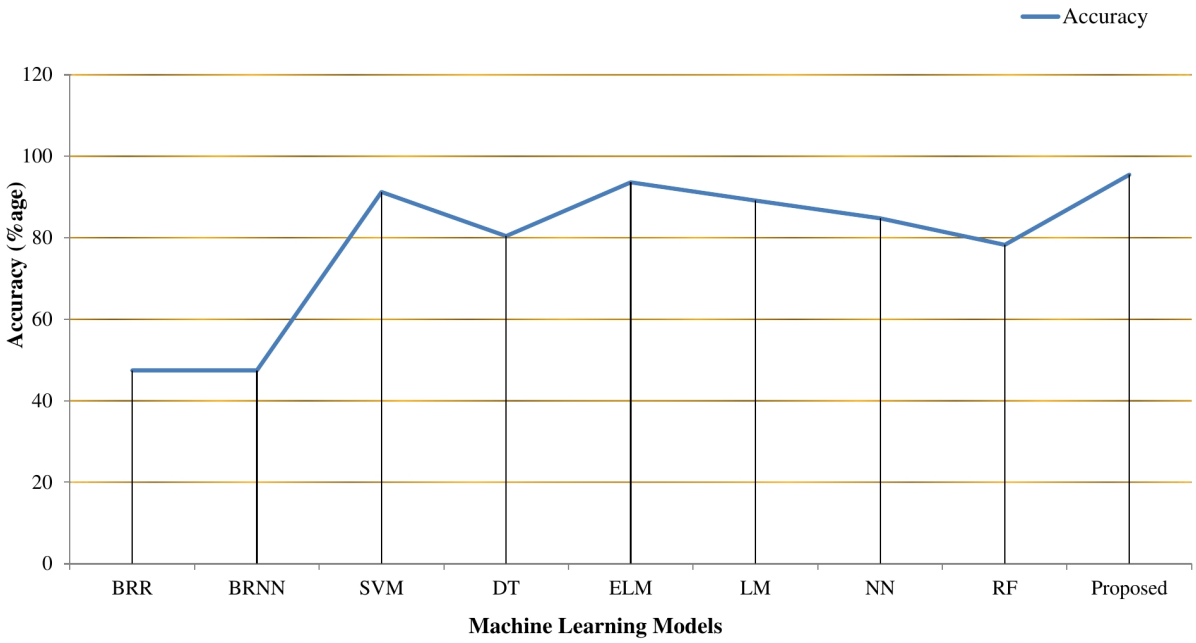

Figure 3.11: Prediction Accuracy

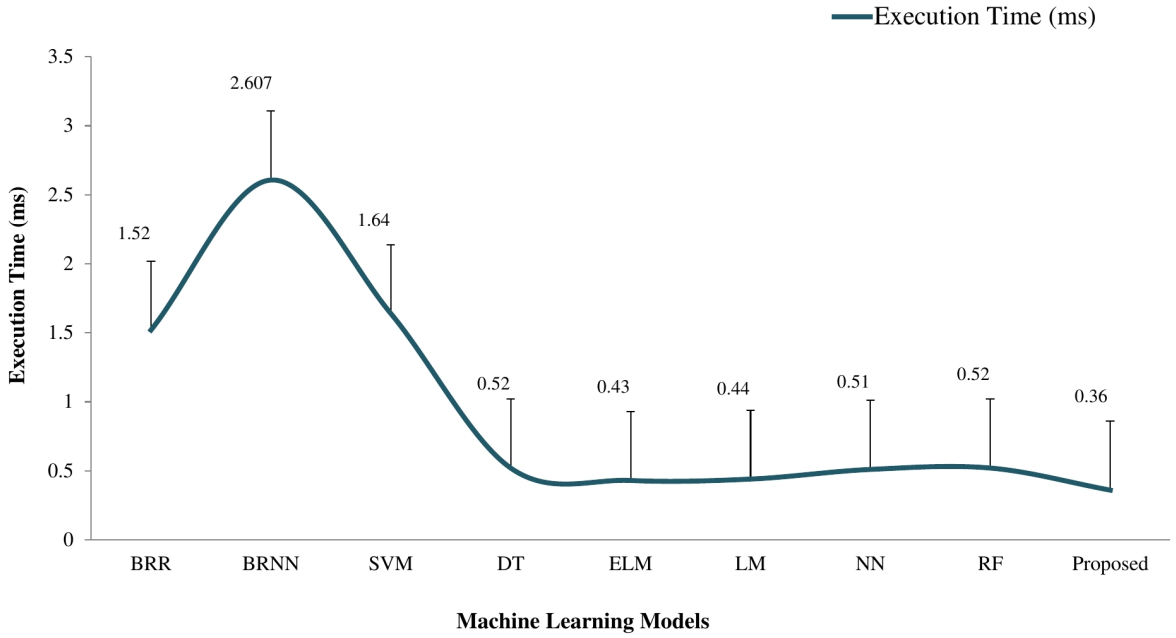

Figure 3.12: Time Taken To Predict the Resource Usage

The results clearly state that the models individually do not exhibit similar performance in regards to any of the evaluation metrics. To be precise, a single prediction model may perform better than the other prediction models in terms of error rate but may have a lower accuracy rate or higher execution time. Taking an example, ELM outperforms

amongst the eight regression models with 93.62% accuracy and takes 0.43ms total execution time but it gives the error rate of 0.75 which is more than the error rate given by five other models namely linear model, support vector machine, neural network, random forest and decision tree. Inorder to enhance the performance, these individual models are combined together on the basis of proposed ensemble algorithm 3.2. The proposed ensemble model gives the best accuracy of 95.45% amongst all with 0.36ms total prediction time as shown in Figure 3.11 and 3.12. The overall accuracy is enhanced by approximately 2% and execution time is reduced by 16.2%. Hence, it is proved that the proposed regressive ensemble approach for prediction (REAP) is better then the existing individual machine learning regression models.

Case II: Validation of the proposed approach

The proposed ensemble approach is validated by comparing it with the existing Learning Automata (LA) based ensemble approach[135] on the basis of error rate. Learning Automata (LA) theory helps in tuning the weights of prediction models. The Single Window (SW) algorithm compares the absolute error of the last prediction with the absolute error of the prediction in the previous time slot. On the other hand, Multiple Window (MW) compares multiple recent prediction accuracies to determine the performance of the prediction models. In the proposed approach the eight prediction models combined based on the proposed algorithm which enhances the overall performance. The comparitive results are shown in Figure 3.13.

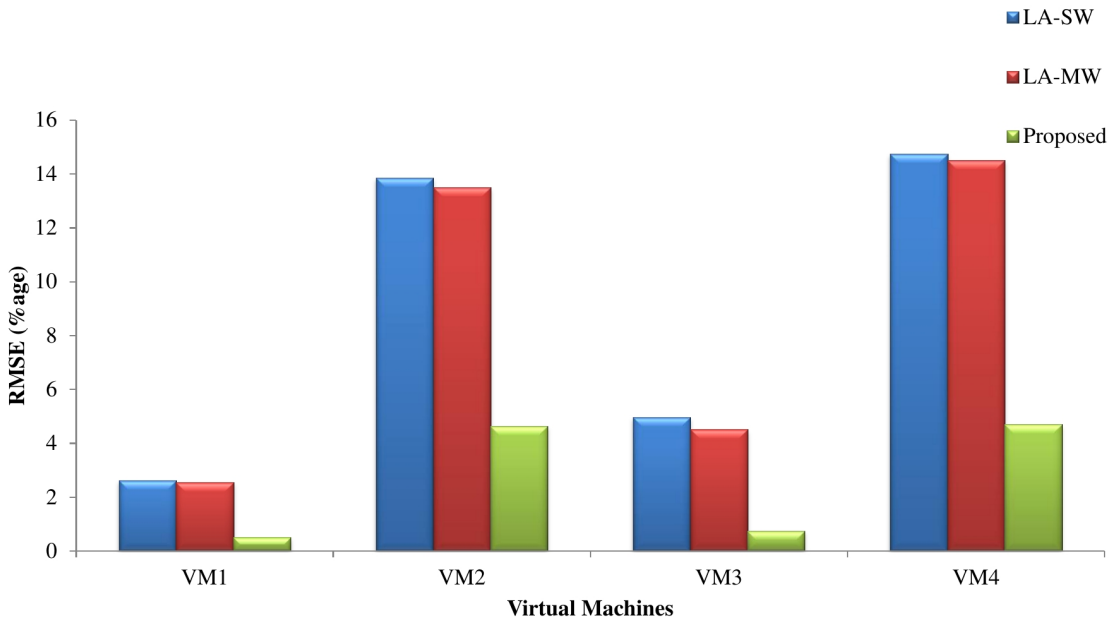

Figure 3.13: Error rate comparison of existing and proposed ensemble approach

LA-SW gave the error rate of 2.62% (VM1), 13.85% (VM2), 4.96% (VM3) and 14.74% (VM4) while using LA-MW the error rate was 2.55% (VM1), 13.51% (VM2), 4.51% (VM3) and 14.50% (VM4). The error rate of the proposed ensemble approach is 0.52% (VM1), 4.64% (VM2), 0.75% (VM3) and 4.7% (VM4). The proposed ensemble approach clearly yields better results than existing ensemble approach on the basis of error rate.

3.3 Conclusion

This chapter discussed in detail that how the resource usage is predicted for a scientific application using proposed Regressive Ensemble Approach for Prediction (REAP). It elaborates the key objectives, traits and QoS requirements of REAP. Further, the design methodology of the proposed ensemble framework is illustrated along with data set description. Also, the feature selection technique and the machine learning models used for prediction are precisely discussed. The performance evaluation for scientific application "Cybershake" have been detailed along with its average resource usage requirements. Finally, the proposed ensemble algorithm for combining the selected machine learning models is presented along with evaluation metrics for validating its performance.

The next chapter explains the scheduling approach for a scientific application and also, illustrates the preciseness of proposed resource usage prediction technique for performing the scheduling.

Chapter 4

RPS: Proposed Prediction Based Scheduling Approach

The previous chapter expatiated the design of proposed regressive ensemble approach for prediction of resources. In this chapter, a scheduling technique based on the predicted resource usage for scientific applications is proposed. Task scheduling (TS) is a multi-objective NP hard optimization problem whose objective is to achieve successful mapping between tasks and VMs by minimizing the execution time, cost and service level agreement (SLA) violations between cloud user and provider.

The scientific applications have dynamic resource demands that affect the resource availability during scheduling. To solve this issue, a resource prediction based scheduling technique has been introduced which automates the resource allocation for scientific application in virtualized cloud environment. The performance of the proposed technique has been evaluated on the basis of execution time, cost and SLA violations.

The motive of this chapter is to design a scheduling algorithm for mapping of tasks scientific applications. The tasks of the application are mapped with the optimal VM using Optimized Prediction-Based Scheduling Algorithm (OPSA).

4.1 Need of Prediction based Scheduling Approach

Cloud systems have gained popularity in hosting complex scientific applications such as montage, cybershake, inspiral, sipht, etc. Applications are deployed on VM's cloud by clients with dedicated resource requirements for performance guarantee and it is recorded in terms of Service Level Agreement (SLA). The workload of VMs varies all the time and some may exhibit weekly or seasonal variability. To guarantee good performance at periods of peak demand, VMs processing capacity is often over-provisioned. This leads to poor scheduling and cloud providers are unable to exploit the benefits out of cloud services. Thus, it poses a challenging scenario for cloud providers to schedule the virtualized resource adaptively in order to handle variable workloads without SLA violation. The significant prediction of resource usage is essential to achieve optimal resource scheduling for cloud computing [130]. In this chapter, to overcome the issues of existing scheduling algorithms an intelligent Resource Prediction based Scheduling (RPS) approach has been proposed.

4.1.1 Problem Formulation

The objective of task scheduling algorithm in this research work is to solve a problem of scheduling n tasks of a scientific application on a set of m heterogeneous VMs to attain certain gaols such as minimizing total execution time, minimizing cost and reducing SLA violations. So, there is a need of an efficient scheduling algorithm which can take into consideration multiple objectives [94]. The following objective problems are taken into account for developing an optimal scheduling algorithm.

- **Total Execution Time:** The time taken by a task to execute on a particular VM is known as execution time. ET_t is the execution time of the tasks running in VMs on the nth node [136] and is defined as Eq 4.1:

$$ET_t = \sum_{m=1}^{v_n} \sum_{k=1}^{j_n} ET_{nmk} \tag{4.1}$$

  Where ET_{nmk} is the execution time for k jobs running on m VMs on the ith node. Hence, the total execution time (ET) is Eq 4.2:

$$ET = \sum_{n=1}^{M} ET_n \tag{4.2}$$

Where M is the total number of nodes.

- **Total Execution Cost:** The cost of executing a task on a VM is computed by Eq 4.3:

$$EC_t = Processingcostperhour * ET_t \tag{4.3}$$

- **SLA Violation (SLAV):** The end users states the QoS requirements to the Cloud Service Providers (CSPs) in the form of Service Level Agreements (SLAs) [137]. It is the responsibility of the CSPs to make sure that an appropriate amount of resources are allocated to an application inorder to fulfill the users demands and minimize the SLA Violations (SLAV). The formula to compute SLAV is given in Eq 4.4:

$$SLAV = \frac{prevRequested - prevAllocated}{prevRequested} \tag{4.4}$$

Here, *prevAllocated* is the total amount of CPU MIPS already allocated to VM and *prevRequested* is the total amount of CPU MIPS requested by the task for execution.

- **Average CPU Utilization:** At any given time, for n^{th} node, the CPU utilization (ACU_n) [136] can be given as Eq 4.5:

$$ACU_n = \sum_{m=1}^{v_n} \sum_{k=1}^{j_n} JCT_{nmk} \tag{4.5}$$

where v_n is the number of VMs running on the n^{th} node and j_n is the number of jobs assigned to v_n VMs. JCT_{nmk} is the CPU utilization of k jobs running on m VMs on the nth node. The CPU utilization in percentage is calculated as Eq 4.6:

$$ACU_n(\%age) = (\sum_{m=1}^{v_n} \sum_{k=1}^{j_n} JCT_{nmk})/TCU_n * 100 \tag{4.6}$$

Where $TotalCPUUtilization(TCU_n) = \frac{ClockCyclesperInstruction * InstructionCount}{ClockRate}$

- **Average Memory Utilization:** At any given time, for nth node, the memory utilization (AMU_n) [136] can be given as Eq 4.7:

$$AMU_n = \sum_{m=1}^{v_n} \sum_{k=1}^{j_n} JMU_{nmk} \tag{4.7}$$

where v_n is the number of VMs running on the n^{th} node and j_n is the number of

jobs assigned to v_n VMs. JMU_{nmk} is the memory of k jobs running on m VMs on the nth node. The memory utilization in percentage is calculated as Eq 4.8:

$$AMU_n(\%age) = (\sum_{m=1}^{v_n} \sum_{k=1}^{j_n} JMU_{nmk})/TMU_n * 100 \tag{4.8}$$

Where TMU_n is the total memory utilization for nth node.

4.1.2 Fitness Value Formulation

In this research work, the problem formulation for minimizing objective criterion mentioned in Eqs. 4.1-4.3 can be optimized. The Relative Closeness Score (RCS) for each task is calculated using multi-criteria decision making algorithm, TOPSIS [138, 139]. This algorithm will enhance the proficiency of task scheduling by supporting multiple objectives. The RCS value computed by TOPSIS is taken as Fitness Value (FV) of the tasks for proposed scheduling algorithm.

FV_{t1}	$=RCS_{t1}$
FV_{t2}	$=RCS_{t2}$
-	-
-	-
-	-
FV_{ti}	$=RCS_{ti}$

where RCS of tasks obtained using TOPSIS is $RCS_t = RCS_{t1}, RCS_{t2}, ..., RCS_{ti}$ which corresponds to the FV of tasks $FV_t = FV_{t1}, FV_{t2}, ..., FV_{ti}$, respectively. The TOPSIS algorithm for computing RCS of tasks is elaborated in next section.

4.2 Resource Prediction based Scheduling (RPS) Approach

Cloud computing offers unlimited resources to it users in the form of services like Infrastructure-as-a-service (IaaS), Software-as-a-service (SaaS) and Platform-as-a-service (PaaS). Virtualization is a key process in cloud computing which segregates the resources of a physical machine to create more than one execution environment and enable the concept of multi-tenancy. These peculiar characteristics of cloud environment lead to certain major

challenges such as load-balancing, fault-tolerance and scheduling. Task scheduling (TS) is a multi-objective NP hard optimization problem whose objective is to achieve successful mapping between tasks and VMs by minimizing the execution time, cost and SLA violations between cloud user and provider.

Inorder to achieve highly effective computations and best Quality of Service (QoS) of cloud, it is vital to perform scheduling of tasks in an efficient manner. The mapping of submitted tasks and VMs is considered to be successful if cloud has attained minimum execution time, cost, SLA violations and maximum utilization of resources. To solve the problem of multi-objective task scheduling, an Optimized Prediction based Scheduling Algorithm (OPSA) has been proposed in this chapter.

4.2.1 Resource Prediction based Scheduling Framework

This section details the framework of the proposed RPS technique as portrayed in Figure 4.1. This framework contains three modules: deployment, prediction and scheduler which are explained further.

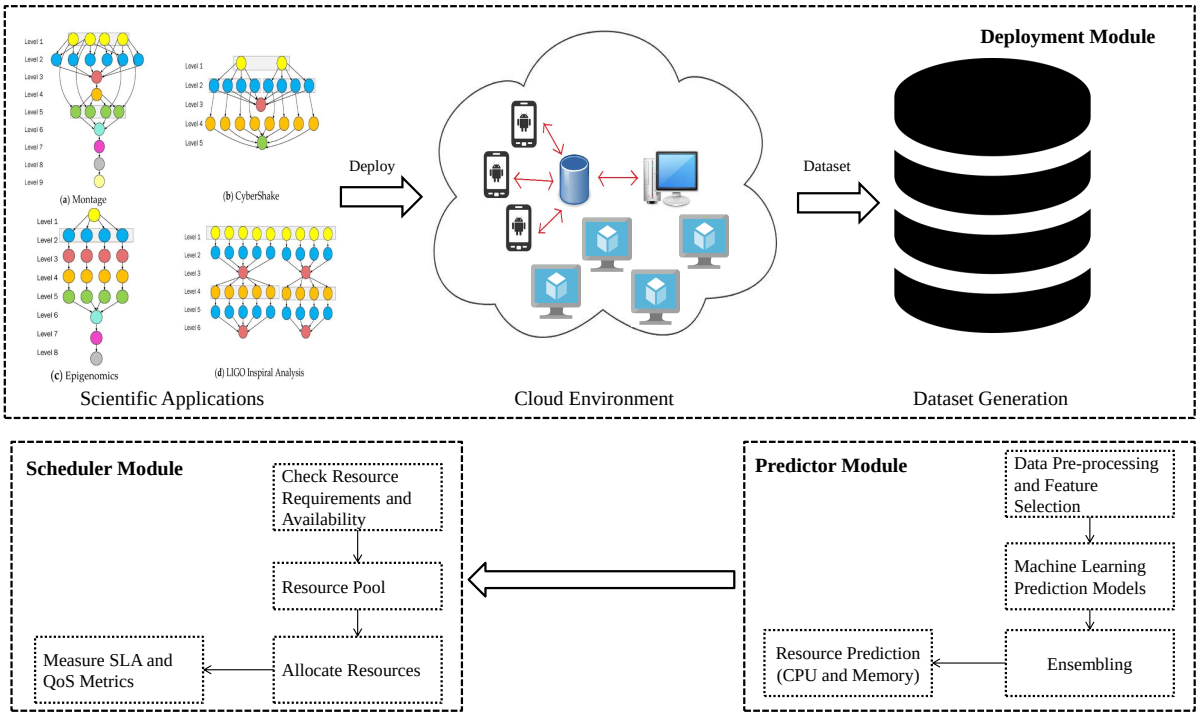

Figure 4.1: Proposed RPS Framework

In deployment module, the scientific application is executed on the workflowSim (a cloud simulator for scientific applications) [140] as discussed in chapter 3. Once the applica-

tion is deployed, a dataset is generated and passed onto prediction module for further processing. The prediction module performs the preprocessing of data so that there are no null values and converts the alphabetic values to numeric for smooth processing. It also selects the relevant features using a meta-heuristic feature selection approach. Finally, this module predicts the usage of resources by implementing REAP algorithm as discussed in Chapter 3. The working of scheduler module depends upon the output of the prediction module. In this module, the availability of the resources is checked from the resource pool. Then, the resources are scheduled efficiently to the application for further processing as discussed in Algorithm 4.1. The aim of scheduling algorithm is to strengthen the performance by reducing the execution time, cost and SLA by efficiently allocating the resources to the tasks.

4.2.2 Proposed Algorithm

The objective of this algorithm is to find an optimal solution by considering multiple criteria. Therefore, the features of an swarm intelligence are combined with TOPSIS. The former technique is very quick at determing the optimal solutions and the latter helps to make a decison based on multiple criteria. In this algorithm, the resources are scheduled on the basis of predicted set of resources by Algorithm 3.2. Initially, the average CPU utilization (Ap_cp) and average memory utilization (Ap_mem) requirement of a scientific application is checked against the available CPU and memory size of the firstIdleVm. If the CPU and memory requirement of the application ae less than the available MIPS and current size of VM, then the application is mapped to that particular VM. Further, to schedule the tasks of mapped application, an optimzation approach is followed. The velocity v_i and position p_i of particles(tasks) are randomly initialized using formula 4.9 and 4.10.

$$V_i[k+1] = w * V_i[k] + c1 * rand1 * (pbest - P_i[k]) + c2 * rand2 * (gbest - P_i[k]) \quad (4.9)$$

$$P_i[k+1] = P_i[k] + V_i[k+1] \quad (4.10)$$

where $V_i[k+1]$ is current velocity and $V_i[k]$ is the previous velocity of particle i. $P_i[k+1]$ is current position and $P_i[k]$ is the previous position of the particle i.

Algorithm 4.1 Optimized Prediction Based Scheduling (OPSA) Algorithm

Input Predicted resource requirements from Algorithm 3.2
TaskList= $[T_1, T_2, ..., T_n]$, VmList= $[Vm_1, Vm_2, ..., Vm_n]$
Output Optimal mapping between applications and VMs
Begin
Set `Ap_cp` = ACU
Set `Ap_mem` = AMU
Set `n` = number of scientific applications
Set `pdim` ← TaskList size
for each i in 1,...,n do
Set vmSize ← getVmList().size()
Set firstIdleVm ← null
for each j in 1,...,vmSize do
Set vm ← getVmList().get(j)
If vm.getState()== IDLE
∧ Ap_i_cp < vm.getAvailableMips()
∧ Ap_i_mem < vm.getCurrentSize()
then
firstIdleVm ← vm
Assign application to firstIdleVm
Set firstIdleVmBUSY.setState() ← BUSY
else
Move to next VM for comparison
endIf
endfor
Randomly initialize the velocity v_i (Eq 4.9)
position p_i (4.10) of particles(tasks)
for each $t \in ATList$
Repeat
Compute FV(RCS) for each particle using Algorithm 4.2 and update the values for p_i
If $FV_{cur} < pbest$
then Assign $pbest \leftarrow p_i$
else Keep $pbest$
endIf
Compare all $pbest$
Assign $gbest \leftarrow$ highest $pbest$
If $gbest_{cur} < FV_{cur}$
then Assign $gbest \leftarrow p_i$
else Keep previous $gbest$
endIf
Assign particle with highest $gbest$ to VM for execution
Update the velocity v_i and position p_i of particles
Until stopping criteria is not satisfied
endfor
endfor
End

$c1$ and $c2$ are acceleration coefficients whose value can be taken between 1 and 2. $rand1$ and $rand2$ are the random number whose value lie between 0 and 1. Particle's best position is denoted by pbest and the position of the best particle in the entire population is denoted as gbest. w is the interia weight usually lie between 0 and 1. Fitness Value (FV) is used as an evaluation tool to measure the performance of a particle. The FV for each task is computed using TOPSIS algorithm which is explained in section 4.2.3. If the current fitness value of a task is less than its personal best value, then current fitness value is assigned as its personal best value and the same process is repeated for all the tasks. Next, all the personal best values are compared and highest personal best value is assigned as global best value. Again, if current global best value is less than current fitness value, then current fitness value is assigned as global best value and task with highest global best value is given to VM for execution. The same procedure is applied to rest of the tasks of all the mapped applications.

4.2.3 TOPSIS- A multi-criteria decision making algorithm

Several heuristic techniques such as PSO, ACO, ABC, etc. have been utilized by various researchers for optimizing single criteria based problems. These optimization techniques lack the ability to handle decision making based on multiple criteria. Inorder to attain better optimized results for problems based on multiple criteria, a multi-objective decision making algorithm named "TOPSIS" is incorporated [138, 139, 141]. This method takes multiple factors into consideration while computing the fitness value for tasks. Algorithm 4.2 depicts the overall process followed by TOPSIS algorithm. Initially, a decision matrix is constructed of size $t * c$, where t are the total tasks (alternatives) and c represents the number of criterion as shown in Table 4.1. Next, the decision matrix is normalized using Eq 4.11.

$$(DM_n[j][i]) \leftarrow (DM[j][i])/\sum \sqrt{i^2} \tag{4.11}$$

where i= 1,2,...,t, j=1,2,...,c and $(DM_n[j][i])$ are the elements of decision matrix corresponding to i^th alternative and j^th criteria. Further, the elements of $(DM_n[j][i])$ are multiplied by inertia weight as shown in Eq 4.12, provided by the decision maker as per the importance of criteria in scheduling process.

$$DM_{nw}[i][j] \leftarrow DM_n[i][j] * inertiaweight[j] \tag{4.12}$$

Algorithm 4.2 TOPSIS Algorithm to Compute RCS

Input t alternatives, c criterion and inertia weight for each task

Output Relative Closeness Score (RCS)

Begin

Construct Decision Matrix(DM)

DM[Execution Time] $ET_t = \sum_{m=1}^{v_n} \sum_{k=1}^{j_n} ET_{nmk}$

DM[Execution Cost] $EC_t = Processingcostpersecond * ET_t$

Calculate normalized DM $[DM_n]$

for each i in alternative

for each j in criteria

$\sum \sqrt{j^2} \leftarrow (\sum (DM[i][j])^2)^{\frac{1}{2}}$

$(DM_n[j][i]) \leftarrow (DM[j][i]) / \sum \sqrt{i^2}$

endfor

endfor

Determine weighted normalized DM $[DM_{nw}]$

for each i in alternative

for each j in criteria

$DM_{nw}[i][j] \leftarrow DM_n[i][j] * inertiaweight[j]$

endfor

endfor

Calculate Att_p and Att_n

$Att_p = setofpositiveattributes$

$Att_n = setofnegativeattributes$

Evaluate the separation measures from Att_p and Att_n for each attribute

for each i in alternative

for each j in criteria

$SM\_Att_p[i] \leftarrow (\sum_j (Att_p[j] - DM_{nw}[i][j])^2)^{\frac{1}{2}}$

$SM\_Att_n[i] \leftarrow (\sum_j (Att_n[j] - DM_{nw}[i][j])^2)^{\frac{1}{2}}$

endfor

endfor

Compute RCS

for each i in alternative

$RCS[i] \leftarrow SM\_Att_n[i] / SM\_Att_n[i] + SM\_Att_p[i]$

endfor

Return RCS

Table 4.1: Decision Matrix

Cloudlet ID	**Time (ms)**	**Cost**
30	0.11	228.03
13	137.03	278.1
20	25.11	130.13
18	29.22	142.46
16	46.15	193.25
14	47.44	197.33
22	31.06	148.27
21	1.36	4.08
19	1.36	4.08
17	1.53	4.59
2	198.22	168.46
15	1.53	4.59
23	1.43	4.29
26	23.99	126.88

Now, calculate the Att_p and Att_n, where Att_p are the set of attributes which have positive impact and Att_n are those set of attributes which have negative impact on the solution.

Next step is to evaluate the separation measure for Att_p and Att_n for each attribute using Eq 4.13 and 4.14.

$$SM\_Att_p[i] \leftarrow (\sum_j (Att_p[c] - DM_{nw}[i][c])^2)^{\frac{1}{2}} \tag{4.13}$$

$$SM\_Att_n[i] \leftarrow (\sum_j (Att_n[c] - DM_{nw}[i][c])^2)^{\frac{1}{2}} \tag{4.14}$$

Finally, compute the relative closeness score (RCS) using Eq 4.15 and update the velocity of tasks(particles) in Algorithm 4.1 for determing the global best value for scheduling.

$$RCS[i] \leftarrow SM\_Att_n[i]/SM\_Att_n[i] + SM\_Att_p[i] \tag{4.15}$$

The final computed RCS is shown in Table 4.2. The score is sent as FV for the tasks in Algorithm 4.1 for scheduling.

Table 4.2: Relative Closeness Score

Cloudlet ID	**RunTime (ms)**	**Start Time (ms)**	**Finish Time (ms)**	**Cost**	**score**
30	0.11	0.1	0.21	228.03	0.4761452
13	137.03	0.21	137.24	278.1	0.3840374
20	25.11	137.24	162.35	130.13	0.9160464
18	29.22	137.24	166.46	142.46	0.9027774
16	46.15	137.24	183.39	193.25	0.8494419
14	47.44	137.24	184.69	197.33	0.8454663
22	31.06	162.35	193.41	148.27	0.8968708
21	1.36	193.41	194.77	4.08	0.9957131
19	1.36	194.77	196.13	4.08	0.9957131
17	1.53	196.13	197.66	4.59	0.9951354
2	198.22	0.21	198.43	168.46	0.2794519
15	1.53	197.66	199.19	4.59	0.9951354
23	1.43	198.43	199.86	4.29	0.9954752
26	23.99	183.39	207.39	126.88	0.9196805

4.3 Experimental Results

The proposed RPS approach has been executed and tested on Workflowsim 1.0 together with Cloudsim 3.0 for the reasons set out below:

- It supports the creation and simulation of multiple VMs on a single data center.
- Concurrent execution of tasks is possible.
- Multiple scheduling heuristics can be evaluated for different parameters.
- It supports both computational intensive and data intensive scientific applications.

The proposed prediction based scheduling approach has been compared with the existing heuristics namely DataAware, FCFS, MaxMin, MinMin and MCT on the basis of execution time and cost. The results are also validated on the basis of SLA violation rate and the comparative analysis is shown between proposed and existing scheduling approaches.

- **Case1- Execution Time:** The performance of proposed approach has been anal-

ysed with Cybershake application with 30, 50, 100 and 1000 tasks. The time taken by existing and proposed scheduling approach for executing cyber30, cyber50, cyber100 and cyber1000 can be seen in Figure 4.2.

The time taken by RPS approach for executing Cyber30 is 30.18 ms while the existing heuristics DataAware, FCFS, MaxMin, MCT and MinMin executed the applications in 62.59 ms, 84.005 ms, 125.76 ms, 71.63 ms and 49.04 ms, respectively. Similarly, for Cyber50, Cyber100 and Cyber1000, the proposed approach took 42.63 ms, 53.70 ms and 73.03 ms respectively. In comparison, the DataAware approach executed Cyber50, Cyber100 and Cyber1000 in 76.83 ms, 83.78 ms and 94.45 ms, respectively. Further, FCFS executed Cyber50 in 137.08 ms, Cyber100 in 154.46 ms and Cyber1000 in 189.99 ms. Also, the execution time taken by MaxMin is 154.73 ms, 155.16 ms and 169.81 ms for Cyber50, Cyber100 and Cyber1000, respectively. Next, MCT accomplished the execution of Cyber50, Cyber100 and Cyber1000 in 85.37 ms, 95.36 ms and 150.40 ms, respectively. At last, MinMin executed Cyber 50 in 63.41 ms, Cyber100 in 93.89 ms and Cyber1000 in 103.49 ms.

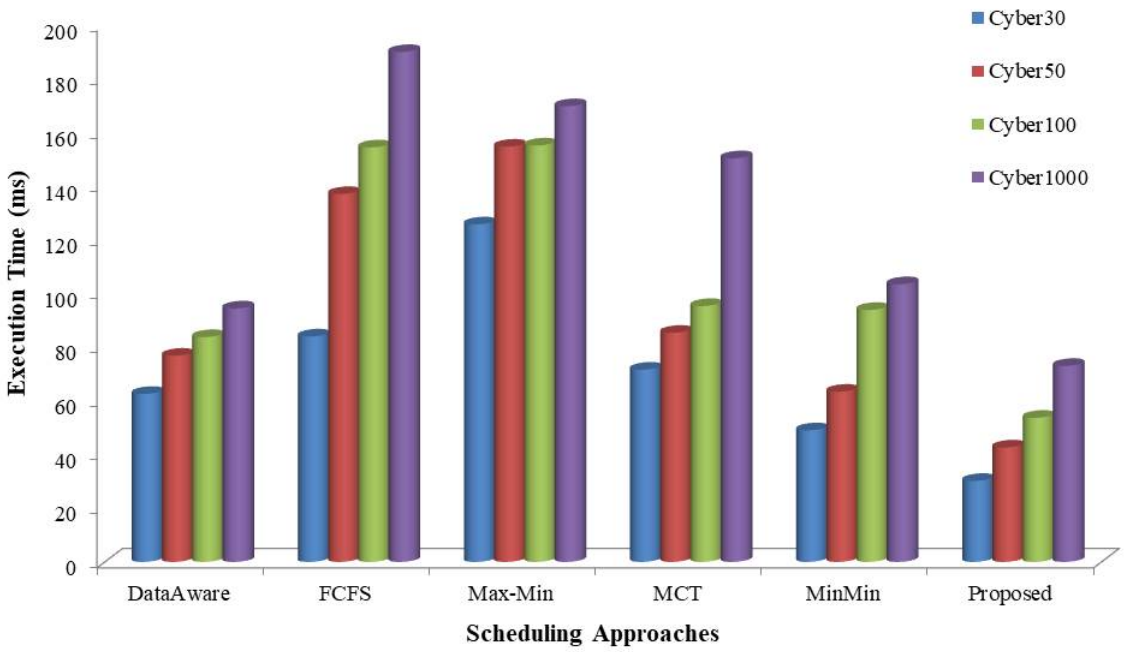

Figure 4.2: Execution time comparison of existing and proposed scheduling approach

The results shown in Figure 4.2 clearly states that proposed prediction based scheduling approach took far less execution time when compared to existing scheduling approaches. The overall execution time is curtailed by 35.59% using proposed approach.

- **Case2- Cost:** With every action during the application execution there is a cost associated with it, for example- cost for resource usage, data transfer cost and execution cost. The cost incurred by existing and proposed approaches is depicted through Figure 4.3. The cost obtained by the proposed approach for executing 30 jobs of Cybershake is 144.54 INR which is least amongst existing scheduling heuristics whereas, MaxMin attained the highest cost of 602.21 INR. For executing 50 jobs, RPS approach obtained cost of 204.16 INR while MaxMin executed the jobs with highest cost of 740.92 INR, FCFS with 656.40 INR. Similarly, for executing 100 & 1000 jobs of cybershake the proposed RPS approach incurred the minimal cost of 257.14 INR and 349.72 INR whereas MaxMin (742.98 INR) and FCFS (909.75 INR) obtained the highest cost to execute 100 & 1000 jobs of cybershake. Hence, the proposed approach is better than the existing approaches as it has minimum execution cost.

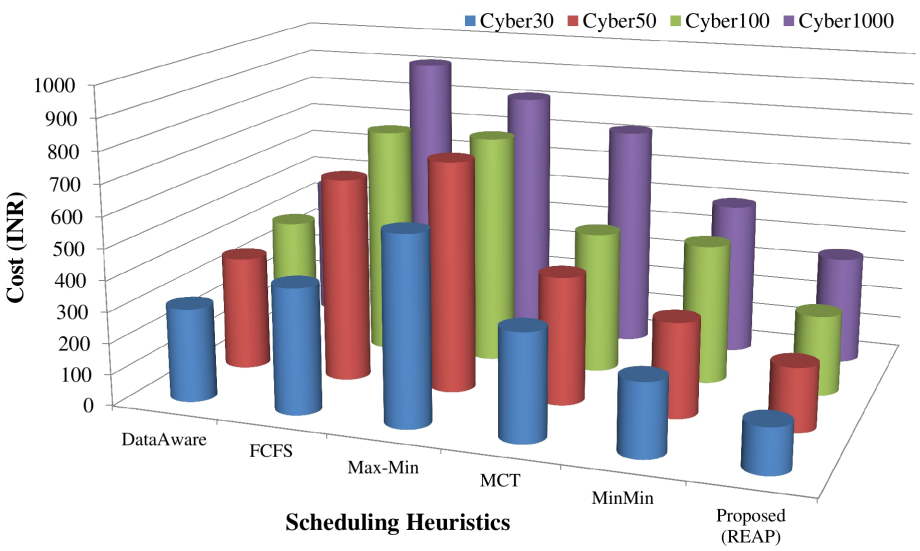

Figure 4.3: Cost comparison of existing and proposed scheduling approach

- **Case3- SLA Violation Rate:** It is very important that there should be minimal violation of SLAs so that cloud providers are able to retain their users. Another major goal of the proposed approach was to reduce the SLA violation. The results of the SLA violation rate can be seen in Figure 4.4.

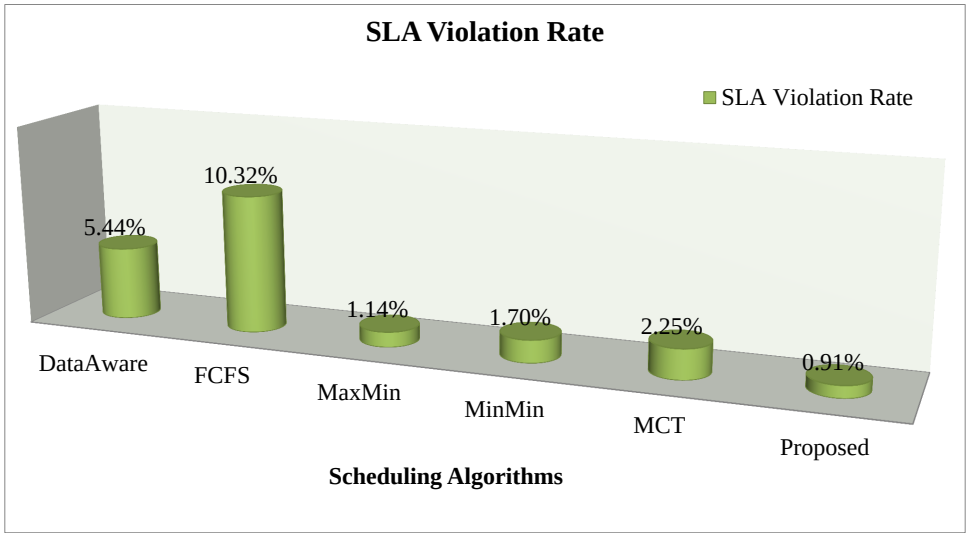

Figure 4.4: SLA Violation rate comparison of existing and proposed scheduling approach

FCFS has the highest SLA violaton rate of 10.32%, followed by DataAware and MCT with 5.44% and 2.25%. Thr MaxMin and MinMin have very minute difference between SLA violation, the former attained 1.14% while the latter obtained 1.70%. The proposed approach has 0.91% of SLA violation rate, which is least amongst existing scheduling approaches. It can be clearly seen that the proposed approach has the minimum rate of SLA violation. Therefore, the proposed prediction based scheduling approach is better than the existing approaches.

4.4 Conclusion

This chapter has examined the problem formulation by focusing on the necessity for prediction based scheduling of scientific applications. Furthermore, the section has discussed the resource prediction based scheduling approach in which the optimized prediction based scheduling algorithm has been proposed. This algorithm has exploited the predicted resource requirements for mapping the applications to VMs and scheduled the application tasks using the combination of swarm intelligence and multi-criteria decision making algorithm. This eventuates in reduced execution time and cost with minimum SLA violations.

The upcoming chapter presents the exploratory outcomes attained by affirming the proposed REAP and RPS approaches on actual cloud platform using another scientific ap-

plication. Firstly, the outcomes of the resource usage prediction technique have been discussed. Next, the results of the prediction based scheduling approach have been detailed and the performance has been validated using different evaluation metrics.

Chapter 5

Verification and Validation of the Proposed REAP and RPS Approaches

The previous chapter explained the prediction based scheduling approach for the scientific application "Cybershake". To verify the effective working of the proposed approach, another scientific application "Floodplain" is considered. An actual cloud testbed is set up for conducting the experiments and the results are evaluated on the basis of various performance metrics. This chapter deals with the verification and validation of the proposed REAP and RPS approaches through the case study of another scientific application "Floodplain" with different number of instances. A description of the cloud testbed is given along with the generated resource usage dataset to assess the working of the proposed approach. Moreover, the implementation of the method is highlighted in two stages.

Firstly, a regressive ensemble approach for predicting the resource usage is evaluated by analyzing the scientific application data using an ensemble form of the machine learning approaches. The obtained results are compared with the existing prediction approaches on the basis of RMSE, accuracy and prediction time. Secondly, an optimized prediction based scheduling approach is validated, which schedules the tasks based on the predicted set of resources. The experimental results are compared with the existing scheduling approaches on the basis of execution time, cost and SLA violation rate. Finally, the section summarizes the results obtained after implementing the resource usage prediction technique and prediction based scheduling approach. The findings clearly state that the proposed approach is better than the existing ones in terms of accuracy, execution time, cost and SLA violation.

5.1 Floodplain - Scientific Application as a Case Study

Floodplain application [142, 143] is committed to develop the accurate simulation for frequent surges in storms at north carolina's coastal regions. Currently, a four model system is deployed which comprises of different models namely Hurricane Boundary Layer which is focused on winds, ADCIRC is for surges in the storms, SWAN and Wavewatch III are directed towards waves generated by winds at near-shore along with oceans. To achieve accuracy in analysis and floodplain mapping in a given region, a broader coverage of parametric space is needed which also describes the characteristics of storms.The dynamic portion of the application is illustrated in Figure 5.1 [142, 143]. The applications's instance executes in about a day, hence demanding large computational and storage resources. Therefore, an efficient management of resources is required to make sure the smooth execution of the application.

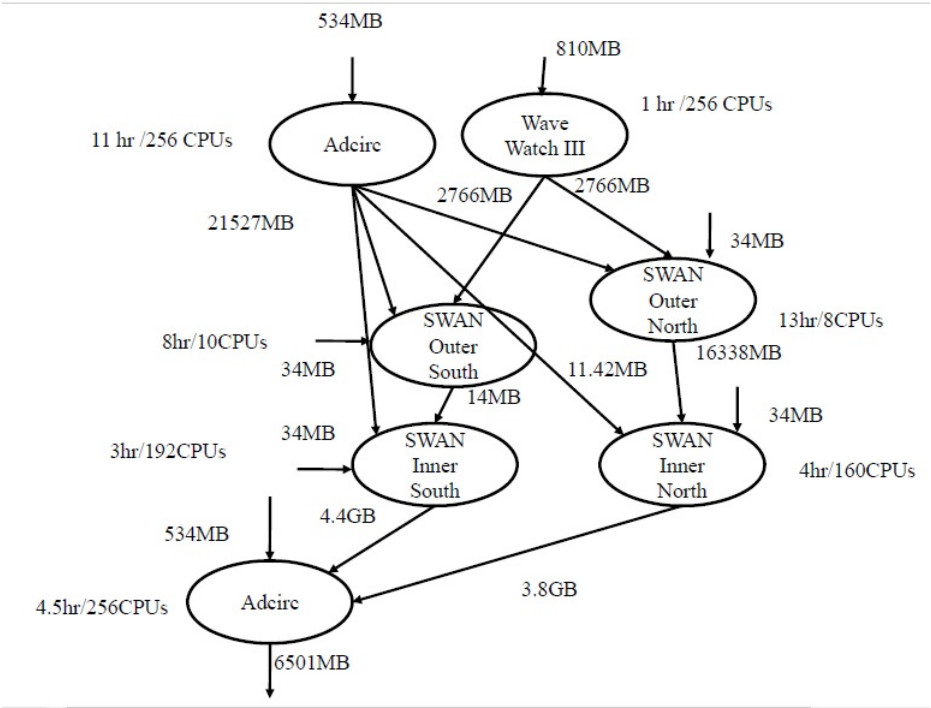

Figure 5.1: Structure of Floodplain

Initially, the application is deployed and a resource usage dataset is generated for further processing. The execution sample dataset of Floodplain application generated after performing an extensive amount of experimentation is shown in Table5.1.

Table 5.1: Floodplain Execution Dataset

id	**count**	**name2**	**run time (ms)**	**cores**	**link**	**size (bytes)**	**CPU usage (%)**	**Memory Usage**
sin	4	SWAN Inner North	14400	160	input	35651584	20.83	34 MB
ww3	2	WaveWatchIII	3600	256	input	849346560	6.49	810 MB
sis	4	SWAN Inner South	10800	192	output	4724464025	14.58	4.4 GB
adcirc2	4	Adcirc	16200	256	input	559939584	62.82	534 MB
adcirc	2	Adcirc	39600	256	output	6816792576	95.01	6501 MB
son	4	SWAN Outer North	46800	8	input	2900361216	92.42	2766 MB
sos	4	SWAN Outer South	28800	10	output	14680064	79.25	14 MB

Further, to select only relevant features from the dataset, Genetic Algorithm was applied as discussed in Chapter 3. Table 5.2 illustrates the features which were selected to perform further experiments.

Table 5.2: Illustration of the Features

Feature	**Description**
Id	The unique id number of the job being executed.
Count	Total number of tasks being executed in background.
name2	This attribute contains 6 names- Swan Inner North, WaveWatchIII, Swan Inner South, Adcirc, Swan Outer North and Swan Outer South. For processing purpose these names are given a numeric values 1,2,3,4,5,6 respectively.
Runtime	It is the execution time taken by a particular jobID. The units used to denote runtime is microseconds(ms).
File	Name of the files being executed.
Cores	Total number of CPU cores utilized for execution.
Link	It indicates whether the file is an input file or an output file.
Size	Defines the size of the job. The unit used to represent size is bytes.
CPU Usage	The total percentage of CPU being utilized while executing a particular job.
Memory Usage	Total bytes of memory used by each job during execution.

After the brief overview of the scientific application, the next section describes the experimental setup used for validating the proposed approaches.

5.2 Experimental Setup

To determine the efficacy of the proposed resource usage prediction model, various machine learning regression models such as BRR, BRNN, SVM, DT, ELM, LM, NN and RF are implemented to evaluate the experimental results. The tools used to set up testbed for experiments include RStudio 1.0.143, Netbeans IDE 8.2, CloudSim 3.0, WorkflowSim 1.0, Java SDK 8 and Microsoft Azure Cloud as shown in Figure 5.2.

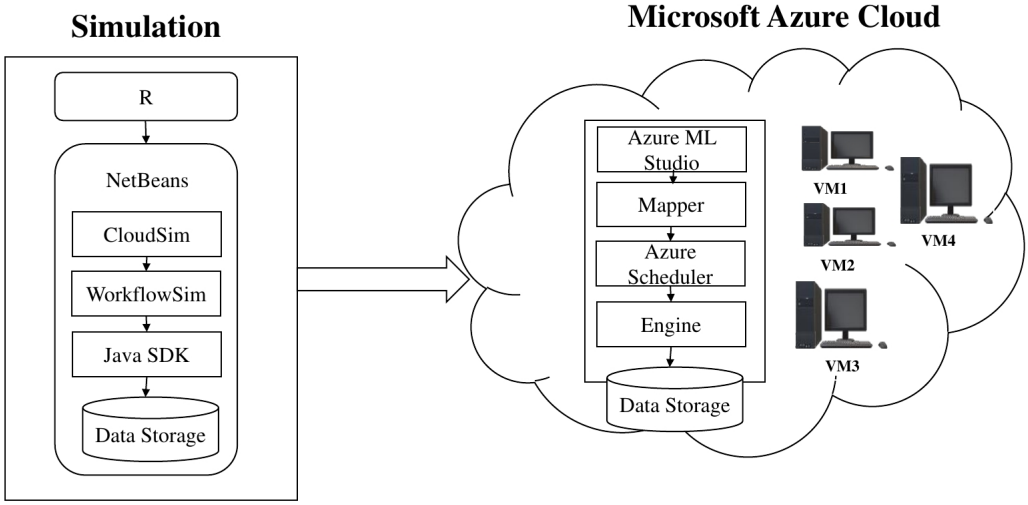

Figure 5.2: TestBed Design

WorkflowSim is routinely used for executing scientific applications such as "Cybershake" and "Floodplain". This tool is employed to collect the resource usage requirements of scientific applications. CloudSim is utilized for log file storage and R tool is applied for implementing machine learning models using scientific application datasets. The computed outcomes are verified after executing the applications in the actual cloud environment.

Table 5.3: Configuration of VMs

VM	VM Type	RAM	Storage Capacity	Data Disks	VCPUs	Max IOPS	OS
VM1	Standard D2s v3	8GB	16GB	4	2	3200	Ubuntu Server 18.04 LTS
VM2	Standard B2ms	8GB	16GB	4	2	2400	CentOS-based 7.5
VM3	Standard B1s	0.5GB	4GB	1	1	200	Windows Server 2016 Datacenter
VM4	Standard DS1 v2	3.5GB	7GB	4	1	3200	Windows Server 2012 R2 Datacenter

Four heterogeneous Virtual Machines (VMs) are created on Microsoft Azure Cloud. The characteristics of the VMs are mentioned in Table 5.3 which clearly indicate that all the four VMs have different configurations and hence create a distributed environment for deploying the applications. The resource usage requirement of Floodplain application with different number of instances such as 10, 20, 30 and 50 is shown in Table 5.4.

Table 5.4: Resource Usage requirement of scientific application

Application	Average CPU Required	Average Memory Required (MB)
flood10	2.39%	4.4
flood20	4.695%	8.162
flood30	8.72%	11.008
flood50	14.33%	14.23

5.3 Implementation Results

The experimental results of the proposed work has been verified and categorized into two sections:

- **REAP:** Regressive Ensemble Approach for Predicting Resource Usage
- **RPS:** Resource Prediction based Scheduling Approach

5.3.1 Results: Regressive Ensemble Approach for Predicting(REAP) Resource Usage for Floodplain

In this segment, an assessment of the presented REAP approach is demonstrated for "Floodplain" scientific application. An intensive experimentation was conducted for carefully analysing the efficacy of proposed approach. Firstly, the performance of machine learning regression models was measured and then the ensemble model was assessed on the basis of RMSE, accuracy and total prediction time. The experimental results are displayed in Table 5.5. of the 25 machine learning regression models, 8 were selected as the best ones to form an ensemble model according to the proposed Algorithm 3.2 discussed in Chapter 3.

Table 5.5: Performance Evaluation of Machine Learning Models

Model Name	**RMSE**	**Accuracy(%)**	**Total Time(ms)**
BRR	1.24	39.12	1.02
BRNN	1.86	36.87	2.107
SVM	0.72	89.32	1.96
DT	0.43	82.68	0.92
ELM	0.52	90.24	0.32
LM	0.61	85.72	0.64
NN	0.39	86.29	1.81
RF	1.46	75.38	0.48
Proposed	**0.325**	**93.56**	**0.26**

Case I: Root Mean Square Error (RMSE)

Figure 5.3 suggests that the computed RMSE is minimal (0.325%) in case of the proposed ensemble approach, whereas BRNN has maximal value (1.86%) and is closely followed by RF (1.46%) and BRR (1.24%). Moreover, it could be discerned NN has lower RMSE value (0.39%) than the existing predicting models DT (0.43%), ELM (0.52%), LM (0.61%) and SVM (0.72%).

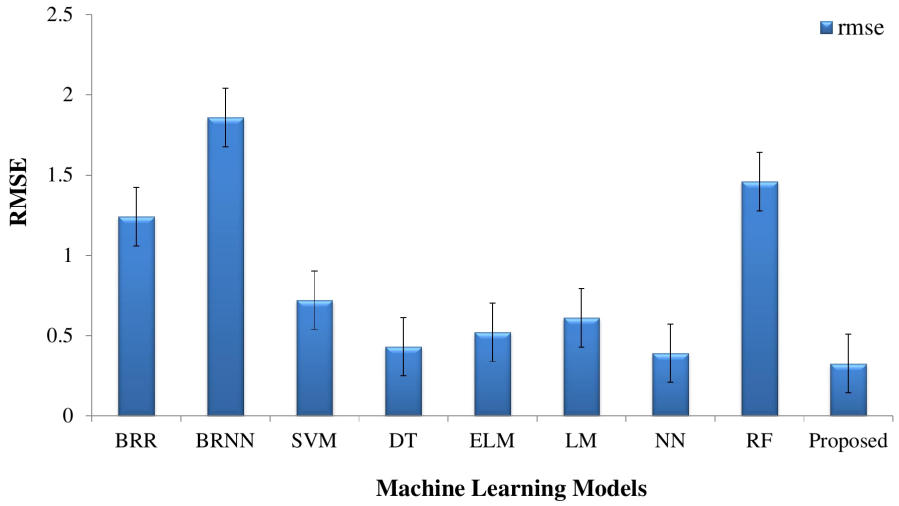

Figure 5.3: Root Mean Square Error

Case II: Prediction Accuracy

To assess the accuracy rate of proposed ensemble approach, various regression based machine learning models were utilized to predict the resource usage for different size of scientific applications. Next, "Floodplain" with 10, 20, 30 and 50 jobs was used for validating the prediction results on Microsoft Azure platform for actual resource usage. Figure 5.4 compares the actual CPU usage with the predicted use for different sizes of scientific application "Floodplain".

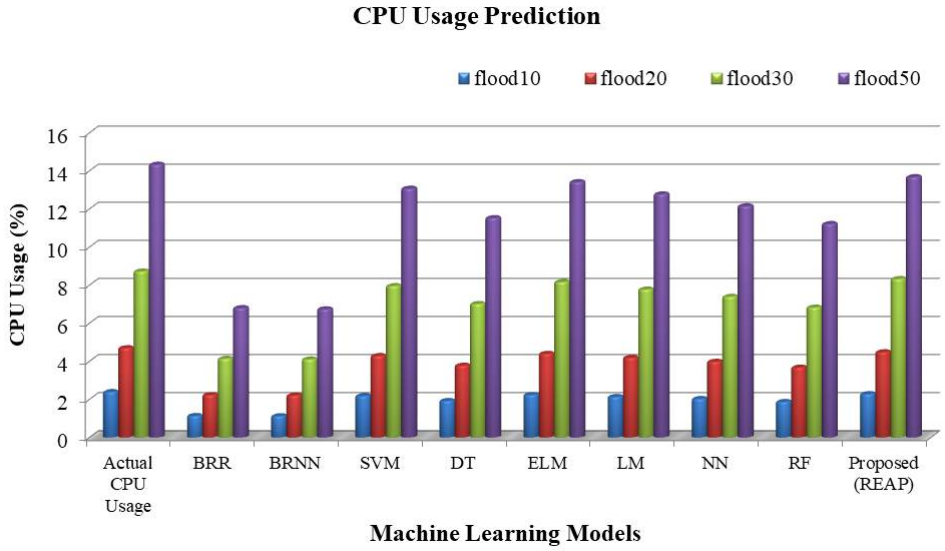

Figure 5.4: Comparison of Actual and Predicted CPU Usage

Similarly, Figure 5.5 compares the actual memory usage with the predicted one for "Floodplain" while varying the sizes. Furthermore, the accuracy rate was computed as deviation of predicted resource usage from the actual value.

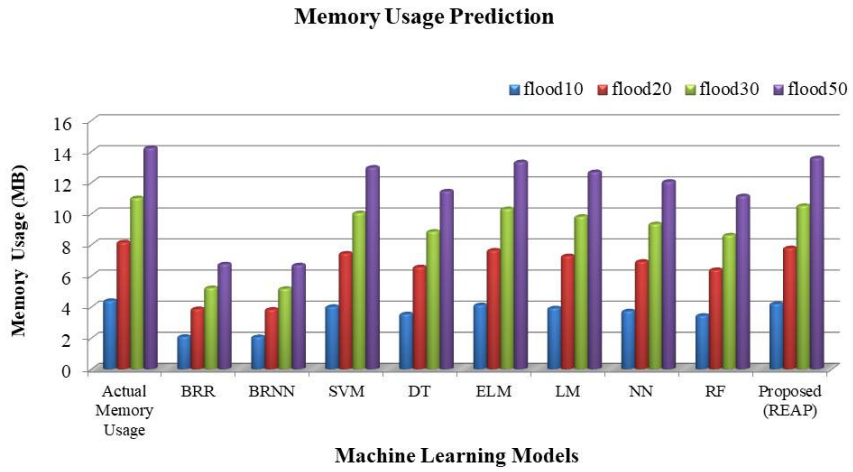

Figure 5.5: Comparison of Actual and Predicted Memory Usage

To evaluate the effectiveness of proposed ensemble approach, the exploratory outcomes were contrasted with the existing ones on the basis of prediction accuracy. Figure 5.6 illustrates the accuracy attained by the proposed ensemble approach is highest (93.56%), while BRNN achieved the least accuracy (36.87%) and was closely followed by BRR (39.12%). Amongst existing models, ELM exhibited greater accuracy (90.24%) than SVM (89.32%), NN (86.29%), LM (85.72%), DT (82.68%) and RF (75.38%).

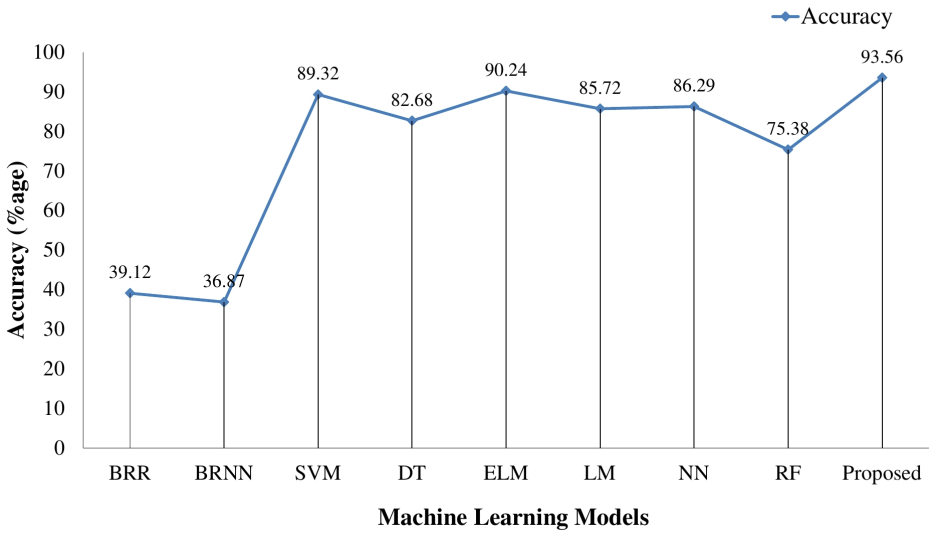

Figure 5.6: Accuracy

Thus, it can be stated that the proposed approach is more effective than the existing models for predicting the resource usage requirements for a scientific application.

Case III: Total Time for Prediction

The total time taken by the proposed approach for predicting the resources was least (0.26 ms) in comparison with the existing models. ELM displyed a prediction time of (0.32 ms), which was followed by RF (0.48 ms), LM (0.64 ms) and DT (0.92 ms). BRNN took the maximum time (2.107 ms)for prediction and was trailed by SVM (1.96 ms), NN (1.81 ms) and BRR (1.02 ms).

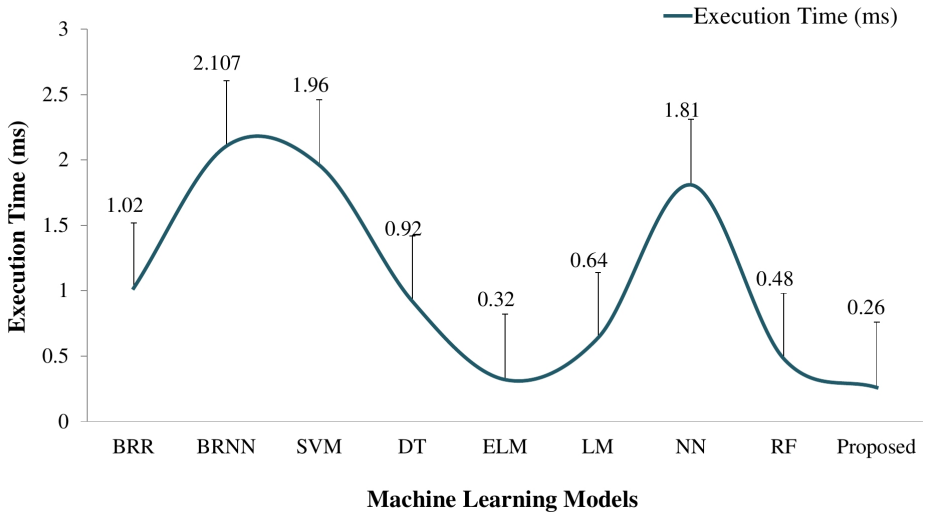

Figure 5.7: Execution Time for Prediction

The results clearly state that the models independently do not impart identical performance regarding any of the evaluation metrics as their functioning depends on the dataset to be learned from. With the change in the dataset, the values of the evaluation metrics also differs. To be precise, a single prediction model may be more suitable than the others in terms of error rate but may have a lower accuracy rate or higher execution time. For example, ELM outperforms the other regression models with 90.24% accuracy and takes only 0.32 ms total execution time but gives an error rate of 0.52%, which is higher than that of NN and DT. Similarly, NN is amongst the lowest three models having the maximum prediction time (1.81 ms) despite possessing the third highest accuracy rate (86.29%).

To enhance the performance, these individual models were combined on the basis of the proposed ensemble algorithm. An analysis of the experimental results using R and

Microsoft Azure proved that the proposed approach has the highest accuracy of 93.56% amongst all with 0.26 ms total prediction time and 0.325% RMSE. The overall accuracy was enhanced by approximately 3% and the prediction time was reduced by 6%. Hence, it is established that the proposed regressive ensemble approach for prediction (REAP) is better than the existing individual machine learning models.

5.3.2 Results: Resource Prediction based Scheduling (RPS) Approach for Floodplain

The proposed RPS approach was executed and tested on Microsoft Azure Cloud by incorporating the Azure Scheduler. Firstly, a resource group was set up in the cloud to create VMs for conducting the scientific applications. Subsequently, the jobs were uploaded in the scheduler job collection directory for execution. Finally, the resources were scheduled efficiently to the application for further processing as discussed in Algorithm 4.1 of Chapter 4.

The proposed prediction based scheduling approach was compared with the existing heuristics, namely DataAware, FCFS, MaxMin, MinMin and MCT in terms of execution time and cost. The results were also validated on the basis of SLA violation rate and the comparative analysis was done between the proposed and existing scheduling approaches.

- **Case1- Execution Time:**

  The performance of the proposed scheduling approach was analyzed for the "floodplain" application with 10, 20, 30 and 50 jobs, in which each one can consist of as many as hundred to thousand tasks. It is evident from Figure 5.8 that the presented scheduling approach has minimal execution time (20.65 ms) for flood application with 10 jobs, whereas Max-Min has the maximal execution time (68.65 ms). The performance of the proposed approach was also verified by incrementing the size of the application to 20, 30 and 50 jobs.

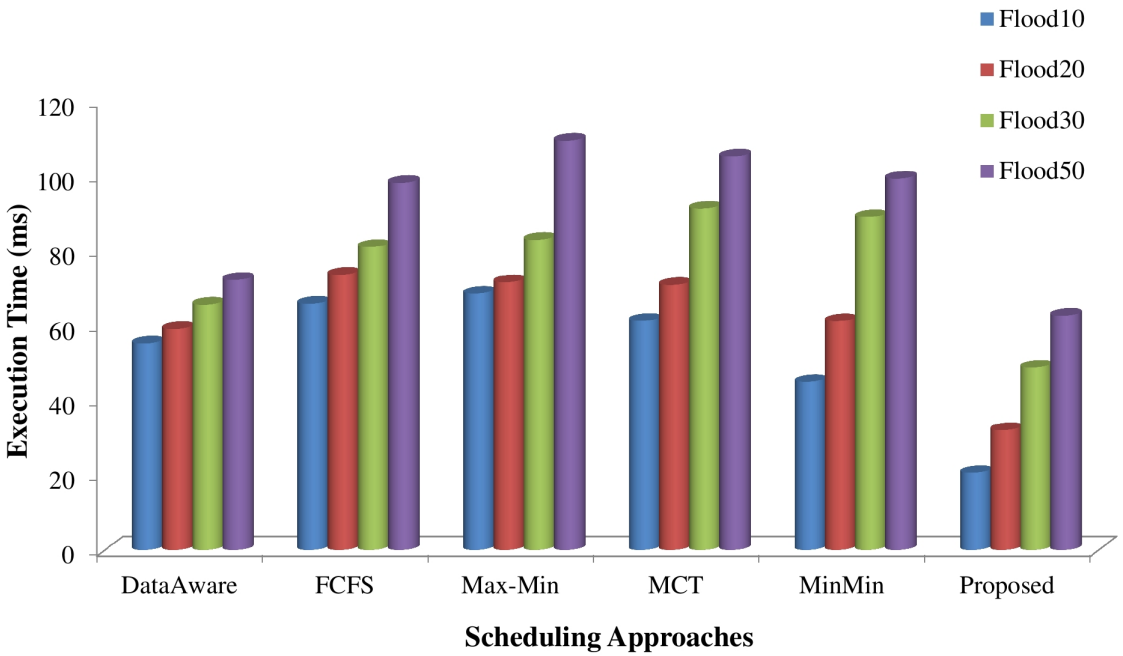

Figure 5.8: Execution time comparison of existing and proposed scheduling approach

The proposed approach had the least execution time of 32.106 ms for 20 jobs, while FCFS had the highest execution time (73.68 ms). Similarly, for 30 and 50 jobs, the proposed approach had the lowest execution times of 48.84 ms and 62.719 ms, respectively, while MCT (91.36 ms) and Max-Min (109.53 ms) possessed the highest execution times. The experimental results shown in Figure 5.8 clearly imply that the execution time taken by the proposed prediction based scheduling approach is considerably lesser than that of the existing scheduling approaches.

- **Case2- Cost:**

A cost is associated with every action during the implementation of the application, for instance, costs related to execution, resource usage and data transfer. The proposed approach amounted to 98.87 INR for executing the flood application with 10 jobs, which is the least amongst the existing approaches. On the other hand, Max-Min scheduling approach incurred the highest cost of 328.71 INR. Similarly, the cost of the proposed RPS approach for 20, 30 and 50 jobs was found to be 153.73 INR, 233.85 INR and 300.31 INR, respectively. On the contrary, FCFS was associated with the maximum cost of 352.80 INR for a flood application with 20 jobs. MCT incurred the highest expense of 437.45 INR for an application with 30 jobs, and Max-Min amounted to 524.45 INR in case of 50 jobs. It is apparent that

for all the different sizes of the application, the proposed approach has the least execution cost. Therefore the proposed RPS approach is better than the existing scheduling heuristics.

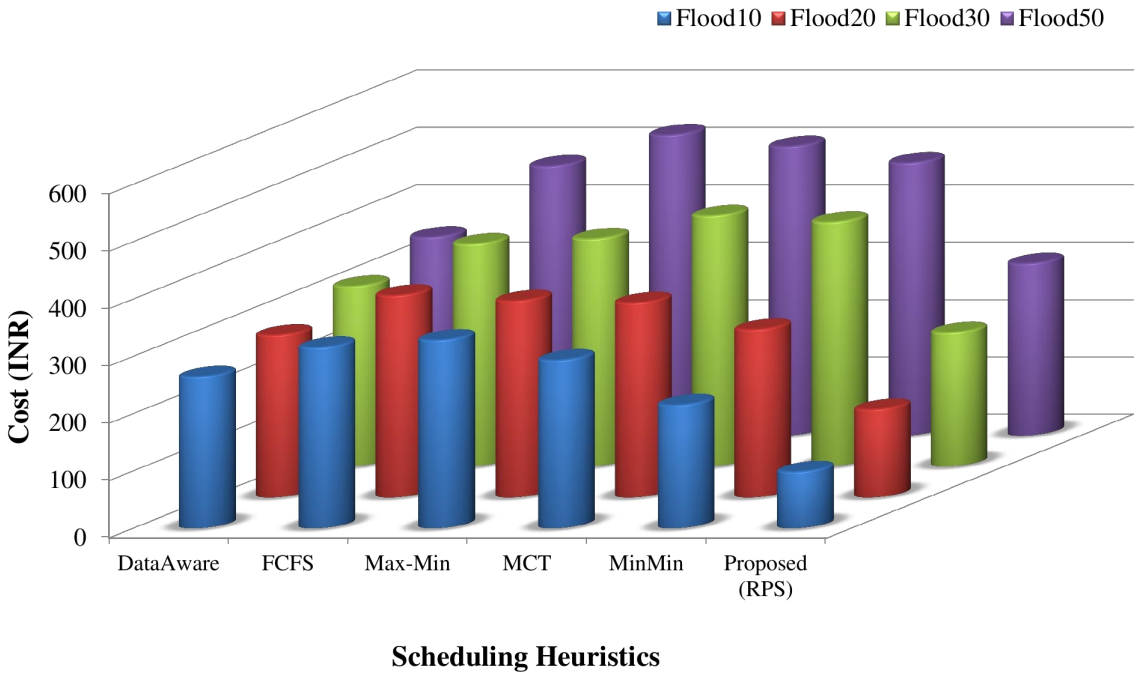

Figure 5.9: Cost comparison of existing and proposed scheduling approach

- **Case3- SLA Violation Rate:**

It is very important to have only a minimal violation of SLAs so that the cloud providers are able to retain their users. Hence, curtailing the SLA violation was another major goal of the proposed approach.

FCFS had the highest SLA violation rate of 8.21%, followed by DataAware and MCT with rates of 6.04% and 2.19%, respectively. MaxMin and MinMin had only minor differences in their SLA violations, with the former reaching 1.92% and the latter reaching 1.13%. The proposed approach had 0.74% of SLA violation rate, which is the least amongst the existing scheduling approaches.

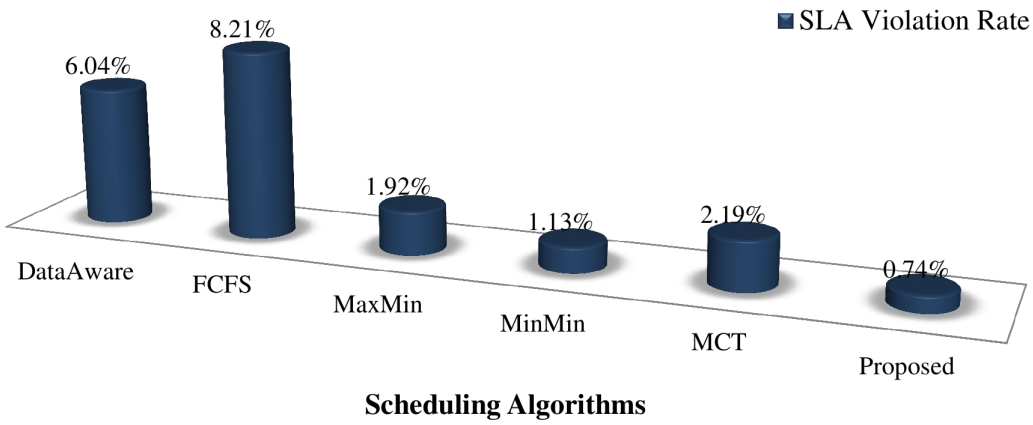

Figure 5.10: SLA Violation rate comparison of existing and proposed scheduling approach

The graphical representation of the above-mentioned results is given in Figure 5.10. It can be understood that the proposed approach has the least rate of SLA violation, and it is thus superior to the existing approaches.

5.4 Conclusion

This chapter has thus validated the maximum accuracy of the ensemble approach by implementing the proposed resource prediction technique for floodplain scientific application on cloud infrastructure. Further, the section has compared the proposed approach with the existing ones on the basis of RMSE, accuracy rate and total time taken for prediction. The proposed REAP approach has been inferred to possess the least error rate, highest accuracy rate and least prediction time. Finally, the results of the proposed prediction based scheduling approach have been validated for floodplain scientific application along with the existing scheduling heuristics. The proposed RPS approach has been found to outperform the existing ones in terms of execution time, cost and SLA violation rate.

The next chapter concludes all the chapters and presents the future directions.

Chapter 6

Conclusion and Future Scope

Efficient management of resources remains a major issue in cloud computing as the service providers endeavor to successfully plan, arrange and schedule the resources for executing a scientific application. The situation gets further complicated owing to the diverse application types requesting a large number of resources at different points of time. Therefore, the existing cloud systems need to be efficient enough to schedule the resources for scientific applications as per the predicted resource usage requirements.

This thesis addresses the need for prediction based scheduling of resources for scientific applications in the cloud environment. The work proposes a Regressive Ensemble Approach for Predicting (REAP) resource usage, such as CPU and memory, for scientific applications including "Cybershake" and "Floodplain". Further, it implements a Resource Prediction based Scheduling (RPS) approach which schedules the tasks of scientific application by taking into consideration the predicted set of resources.

Finally, this chapter summarizes the research work presented in the thesis by portraying the progress made in attaining the objectives of the research work in terms of prediction and scheduling of the resources for scientific applications in the cloud environment. The outcome of each chapter, along with the contributions of the research work, is also specified briefly. Furthermore, this chapter outlines a number of directions for future research work.

6.1 Conclusion

The motive of this research work has been to model and execute an efficient prediction and scheduling approach for scientific applications in the cloud environment. This aim has been addressed by the proposed regressive ensemble approach for prediction and further by the resource prediction based scheduling approach for scientific applications such as "Cybershake" and "Floodplain".The key findings of this research work are summarized below:

This thesis begins with the introductory section of scientific computing in **Chapter 1** that provides an overview of scientific computing, its evolution, scientific paradigms, scientific applications and its characteristics. Further, it discusses that how cloud computing is a viable platform for scientific applications along with its adoption challenges. It briefly mentions the motivation behind this research work and presents its primary contributions. In the end, this chapter describes the organization of the rest of this thesis.

Chapter 2 presents a detailed survey of existing scientific applications which can be deployed on cloud. Further, these applications are classified and categorized based on their computational requirements and task dependencies. Also, the metrics to fulfill the QoS requirements are briefly discussed. Furthermore, existing resource predictions are surveyed and compared on the basis of different parameters such as time, cost, power, SLA violation, scalabilty, CPU load, memory usage and accuracy. In addition, an extensive survey of existing scheduling approaches have been presented along with the problems which were solved and the platforms used. Based on the gaps analyzed from the existing literature, the problem formulation and objectives of this thesis are outlined.

In **Chapter 3**, a Regressive Ensemble Approach for Predicting (REAP) resource usage such as CPU and memory has been proposed. It elaborates the key features and QoS requirements of the proposed approach. Also, the design methodology and the constituents of the proposed framework are discussed in detail. The proposed ensemble framework is splited in two modules. In first module, a scientific application "Cybershake" is deployed and a resource usage data of Cybershake with 30, 50, 100 and 1000 jobs have been collected by deploying the application in WorkflowSim. These data values are stored in data storage for further experimentation. The second module comprises of feature selection and resource usage prediction using machine learning models. A meta-heuristic technique, Genetic Algorithm, has been used to select the relevant features which have significant role in predicting a desired combination of genes or traits. There were total nineteen input features namely version, count, index, name, jobCount, fileCount, childCount, id,

namespace, name2, version3, runtime, file, link, register, transfer, optional, type, size and two target features namely CPU Usage and Memory Usage. As per the results, total eight input features were selected for further processing namely Id, jobCount, fileCount, name2, runtime, file, link and size along with two target features memory usage and CPU Usage. Further, the selected machine learning models are combined using proposed ensemble algorithm which utilizes the capabilities of these models for predicting the CPU and memory usage of "Cybershake". Different evaluation measures such as Root Mean Square Error (RMSE), accuracy and execution time are evaluated and compared. Furthermore, the predicted set of resources [CPU and memory usage] is stored into database for scheduling the application tasks.

The need of prediction based scheduling is discussed in **Chapter 4**. This chapter states the problem and fitness value formulation for the resource prediction based scheduling approach and also proposes a framework for the same. Firstly, it takes the predicted CPU usage & memory usage as input and compares the application execution requirements against the availability of resources on VMs. After mapping the application with different sizes to suitable VMs, the tasks of mapped application are scheduled using a combination of swarm intelligence and TOPSIS. Here, TOPSIS is a multi-criteria decision making algorithm which is used to compute the fitness value of tasks for swarm algorithm to optimize the scheduling of resources. The fitness values calculated by TOPSIS are assigned as local best (pbest) values of tasks, then all the tasks are compared and the task with highest fitness value is considered as global best (gbest) and is assigned to VM for execution. This process is repeated until all the tasks are scheduled. The proposed approach is validated by comparing the results with the existing scheduling heuristics. The main objective of the proposed approach is to reduce the execution time, cost and SLA violation rate.

The proposed approaches in **Chapter 3** and **Chapter 4** have been implemented through another case study of a scientific application as detailed in **chapter 5**. It presents the structure of "floodplain" and also, describes its features. Furthermore, it discusses a cloud test bed that has been set up for testing and validating the proposed approach. Then, it presents the experimental results which are obtained after implementing the proposed approaches. Firstly, the results of Regressive Ensemble Approach for Predicting (REAP) resource usage have been discussed. The performance of the proposed approach has been validated on the basis of RMSE, accuracy and total prediction time. The results clearly state that the proposed ensemble approach is better than the existing approaches as it improves the overall accuracy by 3% and reduces the execution time by 6%. Finally, the results of proposed resource prediction based scheduling (RPS) approach are validated

for 10, 20, 30 and 50 jobs of floodplain along with existing scheduling heuristics. The proposed RPS approach outperforms the existing approaches namely DataAware, FCFS, MaxMin, MinMin, MCT in terms of execution time, cost and SLA violation rate. The results obtained are presented in tabular as well as graphical form in this chapter. The key contributions of this thesis are outlined as follows:

- A detailed survey has been conducted to explore various scientific applications like healthcare, genomics, physical sciences, drug discovery, flood prediction, etc being currently executed in Cloud environment.
- A comprehensive investigation has been conducted to study various existing machine learning approaches for cloud resource usage prediction.
- Scheduling techniques like bio-inspired, nature inspired and other optimization techniques have been explored for resource scheduling.
- A resource prediction approach for Cloud resources based on resource usage patterns (CPU and memory usage) of different applications using machine learning techniques has been proposed and designed. The proposed technique automatically predicts the requisite set of resources for the scientific application being executed in the cloud environment.
- A prediction based scheduling technique has been designed that takes the predicted set of resources as input and further schedules the tasks of scientific applications, thus improving the utilization of cloud resources, while reducing the SLA violations at runtime and achieving cost-effectiveness.
- Finally, the experimental results have been compared with the existing approaches that clearly exhibits the efficacy of the proposed approach in terms of accuracy, execution time, error rate, cost and SLA violations.

6.2 Future Scope

This thesis elevates the knowledge of prediction and scheduling of resources for scientific applications in cloud environment and fosters the advancements through its key contributions. This thesis has uncovered new research areas in cloud computing requiring further research. Some of the future research directions are mentioned below:

- The proposed prediction approach can be extended for predicting anomalies, peak resource usage period for improving the scheduling, load balancing and resource

scaling.

- Various aspects of scientific application execution such as scalability, bandwidth usage, overloads, response time *etc* can be taken into consideration for improving the accuracy rate and scheduling efficiency.
- The RPS approach can be further tested on IoT based cloud applications with consideration of power consumption so that energy efficiency could be improved in cloud environment.
- The resource scheduling approach can be incorporated with load balancing feature so that loads of VM instances can be efficiently balanced on the servers.
- The incorporation of ensemble model with fault tolerant techniques can help to improve the reliability and availability of cloud services.
- The proposed prediction technique can be used for predicting the security threats inorder to upgrade the performance of various applications such as risk assessment, project management, *etc*.
- REAP can be implemented to predict hosts overloading in the VMs consolidation process which can enhance the performance of scheduling heuristics by efficiently mapping the tasks of any application with cloud resources.

References

[1] Stephen G Nash. *A history of scientific computing*. ACM, 1990.

[2] Scientific Domains and Scientific Areas. [online available]: https://www.fct.pt/apoios/projectos/concursos/2012/docs, 2012.

[3] Information Technology Gems: Computer Science Milestones (1900-2011). [online available]: http://infotechgems.blogspot.in/2011/10/computer-science-milestones-from-1900.html.

[4] Jacqueline Ruttimann. Milestones in scientific computing, 2006.

[5] Edwin D Reilly. *Milestones in computer science and information technology*. Greenwood Publishing Group, 2003.

[6] Janet Ellen Abbate. From arpanet to internet: A history of arpa-sponsored computer networks, 1966–1988. 1994.

[7] Michael. Hauben. "history of arpanet", [online available:] http://www. dei. isep. ipp. pt/ acc/docs/arpa. html, 2012.

[8] Dongfang Zhao, Ning Liu, Dries Kimpe, Robert Ross, Xian-He Sun, and Ioan Raicu. Towards exploring data-intensive scientific applications at extreme scales through systems and simulations. *IEEE Transactions on Parallel and Distributed Systems*, 27(6):1824–1837, 2016.

[9] Anthony JG Hey, Stewart Tansley, Kristin M Tolle, et al. *The fourth paradigm: data-intensive scientific discovery*, volume 1. Microsoft research Redmond, WA, 2009.

[10] Kristin M Tolle, D Stewart W Tansley, and Anthony JG Hey. The fourth paradigm: Data-intensive scientific discovery [point of view]. *Proceedings of the IEEE*, 99(8):1334–1337, 2011.

[11] Radu Prodan and Michael Sperk. Scientific computing with google app engine. *Future Generation Computer Systems*, 29(7):1851–1859, 2013.

[12] Junwei Cao and Junwei Li. Large-scale real-time data-driven scientific applications. In *2nd International Conference on Networking and Distributed Computing*, pages 116–121. IEEE, 2011.

[13] Pelle Jakovits and Satish Narayana Srirama. Adapting scientific applications to cloud by using distributed computing frameworks. In *13th International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, pages 164–167. IEEE, 2013.

[14] Jeremy Cohen, Ioannis Filippis, Mark Woodbridge, Daniela Bauer, Neil Chue Hong, Mike Jackson, Sarah Butcher, David Colling, John Darlington, Brian Fuchs, et al.

Rapport: running scientific high-performance computing applications on the cloud. *Phil. Trans. R. Soc. A*, 371(1983):1–15, 2013.

[15] P Brown. A survey of scientific applications using scidb. *New England Database Day Program*, 2015.

[16] Shishir Bharathi, Ann Chervenak, Ewa Deelman, Gaurang Mehta, Mei-Hui Su, and Karan Vahi. Characterization of scientific workflows. In *3rd workshop on workflows in support of large-scale science*, pages 1–10. IEEE, 2008.

[17] Dario Ogrizovic, Zlatan Car, and Bozidar Kovacic. Scientific applications in cloud computing. *The IPSI BgD Transactions on Advanced Research*, 10(1):27–33, 2014.

[18] Yong Zhao, Xubo Fei, Ioan Raicu, and Shiyong Lu. Opportunities and challenges in running scientific workflows on the cloud. In *International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, pages 455–462. IEEE, 2011.

[19] Iman Sadooghi, Jesús Hernández Martin, Tonglin Li, Kevin Brandstatter, Yong Zhao, Ketan Maheshwari, Tiago Pais Pitta de Lacerda Ruivo, Steven Timm, Gabriele Garzoglio, and Ioan Raicu. Understanding the performance and potential of cloud computing for scientific applications. 99:1–14, 2015.

[20] Peter Mell and Tim Grance. The nist definition of cloud computing. 2011.

[21] Michael Armbrust et al. Above the clouds: A berkeley view of cloud computing. 28(13):1–23, 2009.

[22] Anju Bala and Inderveer Chana. Autonomic fault tolerant scheduling approach for scientific workflows in cloud computing. *Concurrent Engineering*, 23(1):27–39, 2015.

[23] Preeti Abrol and Savita Gupta. Social spider foraging-based optimal resource management approach for future cloud. *The Journal of Supercomputing*, pages 1–23, 2018.

[24] Anju Bala and Inderveer Chana. Multilevel priority-based task scheduling algorithm for workflows in cloud computing environment. In *Proceedings of International Conference on ICT for Sustainable Development*, pages 685–693. Springer, 2016.

[25] Mukesh Saraswat and KV Arya. Feature selection and classification of leukocytes using random forest. *Medical & biological engineering & computing*, 52(12):1041–1052, 2014.

[26] Yunliang Chen, Fangyuan Li, Jia Chen, Bo Du, Kim-Kwang Raymond Choo, and Houcine Hassan. Epls: A novel feature extraction method for migration data clustering. *Journal of Parallel and Distributed Computing*, 103:96–103, 2017.

[27] Saima Gulzar Ahmad, Chee Sun Liew, Ehsan Ullah Munir, Tan Fong Ang, and Samee U Khan. A hybrid genetic algorithm for optimization of scheduling work-

flow applications in heterogeneous computing systems. *Journal of Parallel and Distributed Computing*, 87:80–90, 2016.

[28] Seungmin Kang, Bharadwaj Veeravalli, and Khin Mi Mi Aung. Dynamic scheduling strategy with efficient node availability prediction for handling divisible loads in multi-cloud systems. *Journal of Parallel and Distributed Computing*, 113:1–16, 2018.

[29] Amandeep Verma and Sakshi Kaushal. Cost-time efficient scheduling plan for executing workflows in the cloud. *Journal of grid computing*, 13(4):495–506, 2015.

[30] Kai Li, Ke Zhao, and Sha Li. A concept-ontology-based model for resource conflict and task scheduling in concurrent engineering. *Concurrent Engineering*, 25(2):163–173, 2017.

[31] Gideon Juve, Ewa Deelman, Karan Vahi, Gaurang Mehta, Bruce Berriman, Benjamin P Berman, and Phil Maechling. Scientific workflow applications on amazon ec2. In *5th International Conference on E-Science Workshops*, pages 59–66. IEEE, 2009.

[32] DS Akerib, X Bai, S Bedikian, E Bernard, A Bernstein, A Bolozdynya, A Bradley, D Byram, SB Cahn, C Camp, et al. The large underground xenon (lux) experiment. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 704:111–126, 2013.

[33] Emmanuell D Carre, Eduardo Roloff, Philippe OA Navaux, et al. Towards weather forecasting in the cloud. In *24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP)*, pages 659–663. IEEE, 2016.

[34] Mustafa Uysal, Anurag Acharya, and Joel Saltz. Requirements of i/o systems for parallel machines: An application-driven study. pages 1–16, 1998.

[35] Evgenia Smirni, Ruth A Aydt, Andrew A Chien, and Daniel A Reed. I/o requirements of scientific applications: An evolutionary view. In *Proceedings of 5th International Symposium on High Performance Distributed Computing*, pages 49–59. IEEE, 1996.

[36] Li Liu, Miao Zhang, Rajkumar Buyya, and Qi Fan. Deadline-constrained coevolutionary genetic algorithm for scientific workflow scheduling in cloud computing. *Concurrency and Computation: Practice and Experience*, 29(5):1–12, 2016.

[37] G Bruce Berriman, Ewa Deelman, Gideon Juve, Mats Rynge, and Jens-S Vockler. The application of cloud computing to scientific workflows: a study of cost and performance. *Phil. Trans. R. Soc. A*, 371(1983):1–14, 2013.

[38] GB Berriman, AC Laity, JC Good, JC Jacob, DS Katz, E Deelman, G Singh, MH Su, and TA Prince. Montage: The architecture and scientific applications of a national virtual observatory service for computing astronomical image mosaics. In

Proceedings of Earth Sciences Technology Conference, pages 1–8, 2006.

[39] Gerd Heber, David Lifka, and P Stodghil. Post-cluster computing and the next generation of scientific applications. *Proceedings SCI*, pages 14–18, 2002.

[40] Dongfang Zhao, Jian Yin, and Ioan Raicu. Improving the i/o throughput for data-intensive scientific applications with efficient compression mechanisms. In *International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–2, 2013.

[41] Toni Mastelic, Drazen Lucanin, Andreas Ipp, and Ivona Brandic. Methodology for trade-off analysis when moving scientific applications to cloud. In *4th International Conference on Cloud Computing Technology and Science (CloudCom)*, pages 281–286. IEEE, 2012.

[42] Thomas Fahringer, Nicola Mazzocca, Massimiliano Rak, Sabri Pllana, Umberto Villano, and Georg Madsen. Performance modeling of scientific applications: scalability analysis of lapw0. In *11th Euromicro Conference on Parallel, Distributed and Network-Based Processing*, pages 5–12. IEEE, 2003.

[43] Gokcen Kestor, Roberto Gioiosa, Darren J Kerbyson, and Adolfy Hoisie. Quantifying the energy cost of data movement in scientific applications. In *International symposium on workload characterization (IISWC)*, pages 56–65. IEEE, 2013.

[44] Hongbo Zou, Fang Zheng, Matthew Wolf, Greg Eisenhauer, Karsten Schwan, Hasan Abbasi, Qing Liu, Norbert Podhorszki, and Scott Klasky. Quality-aware data management for large scale scientific applications. In *High Performance Computing, Networking, Storage and Analysis (SCC), 2012 SC Companion:*, pages 816–820. IEEE, 2012.

[45] Seung Woo Son, M Kandemir, and A Choudhary. Software-directed disk power management for scientific applications. In *19th International Parallel and Distributed Processing Symposium*, pages 1–10. IEEE, 2005.

[46] W Kirk, Rong Ge, and Xizhou Feng. High-performance, power-aware distributed computing for scientific applications. pages 40–47, 2005.

[47] Hua Liu and Manish Parashar. Enabling self-management of component-based high-performance scientific applications. In *14th International Symposium on High Performance Distributed Computing*, pages 59–68. IEEE, 2005.

[48] Ron A Oldfield, Lee Ward, Rolf Riesen, Arthur B Maccabe, Patrick Widener, and Todd Kordenbrock. Lightweight i/o for scientific applications. In *International Conference on Cluster Computing*, pages 1–11. IEEE, 2006.

[49] Paul G Brown. A survey of scientific applications using scidb. *New England Database Day Program*, pages 1–2, 2015.

[50] Ileana Ober, Marc Palyart, Jean-Michel Bruel, and David Lugato. On the use

of models for high-performance scientific computing applications: an experience report. *Software & Systems Modeling*, pages 1–24, 2016.

[51] Tonglin Li, Ioan Raicu, and Lavanya Ramakrishnan. Scalable state management for scientific applications in the cloud. In *International Congress on Big Data*, pages 204–211. IEEE, 2014.

[52] Dongfang Zhao and Ioan Raicu. Storage support for data-intensive scientific applications on the cloud. In *NSF Workshop on Experimental Support for Cloud Computing*, pages 1–4, 2014.

[53] Blesson Varghese, Ozgur Akgun, Ian Miguel, Long Thai, and Adam Barker. Cloud benchmarking for maximising performance of scientific applications. *IEEE Transactions on Cloud Computing*, 7(1):170–182, 2016.

[54] Junjie Peng, Jinbao Chen, Shuai Kong, Danxu Liu, and Meikang Qiu. Resource optimization strategy for cpu intensive applications in cloud computing environment. In *3rd International Conference on Cyber Security and Cloud Computing (CSCloud)*, pages 124–128. IEEE, 2016.

[55] Jaliya Ekanayake, Xiaohong Qiu, Thilina Gunarathne, Scott Beason, and Geoffrey Fox. High performance parallel computing with cloud and cloud technologies. *Cloud Computing and Software Services: Theory and Techniques*, 34:1–39, 2010.

[56] Diana Moise. Experiences with performing mapreduce analysis of scientific data on hpc platforms. In *Proceedings of International Workshop on Data-Intensive Distributed Computing*, pages 11–18. ACM, 2016.

[57] Jens-Sonke Vockler, Gideon Juve, Ewa Deelman, Mats Rynge, and Bruce Berriman. Experiences using cloud computing for a scientific workflow application. In *Proceedings of the 2nd international workshop on Scientific cloud computing*, pages 15–24. ACM, 2011.

[58] Jaliya Ekanayake, Shrideep Pallickara, and Geoffrey Fox. Mapreduce for data intensive scientific analyses. In *4th International Conference on eScience*, pages 277–284. IEEE, 2008.

[59] Pankaj Deep Kaur and Inderveer Chana. A resource elasticity framework for qos-aware execution of cloud applications. *Future Generation Computer Systems*, 37:14–25, 2014.

[60] Elvin Sindrilaru, Alexandru Costan, and Valentin Cristea. Fault tolerance and recovery in grid workflow management systems. In *International Conference on Complex, Intelligent and Software Intensive Systems (CISIS)*, pages 475–480. IEEE, 2010.

[61] Wenbing Zhao, PM Melliar-Smith, and Louise E Moser. Fault tolerance middleware for cloud computing. In *3rd International Conference on Cloud Computing*, pages

67–74. IEEE, 2010.

[62] Ravi Jhawar, Vincenzo Piuri, and Marco Santambrogio. A comprehensive conceptual system-level approach to fault tolerance in cloud computing. In *International Systems Conference (SysCon)*, pages 1–5. IEEE, 2012.

[63] Qiang Guan, Ziming Zhang, and Song Fu. A failure detection and prediction mechanism for enhancing dependability of data centers. *International Journal of Computer Theory and Engineering*, 4(5):726–730, 2012.

[64] Anju Bala and Inderveer Chana. Intelligent failure prediction models for scientific workflows. *Expert Systems with Applications*, 42(3):980–989, 2015.

[65] James Oly and Daniel A Reed. Markov model prediction of i/o requests for scientific applications. In *Proceedings of the 16th international conference on Supercomputing*, pages 147–155. ACM, 2002.

[66] Toni Mastelic, Walid Fdhila, Ivona Brandic, and Stefanie Rinderle-Ma. Predicting resource allocation and costs for business processes in the cloud. In *World Congress on Services*, pages 47–54. IEEE, 2015.

[67] Yexi Jiang, Chang-shing Perng, Tao Li, and Rong Chang. Asap: A self-adaptive prediction system for instant cloud resource demand provisioning. In *11th International Conference on Data Mining*, pages 1104–1109. IEEE, 2011.

[68] Yuxiang Shi, Xiaohong Jiang, and Kejiang Ye. An energy-efficient scheme for cloud resource provisioning based on cloudsim. In *International Conference on Cluster Computing*, pages 595–599. IEEE, 2011.

[69] Manish Verma, GR Gangadharan, Nanjangud C Narendra, Ravi Vadlamani, Vidyadhar Inamdar, Lakshmi Ramachandran, Rodrigo N Calheiros, and Rajkumar Buyya. Dynamic resource demand prediction and allocation in multi-tenant service clouds. *Concurrency and Computation: Practice and Experience*, 28(17):4429–4442, 2016.

[70] Rodrigo N Calheiros, Rajiv Ranjan, and Rajkumar Buyya. Virtual machine provisioning based on analytical performance and qos in cloud computing environments. In *International Conference on Parallel Processing*, pages 295–304. IEEE, 2011.

[71] Eddy Caron, Frederic Desprez, and Adrian Muresan. Forecasting for grid and cloud computing on-demand resources based on pattern matching. In *2nd International Conference on Cloud Computing Technology and Science (CloudCom)*, pages 456–463. IEEE, 2010.

[72] Jing Jiang, Jie Lu, Guangquan Zhang, and Guodong Long. Optimal cloud resource auto-scaling for web applications. In *13th IEEE International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, pages 58–65. IEEE, 2013.

[73] Song Fu and Cheng-Zhong Xu. Exploring event correlation for failure prediction

in coalitions of clusters. In *Proceedings of IEEE Conference on Supercomputing*, pages 1–12. IEEE, 2007.

[74] Sadeka Islam, Jacky Keung, Kevin Lee, and Anna Liu. Empirical prediction models for adaptive resource provisioning in the cloud. *Future Generation Computer Systems*, 28(1):155–162, 2012.

[75] Anshuman Biswas, Shikharesh Majumdar, Biswajit Nandy, and Ali El-Haraki. Automatic resource provisioning: a machine learning based proactive approach. In *6th International Conference on Cloud Computing Technology and Science (CloudCom)*, pages 168–173. IEEE, 2014.

[76] Ziming Zhang and Song Fu. Failure prediction for autonomic management of networked computer systems with availability assurance. In *International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum (IPDPSW)*, pages 1–8. IEEE, 2010.

[77] George Kousiouris, Andreas Menychtas, Dimosthenis Kyriazis, Spyridon Gogouvitis, and Theodora Varvarigou. Dynamic, behavioral-based estimation of resource provisioning based on high-level application terms in cloud platforms. *Future Generation Computer Systems*, 32:27–40, 2014.

[78] Yanfei Guo, Palden Lama, Jia Rao, and Xiaobo Zhou. V-cache: Towards flexible resource provisioning for multi-tier applications in iaas clouds. In *27th International Symposium on Parallel & Distributed Processing (IPDPS)*, pages 88–99. IEEE, 2013.

[79] Jinhui Huang, Chunlin Li, and Jie Yu. Resource prediction based on double exponential smoothing in cloud computing. In *2nd International Conference on Consumer Electronics, Communications and Networks (CECNet)*, pages 2056–2060. IEEE, 2012.

[80] Eddy Caron, Frederic Desprez, and Adrian Muresan. Pattern matching based forecast of non-periodic repetitive behavior for cloud clients. *Journal of Grid Computing*, 9(1):49–64, 2011.

[81] Zhenhuan Gong, Xiaohui Gu, and John Wilkes. Press: Predictive elastic resource scaling for cloud systems. In *International Conference on Network and Service Management*, pages 9–16. IEEE, 2010.

[82] Zhiming Shen, Sethuraman Subbiah, Xiaohui Gu, and John Wilkes. Cloudscale: elastic resource scaling for multi-tenant cloud systems. In *Proceedings of the 2nd Symposium on Cloud Computing*, pages 1–14. ACM, 2011.

[83] Haibo Mi, Huaimin Wang, Gang Yin, Yangfan Zhou, Dianxi Shi, and Lin Yuan. Online self-reconfiguration with performance guarantee for energy-efficient large-scale cloud computing data centers. In *International Conference on Services Computing*

(SCC), pages 514–521. IEEE, 2010.

[84] Nilabja Roy, Abhishek Dubey, and Aniruddha Gokhale. Efficient autoscaling in the cloud using predictive models for workload forecasting. In *International Conference on Cloud Computing (CLOUD)*, pages 500–507. IEEE, 2011.

[85] Waheed Iqbal, Matthew N Dailey, David Carrera, and Paul Janecek. Adaptive resource provisioning for read intensive multi-tier applications in the cloud. *Future Generation Computer Systems*, 27(6):871–879, 2011.

[86] Abhishek Chandra, Weibo Gong, and Prashant Shenoy. Dynamic resource allocation for shared data centers using online measurements. In *International Workshop on Quality of Service*, pages 381–398. Springer, 2003.

[87] Michael Cardosa and Abhishek Chandra. Resource bundles: using aggregation for statistical large-scale resource discovery and management. *IEEE Transactions on Parallel and Distributed Systems*, 21(8):1089–1102, 2010.

[88] Wei Fang, ZhiHui Lu, Jie Wu, and ZhenYin Cao. Rpps: a novel resource prediction and provisioning scheme in cloud data center. In *9th International Conference on Services Computing (SCC)*, pages 609–616. IEEE, 2012.

[89] Sunirmal Khatua, Anirban Ghosh, and Nandini Mukherjee. Optimizing the utilization of virtual resources in cloud environment. In *International Conference on Virtual Environments, Human-Computer Interfaces and Measurement Systems*, pages 82–87. IEEE, 2010.

[90] Radu Prodan and Vlad Nae. Prediction-based real-time resource provisioning for massively multiplayer online games. *Future Generation Computer Systems*, 25(7):785–793, 2009.

[91] Sourav Dutta, Sankalp Gera, Akshat Verma, and Balaji Viswanathan. Smartscale: Automatic application scaling in enterprise clouds. In *5th International Conference on Cloud Computing (CLOUD)*, pages 221–228. IEEE, 2012.

[92] Gong Chen, Wenbo He, Jie Liu, Suman Nath, Leonidas Rigas, Lin Xiao, and Feng Zhao. Energy-aware server provisioning and load dispatching for connection-intensive internet services. In *NSDI*, volume 8, pages 337–350, 2008.

[93] Suraj Pandey, Linlin Wu, Siddeswara Mayura Guru, and Rajkumar Buyya. A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments. In *24th International conference on advanced information networking and applications*, pages 400–407. IEEE, 2010.

[94] Amandeep Verma and Sakshi Kaushal. A hybrid multi-objective particle swarm optimization for scientific workflow scheduling. *Parallel Computing*, 62:1–19, 2017.

[95] Rong Ge, Xizhou Feng, and Kirk W Cameron. Performance-constrained distributed dvs scheduling for scientific applications on power-aware clusters. In *Proceedings of*

the conference on Supercomputing, pages 34–44. IEEE, 2005.

[96] Muhammad Shafie Abd Latiff, Gaddafi Abdul-Salaam, Syed Hamid Hussain Madni, et al. Secure scientific applications scheduling technique for cloud computing environment using global league championship algorithm. *PloS one*, 11(7):1–18, 2016.

[97] Ioana Banicescu, Vijay Velusamy, and Johnny Devaprasad. On the scalability of dynamic scheduling scientific applications with adaptive weighted factoring. *Cluster Computing*, 6(3):215–226, 2003.

[98] Akindele A Bankole and Samuel A Ajila. Predicting cloud resource provisioning using machine learning techniques. In *26th Annual Canadian Conference on Electrical and Computer Engineering (CCECE)*, pages 1–4. IEEE, 2013.

[99] Jian Li, Sen Su, Xiang Cheng, Meina Song, Liyu Ma, and Jie Wang. Cost-efficient coordinated scheduling for leasing cloud resources on hybrid workloads. *Parallel Computing*, 44:1–17, 2015.

[100] Zhifeng Yu, Chenjia Wang, and Weisong Shi. Flaw: Failure-aware workflow scheduling in high performance computing systems. *Journal of Cluster Computing*, 13(4):421–434, 2010.

[101] Deepak Poola, Kotagiri Ramamohanarao, and Rajkumar Buyya. Fault-tolerant workflow scheduling using spot instances on clouds. *Procedia Computer Science*, 29:523–533, 2014.

[102] Danilo Ardagna, Barbara Panicucci, Marco Trubian, and Li Zhang. Energy-aware autonomic resource allocation in multitier virtualized environments. *IEEE Transactions on Services Computing*, 5(1):2–19, 2012.

[103] Haluk Topcuoglu, Salim Hariri, and Min-You Wu. Task scheduling algorithms for heterogeneous processors. In *Heterogeneous Computing Workshop*, pages 3–14. IEEE, 1999.

[104] Ayda Mazandarani and Hossein Momeni. Qos-aware scientific application scheduling algorithm in cloud environment. *Journal of Computer Engineering and Intelligent Systems (CEIS)*, 4(12):21–30, 2013.

[105] Zhangjun Wu, Xiao Liu, Zhiwei Ni, Dong Yuan, and Yun Yang. A market-oriented hierarchical scheduling strategy in cloud workflow systems. *The Journal of Supercomputing*, 63(1):256–293, 2013.

[106] Xixhou Feng, Rong Ge, and Kirk W Cameron. Power and energy profiling of scientific applications on distributed systems. In *19th International Parallel and Distributed Processing Symposium*, pages 34–34. IEEE, 2005.

[107] Jia Yu, Rajkumar Buyya, and Chen Khong Tham. Cost-based scheduling of scientific workflow applications on utility grids. In *1st International Conference on e-Science and Grid Computing (e-Science'05)*, pages 1–8. IEEE, 2005.

[108] Meng Xu, Lizhen Cui, Haiyang Wang, and Yanbing Bi. A multiple qos constrained scheduling strategy of multiple workflows for cloud computing. In *International Symposium on Parallel and Distributed Processing with Applications*, pages 629–634. IEEE, 2009.

[109] Rizos Sakellariou, Henan Zhao, Eleni Tsiakkouri, and Marios D Dikaiakos. Scheduling workflows with budget constraints. In *Integrated research in GRID computing*, pages 189–202. Springer, 2007.

[110] S Selvarani and G Sudha Sadhasivam. Improved cost-based algorithm for task scheduling in cloud computing. In *International conference on Computational intelligence and computing research (iccic)*, pages 1–5. IEEE, 2010.

[111] Yun Yang, Ke Liu, Jinjun Chen, Xiao Liu, Dong Yuan, and Hai Jin. An algorithm in swindew-c for scheduling transaction-intensive cost-constrained cloud workflows. In *4th International Conference on eScience*, pages 374–375. IEEE, 2008.

[112] Ke Liu, Hai Jin, Jinjun Chen, Xiao Liu, Dong Yuan, and Yun Yang. A compromised-time-cost scheduling algorithm in swindew-c for instance-intensive cost-constrained workflows on cloud computing platform. *International Journal of High Performance Computing Applications*, 24(4):445–456, 2010.

[113] P Varalakshmi, Aravindh Ramaswamy, Aswath Balasubramanian, and Palaniappan Vijaykumar. An optimal workflow based scheduling and resource allocation in cloud. In *International Conference on Advances in Computing and Communications*, pages 411–420. Springer, 2011.

[114] Cui Lin and Shiyong Lu. Scheduling scientific workflows elastically for cloud computing. In *International Conference on Cloud Computing (CLOUD)*, pages 746–747. IEEE, 2011.

[115] Andre Frederico Guilhoto Monteiro. Hpc management and engineering in the hybrid cloud. pages 1–153, 2015.

[116] Sukhpal Singh and Inderveer Chana. A survey on resource scheduling in cloud computing: Issues and challenges. *Journal of Grid Computing*, 14(2):217–264, 2016.

[117] Ji Liu, Esther Pacitti, Patrick Valduriez, and Marta Mattoso. A survey of data-intensive scientific workflow management. *Journal of Grid Computing*, 13(4):457–493, 2015.

[118] Minxian Xu, Wenhong Tian, and Rajkumar Buyya. A survey on load balancing algorithms for virtual machines placement in cloud computing. *Concurrency and Computation: Practice and Experience*, 29(12):1–16, 2017.

[119] Gurleen Kaur, Ranjit Kaur, and Sita Rani. Cloud computing-a new trend in it era. *International journal of Science Technology & Management (IJSTM)*, pages 1–6, 2015.

[120] Weishan Zhang, Pengcheng Duan, Laurence T Yang, Feng Xia, Zhongwei Li, Qinghua Lu, Wenjuan Gong, and Su Yang. Resource requests prediction in the cloud computing environment with a deep belief network. *Software: Practice and Experience*, 47(3):473–488, 2017.

[121] Robert Graves, Thomas H Jordan, Scott Callaghan, Ewa Deelman, Edward Field, Gideon Juve, Carl Kesselman, Philip Maechling, Gaurang Mehta, Kevin Milner, et al. Cybershake: A physics-based seismic hazard model for southern california. *Pure and Applied Geophysics*, 168(3-4):367–381, 2011.

[122] Philip Maechling, Ewa Deelman, Li Zhao, Robert Graves, Gaurang Mehta, Nitin Gupta, John Mehringer, Carl Kesselman, Scott Callaghan, David Okaya, et al. Scec cybershake workflowsautomating probabilistic seismic hazard analysis calculations. In *Workflows for e-Science*, pages 143–163. Springer, 2007.

[123] Pegasus Workflow Gallery (2011). Cybershake dataset, https://pegasus.isi.edu/workflowgallery/gallery/cybershake/index.php, accessed online: 2019-06-14.

[124] Niju P Joseph and B Ramadoss. A genetic algorithm applying single point crossover and uniform mutation to minimize uncertainty in production cost. *World Applied Sciences Journal*, 23(8):1013–1017, 2013.

[125] Nelson Fumo and MA Rafe Biswas. Regression analysis for prediction of residential energy consumption. *Renewable and sustainable energy reviews*, 47:332–343, 2015.

[126] Shun Takai, Tae Yang, and John A Cafeo. A bayesian method for predicting future customer need distributions. *Concurrent Engineering*, 19(3):255–264, 2011.

[127] Shailja Sharma, Jagdeep Singh Lather, and Mayank Dave. Semantic approach for web service classification using machine learning and measures of semantic relatedness. *Service Oriented Computing and Applications*, 10(3):221–231, 2016.

[128] Brijendra Gupta and RB Mishra. Neuro-psychiatric disease prediction using support vector machine. *DEMENTIA*, 200(71):129, 2013.

[129] Nishant Gupta, Naman Ahuja, Shikhar Malhotra, Anju Bala, and Gurleen Kaur. Intelligent heart disease prediction in cloud environment through ensembling. *Expert Systems*, 34(3):1–14, 2017.

[130] Gurleen Kaur and Anju Bala. A survey of prediction-based resource scheduling techniques for physics-based scientific applications. *Modern Physics Letters B*, 32(25):1–26, 2018.

[131] Gao Huang, Guang-Bin Huang, Shiji Song, and Keyou You. Trends in extreme learning machines: A review. *Neural Networks*, 61:32–48, 2015.

[132] Salam Ismaeel and Ali Miri. Using elm techniques to predict data centre vm requests. In *2015 IEEE 2nd International Conference on Cyber Security and Cloud*

Computing, pages 80–86. IEEE, 2015.

[133] R Development Core Team. The r project for statistical computing. *https://www.r-project.org/*, 2018.

[134] Jitendra Kumar and Ashutosh Kumar Singh. Workload prediction in cloud using artificial neural network and adaptive differential evolution. *Future Generation Computer Systems*, 81:41–52, 2018.

[135] Ali Asghar Rahmanian, Mostafa Ghobaei-Arani, and Sajjad Tofighy. A learning automata-based ensemble resource usage prediction algorithm for cloud computing environment. *Future Generation Computer Systems*, 79:54–71, 2018.

[136] Nidhi Jain Kansal and Inderveer Chana. Artificial bee colony based energy-aware resource utilization technique for cloud computing. *Concurrency and Computation: Practice and Experience*, 27(5):1207–1225, 2015.

[137] Vincent C Emeakaroha, Marco AS Netto, Rodrigo N Calheiros, Ivona Brandic, Rajkumar Buyya, and César AF De Rose. Towards autonomic detection of sla violations in cloud infrastructures. *Future Generation Computer Systems*, 28(7):1017–1029, 2012.

[138] Majid Behzadian, S Khanmohammadi Otaghsara, Morteza Yazdani, and Joshua Ignatius. A state-of the-art survey of topsis applications. *Expert Systems with applications*, 39(17):13051–13069, 2012.

[139] Gholam Reza Jahanshahloo, F Hosseinzadeh Lotfi, and Mohammad Izadikhah. An algorithmic method to extend topsis for decision-making problems with interval data. *Applied mathematics and computation*, 175(2):1375–1384, 2006.

[140] Weiwei Chen and Ewa Deelman. Workflowsim: A toolkit for simulating scientific workflows in distributed environments. In *8th International Conference on E-Science*, pages 1–8. IEEE, 2012.

[141] Akshay Jaiswal and RB Mishra. Cloud service selection using topsis and fuzzy topsis with ahp and anp. In *International Conference on Machine Learning and Soft Computing*, pages 136–142. ACM, 2017.

[142] Lavanya Ramakrishnan and Dennis Gannon. A survey of distributed workflow characteristics and resource requirements. *Indiana University*, pages 1–23, 2008.

[143] Lavanya Ramakrishnan and Beth Plale. A multi-dimensional classification model for scientific workflow characteristics. In *Proceedings of the 1st International Workshop on Workflow Approaches to New Data-centric Science*, pages 1–4. ACM, 2010.

List of Publications

International Journal

1. **Gurleen Kaur**, and Anju Bala, "A survey of prediction-based resource scheduling techniques for physics-based scientific applications", *Modern Physics Letters B*, World Scientific Publications, 32(25), 1-26, 2018. [**SCI Indexed, Impact Factor - 0.929**]

2. **Gurleen Kaur**, Anju Bala, and Inderveer Chana, "An intelligent regressive ensemble approach for predicting resource usage in cloud computing", *Journal of Parallel and Distributed Computing*,Elsevier Publications, 123, 1-12, 2018. [**SCI Indexed, Impact Factor - 1.819**]

3. **Gurleen Kaur**, and Anju Bala, "An Efficient Resource Prediction based Scheduling (RPS) Technique for Scientific Applications in Cloud Environment", *Concurrent Engineering: Research and Applications*, Sage Publications, 27(2), 112-125, 2019. [**SCI Indexed, Impact Factor - 1.127**]

International Conference

1. **Gurleen Kaur** and Anju Bala, "An Efficient Automated Hybrid Algorithm to Predict Floods in Cloud Environment", *IEEE Canadian Conference on Electrical And Computer Engineering*, 2019.

Communicated

1. **Gurleen Kaur** and Anju Bala, "OPSA: An Optimized Prediction based Scheduling Approach for Scientific Applications in Cloud Environment", *Cluster Computing*, 2019 [**SCI Indexed, Impact Factor - 1.851**].

2. **Gurleen Kaur** and Anju Bala, "Prediction based Task Scheduling Approach for Floodplain Application in Cloud Environment", *Computing*, 2019 [**SCI Indexed, Impact Factor - 2.063**].