An Enhanced MinMin Resource Scheduling Algorithm for Scientific Applications in Cloud Computing

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

Master of Engineering
in
Software Engineering

By
Resham Kaur
(Roll No: 801531011)

Under the supervision of
Dr. Inderveer Chana
(Professor)

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA, PUNJAB, INDIA
July 2017

Certificate

I hereby certify that the work which is being presented in the thesis entitled, "*An Enhanced MinMin Resource Scheduling Algorithm for Scientific Applications in Cloud Computing*", in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Software Engineering* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of *Dr. Inderveer Chana* and refers other researchers' work which are duly listed in the reference section.

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

Reshamkaur

(Resham Kaur)

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

Inderveer 19/7/17

(Dr.Inderveer Chana)
Professor, CSED

Acknowledgement

I would like to thank God for blessing me with all the strength and resources required to complete this task. I would like to express my deepest gratitude to **Dr. Inderveer Chana** for guiding me through the whole process and providing me with your knowledge and experience.

I am also heartily thankful to **Dr. Maninder Singh**, Associate Professor and Head, Computer Science & Engineering Department for motivation and providing uncanny guidance and support throughout the preparation of the thesis report.

I am also thankful to the entire faculty and staff members of Computer Science and Engineering Department for their direct-indirect help, cooperation, love and affection.

I would like to express my sincere thanks to CSIR (Council of Scientific and Industrial Research), Government of India to carry out my research work under the title "An enhanced MinMin Resource Scheduling Algorithm for Scientific Applications in Cloud Computing" with Scheme no. 22 (0693)/15/EMR-II.

I would like to thank my friend, Pratisha Sarma, for being there for me always.
Last but not the least, I would like to thank my family for their wonderful support and encouragement without which none of this would have been possible.

Resham Kaur 17/07/2017

Resham Kaur
(801531011)

Abstract

Cloud computing is a very well known term which is used commonly in almost every business or research field. It is a young but familiar technology which enables the clients to use its services without being bothered to know how the services run and leaving this job to the providers. Cloud providers offer three main services namely IaaS, PaaS and SaaS where IaaS stands for Infrastructure as a Service, PaaS stands for Platform as a Service and SaaS stands for Software as a Service. A user can choose an appropriate service which suits his/her requirements. The user has to pay according to the amount of resources used in a particular duration of time. This is why it is known as a pay-as-you-go model. It has opened up many new opportunities for researchers as well as business organizations. It is flexible as the users can scale up or scale down resources depending on their requirement.

Various scientific applications face the problem of starvation. Starvation is a condition where a task is in the ready queue, but due to some reason, is not able to get resources. Scheduling algorithms which are priority-based generate such conditions. In such scheduling algorithms, a job with low priority might have to wait for a long time in order to get the resources like processor or input output resources. Efficient scheduling algorithms are required in order to solve this problem thereby reducing the scheduling overhead cost and execution time.

This report discusses various scheduling algorithms and proposes a new scheduling algorithm known as improved minmin scheduling algorithm. The proposed scheduling algorithm considers task size as well as task arrival time. Hence it solves the problem of starvation faced by large sized tasks in the original MinMin scheduling algorithm. The proposed algorithm also performs prediction based on the execution time of the tasks.

Table of Contents

List of Figures

List of Tables

An Enhanced MinMin Resource Scheduling Algorithm for Scientific Applications in Cloud Computing

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

Master of Engineering
in
Software Engineering

By
Resham Kaur
(Roll No: 801531011)

Under the supervision of
Dr. Inderveer Chana
(Professor)

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA, PUNJAB, INDIA
July 2017

Certificate

I hereby certify that the work which is being presented in the thesis entitled, **"***An Enhanced MinMin Resource Scheduling Algorithm for Scientific Applications in Cloud Computing***"**, in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Software Engineering* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of *Dr. Inderveer Chana* and refers other researchers' work which are duly listed in the reference section.

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

(Resham Kaur)

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

(Dr.Inderveer Chana)
Professor, CSED

Acknowledgement

I would like to thank God for blessing me with all the strength and resources required to complete this task. I would like to express my deepest gratitude to **Dr. Inderveer Chana** for guiding me through the whole process and providing me with your knowledge and experience.

I am also heartily thankful to **Dr. Maninder Singh**, Associate Professor and Head, Computer Science & Engineering Department for motivation and providing uncanny guidance and support throughout the preparation of the thesis report.

I am also thankful to the entire faculty and staff members of Computer Science and Engineering Department for their direct-indirect help, cooperation, love and affection.

I would like to express my sincere thanks to CSIR (Council of Scientific and Industrial Research), Government of India to carry out my research work under the title "An enhanced MinMin Resource Scheduling Algorithm for Scientific Applications in Cloud Computing" with Scheme no. 22 (0693)/15/EMR-II.

I would like to thank my friend, Pratisha Sarma, for being there for me always.
Last but not the least, I would like to thank my family for their wonderful support and encouragement without which none of this would have been possible.

Resham Kaur
(801531011)

Abstract

Cloud computing is a very well known term which is used commonly in almost every business or research field. It is a young but familiar technology which enables the clients to use its services without being bothered to know how the services run and leaving this job to the providers. Cloud providers offer three main services namely IaaS, PaaS and SaaS where IaaS stands for Infrastructure as a Service, PaaS stands for Platform as a Service and SaaS stands for Software as a Service. A user can choose an appropriate service which suits his/her requirements. The user has to pay according to the amount of resources used in a particular duration of time. This is why it is known as a pay-as-you-go model. It has opened up many new opportunities for researchers as well as business organizations. It is flexible as the users can scale up or scale down resources depending on their requirement.

Various scientific applications face the problem of starvation. Starvation is a condition where a task is in the ready queue, but due to some reason, is not able to get resources. Scheduling algorithms which are priority-based generate such conditions. In such scheduling algorithms, a job with low priority might have to wait for a long time in order to get the resources like processor or input output resources. Efficient scheduling algorithms are required in order to solve this problem thereby reducing the scheduling overhead cost and execution time.

This report discusses various scheduling algorithms and proposes a new scheduling algorithm known as improved minmin scheduling algorithm. The proposed scheduling algorithm considers task size as well as task arrival time. Hence it solves the problem of starvation faced by large sized tasks in the original MinMin scheduling algorithm. The proposed algorithm also performs prediction based on the execution time of the tasks.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Internet has become a basic necessity in the present times where almost all of the records, data, and information are stored and operations are performed on them online. A lot of resources are required to store or to perform various operations on the data. Before cloud, various organizations needed to buy, maintain and replace these resources after some years. This had a lot of overheads like training the employees to use and maintain these resources, spend on various utilities for these resources and to replace the obsolete system with the new ones.

But with cloud coming into the picture, these overheads are significantly reduced. The organization just needs to pay for an independent pool of resources on a pay-as-you-go basis. The employees do not require any kind of special training to know how the resources work or how to maintain them. They just need to know how to work on these resources to get their mainstream work done. The organizations can spend this training expenditure capital somewhere else.

1.1 Cloud Computing Evolution

A shared pool of resources is referred to as a cloud, which can be rented by a client or multiple clients to ubiquitously access it. It is a model that is easy to maintain and manage. The clients can store their data on a third party server to make it accessible or they can store it on a privately owned cloud. Some organizations have a lot of data intensive jobs at their hands. They require extensive storage and computation resources. Cloud caters to these needs of the organizations.

In the mid 1970's, the concept of virtualization became well-known. Due to this, various operating systems could be run on one system. Then in the 1990's, the telecommunication companies came forward with the concept of virtual private networks and made it accessible to people at a comparatively reduced price. By doing this, they

were able to use the available network bandwidth more efficiently by switching traffic whenever needed. Some more experiments were carried out in order to provide computation power to a wider range of users in a time sharing environment. Attempts were made to augment platform, infrastructure and increase CPU efficiency for end users.

Cloud computing was finally introduced in the early 2000's where OpenNebula of NASA was the first ever software platform to deploy hybrid as well as private clouds in 2008.

After that Interactive Realtime Multimedia Applications on Service Oriented Infrastructures (IRMOS) was started which lead to the creation of real time cloud environment.

On 1 February, Amazon launched Windows Azure as its cloud service which later came to be known as Microsoft Azure. NASA and Rackspace Hosting came together to launch an open service cloud platform software in the month of July 2010. IBM and Oracle announced IBM Smart Cloud and Oracle cloud on1 March 2011 and 7 June 2012 respectively.

1.2 Cloud Service Models

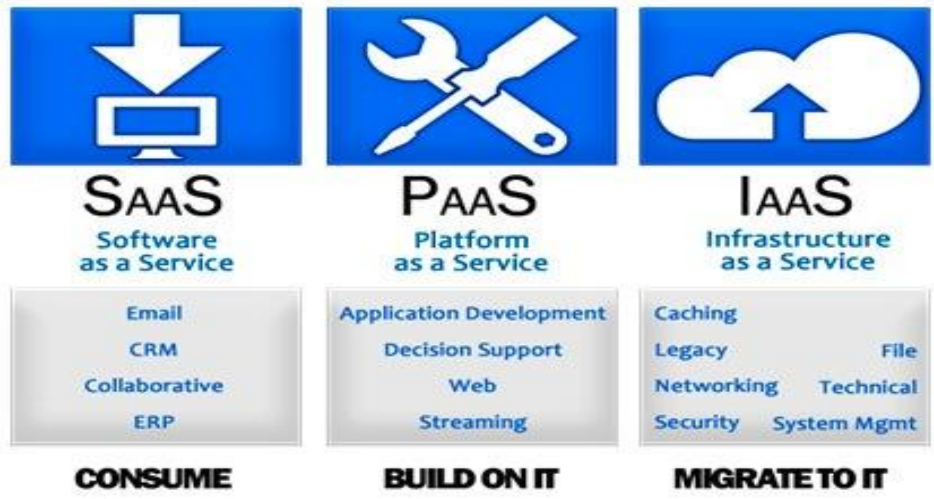

Figure 1.1: Cloud Service Model [41]

A set of services models can be used to access cloud computing. These services are to exhibit various properties in order to satisfy organizational requirements. If an

organization requires a service, then it can pick any service and get it accustomed to their requirements. Following are some of the service models explained in detail:

i. Infrastructure as a Service (IaaS): In this service model, the facility of controlling processes, managing the storage of data is provided to the user. Basically, the user is given the ability so that he can manage any other principle computing resources that help managing random software like applications and/or operating system is entrusted with the user. Some most widely known IaaS providers are Amazon EC2, Eucalyptus and Rackspace.

ii. Platform as a Service (PaaS): If a user selects this service model, the service provider will provide computing resources as well as the working environment which may constitute operating system, any execution environment for a specific programming language or any application platform. Minimal complexity is encountered by the user in putting together or maintaining the infrastructure. Google App Engine, Heroku and Microsoft Azure are some examples of PaaS providers.

iii. Software as a Service (SaaS**):** In this service model, the provider facilitates the user with the running application on the cloud resources. Management of cloud infrastructure is not done by the user. The user does not need any thin interfaces to run the application. Oracle, SAP and Microsoft are some of the SaaS providers.

1.3 Deployment Models

A deployment model depicts the type of environment of a cloud system. The environment may vary in size, ownership or access. In addition to the exact classification of cloud environment, access, size identifiers and assets is also depicted by the deployment models. It helps to relate the goals and nature of the cloud. Most organizations are ready to implement the cloud, as it reduces investment and controlling operating costs. To know if a deployment model is suitable for a particular application or not, it is necessary to know about the four deployment models.

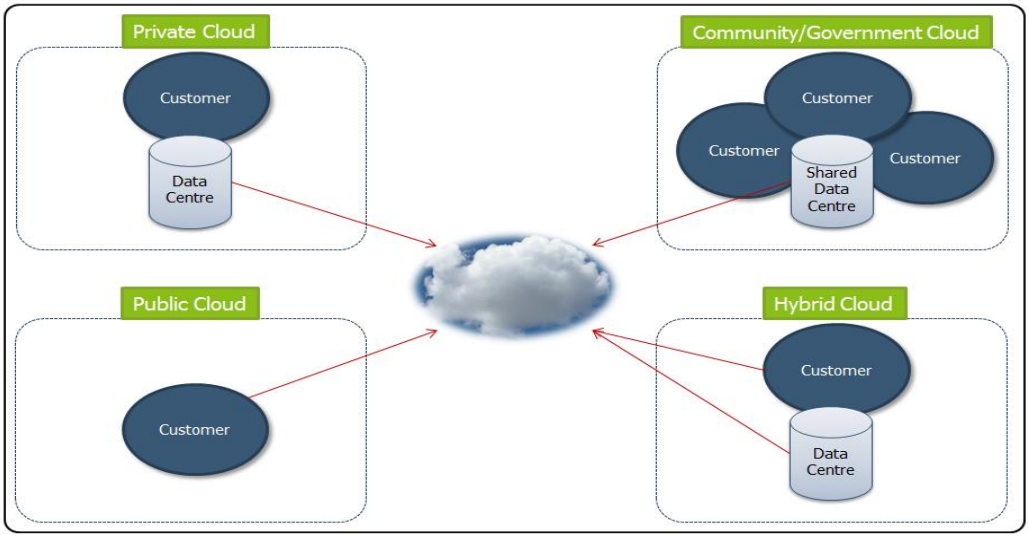

Figure 2: Cloud Deployment Models [41]

i. Public Cloud: In this deployment model, a network open for public usage is used to provide services to the users. Here, services and infrastructure is provided to multiple users. The users have no control over the infrastructure's location and neither can they differentiate between infrastructures from different locations. Public and private clouds are almost similar in the structural design except that private cloud has more security as compared to public cloud. The users share the total cost, therefore, it is an economical model. This model is well suited for business applications which have to manage a lot of traffic, the applications which run on SaaS model and the applications which are widely used by the public.

ii. Private Cloud: This type of cloud can only be accessed by authorized person of an organization. It is secure as data is protected by a firewall administered by that organization. This cloud is suitable for business applications which require transfer of confidential data or which consist of mission critical operations or having security requirements like security alarm, management requirements.

iii. Hybrid Cloud: It is an integration of public and private cloud. This model provides benefits of multiple models. The user can perform aggregation, customization with another package or assimilation in order to increase the capability. Resource management can either be in-house or done by providers externally. As per the need of the user, the workload can be exchanged in between private and public cloud. The public cloud can contain the non-critical workloads where security is not a major

concern. Whereas, the critical workload which require security can be placed in private cloud.

iv. Community Cloud: It is accessed by authorized people of more than one organization belonging to a specific community. The security, privacy and performance concerns of the community members are similar. The cost is shared among the community members. This kind of cloud is well suited for business organizations which work on a research problem that requires extensive computing and/or storage resources or business organization which work on joint ventures.

1.4 Cloud Applications

Cloud has many applications in many fields. These applications can be categorized into two main types: business applications and scientific applications.

i. Business Applications: These are the applications which comprise of a set of programs that are used to perform some specific business activity. These business activities include measuring productivity, performing other business functions and increasing productivity. There are some business applications which can perform functions like query, input and modify data using their own GUI (Graphical User Interface). But some business applications do not have their own GUI. So they are performed in the form of batch processes. These business applications perform very different tasks and thus are not transferable in most cases. But given that similar functions are to be performed, one application can be modified to fit the requirements of another business. Some of the examples of business applications are Microsoft Office, human resource management and enterprise resource planning.

ii. Scientific Applications: These applications that simulate the real world activities using mathematical models. These applications transform real world objects into mathematical models and encode their behavioral traits in formulas. These formulas are simulated to duplicate their real world activities. The applications can be computation intensive or data intensive. A computation intensive scientific application has a lot of processing steps. It requires a lot of computation resources. For example, image processing, bioinformatics and data mining. Whereas, a data

intensive application has storage requirements and need extensive storage resources. For example, meteorological applications like montage.

1.5 Benefits of Cloud Computing

Cloud computing has the following advantages:

i. Flexibility: The cloud is flexible in the sense that it can respond to changing needs of the client. If the client has dynamic resource requirements that tend to change after a time period then cloud is the best option. Cloud resources can be scaled up or down as and when required by the client. So the cloud can be used for growing businesses or businesses having fluctuating demands.
ii. Data Recovery: If the data is stored in a cloud, the data will be distributed and duplicated across different locations and servers. So if one server fails due to any natural or man-made disaster, it can be recovered from other server.
iii. Easy Maintenance: As the cloud is handled by a third party, it is the sole responsibility of the provider to maintain and update the system over time. This way the client saves maintenance cost. The client need not bother about knowing the latest security or performance updates.
iv. Cost Savvy: The client need not buy the infrastructure. He/she just needs to pay the rent for using the services according to a pay-as-you-go model. This saves the capital investment on the infrastructure.
v. Increased Collaboration: Cloud contains all the data of a particular organization or research center. So it becomes easy for the organization members to access the data and work on it simultaneously in a real time environment.
vi. Security: The data of an organization is safe on cloud. The data is not kept on a single machine like a laptop or a desktop. But it is kept in servers, duplicated and distributed across locations. Even if the machine from which the data is being accessed goes missing or gets corrupted, the data can be removed remotely from that machine. This way it helps to avoids unauthorized access to data.

1.6 Research Motivation

Resource management is carried on in different steps of workloads and resources starting from the submitting of the workload to the execution of the workload. Cloud resource management consists of two steps: a) resource planning; and b) resource provisioning. Resource provisioning is defined as the process of identifying enough resources for a specific workload depending on the QoS (Quality of Services) requirements provided by cloud consumers. Whereas resource planning is mapping the performance of workloads of cloud-based consumers based on resources selected on the basis of acquirement. First, the consumer cloud submits a request to perform the workload as a set of workload details. Then these details form the basis for the broker to find out the appropriate resource for a specific task and to find out the viability of the supply of resources depending on QoS requirements [10]. Broker sends queries to the scheduling unit for planning. After the supply of resources, the broker's duties include: the release of the resources which are spare, back to the pool of resources, to monitor the performance of the system, it contains information about procured resources and to add up or withdraw resources. After the supply of resources, resource planning is done in the second step.

The workloads that are submitted are operated on in the workload queue while the rest of resources are in the pool of resources. At this stage, the planner forms the provision of resources for the provided workloads. The workloads resume resource pool after execution of workloads. According to QoS (Quality of Service) needs, allocation of suitable count of resources for workloads is a challenging problem. If resources are to be scheduled in an efficient manner, QoS requirements need to be taken into account [3]. Scheduling of resources is an area which has a lot of scope as it has a direct impact on the time of execution and cost of resource. For different algorithms for scheduling the resources, there are separate criterion and parameters. This research discusses the second resource management stage i.e. resource scheduling.

1.7 Organization of Thesis

The rest of the thesis is organized as follows:

Chapter 2 – It entails the literature survey done in detail about the challenges faced by scientific applications, solutions implemented to tackle the challenges and the importance of efficient scheduling.

Chapter 3 – It discusses the problem statement.

Chapter 4 – It presents the solution proposed to the given problem statement by discussing the algorithm.

Chapter 5 – It represents all the results achieved by implementing the proposed system.

Chapter 2

Literature Survey

It is very important for providers to provide quality of service to the clients so that the clients can have maximum profit. The quality of service is decided on the basis of various parameters. The parameters are specified in a document called an "SLA" which stands for Service Level Agreement. An SLA entails Service name, Clearance information (with location and date), Contract duration, Description/ desired customer outcome, Communication between customer and service provider, Service and asset criticality, Service times, Required types and levels of support, Service level requirements/ targets, Technical standards/ specification of the service interface, Responsibilities, Pricing model, Change history, List of annexes and references and Glossary. So SLA specifies all the services and their details that the provider and the customer agreed upon. Also, scientific applications either have a lot of computations to perform or have a lot of data to analyze. For this reason, scheduling becomes all the more complex. Scheduling overhead increases significantly in such applications. This increases the overall cost of operation. If scheduling is inefficient, the tasks may take longer to complete than they should. This in turn increases the execution time. So if SLA is to be guaranteed, efficient scheduling mechanisms need to be developed. For developing an efficient scheduling technique, the challenges faced by scientific applications need to be studied and analyzed in detail.

2.1 Research Questions

The literature review depicts the present scenario of scheduling of scientific applications in cloud computing by answering the following questions:

i. What are the major challenges faced by scientific applications?
ii. How can these challenges be tackled?
iii. What are the efficient solutions presently being used?
iv. How can scheduling help to overcome these challenges?
v. What is the present status of scheduling techniques?

vi. Which new scheduling technique can be implemented in order to overcome the challenges?

In this thesis, following is the table enlisting all the keywords needed to find the answers to the research questions:

Table 1.1: Keywords and Synonyms

Keywords	**Synonyms**
Cloud Computing	Distributed computing
Scientific Applications	Data intensive application, Compute intensive application
Prediction Models	Prediction techniques, Resource prediction
Scheduling	Resource provisioning, Resource allocation

2.2 Challenges faced by Scientific Applications

Qi Zhang, et al.,[1], have discussed various cloud technologies and commercial products in detail in "Cloud Computing: State-of-the-Art and Research Challenges". The commercial products have been compared on parameters like cloud provider, computing classes, target applications, computation, storage and auto-scaling. Research challenges such as service provisioning automation, server consolidation, virtual machine migration, management and analysis of traffic, management of energy, data security, technologies for storage and management of data and cloud architectures have been discussed.

Christian Vecchiola, et al., [2], have given a comparison of cloud computing solutions such as Amazon EC2, Google AppEngine, Microsoft Azure, and Manjrasoft Aneka in "High-Performance Cloud Computing: A View of Scientific Applications". The comparison is done on the basis of various features such as service type, value added provider, if PaaS, ability to deploy on third party IaaS, platform (OS and runtime), virtualization, deployment model, interface for user access. The Aneka architecture, deployment model and application model are discussed in detail. Programming models like task model, map-reduce model, thread model, parameter sweep model, workflow,

message passing interface actors have been compared on the basis of execution services, applications and execution units.

A case study of two applications namely classification of datasets of gene expression using Aneka cloud and functional magnetic resonance imaging workflows has also been presented. The challenges faced in these applications have been discussed.

Yong Zhao, et al.,[3], have given many opportunities that the cloud has brought in, such as better utilization of resources, improved responsiveness thereby improving user experience, enabling a generation of collaborative scientific workflows and reducing the cost in challenges and opportunities in running scientific workflows on cloud. Many challenges faced by scientific application like architectural challenges (for e.g. reproducibility support; software tools integration and distributed and heterogeneous services; user interaction support and interface customizability; heterogeneous distributed data product management; failure handling and interoperability and workflow monitoring), services and challenge for integration of tools, high-end computing support language-conversion challenge, challenge for compute intensive applications, challenge for data management (mainly due to increasing dataset size and data locality to minimize movement of data), applications and service management challenge (managing large number of service instances).

Jens-Sönke Vöckler, et al., [4], a scientific application is executed on FutureGrid, Amazon EC2 cloud and NERSC's Magellan cloud in "Experiences using Cloud Computing for a Scientific Workflow Application". There results have been compared and analyzed to comprehend various challenges that came across during the process. Pegasus workflow management system has been used to execute a scientific application which was used to process data from Kepler project by NASA to find out planets similar to the earth.

Suraj Pandey, et al., [5], have focused on reducing the total operating cost of a scientific application in "A Particle Swarm Optimization-Based Heuristic for Scheduling Workflow Applications in Cloud Computing Environments". Particle Swarm Optimization (PSO) algorithm has been applied in combination with a heuristic

scheduling technique. The fitness function is calculated by the PSO. The total cost calculated is the addition of cost of execution and the transfer cost of data. The PSO technique ensures that the highest of all the task costs is reduced. Then heuristic scheduling is applied to schedule the resources according to the mapping formed by the PSO. The results of PSO were compared to that of best resource selection algorithm where PSO produces better results.

Alexandru Iosup, et al., [6] have explained the differences between the actual scope field of cloud and the requirements of scientific applications in "Performance Analysis of Cloud Computing Services for Many-Tasks Scientific Computing". It evaluates cloud and checks the capability of a cloud to run a scientific application efficiently. The evaluation is done by quantifying the number of users that require scientific computing services followed by evaluating four cloud services mostly used for scientific computing and then comparing cloud performance with grid and parallel computing for scientific applications.

Simon Ostermann, et al., [7] have discussed about various features of cloud computing which help ease the execution of scientific applications in "A Performance Analysis of EC2 Cloud Computing Services for Scientific Computing". It evaluates these features by executing scientific workloads like SJSI, MJSI and SJMI on Amazon EC2 cloud. Benchmarks like Lmbench, Bonnie, CacheBench and HPC Challenge Benchmark are used to evaluate the performance of EC2 cloud for scientific application.

Ewa Deelman, et al., [8], have discussed dependency of cost on execution models in "The cost of doing science on the cloud: the Montage Example". Here cost is calculated as a function of number of processors. The cost of executing montage workflow has been estimated by running simulation using GridSim tool. Three data management models have been explored namely Remote I/O, Regular and Dynamic Cleanup. Three montage workflows are executed i.e. montage degree 1 (203 application tasks), montage degree 2 (731 application tasks) and montage degree 4 (3027 application tasks). The cost of running each of these in each of the data management models have been compared graphically. To maintain the trade-off between the number of processors and reduction in the execution time.

Christina Hoffa, et al., [9], have run four different sized workflows in four different environments in order to compare the performances in "On the Use of Cloud Computing for Scientific Workflows". Tools like pegasus WMS, DAGman, GridFTP, condor and GRAM and their roles have been discussed. Four different montage mosaics i.e. 0.1 degree2 (50 application tasks), 0.2 degree2 (74 application tasks), 0.5 degree2 (150 application tasks), 1.0 degree2 (426 application tasks) are executed on local machine, multiple virtual machines, a local cluster and a virtual cluster.

Scott Callaghan [10], have discussed and tackled the problems relating to managing a workflow in "Scaling Up Workflow-Based Applications". To discover, study and eliminate the problems, CyberShake workflow has been implemented on TeraGrid. This workflow requires PSHA (Probabilistic Seismic Hazard Analysis) calculation estimates. PSHA requires ground motions caused by past earthquakes as input. The ground motions are calculated by CyberShake using 3D ground motion simulations with analytic wave propagation model.

Table 2.1: Challenges Faced By Scientific Applications in Cloud Computing

Author and Year	**Scientific Application**	**Challenges**
Qi Zhang, Lu Cheng, Raouf Boutaba 20 April 2010	GoogleApp Engine[1]	Automation of Service Provisioning, Migration of, Energy management and Server Consolidation
Christian Vecchiola, Suraj Pandey, and Rajkumar Buyya. Melbourne	FMRI (Functional Magnetic Resonance Imaging (FMRI) brain imaging workflow[2]	Scheduling, Prediction, Computation Time, Pricing.
Yong Zhao, Xubo Fei, I. Raicu, Shiyong Lu and 18 November 2011	Amazon's mapReduce Workflows [3]	Workflow Scheduling and Management, Computation
Jens-Sönke Vöckler, Gideon Juve, Ewa Deelman, Mats Rynge, Bruce Berriman 08 June 2011	Abstract Workflow represented by DAG (Direct Acyclic Graph) [4]	Scheduling, Computation
Suraj Pandey, Linlin Wu, Siddeswara Mayura Guru, Rajkumar Buyya 2010.	Bio Informatics workflow [5]	Scheduling, Cost of execution
M. Nezih Yigitbasi, Radu Prodan, Thomas Fahringer, Dick Epema june 2011	Grid Workload Archives, Parallel Workload Archives [6]	Cost of execution, Response Time, Storage
Simon Ostermann, Alexandria Iosup, Nezih Yigitbasi, Radu	SJSI workload, SJMI workload, MJSI workload,	Computation Time, Response Time, Throughput,

Prodan, Thomas Fahringer, Dick Epema 2009	SJSI/MJSI workload [7]	Reliability
Ewa Deelman, Gurmeet Singh, Miron Livny, Bruce Berriman. John Good. 2008.	Montage 1 degree [8]	Computation cost, Execution Time
Christina Hoffa, Gaurang Mehta, Timothy Freeman, Ewa Deelman, Kate Keahey, Bruce Berriman, John Good. 2008.	Montage[9]	Scheduling overhead, Computation Time
Scott Callaghan, Ewa Deelman, Dan Gunter , Gideon Juve,Philip Maechling,Christopher Brooks, Karan Vahi, Kevin Milne,, Robert Graves, Edward Field,David Okaya, Thomas Jordan. August 2009	CyberShake workflow [10]	Scaling up resources, Execution Time
Ewa Deelman, Scott Callaghan, Edward Field, Hunter Francoeur, Robert Graves, Nitin Gupta, Vipin Gupta, Thomas H. Jordan, Carl Kesselman, John Mehringer, Philip Maechling, Gaurang Mehta, Karan Vahi, David Okaya, Li Zhao December 2006	CyberShake workflow11]	Execution time, Computation
Robert Graves, Thomas H. Jordan,Scott Callaghan, Ewa Deelman,Edward Field,Gideon Juve, Carl Kesselman,Philip Maechling,Gaurang Mehta,Kevin Milner,David Okaya,Patrick Small,Karan Vahi	CyberShake [12]	Computation, Storage

2.3 Existing Prediction Models

The above challenges like execution time, operational cost, computation problem, scaling of resources point towards one solution, that is, predicting the user requirements in advance and scheduling the resources accordingly. Following are some of the prediction models implemented to solve the above problems.

Christopher Moretti, et al., [13] have discussed about modified all pairs abstraction technique in order to predict resource requirements in “All-Pairs: An Abstraction for Data-Intensive Cloud Computing”. It discusses about various performances related drawbacks like cost of dispatching tasks, failure probability for shared hardware,

maintaining the number of suitable compute nodes and how to distribute the data and communicate it through the system of using the original all pairs abstraction method. The major principle of this technique is to communicate to the system the data needed by a job and the number of computations required by it. This way the system does not run blind and has ideas on how to distribute the jobs among the resources. This is done in four steps where first all the values like transfer time, computation time and turnaround time. Then all the data is distributed via a file distributor module. After that, scripts are sent to the nodes to execute the data. In the end, all the results are collected.

Lavanya Ramakrishnan, et al., [14], have used LEAD (Linked Environments for Atmospheric Discovery) workflow as an example to see if VGrADS (Virtual Grid Application Development Software) is able to perform better scheduling and fault tolerance in "VGrADS: Enabling e-Science Workflows on Grids and Clouds with Fault Tolerance". LEAD is an application that generates as well as processes a large amount of data. It has various heterogeneous components that communicate with each other using different exclusive interfaces. This is why, scheduling poses a challenge. But VGrADS provides a common interface which helps solve the problem of scheduling. Here the components of VGrADS are discussed in detail. VGrADS is applied on LEAD which is executed on grid as well IAAS cloud infrastructure.

Norman W. Paton, et al.,[15], have discussed and explored the concept of utility functions in "Optimizing Utility in Cloud Computing through Autonomic Workload Execution". Utility function helps to measure the utility of the services provided by the cloud. The utility function is made to work with the optimization algorithms so as to use the service to its maximum potential to maximize the efficiency. This technique has been applied to a utility function created to consider response time and profit.

Zhenhuan Gong, et al.,[16], have discussed the problem of deciding the number of resources to be scaled in order to maximize profit and avoid violations in the SLO(Service Level Objectives) in "PRESS: PRedictive Elastic ReSource Scaling for Cloud Systems". A new technique has been proposed in order to do so. This technique is PRESS (PRedictive Elastic reSource Scheduling). This technique works by processing the request signals and identifying a pattern. This pattern helps to predict the future

requirements. The experiments are run on NCSU's virtual communication lab where there is dual core of each host, CentOs 5.2 64 bit. RUBiS benchmark has been used to check for any violations.

Fortes, et al.,[17], have discussed and compared various machine learning techniques like k-nearest neighbor algorithm, linear regression, decision tree algorithm, artificial neural networks, support vector machine algorithm and time series methods to predict the resource needs in "On the Use of Machine Learning to Predict the Time and Resources Consumed by Applications". This paper aims at finding best algorithms to generate opimum results for BLAST (Basic Local Alignment Search Tool) and RAxML (Randomized Axelerated Maximum Likelihood) applications. It also proposes a new technique called redicting query runtime regression. This technique predicts three attributes i.e. execution time, memory and output size. In order to implement this technique, BLAST and RAxML applications are run on the nodes. All the experiments are run on WEKA.

Ming Mao and Marty Humphrey [18], have emphasized the importance of completing the task before deadline in "Auto-Scaling to Minimize Cost and Meet Application Deadlines in Cloud Workflows". This paper discusses the scaling and scheduling problem. It uses the Earliest Deadline First (EDF) algorithm in order to solve this problem. It is under cost is checked by pre-analyzing the jobs in order to find out their deadlines. This technique was implemented on pipeline, parallel and hybrid applications and the results were compared. The results were compared to that of two techniques namely GAIN and Greedy. Four workload models – growing, cycle/busting, stable and on/off are tested for each of the three applications for the three techniques.

Zhiming Shen, et al., [19], have discussed the problem of deciding to upscale or down scale and to what amount due to varying workloads in "CloudScale: Elastic Resource Scaling for Multi-Tenant Cloud Systems". CloudScale is introduced as a solution to this problem. CloudScale performs auto scaling by predicting the resource requirements. For over-estimation errors, online resource prediction is used. For under-estimation errors, reactive strategy for error correction and online adaptation for padding is applied. The technique is applied to two traces – one form world cup 98 web server and second from

EPA web server. RUBiS benchmark is used to evaluate the response time, conflict resolving capacity, energy consumption and performance of the system.

Qi Zhang [20],have emphasized the problem of saving the energy of huge data centres in "Dynamic Energy-Aware Capacity Provisioning for Cloud Computing Environments". Here online monitoring of resource usage is done and tasks are assigned accordingly. Also feedback is given to find out if there are servers that can be switched off to save energy. The model uses Auto Regressive Integrated Moving Average (ARIMA) in order to predict the resource requirements.

Hiep Nguyen, et al.,[21],have proposed agile technique to predict the workload and schedule resources in "AGILE: elastic distributed resource scaling for Infrastructure-as-a-Service". Agile provides an efficient elastic way to schedule distributed resources. Online profiling is used to record patterns of incoming requests and then these patterns are further processed into wavelet transforms. Markov model is used to make predictions using the wavelets. Proactive virtual machine cloning is performed based on the predictions. These experiments are performed on Cassandra value key-store, evaluated by RUBiS benchmark and input traces are taken from Google cluster.

Table 2.2: Existing Prediction Models

Author and Year	**Name of Scientific Application**	**Prediction Technique Applied**	**Parameter Improved/ Outcome**
DouglasThain, Patrick J. Flynn	4010x4010 standard biometric workload and 1000x1000 data mining workload[13]	Abstract all pairs shortest path algorithm	Reduced turnaround time and improved storage efficiency
Lavanya Ramakrishnan, Charles Koelbel, Yang-Suk Kee,Rich Wolski, Danioel Nurmi, Dannis Gannon, Ghaziano, Orbetelli, AsimYarKhan, Anirban Mandal, T.Mark Huang, Kiran Thyagaraja, Dmitrii Zagorodnov. 2009	Mesoscale meterology workflows in the Linked Environments for Atmospheric Discovery (LEAD) [14]	Virtual grid execution system (vgES)	Reduced execution time
Norman W. Paton, Marcelo A. T. de Araga˜o, Kevin Lee, Alvaro A. A. Fernandes, Rizos Sakellariou. 2009.	Workflow consisting of a set of queries [15]	Utility -based techniques	Reduced response time and increased profit

Zhenhuan Gong, Xiaohui Gu, John Wilkes. 2010. s.l. : IEEE, 2010,	statistical learning algorithms [16]	PRedictive Elastic resource Scaling (PRESS)	Reduced over-estimation error and reduced resource wastage
Fortes, Andréa Matsunaga and José. 2010	BLAST and RAxML [17]	PQR2 algorithm	Improved accuracy
Ming Mao, Marty Humphrey. 2011	Stable, growing, cycle and bursting workloads [18]	Scaling consolidation scheduling	Reduced overall cost significantly
Zhiming Shen, Sethuraman Subbiah, Xiaohui Gu, John Wilkes. 2011	CPU and memory intensive applications [19]	CloudScale algorithm	Increased accuracy of predicting the frequency/voltage setting
Qi Zhang, Shuo Zhang, Quanyan Zhu, Mohamed Faten Zhani, Raouf Boutaba,Joseph L Hellerstein. 2012	CPU and memory intensive applications [20]	Predictive Control (MPC) algorithm	Reduced prediction error, optimized memory usage
Hiep Nguyen, Zhiming Shen, Xiaohui Gu, Sethuraman Subbiah, John Wilkes. 2013	Memory intensive applications[21]	Agile	Optimized resource distribution

2.4 Existing Scheduling Models

After implementing prediction models, efficient scheduling technique is required to allocate resources to the user tasks accordingly. Following are some scheduling techniques currently being used for scientific applications.

In [22], Ye Hu et. al. have divided the jobs into two categories, that is, shared server allocation and dedicated server allocation jobs. So a pool of resources is assigned to each of these two classes of jobs. This paper contains the comparison of number of servers required to fulfill the objectives in shared allocation and in dedicated allocation. It applies FCFS (First Come First Serve) scheduling algorithms to dedicated allocation jobs. Whereas, on shared allocation, it applies various scheduling algorithms like FCFS, probability dependent priority (PDP) and head of the line (HOL) priority. These results are compared to see which allocation strategy fulfills the user objectives with minimum number of servers.

In [23], Chenhong Zhao et al. have introduced a task scheduling genetic algorithm. Here, a task is divided into processes. Each task is supposed to have n processes and there are n processors. A task can select or choose one task at a time and that process can be assigned to one processor. In this scheduling algorithm, a selection procedure is introduced in order to select the highest fit value. Crossover of population is done to create new generation. This way a schedule is made according to which, only the fittest chromosome is implemented.

In [24], Jinhua Hu et al. have introduced a genetic scheduling algorithm. Here chromosomes are represented in the form of a spanning tree. The selection of order of nodes in the spanning tree is done according to the average virtual machine load. The selection of a node is done by using fitness function. Crossover operation is performed by selecting two chromosomes with high fitness value and then making a spanning tree out of them.

In [27], Xin Lu and Zilong Gu have proposed a mdel for resource scheduling which adapts to the changing load. If load on a cluster increases from the threshold value, the controller is responsible to find out which resource is getting affected the most by the load. This resource is called a hot spot. Once the hot spot is found, the rest of the nodes start to calculate their remaining resource time with the help of the formula. Then the node with minimum value is assigned the task of hot spot. If no such node is found, nodes of other clusters are checked and same procedure is repeated.

In [28], Wei Wang et al. have proposed a reliability based scheduling algorithm called dynamic trusted scheduling where trust forms the basis of scheduling. Bayesian method is used to identify the trustworthy traits of nodes in order to check if they are working and coordinating properly or not. The DLS (Dynamic Level Scheduling) algorithm has been extended by using the trust model.

In [29], Liang Luo et al. have proposed an energy efficient scheduling algorithm. This algorithm helps to adjust the frequency of various resources in order to save energy. In this paper, the relation between consumption of energy and components of infrastructure has been discussed. It is also mentioned if the frequency keeps on increasing, after a certain point the energy consumption will start increasing too. So this technique maintains a balance between the frequency of the resources and the power consumption.

In [30], a priority based scheduling algorithm has been proposed by Kamalakar. M, Moulika.T. The AHP (Analytical Hierarchy Process) forms the basis of this algorithm. It focuses on the user

provided priority. It schedules the processes not on the basis of finish time, but on the basis of priority of that task.

In [31], Dzmitry Kliazovich et al. have discussed the importance of traffic handling in cloud. A new scheduling algorithm named e-STAB has been proposed. This algorithm considers the load balancing of traffic across the servers and thus tries to reduce the energy consumption by providing resource allocation accordingly. It aims at reducing delays caused due to poor communication and lss of packets due to congestion.

In [32], Mohamed Abu Sharkh, Abdelkader Ouda, and Abdallah Shami have emphasized on the need to consider network resource utilization in addition to resource utilization. They have proposed three virtual machine reservation techniques and two connection reservation and have tried to reduce the effect of users requesting the resources very late. The first virtual machine reservation technique is equal time distribution technique, where resources are allocated to servers according to their share of time reserved. The second technique proposed in the node distance technique. In this technique, two server nodes at maximum distance from each other are allocated virtual machines. In the third technique, namely, resource based distribution technique, virtual machines are categorized into three types: (1) High Memory Extra Large, (2) High CPU Extra Large, (3) Standard Extra Large. The user can request any kind of virtual machine, the virtual machine of that type with highest available resource will be allocated to the server. The two connection reservation techniques are duration priority technique where the smallest connection request has the highest priority and greedy algorithm where connection requests with early request start time have the highest priority.

In [33], AV.Karthick, Dr.E.Ramaraj, R.Ganapathy Subramanian proposed a multi-queue scheduling algorithm. This algorithm aims at utilizing the unused memory space and reduces fragmentation. The traditional scheduling algorithms like shortest job first, combinational backfill and First Come First Serve and their drawbacks have been discussed in detail. MQS is introduced to overcome these drawbacks.

In [34], Saurabh Bilgaiyan, Santwana Sagnika and Madhabananda Das have proposed a cat swarm optimization technique which is aimed at reducing the cost of execution of each operation as well as to reduce the transmission cost. The cats are kept in two modes i.e., seeking mode and tracing mode. In seeking mode, the cats are at a fixed position whereas in tracing mode, they are moving at some specific velocity. The fitness of a cat is checked based on its position. This way at the end of each loop, the population with best fitness is given. Hence the allocation that will

result in minimum cost is given out. These results are compared to that of particle swarm application by implementing them both on a hypothetical workflow.

In [35], Hai-Hao Li and Zhi-Hui Zhan have proposed a new cost minimization and deadline-constrained workflow scheduling (CMDCWS) technique. They have discussed in detail about the particle swarm optimization (PSO) technique which was combined with (CMDCWS) in order to enhance the CMDCWS algorithm. They have pointed out various drawbacks in PSO combined CMDCWS. To overcome these drawbacks, they have proposed a new technique called renumbered PSO with CMDCWS. This technique involves each task being represented as a particle position and their computation capability being represented by the dimensions of the particle. The results of both the algorithms have been compared.

In [36], Zhaobin Liu et al. have discussed the earliest finish time scheduling algorithms in detail. Their drawbacks have also been studied. Therefore to remove these drawbacks, a new scheduling methodology called fuzzy clustering method is proposed. It helps to reduce the idle waiting time of resources and thus helps to increase the productivity. This method preprocesses the resources and creates a list of available resources. This way scheduling overhead is greatly reduced. This list is then used for scheduling by using the proposed scheduling technique of task duplication scheduling.

In [37], Mohammed Abdullahi, Md Asri Ngadi , Shafi Muhammad Abdulhamid have proposed a symbiotic Organism Search (SOS). Three basic behaviors of mutualism (where both the organisms gain from their relationship), commensalism (where only one of the two organisms profit from their relationship) and parasitism (where one organism benefits and the other bears a loss from the relationship) of any organism are considered. These three behaviors are transformed into mathematical model and are simulated. These three behaviors are implemented on virtual machines and resources.

In [38], Dzmitry Kliazovich et al. have emphasized on considering the data communication operations when considering the time of execution. They have proposed a communication aware directed acyclic graph (CA-DAG) scheduling model. In this model, vertices represent computation task just like normal DAG model. In addition to this, it also represents communication tasks through vertices. This way, the network resource utilization is also kept under check. The communication delays can be reduced significantly. The results of DAG models were compared to the CA-DAG model.

In [39], Rui Zhang, Kui Wu and Jianping Wang have emphasized on cost effective computing. They have proposed a randomized online stack centric scheduling model in order to efficiently reduce cost of operation. This scheduling model uses a cost function to check the cost of operation.

Table 2.3: Existing Scheduling Models

Author(s)	**Research Paper Title**	**Scientific Applications**	**Techniques Proposed**	**Parameter/ Outcome**
Ye Hu, Johnny Wong, Gabriel Iszlai and Martin Litoiu	Resource Provisioning for Cloud Computing [23]	Multi Server Queuing Model	Probability Dependent Priority	No. of servers
Chenhong Zhao, Shanshan Zhang, Qingfeng Liu, Jian Xie and Jicheng Hu	Independent Tasks Scheduling Based on Genetic Algorithm in Cloud Computing [24]	Mapping and Scheduling of macro data flow graphs	Genetic Algorithm	Response Time
Jinhua Hu, Jianhua Gu, Guofei Sun and Tianhai Zhao	A Scheduling Strategy on Load Balancing of Virtual Machine Resources in Cloud Computing Environment [25]	OpenNebula Agent	Load Balancing on the basis of Genetic Algorithm	Migration Cost
Hai Zhong, Kun Tao and Xuejie Zhang	An Approach to Optimized Resource Scheduling Algorithm for Open-source Cloud Systems [26]	Experimental Data	Improved Genetic Algorithm For Resource Scheduling	Productivity
Zhongni Zheng ,Rui Wang, Hai Zhong and Xuejie Zhang	An Approach for Cloud Resource Scheduling Based on Parallel Genetic Algorithm [27]	Experimental Data	Parallel Genetic Algorithm	Utilization of resources and speed of allocation

Xin Lu and Zilong Gu	A load-adapative cloud resource Scheduling model based on Ant Colony Algorithm [28]	Jmeter	Ant colony based load-adaptive scheduling model	Resource allocation efficiency and speed and adapts to changing workloads
Wei Wang, Guosun Zeng, Daizhong Tang and Jing Yao	Cloud-DLS: Dynamic trusted scheduling for Cloud Computing [29]	Web and Image data analysis	Cognitive trust model based on Bayesian method	Length of scheduling and no of successful executions
Liang Luo, Wenjun Wu, Dichen Di,Fei Zhang, Yizhou Yan, and Yaokuan Mao	A Resource Scheduling Algorithm of Cloud Computing based on Energy Efficient Optimization Methods [30]	iozone , netperf and Spec 2006	Job Classification And Resource Scheduling Algorithm	Execution Time, Energy Consumption and Resource Usage
Shamsollah, Ghanbari, Mohamed Othman	A Priority based Job Scheduling Algorithm in Cloud Computing [31]	Experimental Data	Job Scheduling Based on Priority	Analytical Hierarchy Process (AHP)
Dzmitry Kliazovich, Sisay T. Arzo, Fabrizio Granelli, Pascal Bouvry and Samee Ullah Khan	e-STAB: Energy-Efficient Scheduling for Cloud Computing Applications with Traffic Load Balancing [32]	Network tasks	e-STAB scheduler	Energy consumed
Mohamed Abu Sharkh, Abdelkader Ouda, and Abdallah Shami	A Resource Scheduling Model for Cloud Computing Data centers [32]	NFS (National Science Foundation)Network Application	Heuristic Techniques For Scheduling	Request Blocking Percentage
AV.Karthick, Dr.E.Ramaraj, R.Ganapathy Subramanian	An Efficient Multi Queue Job Scheduling for Cloud Computing [33]	Experimental Data	Multi-Queue Job Partitioning	Load Balancing
Saurabh Bilgaiyan, Santwana Sagnika, Madhabananda Das	Workflow Scheduling in Cloud Computing Environment Using Cat	Hypothetical Workflow with task size varying within the range of 64 MB to 1024	Cat Swarm Optimization Scheduling Algorithm	Operational Cost

	Swarm Optimization[34]	MB.		
Hai-Hao Li, Yu-Wen Fu, Zhi-Hui Zhan and Jing-Jing Li	Renumber Strategy Enhanced Particle Swarm Optimization for Cloud Computing Resource Scheduling [35]	Test cases consisting of large (50 tasks), middle (100 tasks) and large scale (200 tasks)	cost-minimization and dead line-constrained workflow scheduling (CMDCWS) model	Cost and Deadline
haobin Liu, Wenyu Qu, Weijiang Liu, Zhiyang Li and Yujie Xu	Resource preprocessing and optimal task scheduling in cloud computing environments [36]	Experimental Data	Fuzzy Clustering Based Resource Preprocessing	Normalised Length of Scheduling and Speedup
Mohammed Abdullahi, Md Asri Ngadi, Shafi'I Muhammad Abdulhamid	Symbiotic Organism Search optimization based task scheduling in cloud computing environment [37]	Self Generated Data	Discrete symbiotic organisms search algorithm	Makespan Time, Response Time and Degree of Imbalance Among VM's
Dzmitry, Kliazovich, Johnatan, E. Pecero, Andrei Tchernykh, Pascal Bouvry Samee, U. Khan Albert, Y. Zomaya	CA-DAG: Modeling Communication-Aware Applications for Scheduling in Cloud Computing [38]	Communication intensive applications Cloud Gaming, Remote Desktop and Cloud Synchronization	Communication-Aware DAG Model	Energy Consumption and Efficiency
Rui Zhang, Kui Wu, Minming Li, and Jianping Wang	Online Resource Scheduling Under Concave Pricing for Cloud Computing [39]	Google Cluster Data	Scheduling Under A Linear Function With A Fixed Activation Cost	Operational Cost and Deadline

2.5 Conclusion

This section discussed about various challenges faced by scientific applications in cloud computing. Most of these challenges are directly linked to resource allocation prediction and resource scheduling. The existing solutions such as existing prediction models and scheduling techniques were also discussed.

In the next chapter, research gap between the existing techniques and the proposed methodology has been discussed. The problem statement and its scope have also been defined.

Chapter 3

Research Gap and Problem Statement

The above literature survey consists of various challenges faced by scientific applications as well as solution to tackle these challenges. Various prediction and scheduling techniques have been discussed. The research that has been done in the field of resource prediction and resource scheduling has been included. This chapter contains the research gap between the above techniques and the proposed technique and the proposed methodology has been explained.

3.1 Research Gap

In the last chapter, various challenges, resource allocation prediction models and scheduling techniques have been discussed. In all the prediction models that have been discussed, it is observed that prediction models have a traditional approach and are not suitable for a dynamic environment. These prediction models predict resources according to the previous records. These are not suitable for workloads which have varying resource requirements over a short span of time. The prediction technique proposed in this thesis is suitable for changing resource requirements. It monitors the tasks closely and according to the execution time of all the tasks, allocates the resources.

Moreover, the scheduling techniques focus mainly on reducing the cost or reducing the execution time. The techniques like particle swarm optimization and cat swarm optimization have tried to reduce the cost of execution. Models like PQR2, heuristic techniques, parallel genetic algorithm etc. have considered tasks in general. There are no specifications involved for the kind of tasks. Tasks can vary from application to application. Scheduling algorithm should be based on the types of task and the type of efficiency the application requires. For example in a meteorological application like montage, the size of the jobs is quiet large and varies from each other. So it is very important to not be biased towards a small sized task or a large sized task. The scheduling mechanism should be specific to various characteristics of the tasks. Therefore, the task scheduling algorithm proposed in this report is characteristic specific. Improved MinMin

algorithm considers the size of the tasks as well as the arrival time. This way it helps to remove any bias based on the size of the tasks. This helps to avoid starvation and hence avoids under utilization of resources. Proper utilization of resources leads to increased productivity of the algorithm. Thus, this algorithm helps to achieve higher productivity and reduced resource wastage.

3.2 Problem Formulation

An efficient prediction and scheduling algorithm should be able to improve the overall efficiency of the cloud system. Scheduling is the core of cloud computing. By improving the scheduling algorithm, many cost overheads can be reduced. The main job is to develop a scheduling algorithm which not only considers the size of the tasks but also considers their arrival time in order to avoid starvation. The algorithm should also be able to perform resource allocation prediction.

3.3 Objectives

Following are the objectives of thesis:

i. To study and analyze various challenges of cloud computing for scientific applications.
ii. To study various prediction and scheduling models used for executing scientific applications in cloud computing.
iii. To develop an improved prediction and scheduling algorithm to resolve the problems of starvation and under utilization of resources faced by scientific applications in cloud computing.
iv. To implement the proposed algorithm in the WorkflowSim environment on a scientific application.

3.4 Conclusion

The analysis of various existing models helped in finding the flaws occurring in these models and the cause of these faults was identified. This identification led to the formulation of problem statement. Research gap analysis helped to formulate the objectives of the proposed model.

In this chapter, the major objectives were discussed. In the next chapter, the proposed model is discussed in detail.

Chapter 4

Proposed Methodology

It was discussed in the previous chapter that the research gap led to the formulation of objectives. This chapter includes all the details of the proposed model.

4.1 Existing Model

The original MinMin scheduling algorithm allocates resources on the basis of size of the job. The job with small size is allocated to the resource of minimum execution time. The main concept if MinMin is to complete the shortest task as soon as possible. MinMin algorithm is fast and performs well if the priorities are set according to the size of the tasks. But it has some issues. If small sized tasks keep on entering the queue, relatively large tasks will suffer from starvation. This will most probably lead to congestion on the network. In such case, execution time and operational cost increases.

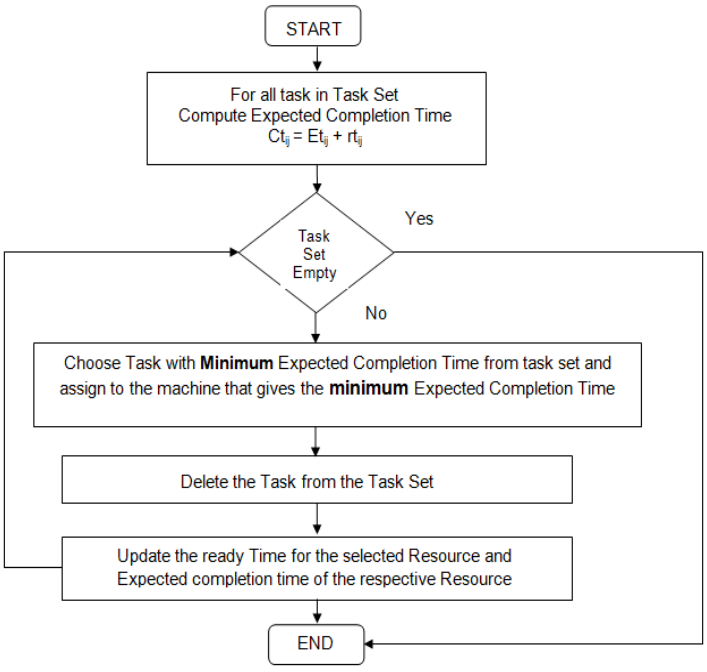

Figure 4.1: Flowchart of MinMin Algorithm [40]

4.2 Design of Proposed System

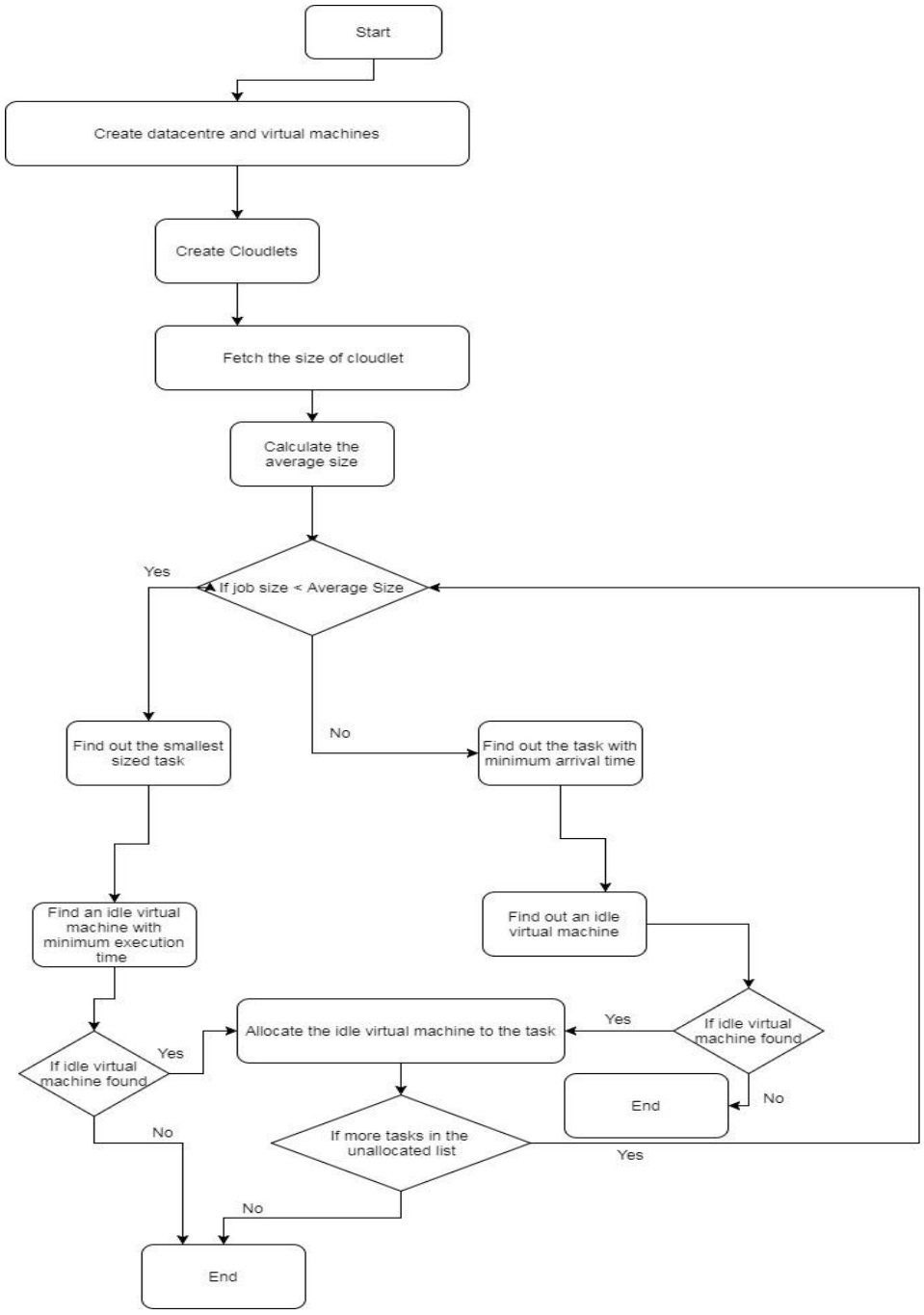

Figure 4.2: Flowchart of Enhanced MinMin Algorithm

4.3 Improved MinMin Algorithm for Scientific Applications

The proposed model tries to resolve the issues of the MinMin algorithm. In the original MinMin scheduling technique, the resources are allocated on the basis of the size of the job. The job with relatively minimum Million Instructions Per Second (MIPS) will be allocated the resource having minimum execution time. The job with relatively maximum Number of MIPS will be allocated to the resources with highest execution time. This way the small sized jobs are completed early and the large sized jobs are completed late. This technique causes starvation of large sized processes. If a set of jobs are dependent on the completion of a large sized job, then it will lead to congestion on the network. This will cause other resources to be idle and thus lead to under utilization of resources. This scenario will also cause low throughput and decrease the productivity. It will lead to an increase in the execution time. Hence the cost of operation will increase. To avoid this scenario, optimized MinMin algorithm has been proposed. In this algorithm, the problem of congestion and starvation has been removed. This algorithm considers arrival time in addition to the size of the job.

4.4 Concept of Improved MinMin Algorithm

In this algorithm, the major objective is to remove the problem of starvation and congestion. To achieve this objective, the arrival time of the jobs has been considered. This algorithm also predicts resource allocation on the basis of the execution time of the resources. It calculates the execution time of the task according to the size i.e. MIPS of the tasks. When the jobs enter the ready queue, they are sorted on the basis of their size taken as million instructions per second. Then average size of the jobs is calculated. Any job which is of size less than average size is scheduled according to their size. Any job which has size greater than the average size is scheduled according to its arrival time. For example, suppose there are 10 jobs J_i (where $0< i<=10$) and 3 virtual machines VM_k where $1<=k<=3$. Now all the jobs have different sizes and have to be scheduled. If these jobs are to be scheduled by using original MinMin algorithm:

VM1	VM2	VM3

J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8	J_9	J_{10}

If J_5 has minimum size and VM_1 has maximum workload processing capacity, then j_5 will be assigned to VM_1. Now we are left with two virtual machines where if workload processing capacity of $VM_3 > VM_2$. Now suppose J_3 has maximum size. Now all the other tasks will be scheduled one by one according to their size. If any other task J_{11} comes in which is smaller in size than J_3 then J_3 will again wait. This way J_3 will go under starvation. All the tasks which are dependent on J_3 will also be in waiting state. This will create congestion on the network which will eventually lead to a significant decrease in throughput and increase in execution time.

However if improved MinMin is used in this situation, J_3 will not have to wait for so long. According to arrival time of J_3 and a new task $J_{11,}$ J_3 will be allocated resources first. Therefore there will be no congestion. Following is the algorithm proposed:

1. Declare int minindex, earlyindex, size;
2. Declare an empty cloudlet named cloudlet;
3. Declare an empty cloudlet named earlycloudlet;
4. Declare an empty cloudlet called mincloudlet;
5. Fetch cloudletlist size into size;
6. Initialize minCloudlet with a cloudlet;
7. Initialize earlyCloudlet with a cloudlet;
8. For i less than size {
9. If minCloudelt size > cloudlet size {
10. minCloudlet=cloudlet;
11. minIndex=i; }}

12. For j is less than size {

13. If earlyCloudlet arrival time < cloudlet arrival time{

14. earlyCloudlet=cloudlet;

15. earlyIndex=j; }}

16. Calculate average size of the cloudlet list;

17. If size of cloudlet is less than average size{

18. Find a VM which has maximum workload processing capacity and is idle;

19. If such VM exists{

20. Allocate that cloudlet to the VM}

21. If such VM does not exist{

22. Break; }}

23. If size of cloudlet is greater than average size{

24. Find the task with minimum arrival time;

25. Find an idle VM ;

26. If such VM exists{

27. Allocate the cloudlet to the VM ;}

28. Else if such VM does not exist{

29. Break;}}

27. End;

4.5 Conclusion

In this chapter, the original MinMin algorithm and enhanced MinMin algorithm have been discussed in detail. The designs of the two approaches have been explained through flowcharts. Algorithm for the proposed model has also been discussed.

In the next chapter, the implementation of the two approaches and their results have been discussed.

Chapter 5

Implementation and Results

5.1 Tools Used

5.1.1 WorkflowSim

WorkflowSim has been used to simulate the application for implementing the optimized MinMin approach. WorkflowSim is a workflow simulator. It extends the features of cloudsim as it provides workflow support for simulation. It is developed by WeiWei Chen, a University of Southern California student pursuing Phd. Workflows are modeled here in the form of DAG (Directed Acyclic Graph). It models the failure node, WMS stack delay in various levels and implements various dynamic and static schedulers of workflow and algorithms of task clustering. Real executions form the source of parameters used in the workflows.

Workflow is formed of various components:

i. Workflow Mapper: It is responsible for creating a list of user activities called as tasks and assigning this list to the execution site. It coordinates with the workflow generator and imports xml format DAG files and metadata from there.

ii. Workflow Engine: It releases free tasks to the clustering engine. It also makes sure that dependencies are kept intact while releasing them. This means that only that task is released whose parent task has been completed.

iii. Clustering Engine: Its main task is to get various tasks together to form a job. While merging the tasks, it also makes sure that the scheduling overhead is reduced. It also merges failed tasks which are sent back by workflow scheduler.

iv. Workflow Scheduler: It considers the user criteria and matches jobs with the worker node. Simulation accuracy is greatly improved by workflow scheduler.

v. Failure Generator: A user can specify the average failure rate and the failure generator will insert failures according to the failure rate specified and distribution. It introduces job/task failures during the simulation at the execution site.

vi. Failure Monitor*:* It monitors the failures that have occurred during the simulation and maintains a record for these failures. These records are then returned to the clustering engine which will help adapt the scheduling strategy accordingly.

5.1.2 NetBeans IDE

NetBeans is a platform for software development written in Java. The NetBeans Platform enables the development of applications from a set of modular software components called modules. The NetBeans IDE is designed primarily for Java development, but it also supports other languages, especially HTML5, C / C ++ and PHP. It has various tools like converters, editors and code analyzers. These tools help in using various java constructs like functional operations, lambda and method references easily. Following are some of the features of Netbeans IDE:

i. Fast and Smart Code Editing: The Netbeans editor provides for matching brackets and words. It helps to draw indented lines. The code can be highlighted both semantically and semantically. The source code can be easily refracted. It also provides code tips, code generators and coding tips.

ii. Efficient Project Management: As there are millions and billions of lines of code, folders and files, it is very difficult to manage them. To help manage and understand those, data can be viewed in multiple ways and can be opened in multiple windows.

iii. Easy-to-Use Graphical User Interface: The NetBeans GUI constructor automatically supports spacing and alignment, while also supporting assembly. The Constructor GUI is easy to use and intuitive to use for prototypes for live presentations of client GUIs.

iv. Writing Bug-Free Code: The cost of corrupt code increases if it remains unchanged for a longer duration. NetBeans offers static analysis tools, mainly integrating with FindBugs tool to identify and solve common problems used in Java code. In

addition, NetBeans Debugger allows you to set breakpoints in your source code, add field watch, take snapshots, run code methods, and monitor the execution while it occurs. NetBeans Profiler provides specific support for speed and memory usage to optimize the application and allows the creation of Java SE applications, JavaFX and Java EE reliable and scalable. NetBeans IDE includes a visual debugger for Java SE applications, allowing debugging user interfaces without looking for the source code. Snapshots of a GUI application can be taken in order to get its source code by clicking on a specific element.

5.2 Scientific Application Used

Montage is the scientific application on which the optimized MinMin approach has been implemented. Montage (Montage Assembly Astronomical Image Mosaic Engine) is a collection of astrophotography software tools for composing FITS astronomical images in composite images, called mosaics, which retain the calibration and fidelity of the original input image [22]. Montage was designed to support scientific research. It allows astronomers to create images of the sky too big to be generated by astronomical cameras. It also creates a composite image of the sky with different wavelengths measured and using different instruments; Composite image seems as if the area measured with the same instruments on the telescope. There is also a link for software user community to bash shell scripts and C to create the mosaic and application programming interface Python wrapper to create a part of the Astropy project.

Table 5.1: General Information Regarding Montage

Application Name	Developers	Operating System	License	Price	Website	Comments
Montage Image Mosaic	Caltech/IPAC, JPL, CACR, ISI	Unix	3-Clause BSD	Free	official Web site	Produces science-grade mosaics for astronomical research

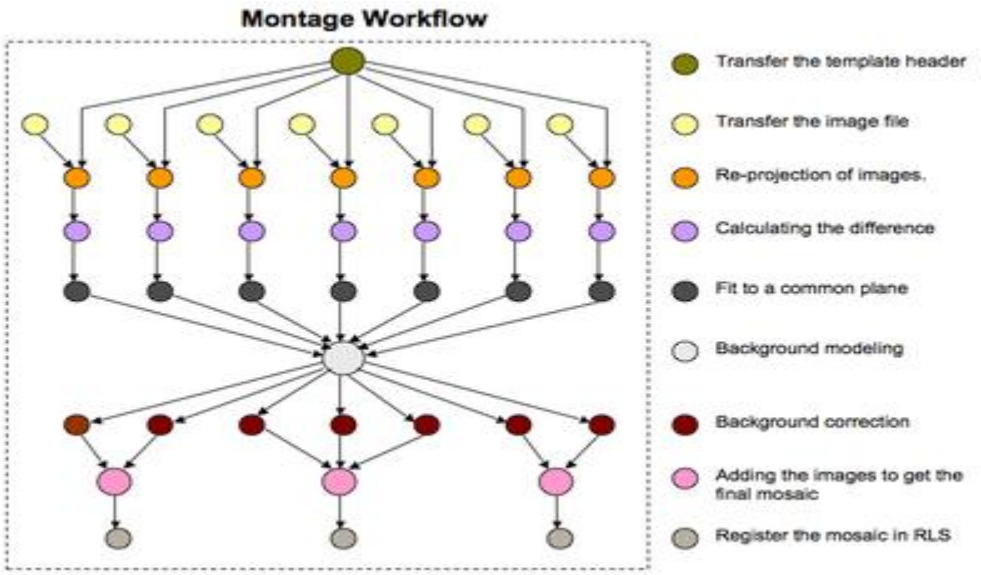

Figure 5.1: Montage Workflow [42]

5.3 Experimental Setup

In order to implement the optimized MinMin algorithm, 5, 10 and 15 virtual machines have been used to create different scenarios. Montage workflow consisting of 100 nodes is used. Three datacenters are created. The results of original MinMin and optimized MinMin are compared for the three scenarios.

5.4 Results

In order to test and compare the performance of enhanced MinMin algorithm with the original MinMin algorithm, both the algorithms have been implemented on WorkflowSim. These algorithms have been applied on Montage scientific workflow and the results have been analyzed. According to the results, the enhanced MinMin algorithm performs better than the original MinMin algorithm. The proposed algorithm takes into account the problem of starvation faced by large sized jobs. The results show that average finish time of the jobs is significantly reduced. This indicates that the proposed algorithm can help in reducing the execution time as well as operational cost.

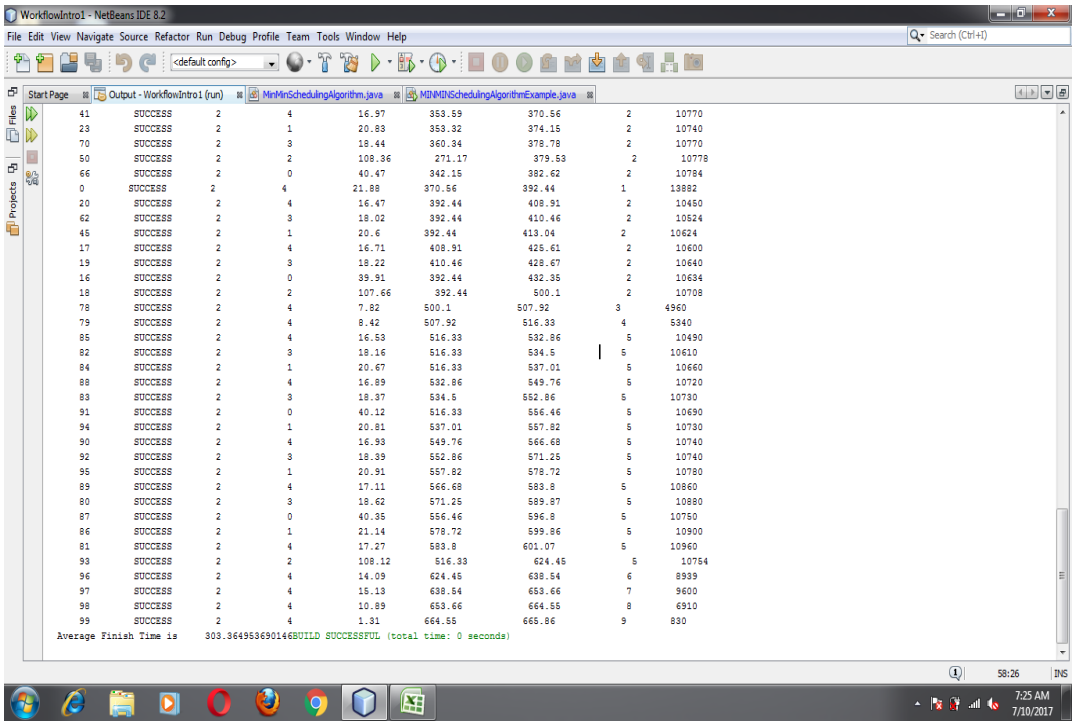

Figure 5.2: Output of Original Minmin Scheduling For 5 Virtual Machines

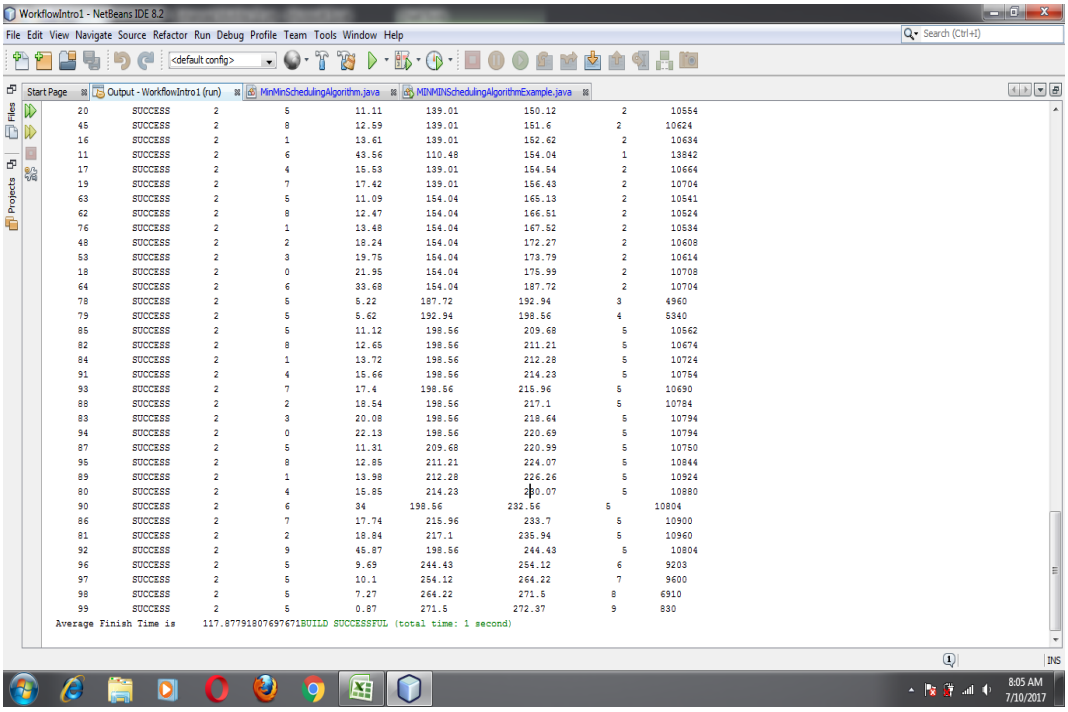

Figure 5.3: Output of Original Minmin Scheduling For 10 Virtual Machines

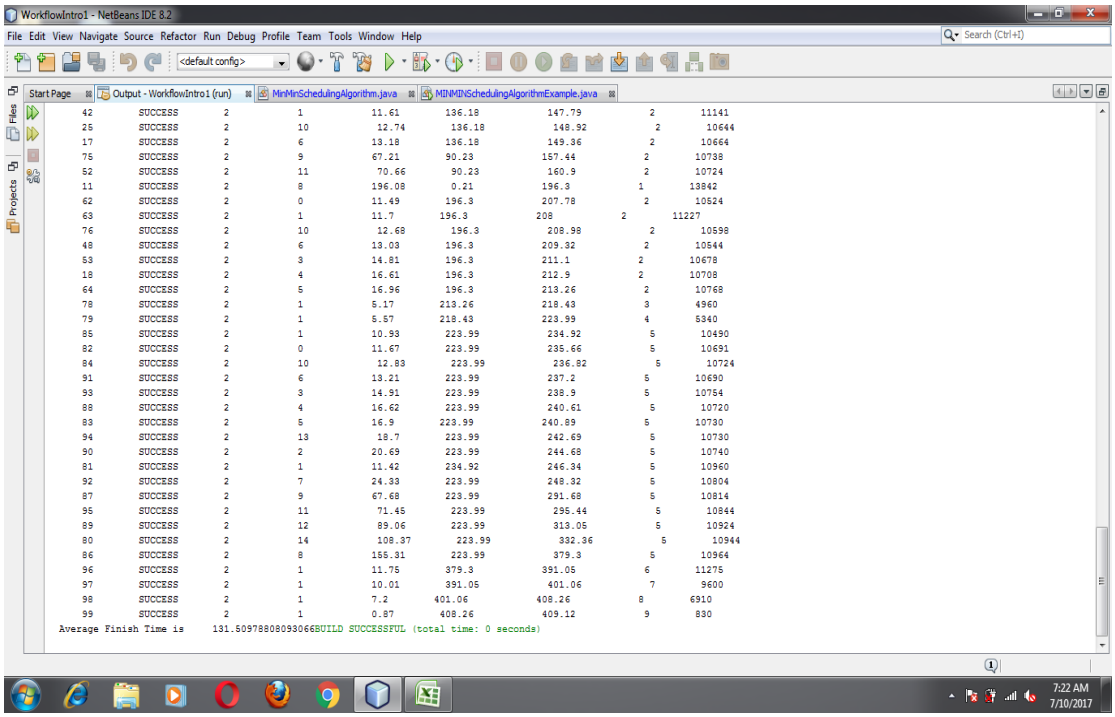

Figure 5.4: Output of Original Minmin Scheduling For 15 Virtual Machines

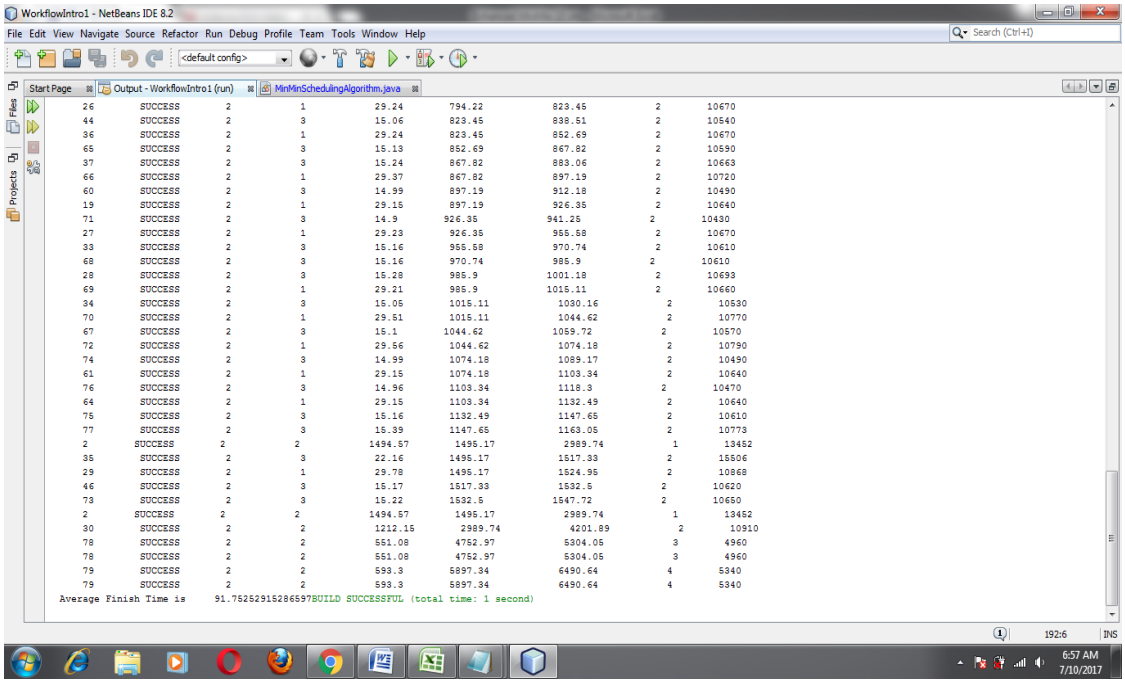

Figure 5.5: Output of Enhanced Minmin Scheduling For 5 Virtual Machine

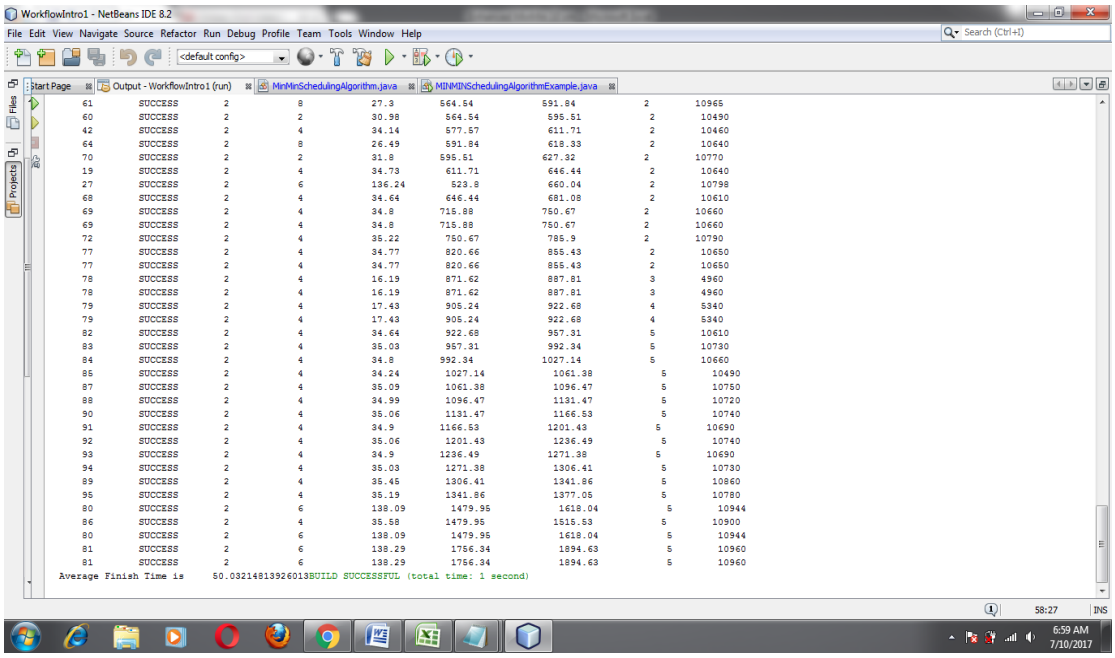

Figure 5.6: Output of Enhanced Minmin Scheduling For 10 Virtual Machines

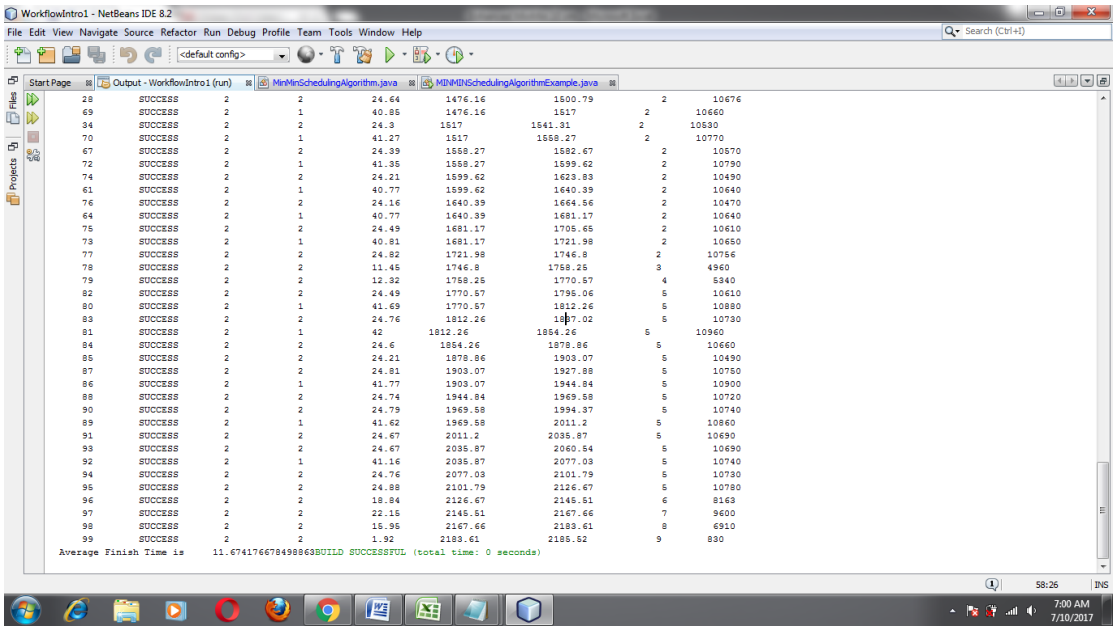

Figure 5.7: Output of Enhanced Minmin Scheduling For 15 Virtual Machines

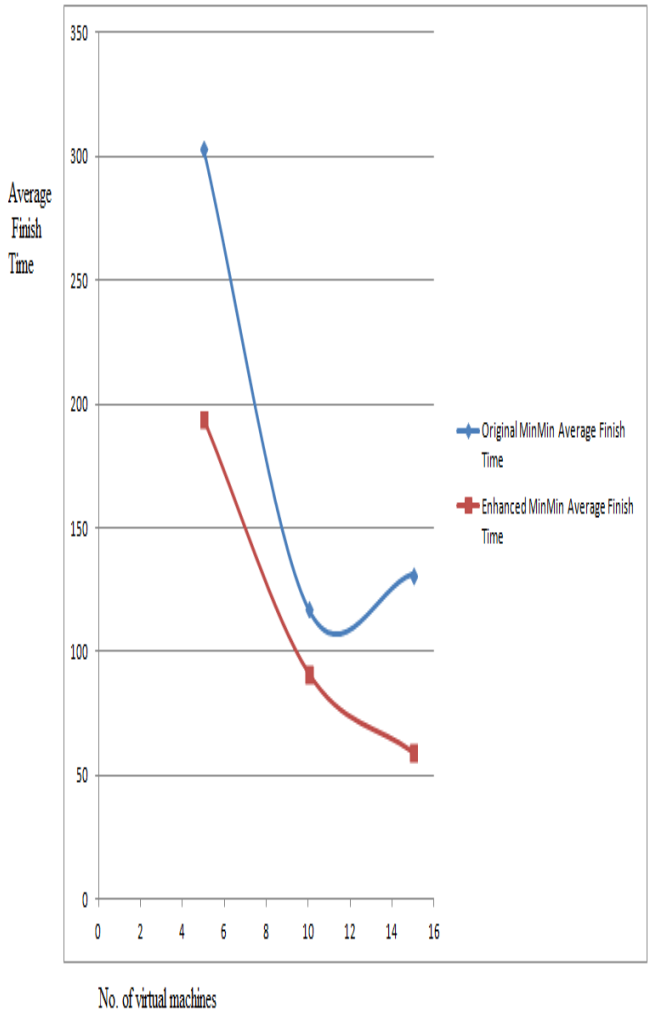

Figure 5.8: Comparison of Average Finish Time of Original And Enhanced Minmin Scheduling Algorithm

The above graph depicts that for 5 virtual machines, the average finish time in the original MinMin is approximately 300 seconds. Whereas, the average finish time for enhanced MinMin is less than 200 seconds. If 10 virtual machines are used, the average finish time for the original MinMin is approximately 120 seconds and for enhanced algorithm, it is less than 100 seconds. When 15 virtual machines are used, the average finish time for the original MinMin approach is approximately 130 seconds and for enhanced MinMin approach, the average finish time is approximately 60 seconds. Therefore, the enhanced MinMin algorithm performs better than the original MinMin algorithm.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

Cloud computing has various challenges to face. Some challenges are specific to scientific applications. In this thesis work, various challenges faced by scientific applications have been studied and discussed. Major challenges discovered include automation of service provisioning, energy management and server consolidation, resource allocation prediction, computation time, pricing, workflow scheduling and management, scheduling, computation, cost of execution, response time, storage, response time, throughput, reliability, computation cost, execution time, scheduling overhead, scaling of resources and intensive computation. But majority of the problems indicated towards resource allocation prediction and resource scheduling. Various currently used resource allocation prediction models have been discussed in this research work. These prediction models helped to improve various parameters like execution time, cost, throughput and scheduling overhead. An efficient resource allocation prediction model needs to be implemented with a suitable scheduling model. An optimized resource scheduling model can bring about a lot of reduction in scheduling overhead, execution time and cost. Various existing scheduling have been discussed. Major existing scheduling models include Round Robin, MinMin, MaxMin, Cat Swarm Optimization (CSO) and Particle Swarm Optimization (PSO). The focus of this thesis is MinMin scheduling algorithm. It had various drawbacks like starvation, network congestion and under utilization of resources. These drawbacks have been removed to a significant extent by the proposed enhanced MinMin scheduling algorithm. The average finish time is significantly improved. The proposed algorithm avoids starvation of large sized tasks.

6.2 Thesis Contributions

i. A new algorithm called enhanced MinMin scheduling algorithm is proposed which aims at removing the drawbacks of the original MinMin algorithm.
ii. The proposed algorithm considers the size of the tasks as well as their arrival time while scheduling the resources in order to remove starvation of large sized tasks.
iii. The proposed algorithm also performs resource prediction on the basis of execution time of the tasks to schedule Montage scientific workflow in WorkflowSim.
iv. The results of original algorithm were compared to the results of the proposed algorithm in three different scenarios.
v. The proposed algorithm reduces the average finish time significantly as well as helps to avoid starvation of large sized tasks.

6.3 Future Work

i. Other heuristics based algorithms like Particle Swarm Optimization (PSO), Cat Swarm Optimization (CSO) and Ant Colony Optimization (ACO) can be explored and merged with MinMin algorithm.
ii. Resource scheduling algorithms for handling dynamic workloads can be developed.
iii. Memory storage can be taken into consideration and scheduling algorithm can be developed to reduce memory wastage.

REFERENCES

[1]*Cloud computing: state-of-the-art and research challenges.* Qi Zhang, Lu Cheng, Raouf Boutaba. 2010. s.l. : The Brazilian Computer Society 2010, 20 April 2010, Springer, pp. 7-18.

[2]*High-Performance Cloud Computing: A View of Scientific Applications.* Christian Vecchiola, Suraj Pandey, and Rajkumar Buyya. Melbourne : Manjrasoft Pty Ltd.

[3]*Opportunities and Challenges in Running Scientific Workflows on the Cloud.* Yong Zhao, Xubo Fei, I. Raicu, Shiyong Lu. 2011. Beijing, China : IEEE, 18 November 2011, IEEE. 978-1-4577-1827-4.

[4]*Experiences using cloud computing for a scientific workflow application.* Jens-Sönke Vöckler, Gideon Juve, Ewa Deelman, Mats Rynge, Bruce Berriman. 2011. California, USA : s.n., 08 June 2011, ScienceCloud '11 Proceedings of the 2nd international workshop on Scientific cloud computing, pp. 15-24. 978-1-4503-0699-7.

[5] *A Particle Swarm Optimization-Based Heuristic for Scheduling Workflow Applications in Cloud Computing Environments* . Suraj Pandey, Linlin Wu, Siddeswara Mayura Guru, Rajkumar Buyya. 2010. Perth, Australia : Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference, 2010. 978-1-4244-6695-5.

[6]*Performance Analysis of Cloud Computing Services for Many-Tasks Scientific Computing.* Alexandru Iosup, Simon Ostermann, M. Nezih Yigitbasi, Radu Prodan, Thomas Fahringer, Dick Epema. 2011. 6, s.l. : IEEE, june 2011, IEEE Transactions on Parallel and Distributed Systems, Vol. 22, pp. 931 - 945. 1045-9219.

[7] *A Performance Analysis of EC2 Cloud Computing Services for Scientific Computing.* Simon Ostermann, Alexandria Iosup, Nezih Yigitbasi, Radu Prodan, Thomas Fahringer, Dick Epema. 2009. s.l. : Springer, 2009. International Conference on Cloud Computing. pp. 115-131.

[8]*The cost of doing science on the cloud: the Montage example.* Ewa Deelman, Gurmeet Singh, Miron Livny, Bruce Berriman. John Good. 2008. s.l. : IEEE, 2008. Proceeding

SC'08 Proceedings of the 2008 ACM/IEEE conference on Supercomputing. 978-1-4244-2835-9.

[9]*On the Use of Cloud Computing for Scientific Workflows.* Christina Hoffa, Gaurang Mehta, Timothy Freeman, Ewa Deelman, Kate Keahey, Bruce Berriman, John Good. 2008. s.l. : IEEE, Dec 2008, eScience, 2008. eScience '08. IEEE Fourth International Conference. 978-1-4244-3380-3.

[10]*Scaling up workflow-based applications.* Scott Callaghan, Ewa Deelman, Dan Gunter , Gideon Juve , Philip Maechling ,Christopher Brooks,Karan Vahi, Kevin Milne,, Robert Graves, Edward Field,David Okaya, Thomas Jordan. 2009. s.l. : ELSEVIER, 22 August 2009, Journal of Computer and System Sciences, pp. 428-446.

[11]*Managing Large-Scale Workflow Execution from Resource Provisioning to Provenance Tracking: The CyberShake Example.* Ewa Deelman, Scott Callaghan, Edward Field, Hunter Francoeur, Robert Graves, Nitin Gupta, Vipin Gupta, Thomas H. Jordan, Carl Kesselman, John Mehringer, Philip Maechling, Gaurang Mehta, Karan Vahi, David Okaya, Li Zhao. 2006. s.l. : IEEE, December 2006, e-Science and Grid Computing, 2006. e-Science '06. Second IEEE International Conference. 0-7695-2734-5.

[12]*CyberShake: A Physics-Based Seismic Hazard Model for Southern California.* Robert Graves, Thomas H. Jordan,Scott Callaghan, Ewa Deelman,Edward Field,Gideon Juve, Carl Kesselman,Philip Maechling,Gaurang Mehta,Kevin Milner, David Okaya,Patrick Small,Karan Vahi. 3, s.l. : Springer, Pure and Applied Geophysics, Vol. 168, pp. 367–381.

[13]*All-Pairs:An Abstraction for Data Intensive Cloud Computing* Christopher Moretti, Jared Bulosan, DouglasThain, Patrick J. Flynn. IEEE.

[14]*VGrADS: Enabling e-Science Workflows on Grids and Clouds with Fault Tolerance.* Lavanya Ramakrishnan, Charles Koelbel,Yang-Suk Kee,Rich Wolski,Danioel Nurmi, Dannis Gannon, Ghaziano, Orbetelli, AsimYarKhan, Anirban Mandal, T.Mark Huang, Kiran Thyagaraja, Dmitrii Zagorodnov. 2009. 2009, SC'09.

[15]*Optimizing Utility in Cloud Computing through Autonomic Workload Executio.* Norman W. Paton, Marcelo A. T. de Aragaˆo, Kevin Lee, Alvaro A. A. Fernandes, Rizos Sakellariou. 2009. 2009, IEEE Computer Society Technical Committee on Data Engineering.

[16]*PRESS: PRedictive Elastic ReSource Scaling for cloud systems.* Zhenhuan Gong, Xiaohui Gu, John Wilkes. 2010. s.l. : IEEE, 2010, 2010 International Conference on Network and Service Management – CNSM 2010, pp. 9-16. 978-1-4244-8909-1.

[17]*On the use of machine learning to predict the time and resources consumed by applications* . Fortes, Andréa Matsunaga and José. 2010. 2010, 2010 10th IEEE/ACM International Conference on Cluster, Cloud, Grid Computing, pp. 495-504.

[18]*Auto-Scaling to Minimize Cost and Meet Application Deadlines in Cloud Workflows.* Ming Mao, Marty Humphrey. 2011. 2011, SC11.

[19]*CloudScale: Elastic Resource Scaling for Multi-Tenant Cloud Systems.* Zhiming Shen, Sethuraman Subbiah, Xiaohui Gu,John Wilkes. 2011. 2011, SOCC'11.

[20]*Dynamic Energy-Aware Capacity Provisioning for Cloud ComputingEnvironments.* Qi Zhang, Shuo Zhang, Quanyan Zhu,Mohamed Faten Zhani ,Raouf Boutaba,Joseph L. Hellerstein. 2012. 2012, ICAC'12.

[21] *AGILE: elastic distributed resource scaling for Infrastructure-as-a-Service.* Hiep Nguyen, Zhiming Shen, Xiaohui Gu, Sethuraman Subbiah, John Wilkes. 2013. 10th International Conference on Autonomic Computing (ICAC).

[22] AV.KARTHICK, D. R. (2014). AN EFFICIENT MULTI QUEUE JOB SCHEDULING FOR CLOUD COMPUTING. *COMPUTING AND COMMUNICATION TECHNOLOGIES (WCCCT), 2014 WORLD CONGRESS ON* (PP. 164-166). TRICHIRAPPALLI, INDIA: IEEE.

[23] Chenhong Zhao, S. Z. (2009). Independent Tasks Scheduling Based on Genetic. *IEEE* (pp. 1-4). Beijing, China: IEEE.

[24] Dzmitry Kliazovich, J. E. (2016). CA-DAG: Modeling Communication-Aware Applications for Scheduling in Cloud Computing. *Journal of Grid Computing* , 1-17.

[25] Dzmitry Kliazovich, S. T. (2013). e-STAB: Energy-Efficient Scheduling for Cloud Computing Applications with Traffic Load Balancing. *IEEE International Conference on and IEEE Cyber, Physical and Social Computing* (pp. 1-7). Beijing, China: IEEE.

[26] Hai Zhong, K. T. (2010). An Approach to Optimized Resource Scheduling Algorithm for Open-source. (pp. 125-129). Guangzhou, TBD, China, China: IEEE.

[27] Hai-Hao Li, Y.-W. F.-H.-J. (2015). Renumber Strategy Enhanced Particle Swarm Optimization for Cloud Computing Resource Scheduling. *Evolutionary Computation (CEC), 2015 IEEE Congress on* (pp. 870-876). Sendai, Japan: IEEE.

[28] Jinhua Hu, J. G. (2010). A Scheduling Strategy on Load Balancing of Virtual Machine Resources in Cloud. (pp. 1-8). Dalian, China: IEEE.

[29] KAMALAKAR. M, M. (2015). A Priority Based Job Scheduling Algorithm in Cloud Computing. *International Journal of Innovative Technologies* , 0019-0021.

[30] Liang Luo, W. W. (2012). A Resource Scheduling Algorithm of Cloud Computing based on Energy Efficient Optimization Methods. *Green Computing Conference (IGCC), 2012 International* (pp. 1-6). San Jose, CA, USA: IEEE.

[31] Mohamed Abu Sharkh, A. O. (2013). A Resource Scheduling Model for Cloud Computing Data centers. *Wireless Communications and Mobile Computing Conference (IWCMC), 2013 9th International* (pp. 213-218). Sardinia, Italy: IEEE.

[32] Mohammed Abdullahi, M. A. (2016). Symbiotic Organism Search optimization based task scheduling in cloud computing environment. *Elsevier* , 640-650.

[33] Resource Provisioning for Cloud Computing. (2009). York: Conference of the Center for Advanced Studies on Collaborative Research - CASCON '09.

[34] Rui Zhang, K. W. (1131 - 1145). Online Resource Scheduling under Concave Pricing for Cloud Computing. *IEEE Transactions on Parallel and Distributed Systems (Volume: 27, Issue: 4, April 1 2016)* , 2015.

[35] Saurabh Bilgaiyan, S. S. (2014). Workflow Scheduling in Cloud Computing Environment Using Cat Swarm Optimization. *2014 IEEE International Advance Computing Conference (IACC)* (pp. 680-685). Odisha, India: IEEE.

[36] Wei Wang, G. Z. (2012). Cloud-DLS: Dynamic trusted scheduling for Cloud computing. *Elsevier* , 2321–2329.

[37] Xin Lu, Z. G. (2011). A Load-Adaptive Cloud Resource Scheduling Model Based on Ant Colony Algorithm. *Cloud Computing and Intelligence Systems (CCIS), 2011 IEEE International Conference on* (pp. 296-300). Beijing, China: IEEE.

[38] Zhaobin Liu, W. Q. (2014). Resource preprocessing and optimal task scheduling in cloud computing environments. John Wiley and Sons.

[39] Zhongni Zheng, R. W. (2011). An Approach for Cloud Resource Scheduling Based on Parallel Genetic Algorithm. (pp. 444-447). Shanghai, China: IEEE.

[40] "Montage Image Mosaic Software." Wikipedia [Online]. Available: https://en.wikipedia.org/wiki/Montage_Image_Mosaic_Software. [Accessed July 10, 2017].

[41] "VGrADS at Rice University." Montage Workflow figure — VGrADS at Rice University[Online].http://vgrads.rice.edu/research/applications/images/montage-workflow/view [AccessedJuly10,2017].

List of Publications

[1] Resham Kaur, Dr. Inderveer Chana. "A Survey on Different Pricing Schemes in Cloud Computing." *ISRD Recent Advances in Engineering and Technology Summit (RAETS).* 2017. [Published].