SODA-M: SCHEDULE OPTIMIZATION WITH DUPLICATION-BASED SCHEDULING ALGORITHM FOR MULTIPLE WORKFLOWS IN CLOUD COMPUTING

A

Dissertation

submitted in partial fulfillment of the

requirements for the award of the degree of

MASTER OF TECHNOLOGY

In

ARTIFICIAL INTELLIGENCE AND ARTIFICIAL NEURAL NETWORK

By

Kuber Sangal

Roll No.: R102213004

Under the guidance of

Dr. Amit Agarwal

Associate Professor, CIT, UPES, Dehradun

UPES
THE NATION BUILDERS UNIVERSITY

Department of Computer Science & Engineering

Centre for Information Technology

University of Petroleum & Energy Studies

Bidholi, Via Prem Nagar, Dehradun, UK

May – 2015

UPES COLLEGE OF ENGINEERING STUDIES

The innovation driven
E-School

CANDIDATE'S DECLARATION

I hereby certify that the dissertation entitled **"SODA-M: SCHEDULE OPTIMIZATION WITH DUPLICATION-BASED SCHEDULING ALGORITHM FOR MULTIPLE WORKFLOWS IN CLOUD COMPUTING"** in partial fulfilment of the requirements for the award of the Degree of MASTER OF TECHNOLOGY in ARTIFICIAL INTELLIGENCE AND ARTIFICIAL NEURAL NETWORK and submitted to the Department of Computer Science & Engineering at Center for Information Technology, University of Petroleum & Energy Studies, Dehradun, is an authentic record of my work carried out during a period from **January, 2015** to **April, 2015** under the supervision of **Dr. Amit Agarwal, Associate Professor, CIT, UPES Dehradun**.

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

[illegible]

(Kuber Sangal)
Roll No:R102213004

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

[illegible]

(Dr. Amit Agarwal)
Project Guide

Date: 04/05/2015

[illegible]

Dr. Amit Agarwal
Program Head – M.Tech (AI & ANN)
Center for Information Technology
University of Petroleum & Energy Studies
Dehradun – 248 001 (Uttarakhand)

ACKNOWLEDGEMENT

I wish to express my deep gratitude to my guide **Dr. Amit Agarwal**, for all advice, encouragement and constant support he has given us throughout our project work. This work would not have been possible without his support and valuable suggestions. I sincerely thanks to him, for his great support in doing my project in **Cloud Computing** at **CIT**.

I am also grateful to **Dr. Manish Prateek, Associate Dean** and **Dr. Kamal Bansal, Dean CoES, UPES** for giving me the necessary facilities to carry out my project work successfully.

I would like to thank all my **friends** for their help and constructive criticism during my project work. Finally I have no words to express my sincere gratitude to my **parents** who have shown me this world and for every support they have given me.

[illegible]

Name **Kuber Sangal**

Roll No. **R102213004**

ABSTRACT

Optimal scheduling of Workflows in Cloud Computing (Distributed System) can be done by consuming computing facilities effectively. In Distributed Memory Systems (DMS) like cloud computing, it is very important to schedule the precedence constrained task graphs for the sake of effective use of computing resources. To obtain an optimal schedule of tasks of a Directed Acyclic Graph (DAG) with duplication or without duplication is one of the most challenging; and a NP-Complete problem. The proposed work presents an improved task duplication approach for multiple workflows (dependent and independent) that provides a cost effective inter processor communication using limited no. of processors. In this work, Schedule Optimization With Duplication-based Scheduling Algorithm of Multiple Workflows (SODA-M) for homogeneous systems are suggested for minimizing the total execution time of job, and maximizing the machine utilization by assigning tasks of independent workflow. In these algorithms insertion-based task-duplication scheduling strategy is used for generating static task schedules. Experimental result shows that SODA-M algorithms can achieve less computation time with lesser no. of duplications and remarkable less processor consumption as compared with SD (Selective Duplication) for homogeneous system, and more machine consumption as compared to the Reduced Duplication.

Table of Contents

List of Figures

List of Table

CHAPTER 1

INTRODUCTION

This chapter starts off by elaborating the basis of Cloud computing, its evolution and background with related technologies such as grid computing. Types and characteristics are also described in this chapter. The chapter also describes various research issues related to Cloud computing and at the end of this chapter structure of the thesis is given.

1.1. Background

With the advancement in processing, storage technologies and the success of the Internet, new challenges are arising. One of those challenges is availability of resources and demand of connectivity is increasing day by day. Now due to globalization the need of dynamic resource and access is become mandatory for the IT infrastructure. In result of these demands there is a need of elastic infrastructure that can manage the demand accordingly. These factors are responsible for the realization of a new computing model known as Cloud Computing. The concept of cloud computing is not fully new. The concept of cloud computing came into existence in early 1960 by John McCarthy portrayed that "computation may someday be organized as a public utility" in this way utility computing came into existence.

Technologies like Utility computing, Grid Computing and Cloud Computing focus on delivering computing power in a reliable, efficient and scalable manner to a large number of end-users. More or less, cloud computing makes the delivery of these computing power/resources as a utility, much like how electricity and water are deliver to the houses. The concept of utility computing can be tracked in 1969 by Leonard Kleinrock, one of the former scientists of original Advanced Research Projects Agency Network (ARPANET) project, which sown the Internet, said [1]: "As of now, computer networks are still in their infancy, but as they grow up and become sophisticated, we will probably see the spread of, 'computer utilities', which, like present electric and telephone utilities, will service individual homes and offices across the country". The name 'cloud' comes from the metaphor for the internet. Generally cloud icon is used to represent internet in network diagrams as shown in figure 1.1. According to these vision for the computing utilities, there would be massive transformation in entire computing industry can be seen, where on demand computing services are available.

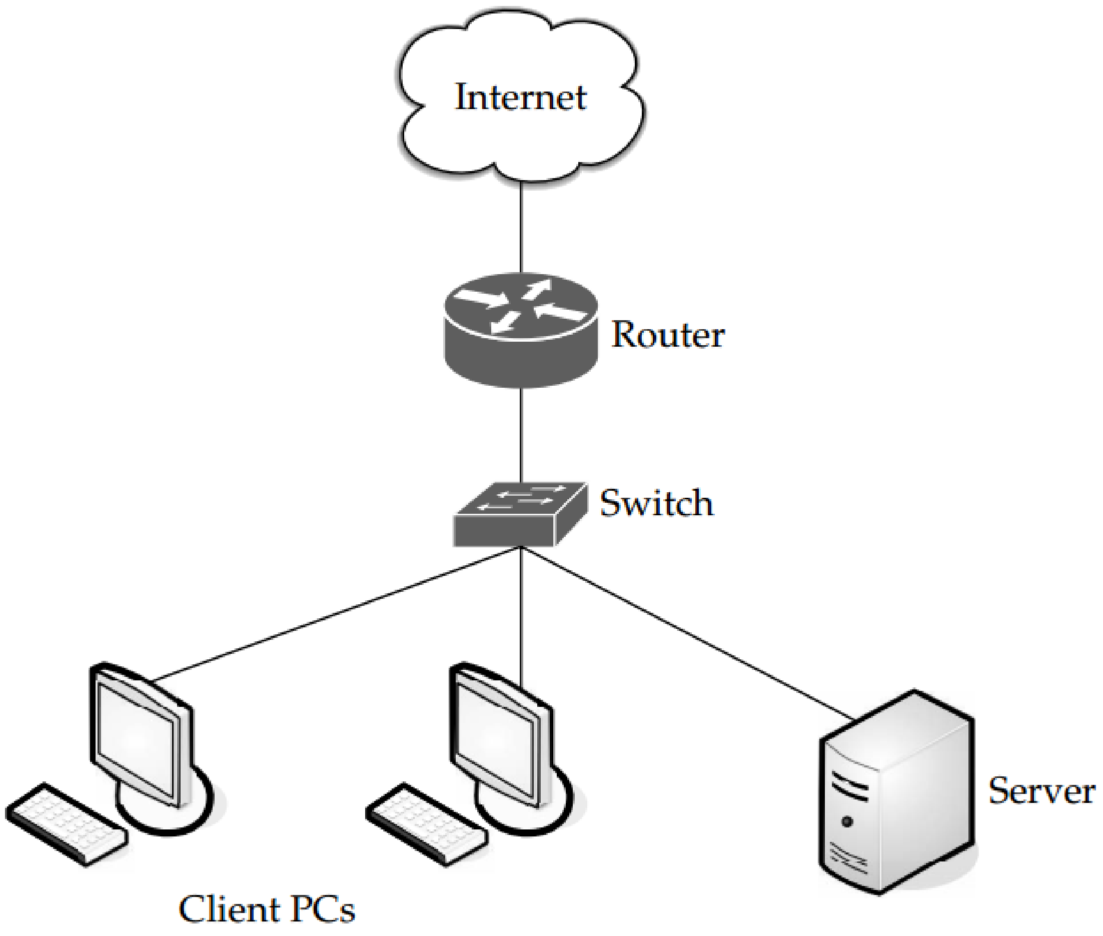


Figure 1.1: Internet is idiomized cloud [1]

1.2. Introduction to Cloud Computing

According to the National Institute of Standards and Technology (NIST), defines Cloud computing as "Cloud computing is a model for enabling 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". This Cloud model promotes availability and is composed of five essential characteristics, three service models, and four deployment models [2].

According to Garner, A style of computing where service is provided using different model and layers of abstraction across the Internet.

Simply, the term cloud computing means delivering the computing service over the internet according to the demand, can access from any remote location. It also involves the deploying software networks and group of remote servers that allows to upload the different kind of data sources on the cloud so that real time processing can be done without storing the processed data on the cloud for computing results generation.

1.3. Evolution of Cloud Computing

There has always been a confusion about the evolution of cloud computing and most importantly the involvement of grid computing in it. Some people say that grid computing and cloud computing are the same concept while some says that grid computing get extended and become cloud computing. To know exactly what the truth is we need to understand grid computing first then we can know how cloud computing evolves from the grid computing.

The development of HPC (High Performance Computing), parallel and distributed computing lead to the evolution of a complex phenomenon called Grid computing.

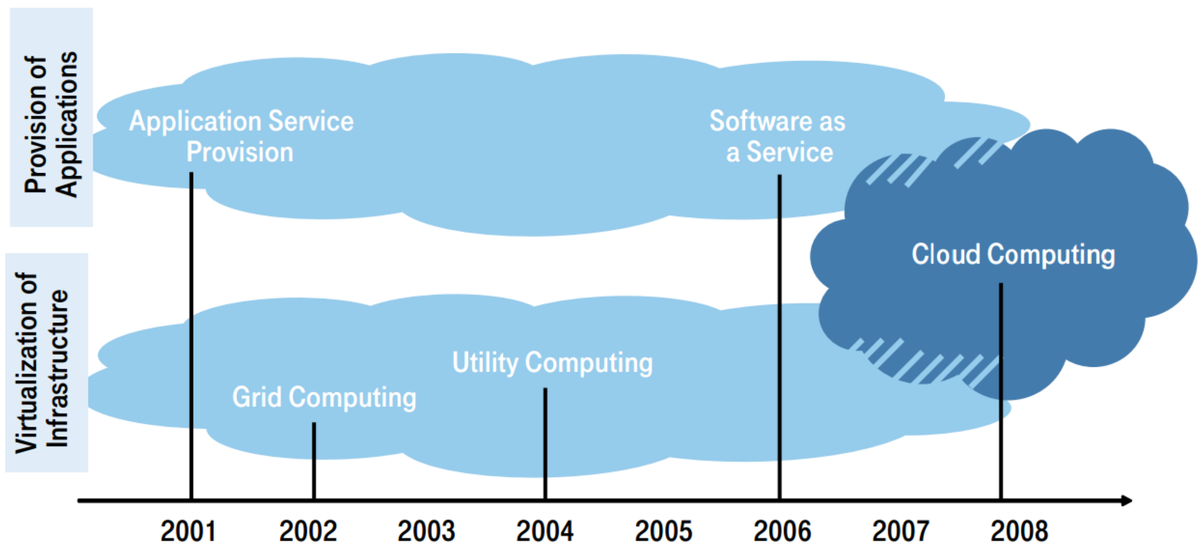


Figure 1.2: Two Merged Concepts and become Cloud Computing [3]

“A computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities [1].”

Figure 1.2 shows that how cloud computing had evolved. Basically grid computing focuses on the concept of interlinking the systems so that the availability of resources and scalability will increase. As the roots of grid computing lies in the HPC, Parallel and distributed computing, it can also infer that the concept of cloud computing is directly link with these concept/computing. After that utility computing came into existence and that become the base of cloud computing.

Utility Computing gives the concept of pay-per-use, i.e. resources like computing power, storage, infrastructure, platform etc. for computing services, are measured / available on the pay-per-use basis. Then before cloud computing came into existence there is one concept that is one of the key concept of cloud computing is software as a service. And all these concepts got merge into a single entity called “Cloud Computing” having the features from distributed computing to the software as a service.

Here are the three new dimensions uncover by the cloud computing

- The cloud computing creates an illusion of having infinite computing resources available on demand, thereby the user need not to plan beforehand for provisioning.
- The cloud user needs not to do any up-front commitment, thereby it allows companies to increase or decrease the resources according to the demand.
- The feature to pay for use of computing resources on a temporary basis and release them as required, thus when the resources are no longer useful it allows user to let them go.

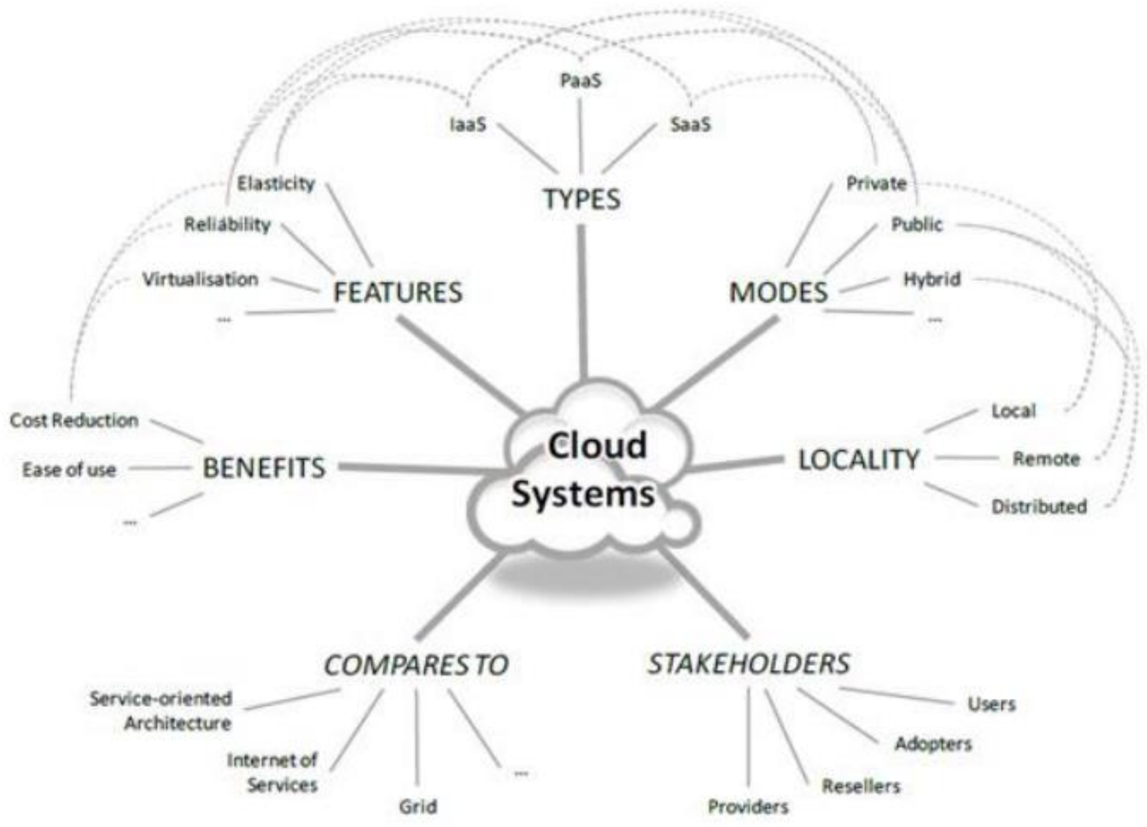


Figure 1.3: Coverage of cloud [4]

Figure 1.3 shows the most of the aspect and different level of the customers present in the cloud computing. Some of the benefits of Cloud computing [3] [5] are described below:

- **Scalability** – Now by using Cloud computing users are able to get the new computing resources and software applications as require, simply user can quickly scale up their IT operations. Furthermore, now in traditional IT there doesn’t need of pre-purchasing of resources to meet peak requirement.
- **Flexibility** - Cloud computing allows users to decrease / increase computing resources as required and allows to utilize huge amount of computing power on an "on-demand" basis. It also helps in ensuring that currently operating business processes and computing services will not get affected (like getting slow) by the change in resource. Pay per use model allows user to pay according to the demand for computational power, storage and other infrastructure.

- **Economics** – Traditionally for industries to get IT services they have to invest a large amount for IT infrastructure establishment to sustain day-to-day business operation and also continue to spend over hardware and software maintenance. By using cloud computing user's overall IT expenditure got reduces which involve on-going support, infrastructure development, software licensing and upgrades.
- **Inherited Resiliency** - Since the Internet is a highly resilient computing environment Cloud computing removes single points of failure. To enhance resiliency some service providers had added extra functionalities. For example, the "Elastic IP Address" and "Availability Zones" features of Amazon.com EC2 allow dynamic IP address re-mapping mechanism and multi-location of application software in an event of service interruption.
- **Highly Automated** – As the maintenance and upgrades of the services are done by the IT professionals of cloud service provider, so there is no longer need of IT staff to worry about the complex structure behind the delivered computing services.
- **Improved cost and resource utilization:** Now user can use the resources whenever needed and also pay per use, this would improve the utilization of the resource and cost to the user is also improved/reduced because users can return the resources back to the cloud provider whenever they don't need them.
- **Resource pooling:** Typically a user doesn't have any information about the locality of the service provider. Thus by assigning resources dynamically and virtually the provider can serves multiple consumers. Also by this the availability of resources will increase.

1.4. General characteristics

Here are some general characteristics which Cloud computing offers [3] [4]:

- **Utility-based pricing**: The Model used by the cloud computing is pay-per-use pricing model which comes from the utility computing and that results in lower the cost. From service to service the pricing scheme can vary accordingly like a wholesale and retail market e.g. one cloud service provider(say SaaS) may rent a services from another service provider(say IaaS) on the hourly basis and now the service provider (SaaS) act as a retailer and further provide that service to the end level consumers. In this way it can serve simultaneously many consumers.

- **Broad network access**: User can reach to a cloud services only by using internet. So cloud services can be accessible by any device which is able to connect with internet such as a PDA, a mobile phone, a laptop, or a tablet etc. Today for localization and high network performance many of service providers are spreading there datacenter at many locations over the globe.
- **Shared resource pooling**: A pool of computing power is available to the IaaS provider that can be allotted dynamically to use by multiple consumers for utilization. By dynamic resource assignment, Infrastructure provider will get a much flexible model for managing their operating costs and resource usage.
- **Dynamic resource provisioning**: One of the key ability of cloud computing is that user can obtain and release the computing resources very frequently. As compare to the traditional model in which resources are supplied according to peak demand, while dynamic resource provisioning allows user to get the resources based on the current demand, which can result in considerably lowering the operating cost.
- **Service oriented**: As the concept of cloud computing is fully based on service. Hence it mainly signifies on service management. By using SLA, consumer will get a clear objective/view of service provider. It also helps the provider by gaining the trust of the consumer.
- **Self-organizing**: Since cloud computing provides on-demand allocation and de-allocation, so cloud service providers should manage their resources accordingly. The feature of automated resource management help the provider in making the model highly agile, by which provider is able to respond quickly to frequently change in demand.

1.5. Service model

Basically there exist three service models named as:

- SaaS (Software as a Service)
- PaaS (Platform as a Service)
- IaaS (Infrastructure as a Service)

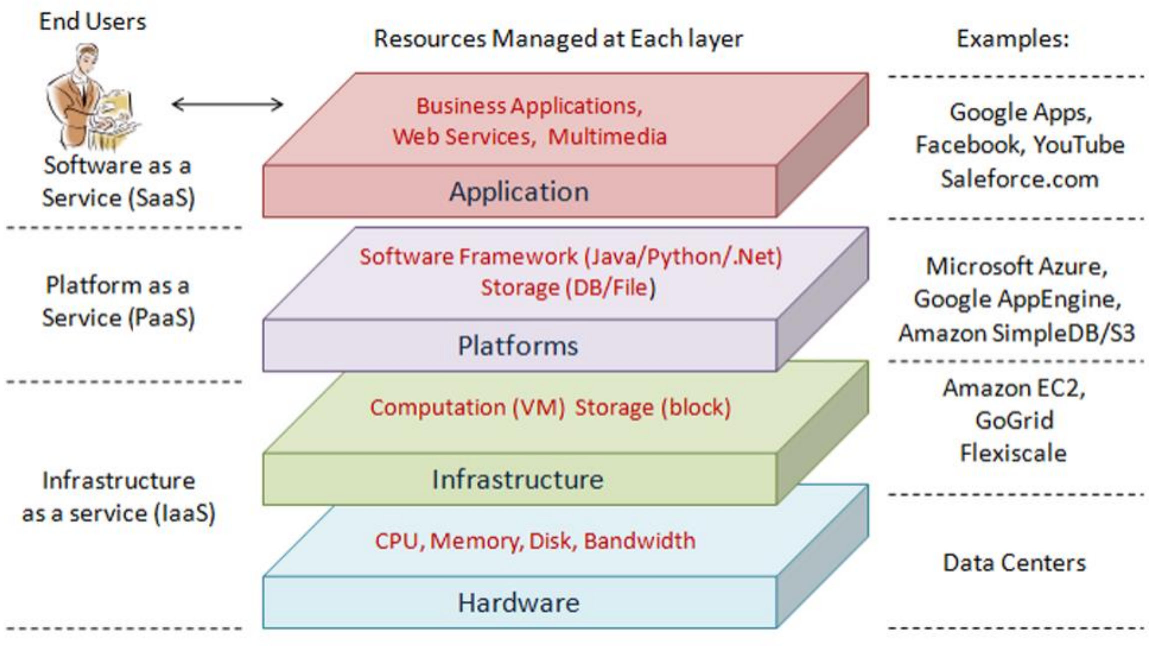

End Users
Resources Managed at Each layer
Examples:
Business Applications, Web Services, Multimedia
Application
Software as a Service (SaaS)
Google Apps, Facebook, YouTube Saleforce.com
Software Framework (Java/Python/.Net) Storage (DB/File)
Platforms
Platform as a Service (PaaS)
Microsoft Azure, Google AppEngine, Amazon SimpleDB/S3
Computation (VM) Storage (block)
Infrastructure
Infrastructure as a service (IaaS)
Amazon EC2, GoGrid Flexiscale
CPU, Memory, Disk, Bandwidth
Hardware
Data Centers

Figure 1.4: Architecture of Cloud computing with various services [4].

There are different types of services that are divided by cloud provider. These services are shown in Figure 1.4. Generally these services are called as XaaS (everything as a Service). The service provider maintain a level of abstraction i.e. the services are provided as “black boxes” to the end user and also an instruction set is given which tells what and how to use these services and how these service is available to the them but the internal working detail of the service is hidden from the consumer/end user. A brief description about these services and there benefits are given below:

- **Infrastructure as a Service** – In IaaS [2] [4] services provided to the consumer is infrastructure i.e. storage, networks, processing, and other fundamental computing resources which enables the end user to deploy and run arbitrary software, including operating systems and applications. The end user does not allow control or manage the infrastructure but has can control over storage, operating systems and deployed applications. End user can obtain as many resources (Infrastructure) according to need and released them back to the provider accordingly. The basic concept behind the providing the infrastructure as a service is virtualization. Basically the strategy of virtualization is to creating isolated virtual machines

(VM) that are independent from both the other VMs and underlying hardware. Examples of IaaS providers are: Amazon EC2 , GoGrid and Flexiscale. Benefits of IaaS are :

  - Less power consumption.
  - Less hardware cost.
  - Higher resource utilization.

- **Platform as a Service** – In PaaS [2] [4] services provided to the consumer is platform layer resources i.e. deploying consumer applications created using programming languages, tools, libraries, and services supported by the provider like software development frameworks and operating system support onto the cloud infrastructure. For the application-hosting environment, the consumer/end user has control over the deployed applications and possibly configuration settings, not on the detail about the platform provided. Simply, it provides a platform (Tools and Technologies) to develop cloud services and applications (e.g. SaaS). Example of PaaS providers are: Microsoft Azure Services Platform, Google AppEngine and Amazon S3 etc. Benefits of PaaS are:
  - Lower development cost.
  - Developers need not to worry about any failure or other resource related issues only need to focus on application code.
  - Can hosts both completed and in-progress cloud applications.
- **Software as a Service** – In SaaS [2] [4] services provided to the consumer is provider's applications running on a cloud infrastructure. As discussed earlier in cloud computing resource pooling is there so various clients devices can access the applications through either a program/application interface, or by a thin client interface like a web browser (e.g., web-based email). SaaS [6] [7] is a multi-tenant platform. The consumers need not to purchase any proprietary rights of software. They just have to use the application which is provided by the provider; according to the term and condition defines in the SLA (Service Level Agreement). Example of the SaaS providers are: CRM application service by salesforce.com, web based calendar application by Google, and online SharePoint by Microsoft etc. The provider charges the services as per the usage on monthly basis. The benefits of SaaS are:
  - No need of software licensing.
  - Free from version compatibility issues.
  - Reduced upfront cost.

1.6. Cloud Computing Deployment Model

Basically there exist four types of cloud deployment models [4] [6] which have been defined in the cloud community as shown in Figure 1.5:

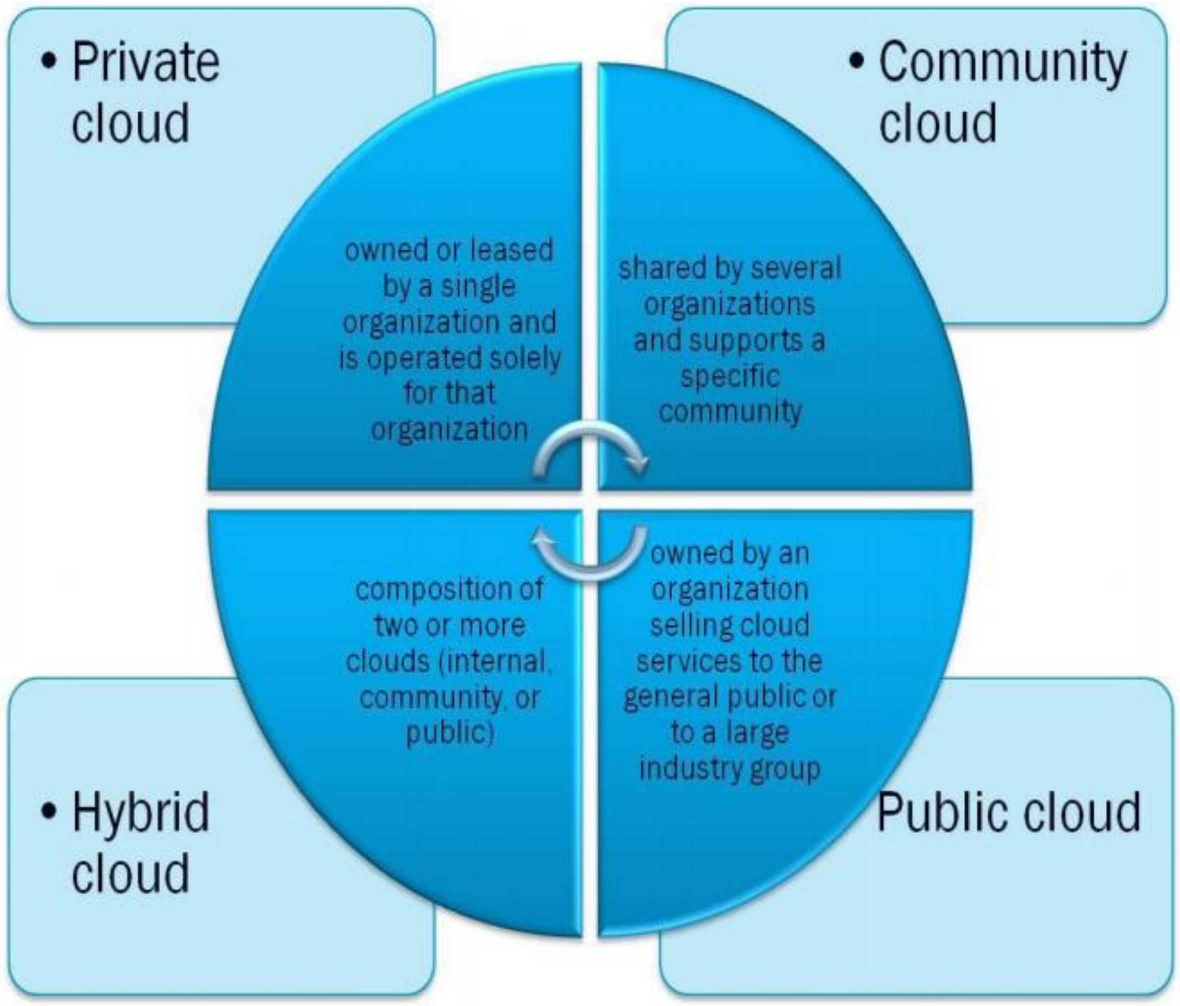

• Private cloud
owned or leased by a single organization and is operated solely for that organization
• Community cloud
shared by several organizations and supports a specific community
composition of two or more clouds (internal, community, or public)
owned by an organization selling cloud services to the general public or to a large industry group
• Hybrid cloud
Public cloud

Figure 1.5: Deployment models of cloud computing [6]

- **Private cloud**

  In simple words, a cloud designed solely for a single organization or dedicated to a single organization is known as private cloud or internal cloud. A private cloud can be built and managed by the third party or by external providers or by the organization. It offers the highest degree of control over security, performance and reliability. However, as compare to the traditional proprietary server farms it is quiet similar, so benefits such as no up-front capital costs does not provided. Benefits of private cloud are:

  - To optimize and maximize the in-house resource utilization.
  - Data privacy makes the cloud more secure.
  - More Reliable

- **Public cloud**

  A cloud which allows the general public or a large industry group to access the services provided by the cloud owner, such type of cloud is known as public cloud. The owner of this type of cloud can be a business, government organization, or academics, or some combination of them. The cloud is operated and maintained by the owner not by general public, also the owner have full control over cloud with its own charge model, costing and profits etc. Examples of public cloud are: Amazon EC2 and Google AppEngine etc. Benefits of public cloud are:

  - General public can access the cloud
  - Less data transfer risk

- **Community cloud**

  A cloud which allows only the specific community of consumers that have shared concerns to access the cloud services, such type of cloud is known as community cloud. The owner of this type of cloud can be a third party, one or more organizations within the community, or some combination of them. This cloud is different from private cloud, which designed solely for a single organization and also different from public clouds, which serve the different needs of multiple users. Some Benefits of Community clouds are:

  - The cloud infrastructure can be hosted by a third party organization
  - Economically scalable

- **Hybrid cloud**

  A cloud which is made by combing two or more unique clouds (private, community, or public cloud) such type of cloud is known as hybrid cloud. After combining they should remain unique entities but are bound together by standardized or proprietary technology. By combing the cloud it is able to take benefits of the clouds by which it is made. Benefits of hybrid clouds are:

  - More flexible.

- More security over data as compare to public cloud

1.7. Research Issues in Cloud Computing

Recently, the Cloud computing is fast growing technology which has high impact on IT industries but still it is in initial phase, so it suffers from many challenges. Some research issues in cloud computing are briefly described over here. This section briefly describes the research issues [4] [6] in Cloud computing environment.

1.7.1. Security Issues

Security is one of the major issues of cloud computing as well as traditional computing. Since the physical security system is only access by the infrastructure provider, so the service provider is rely on the infrastructure provider for the full data security. There are many security issues some of them are as follows:

- Security Policy and Compliance
- Data Security
- Virtual Machine Security
- Network Security
- Data Transmission
- Data Availability
- Data Privacy
- Data Location
- Data Integrity
- Patch management

1.7.2. Automated Service Provisioning

As the demand of user/consumer varies so the cloud service provider is able to frequently allocate and de-allocate resources dynamically to satisfy the need of consumer according to the service level agreement. It mainly involves:

- To handle the dynamic request for resources it needs an application model at each level.
- Periodically it should able to predict the future demand for the resources requirement.
- For the predicted requirement automatically allocating the resources [4].

1.7.3. Energy Management

As per the government rules and environmental standards, the service provider has to reduce the energy consumption so that it can meet the requirement. For this service provider should manage energy properly, the designing of efficient data centers is needed physically as well as technically, for example placing a datacenter at a cooler place will cause less cost on cooling, or by using energy aware job scheduling to manage the energy effectively by turning off the unused machines, or server consolidation [4].

1.7.4. Virtual Machine Migration

The main aim of virtual machine (VM) migration in cloud is to balance the load across the datacenters. VM migration came into existence by process migration techniques. So there is certain limitation to the virtual machine migration that should be improved. One of the benefits of VM migration is avoiding hotspots. Currently, to respond to sudden workload changes lacks in agility, while detecting workload hotspots and initiating a migration. [4].

1.7.5. Server Consolidation

By using virtualization (VM migration) multiple under-utilized servers can be migrated to single server, so that energy can be saved as other servers are in energy saving state. But for controlling energy consumption and resource utilization, using VM migration alone is not sufficient. So there is need of server consolidation, which is used to maximize resource utilization and also helps in minimizing the energy consumption in the cloud. Server consolidation will not harm or affect the performance of application or service running on that. Recently, the dependencies among VMs are also being considered, such as communication requirements etc. [4].

1.7.6. Data Issues

The issues related to data are mainly about the security, but there is also some other issues like interoperability between platforms. So for consumer it is difficult to accept the change i.e. customer may find the difficulty in switching. Already there are researches going on regarding the interoperability of data. For interoperability solutions like intermediary layer, implementation can work. Also to meet the standards is also a challenge. Also, there are problems in data placement and transport within the applications and also as the best of my knowledge till now there is no system that solves the PaaS interoperability issues. There is need to work on the idea of the transportation and placement of data at each level of the system so that the cost can be minimize in this process. All this makes the data issue, one of the major challenges in the cloud computing.

1.7.7. Workflow Management Issue

The management of workflow is one of the issues in Cloud computing. A workflow can be used to represent most of the business process. A workflow is consist of tasks that to be done in order to complete the process. These tasks may or may not be dependent on another task. The aim of workflow management system is to ensure that the information flow, Task invocation, virtual machine allocation, and task synchronization, should done in a specific order. Workflow scheduling is the main issue in workflow management system. Workflow scheduling is a NP-Complete problem; it is define as to find an optimum execution sequence for the tasks in the workflow, i.e., an execution sequence which obeys the constraints (conditions), represented by the business logic of the workflow. Workflow scheduling also Maps and manage the workflow task's execution on shared resources [7]. Workflow scheduling and Workflow Management System has been discussed in detail in the next chapter.

The issue workflow management has been considered in this work. The main objective of this work is to do scheduling on homogenous processors, so that it minimizes the resource utilization and execution time to complete the tasks of workflow.

1.8. Structure of the Thesis

Here is brief description about the chapters which are present in this thesis:

Chapter 2 – This chapter emphasis on the literature review of the WMS(Work flow management System) and also discusses its various element like fault tolerance. After that a classification of workflow scheduling in cloud computing is done. In the end of the chapter there is discussion about various existing algorithm.

Chapter 3 – This chapter starts with defining the problem description with its model i.e. task model and machine model. And the chapter ends with the gap analysis and motivation behind the work.

Chapter 4 – The solution for the problem is discussed here i.e. for the problem describe in previous chapter get solved here using proposed algorithm.

Chapter 5 – The implementation notes and the analysis of the algorithm is done here. The chapter starts with the little description about the eclipse (IDE), and workflowsim toolkit for creating a cloud environment.

Chapter 6 – This chapter starts with the conclusion of the thesis and after that future possibility of the project is given.

CHAPTER 2

REVIEW OF LITERATURE

This chapter discusses about analysis of Workflow Management System, a taxonomy on Existing Workflow Scheduling algorithms and also discusses about the various existing simulator cloud environment.

2.1. Workflow Management System

WMS (Workflow Management System)[14] is used to assignment, management or switching of resources for a proper execution of the workflow tasks. The major components of Workflow Management System are:

- Workflow design
- Fault tolerance
- Workflow scheduling
- Information retrieval
- Data movement

The Workflow design used to describe the structure and composition of the components of a

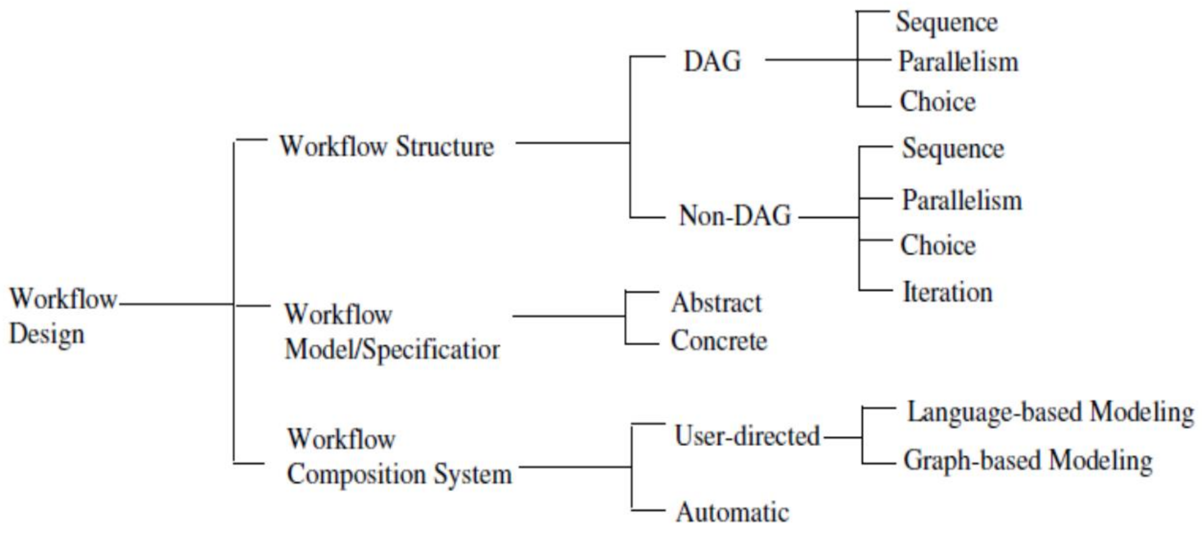


Figure 2.1: Taxonomy of Workflow Design [15]

workflow. The figure 2.1 shows the elements of workflow design.

A workflow consists of tasks, so the relation between these tasks is described by the workflow structure. The structure of the workflow can be of two types:

- Directed Acyclic Graph (DAG
- Non DAG.

Then also in DAG based structure workflow can further be divided into Sequence, Parallelism and Choice. In sequence structure, as the name suggest the tasks are ordered in a serial way, i.e. tasks are executed in sequence that's why, it is called as sequence structure. In this new task can be executed if and only if the previous task already been successfully executed. But in case of parallelism structure, concurrent execution of tasks can be done in the workflow. And the choice structure is the mixture of both the workflow tasks can be executed in series as well as concurrently. The Non DAG based structure, contains all the DAG based patterns along with Iteration pattern. In Iteration pattern, the task can executes in an iterative manner. A complex workflow can be formed with the combination of these four types of patterns in multiple ways.

- Workflow specification/model is used to define the type of workflow, and also characterize the resources that are need by the workflow for executing the tasks. Workflow models categorize the workflow in two types concrete and abstract. For executing the tasks of a workflow; it needs resources, if the tasks are binded to specific resources then those type of workflow are said to be concrete workflow. And if the specific resources are not referred during the task execution, it means that workflow is in abstract form those type of workflow is called as abstract workflow.
- To organize the workflow i.e. to create, update, or modify the workflow there is a need of Workflow composition system in workflow design. The workflow either be generate by the user or automatically. In automatic, system itself generates a workflow automatically. But in the user directed system, user is responsible for creating the workflow. In this user can create, update, and modify the workflow structure directly. The user can modify the workflow by using two different approaches either by language based modeling or by graph based modeling. In language based modeling user have to use some markup language like XML for

building the workflow. Also it is difficult to remember a lot of syntax for a user having non-IT background. But there is also a simple approach that is Graph based modeling [15].

- A fault Tolerance is similar to the general distributed system, one of the key features of workflow management system is fault tolerance. A system which is able to handle or control the failure due to several reasons like insufficient memory, overloaded resource, network failure, task failure, and resource unavailability/failure etc. which occurs during scheduling. Figure 2.2 shows the key elements of Fault tolerance system.

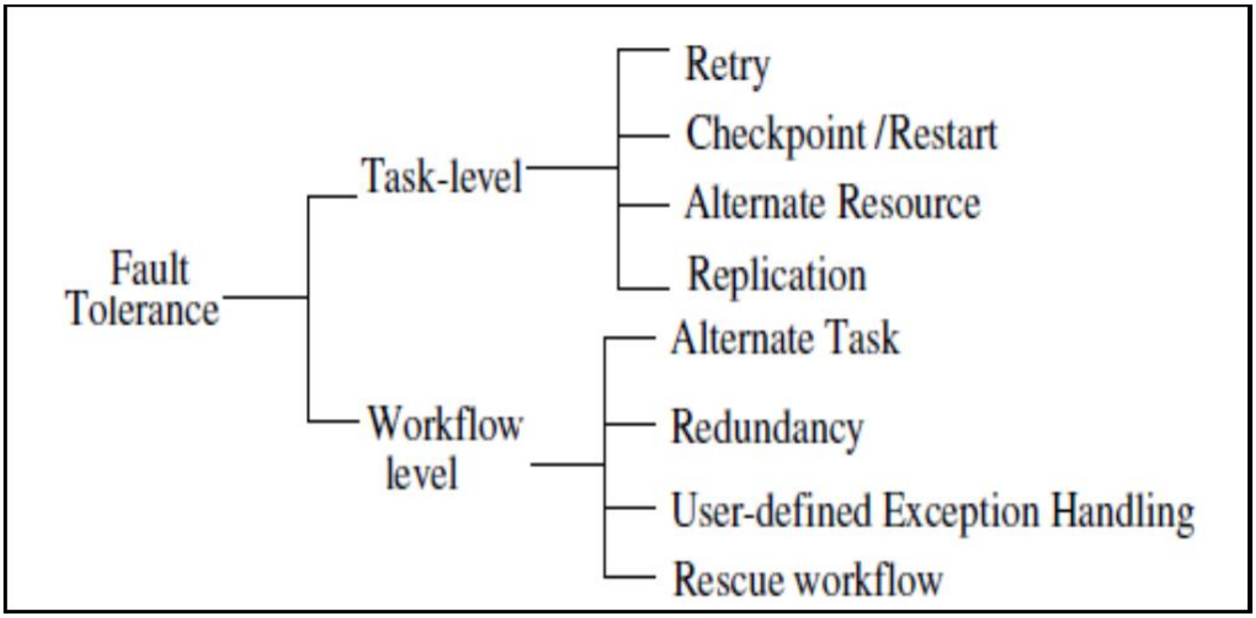


Figure 2.2: Classification of Fault tolerance system [15]

- On the next level of fault tolerance elements, there are two categories one is task level failure tolerance and another is workflow level fault tolerance. As the name suggests if the failure occur due the tasks in the workflow; those failure are come under task level fault tolerance and if the failure occur due to the structure of workflow than it is handle by workflow level fault tolerance. Task level is further divided into four aspects [15]:
  - **Retry**: One of the simplest technique to overcome the failure is Retry. Basically, it is an hit and trail method. In which the system try to execute the same task again over the same system where the failure occurs. It is not expected to work always but it sometimes work.
  - **Alternate resource technique:** Whenever a failure happens it tries to execute the same task (failed task) on other resource.

- **Checkpoint technique:** Similar to databases management system, checkpoints are created so that if a failure occurs it can roll back to the checkpoint i.e. saved state. Basically, Checkpoint stores the state of the system. Whenever a failure occurs; the system tries to start the execution of task from that saved state of the same task on the other resource.
- **Replication technique**: In this type, the system replicates the task on different resources so if the failure happens on one of the resource it will not affect the process.

- Now fault tolerance at Workflow level does not consider the tasks of the workflow it works on the structure of workflow to handle the failure. Workflow level fault tolerance is further divided into four aspects [15]:
  - **Alternate task technique**: In case of failure it tries to execute another or alternate task in place of task which is failed to execute.
  - **Redundancy technique:** Similar to replication, it also tries to execute the tasks simultaneously on more than one resource but here instead of single alternate task it replicates the multiple tasks in place of task that are previously failed.
  - **User defined exception handling**: A better way to handle the exception is exception handling if user thinks that the workflow can go to the state of failure. So user itself define some mechanism what to do when a failure occurs.
  - **Rescue workflow:** In rescue technique if a failure occurs then it continues the execution of other tasks without considering the failed task of the workflow until the execution is possible. The process continues, if the execution is completed then it identifies the failed task and request it to submit again.

2.2. Workflow Scheduling

The term workflow scheduling in the cloud computing environment is defined as, since a workflow is consist of task; and a task is defined as smallest unit of a workflow which is to be scheduled i.e. assigned it, to a resource. Simply, mapping the tasks with the specific group of resources which are distributed in different location or geographical area and having different administrative privileges [13][16]. The resource can be a network link for transporting data, or a data storage device, or a virtual machine for data processing. The characteristics of a task are like computation cost, priority, transportation cost, etc. So there are also some characteristics of schedule are like makespan, resource utilization, cost, etc. Here in figure taxonomy on cloud computing is given.

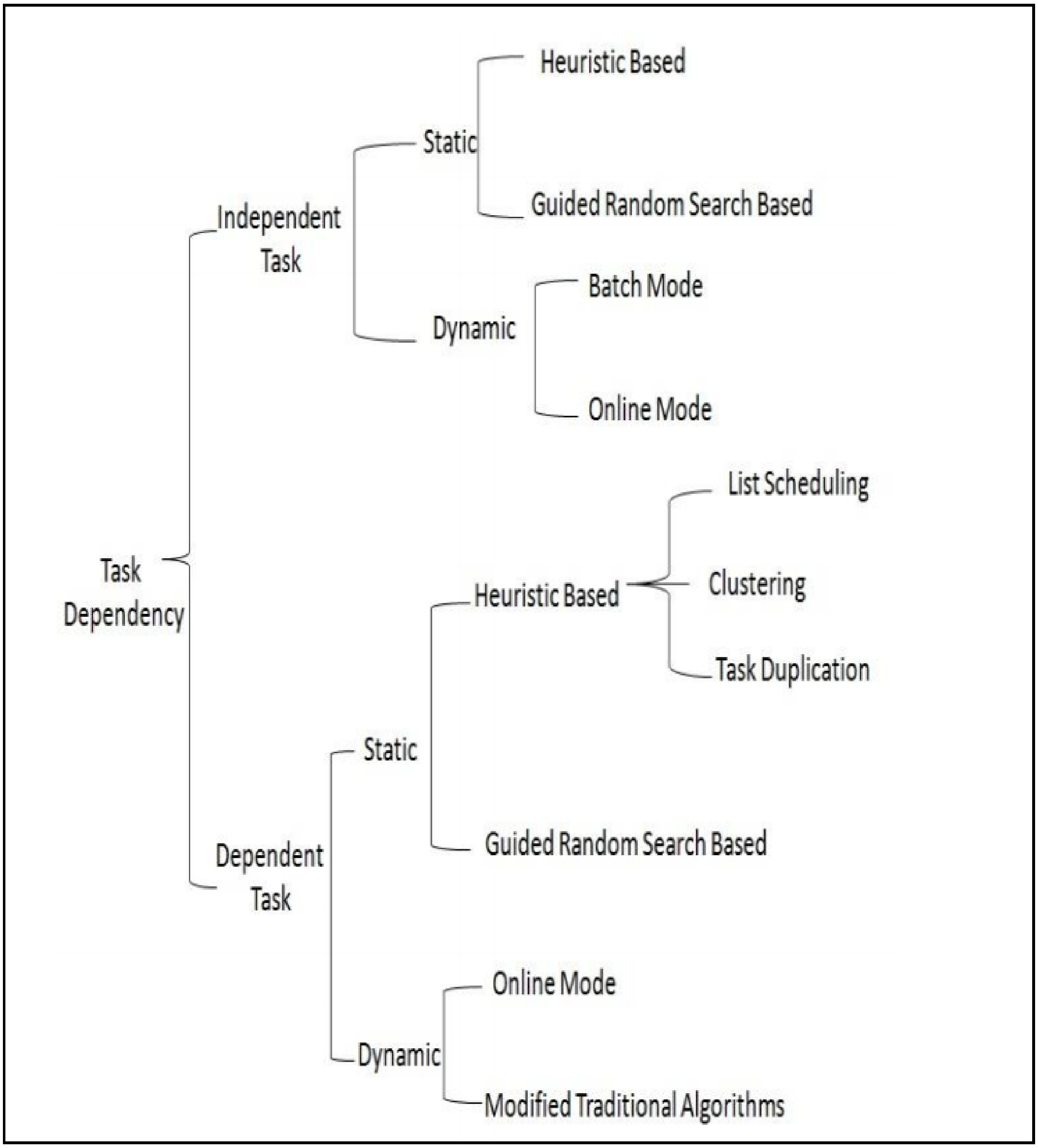


Figure 2.3: Dependency Based Taxonomy of scheduling algorithms in Cloud computing [16]

On the basis of task dependency, it is classified as dependent and independent task. The tasks are said to independent, if there is no need of communication between the tasks. But in case of dependent task, the tasks are related to each other, i.e. there is need of communication between tasks. So during the scheduling of dependent task there is need to make some precedence order to ensure the constraints define by the user. Objective of scheduling is to minimize the execution time or total length or makespan of the schedule. Thus dependency plays a vital role in scheduling. Dependency induces an extra factor for consideration that is communication cost, which the time is taken by tasks to transfer the data.

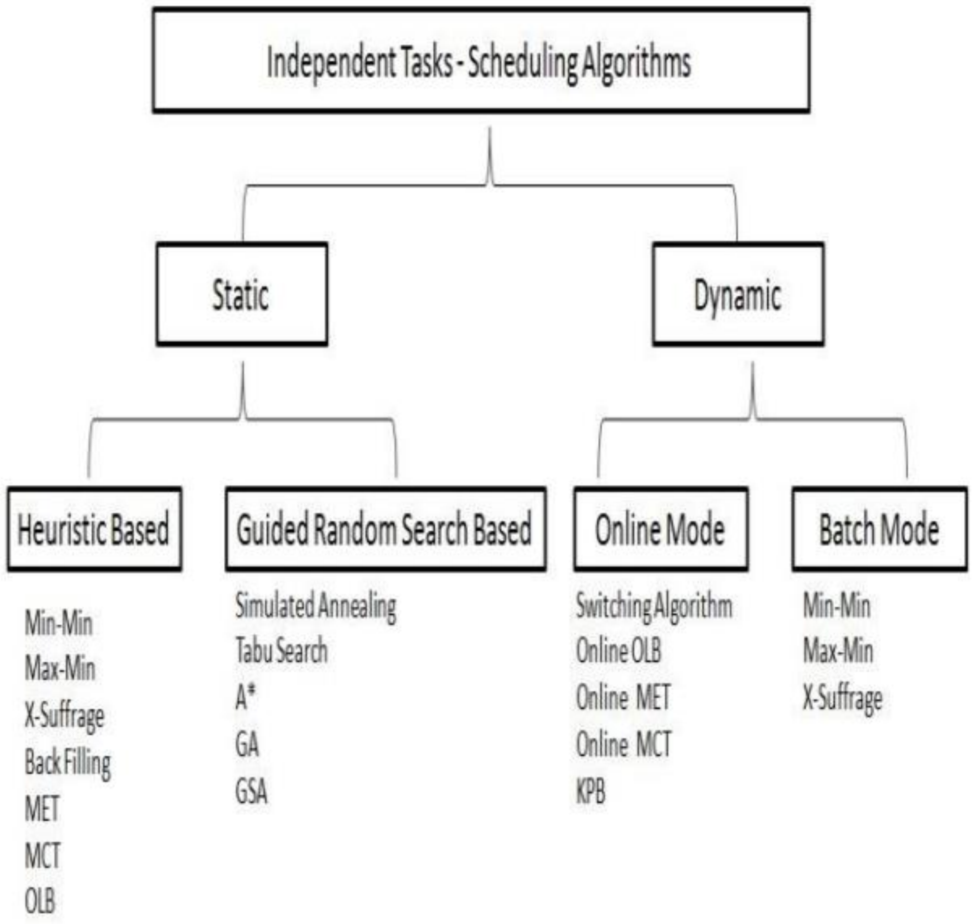


Figure 2.4: Classification of Scheduling Algorithms for independent tasks [16]

It is further classified as Static and Dynamic scheduling algorithms. As shown in figure 2.3, for a static scheduling information about all the resources which are going to be schedule in the cloud Simply, before scheduling the cloud in static manner all the independent subtasks should be available before scheduling. But a dynamic scheduling is that in which the prior knowledge about the task is absent like resources needed by the task, execution environment, etc. as the jobs arrive in the system dynamically.

Figure 2.4 shows the further classification of scheduling algorithms for independent task, it is divided into two types of algorithms: static and dynamic. These static dynamic algorithms are further classified for the independent tasks. The independent task can be further classified in fine and coarsely grained. In fine grained the nature of tasks are fine i.e. small tasks in term of execution time and the communication time is not required. But in coarsely grained the task can communicate with other tasks but very infrequently.

2.2.1. Static algorithms for independent tasks

For independent tasks [16], the static algorithms used for scheduling is further classified as Heuristic based and Random search based algorithms.

- **Heuristic based algorithms:** There are various heuristic based algorithms like OLB, MET, MCT, Max-Min etc. these are the basic algorithm which uses heuristic for scheduling the tasks on the cloud. Simply, a heuristic is an instinct of a human. In term of computing, Heuristics is predefine or pre-calculated cost for any process that can be used as a priority or a solution to the problem.
- **Random search based algorithms:** There are various random search algorithm exist like GA (Genetic algorithm), SA(Simulated Annealing), A*, GSA, Tabu search etc. In these algorithm there exist large set of solution are present; for finding the best path or best solution these techniques are applied to get an optimum schedule.

2.2.2. Dynamic algorithms for independent tasks

For independent tasks [16], the dynamic algorithms used for scheduling is further classified as Online mode and Batch mode algorithms.

- **Online mode:** The static algorithms like OLB, MET, MCT can also be used to dynamically schedule the independent tasks. Also, there are other algorithms for dynamically scheduling the independent task like Switching Algorithm and the k-percent best (KPB).

- **Batch mode:** If the algorithms like Min-min heuristic, Suffrage heuristic, Max-min heuristic, etc. can dynamically schedule the tasks continuously or in batch mode.

So these are the types of algorithms that are used to schedule the independent tasks of workflow. Now for the dependent tasks the algorithms are classified as shown in figure. Similar to the independent these algorithms are also further classified as static and dynamic algorithm. Now here is a brief description about the algorithms that are further classified under this category i.e. dependent task scheduling algorithms shown in figure 2.5.

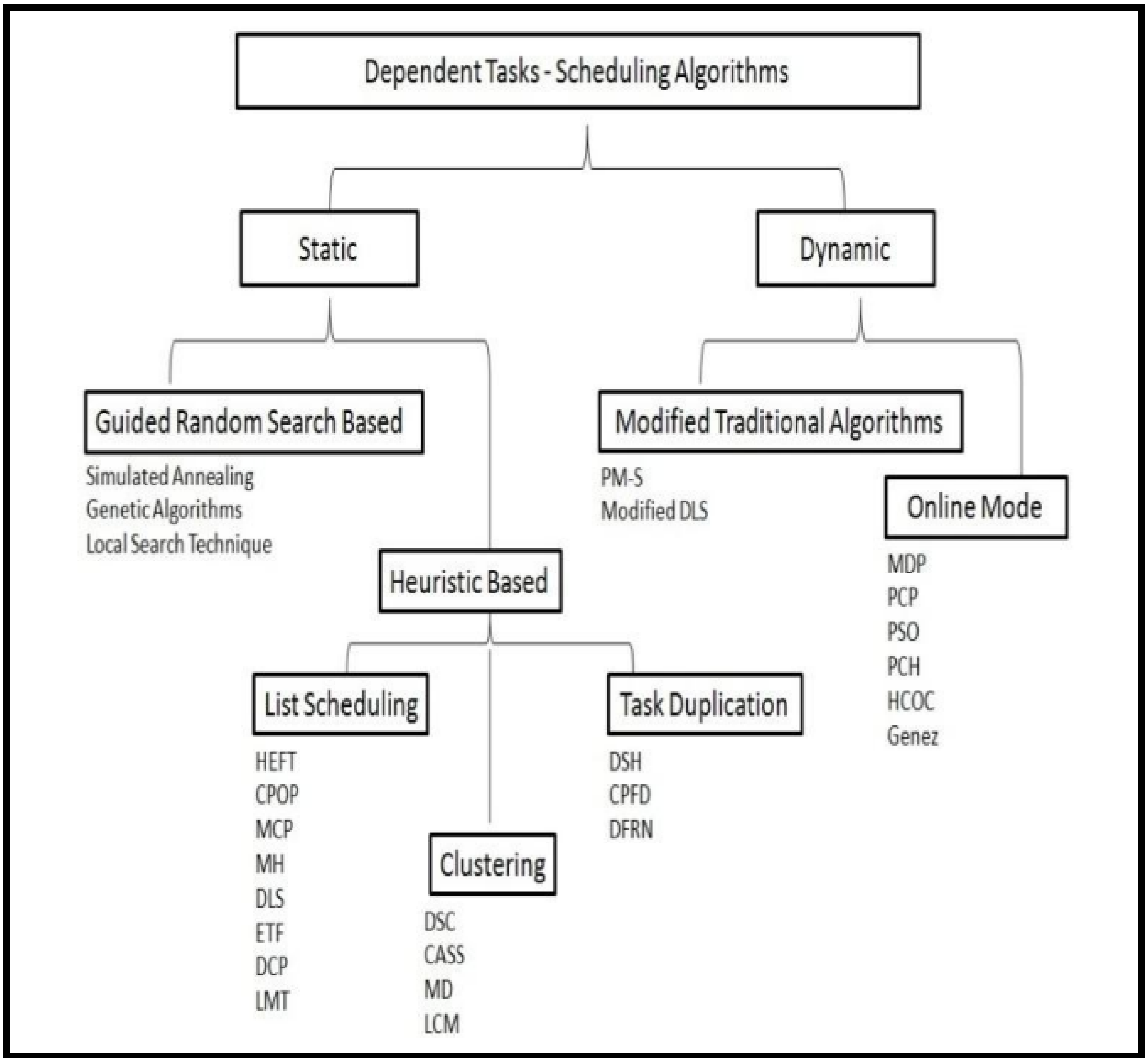


Figure 2.5: Classification of Scheduling Algorithms for dependent tasks [16]

2.2.3 Static scheduling algorithms for dependent task

As discussed in previously that there are coarsely grained independent tasks which are also dependent on the previously executed task [16]. Now further these static algorithms are classified as heuristic based and guided random search based algorithm.

- **Guided Random Search Algorithm:** It is similar to the algorithm as discussed in scheduling for independent task, algorithm like Genetic algorithm, simulated annealing, A* search based etc.
- **Heuristic Based algorithms:** It is also similar to the independent task scheduling, these algorithm works on the basis of heuristic. Heuristic can be anything like rank, order, cluster head, etc. So in here it is further classified as
  - List based scheduling
  - Clustering based scheduling
  - Task duplication based scheduling

Here is the brief description about these scheduling techniques [16]

- **List based algorithms**: Levelized - Min Time (LMT), Modified Critical Path (MCP), Mapping Heuristic (MH), the Critical-Path-on-a-Processor (CPOP), The Heterogeneous-Earliest-Finish-Time (HEFT) and Dynamic level Scheduling (DLS) are some example of the list based scheduling. List based scheduling is that in which a list of task is prepared according to the priority of the tasks then according to the list a schedule is created or tasks are assigned to the machine according to the list order. One of the major advantages of this type of algorithm is: the time taken by these algorithms, to create the schedule is less that makes it a more practical approach.
- **Clustering based scheduling:** Linear Clustering method, Clustering and Scheduling system (CASS), Mobility Directed and Dominant Sequence Clustering (DSC) are the examples of clustering based scheduling. Before executing the cluster algorithm it is not known that how many clusters it can form. So in this approach scheduling of task are done by using cluster technique the task lies in same cluster will schedule on the same machine. But as we do not know the number of cluster so the number of machine should

be unbounded. To make the solution it needs two more steps: first is merging of clusters to make the machine requirement bounded and second is prioritizing the task of same cluster to schedule the task execution on machine.

- **Task duplication based scheduling:** Bottom-Up Top-Down Duplication Heuristic, DSH (Duplication Scheduling Heuristic), DFRN (Duplication First and Reduction Next), CPFD (Critical Path Fast Duplication) are the some examples of the task duplication based algorithm. The basic concept of these algorithms is to reduce the makespan, i.e. to minimizing the communication overhead between the dependent tasks, by duplicating the dependent tasks. These different duplication algorithms differ according to the selection of the task to be duplicated.

2.2.4 Dynamic scheduling algorithms for dependent task

It is further classified as: modified traditional algorithms and online mode [16].

- **Modified traditional algorithms:** These algorithms are generally derived from any other existing algorithm like [17] an algorithm is proposed (pM-S) which is the extended version of the traditional dynamic Master-Slave scheduling model. Similarly in [20], a Dynamic Level Scheduling (DLS) based algorithm is proposed. Traditional DLS for independent task is based on scheduling the task on the basis of heterogeneity in resource, but in this newly proposed algorithm, for computing the task level dynamic length of the queue is also considered.
- **Online Mode:** MDP (Deadline-Markov Decision Process), PCP (Partial critical paths), PSO (Particle swarm optimization), HCOC (Hybrid Cloud Optimized Cost) are some of the best example of the online mode based scheduling algorithm. These algorithm dynamically schedule the dependent tasks to the available machine, some of the take longer time for execution but can make an optimal solution.

The objective of scheduling is to maintain the QoS parameters and to get the optimal system throughput by high performance and proper resource utilization. There are various scheduling algorithm in cloud computing environment. Now here is the brief description about various algorithms used in cloud computing.

- **First Come First Serve (FCFS)**

  As in the traditional system it works similarly in the cloud environment. The tasks are prioritize according the arrival time of the task i.e. the task which comes first will be serve or schedule first. Fast and simple make this algorithm more useful, but if a longer job comes first than shorter job will have to stay in queue for a long time which increase the waiting time of the algorithm, thereby decrease in overall performance of the scheduled workflow.

- **Min-min**

  As the name suggest, the algorithm first execute the tasks having minimum execution time. This approach is similar to the shortest job first. The tasks having small executing time are executed first so the large task waits for a long time. This can be done by making a list of unscheduled tasks, first sort the list in non-deceasing order of their completion time, then remove the task from the list which is having shortest execution time. [19].

- **Max-min**

  This algorithm is very similar to the min-min, in this first for every task and resource calculate minimum of all expected completion time, and then task having the maximum value in this set is selected for execution and this process is repeated until all the tasks are not assigned to the machine [19].

- **Minimum Completion Time (MCT)**

  This approach is very similar to the min-min, in this algorithm tasks having minimum completion time is randomly allocated to a machine. However, in min-min scheduling algorithm it takes all the unmapped tasks into the account, but in the case of MCT, it considers one task at a time [19].

- **Heterogeneous Earliest Finish Time**

  Wieczorek [18] and Durillo focused on cost and overall execution time. In this the priority to the tasks is assign on the basis of average communication time and average execution time of each task. Then according to the priority tasks are scheduled i.e. task with highest priority will schedule first.

- **Genetic Algorithms**

  These algorithms [21] are generally used to multiple objectives like load balance, makespan, cost etc. It uses various mutations and crossover functions to optimize the various parameters. Generally the genetic algorithm is used in the unreliable cloud environment it helps in giving a solution which reduces the fault and makespan of the workflow.

- **Critical Path based Scheduling**

  Algorithm based on critical path[11] i.e. these algorithms generally work, by dynamically finding the critical path of a workflow, which is having lowest execution time. Whereas some derivative of critical path based algorithm used to find the precedence of the node or task of the workflow. This approach mainly focuses on the reducing the makespan for workflow.

- **Multiple QoS with Multiple workflow**

  There are various algorithms based on the multiple workflow scheduling, generally they tries to maintain multiple the various QoS parameter i.e. multiple QoS. The algorithm by Xu et. al. [22] focuses on QoS parameters that are makespan and cost. For all workflow it calculate the average execution cost and average execution time and then the task having less covariance of cost and time is schedule first.

- **Ant Colony Optimization (ACO) based scheduling algorithms**

  Ant colony optimization is a part of swarm intelligence which is based on ants i.e. how ants used find the food and when they find the food, how does the other ants will know about the food. Actually whenever ant finds food, while returning to the colony it spread the pheromone on that path so that other ants will know where the food is. So the ACO based algorithms generally consider multiple objectives or multiple QoS parameters like load balance and cost etc. to find the optimal solution.

- **Particle Swarm Optimization (PSO) based Scheduling Algorithms**

  PSO is also a swarm intelligence technique, which is based on flocking behavior of birds. Similar to the ACO, if a bird (in PSO; particle/individual) finds food then all other birds start moving towards it. So in PSO, all the particles tend to move towards the best individual and hence optimizing the problem. So the PSO based algorithms tries to optimize the multiple QOS parameter. Generally after apply some other scheduling technique a schedule is generated and now this schedule is taken as input to the PSO algorithm to find the optimum schedule.

➢ **Selective Duplication (SD)**

  As the name suggests selective duplication, this algorithm uses less number of duplication as compare to other duplication based algorithm, by duplicating the task selectively. By this there is less duplication which results in the less time and space complexity. This algorithm generates a schedule having less machine consumption and less makespan as compare to other traditional technique.

2.3 Simulators:

It is difficult as well as costly to deploy the application or (in this case) algorithm, on the real environment to test the capability of the application or performance of an algorithm there is a need of similar environment which can be used to deploy and test the application or algorithm and produces the similar result as if it is run in real environment, so that environment can be created by using simulators. There are various simulator are present for simulating grid and cloud environment like CloudSim, Workflowsim, CloudAnalyst, and GridSim.

- **GridSim**: Gridsim is a simulator which is used to simulate the grid environment. R.Buyya proposed the concept of gridsim because it was not efficient to evaluate the performance during testing the application or algorithm by repeatedly deploying it, so there is need of simulator for testing the algorithms in grid environment. Also because there are various types of user having different resource and different QoS parameter so in real time it is very difficult or not possible. Now this simulator will generate the similar environment like homogenous or heterogeneous resources, task creation and mapping of tasks to the machines etc.

- **CloudSim:** Cloudsim can be seems as the extended version of gridsim. It allows to create the cloud environment, need of cloudsim is similar to the gridsim. Here in cloudsim it simulates the virtual machine, datacenters, cloudlets etc. R. Buyya is also the concept proposer behind the cloudsim. CloudSim only simulates the environment by giving by making datacenters, and just try to simulate the environment. There is only single workflow to schedule; also no dependency is defined between tasks.

- **Workflowsim:** Workflowsim is the extension of cloudsim which is designed to simulate the workflow design system for the cloud environment. It is defined there in workflowsim like dependency of tasks on resources, schedule time, cost for machines etc. then makespan and all the characteristics about the virtual machine and tasks are already define in workflowsim and in this it is easy to evaluate the performance in workflowsim.

- **Java Environment:** Many algorithms are implemented using the java environment. In java it is easy to compare the new algorithms with other algorithms. Also the constraint free environment makes it easy to implement and easy to directly upload on any java based simulator.

CHAPTER 3

PROBLEM DESCRIPTION AND MOTIVATION

The last chapter discusses about the some of the existing scheduling algorithm for workflow that are present in Cloud environment. This chapter, is about the discussion on the analysis of gaps of existing workflow scheduling algorithms emphasis on the duplication based algorithms and the problem description.

3.1 Problem Description

Scientific applications like the project at Grid Physics Network [23] created a large amount of data i.e. in gega-bytes or peta-bytes. Workflows are used to represent these experiments and the links between the tasks represent the dataflow and compute dependencies. Workflow is said to be data intensive, when requirements of the data (e.g. no of data storage used, size of data files etc.) are high. Now these scientific experiment or say workflows are generally executed using distributed resources, as there exist replicas of data, so it can retrieve from the different data sources. Data should be available for each task before it gets started. Also the produced output data by a task may also be in the scale of similar sizes to the input data i.e. in gega-bytes and peta-bytes. So transferring of the output of one task from one processor can take much more time as compare to the computation time. A Directed acyclic graph (DAG) can be used to represent workflow, where cycles and conditional dependencies are absent.

3.1.1 A Task scheduling model

In cloud computing scheduling refers to the assignment of the cloudlets to a set of available virtual machines, and then creating execution time order or a schedule for them while maintaining the precedence among the tasks and optimizing the overall execution time and cost of the workflow**.** A workflow task model is represented by a (C, R, E, W), where C = {n_i : i=1, 2,….., v}say vertices of the DAG i.e. represent the tasks or cloudlets,and R represent a relation(edge) between the cloudlets such that if n_i R n_j, means before n_j starts the execution the task n_i must finished; where E (vector of length *v*) is the execution cost of each task i.e. if E[n_i]>0 it implies task n_i,1< i < v needs a number of instruction to execute and W is a matrix of v x v gives the cost of communication between task n_i and n_j for all 1< i ; j < v. This task

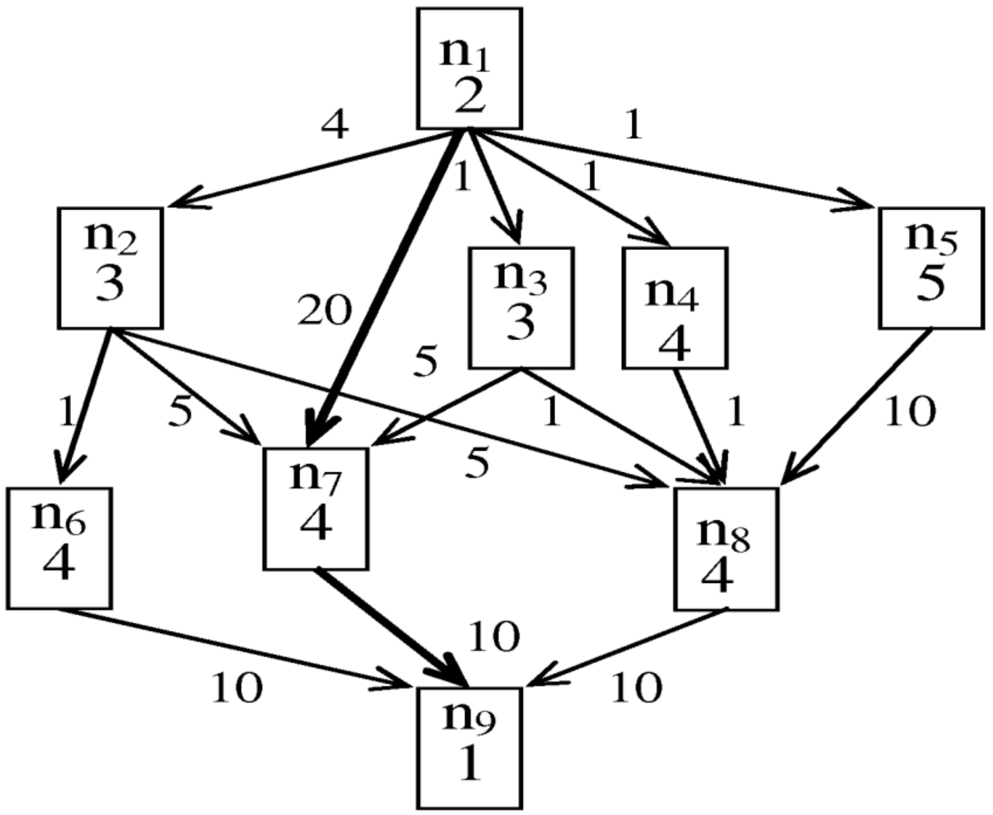


Figure 3.1: A Simple DAG

scheduling model can easily represented in DAG figure 3.1 (example DAG taken from [8]), tasks as vertices, dataflow path as edges, and the computation and communication costs as corresponding weights on them. In this report, the term task, node, and cloudlet have been used interchangeably. Also the term virtual machines and processors can be used interchangeably.

3.1.2 Workflow target machine model

The workflow target machine model is represented by (V, L, H); where V= {v_i; where i=1, 2,…, v} is a set of "v" homogeneous virtual machines; L is the v x v matrix describing interconnection between machine i.e. a network topology of virtual machine with a 1 / 0 indicating if a direct link is present/absent between v_i and v_j; and H is the v x v matrix giving distance between virtual machine v_i and v_j in number of hopes. The objective of scheduling algorithm is to create a optimize schedule by allocating each cloudlet (task) in the workflow to the virtual machine (processor) in the machine network, in such a way that it should minimize the schedule length (SL) or makespan. The finish time of task n_i on virtual machine v_j is termed as F_{ij}.

$SL=max\{ F_{ij} \}$ for all 1<i<v and for all 1<j<v;

And Machine Cost (MC) is

$MC=\sum_{j=1}^{m} U_j * C_j$ for all jth machines.

Here C is the cost of using jth resource and U_j is the usage of jth resource.

3.2 Assumptions

Before defining the solutions, there are certain assumptions are made before doing the work:

- This work does not consider conditional and cycles in the workflow structure.
- This work assumes that data required by workflow applications for executing the tasks are replicated at all virtual machines distributed geographically around the world.
- Communication links between machines are contention free to accommodate the deterministic nature of scheduling.
- If two tasks are schedule on the same machine, so the communication overhead between them is taken as zero.

3.3 Gap analysis

For an arbitrary DAGs, the scheduling problem has been shown to be a NP-Complete problem. For arbitrary DAGs there are various duplication algorithms exists which perform without any guaranteed performance ratio [8], [9], their optimality is restricted for some workflow types has been proved in literature. The duplication-based heuristics based algorithms have been shown to give better performance than nonduplication-based heuristics [10]. The duplication algorithm proposed by Bansal et. al. [12] has emphasis on that by duplicating the tasks on various processors, the overall execution time or SL (schedule length) or makespan time can be optimized. The algorithm optimized the makespan of the workflow but the resource utilization is increased.

3.4 Problem Statement

As the resource utilization is increased by selective duplication algorithm [12] as well as the economic cost is increases. So there is need of improvement in this algorithm which can optimize the schedule length as well as the resource utilization.

CHAPTER 4

PROPOSED METHODOLOGY

Here a new scheduling algorithm is proposed or simply, a new solution is proposed using [23] [24] to the problem described in the last chapter. And later in this chapter description of the proposed algorithm is also given.

4.1 Algorithm Description

The "Schedule optimization with duplication-based scheduling algorithm for multiple workflows (SODA-M)" is divided into 2 parts:

- Firstly, we take one schedule having dependent task then find the optimum time and resource utilization for that workflow.
- Secondly, we take the schedule having independent task then find the schedules for it for more resource utilization.

Now for the first part of algorithm has two major phases:

4.1.1 Makespan-phase: In this phase all the cloudlets are assigned to a virtual machine; then get the optimum execution time or makespan of the workflow.

4.1.2 Duplication Reduction phase: In this phase, the optimization of resource utilization takes place without affecting the makespan of the schedule generated in previous phase.

For makespan phase, an efficient and optimal duplication-based algorithm, meta-heuristic should be applied to optimize the primary criterion (makespan). The output of this phase is a schedule having solution to the problem i.e. initial cost (say SC) which is the total cost of the schedule. Now in the second phase (i.e. Duplication reduction phase), the algorithm should apply, in order to meet the secondary criterion (i.e. Machine utilization cost).

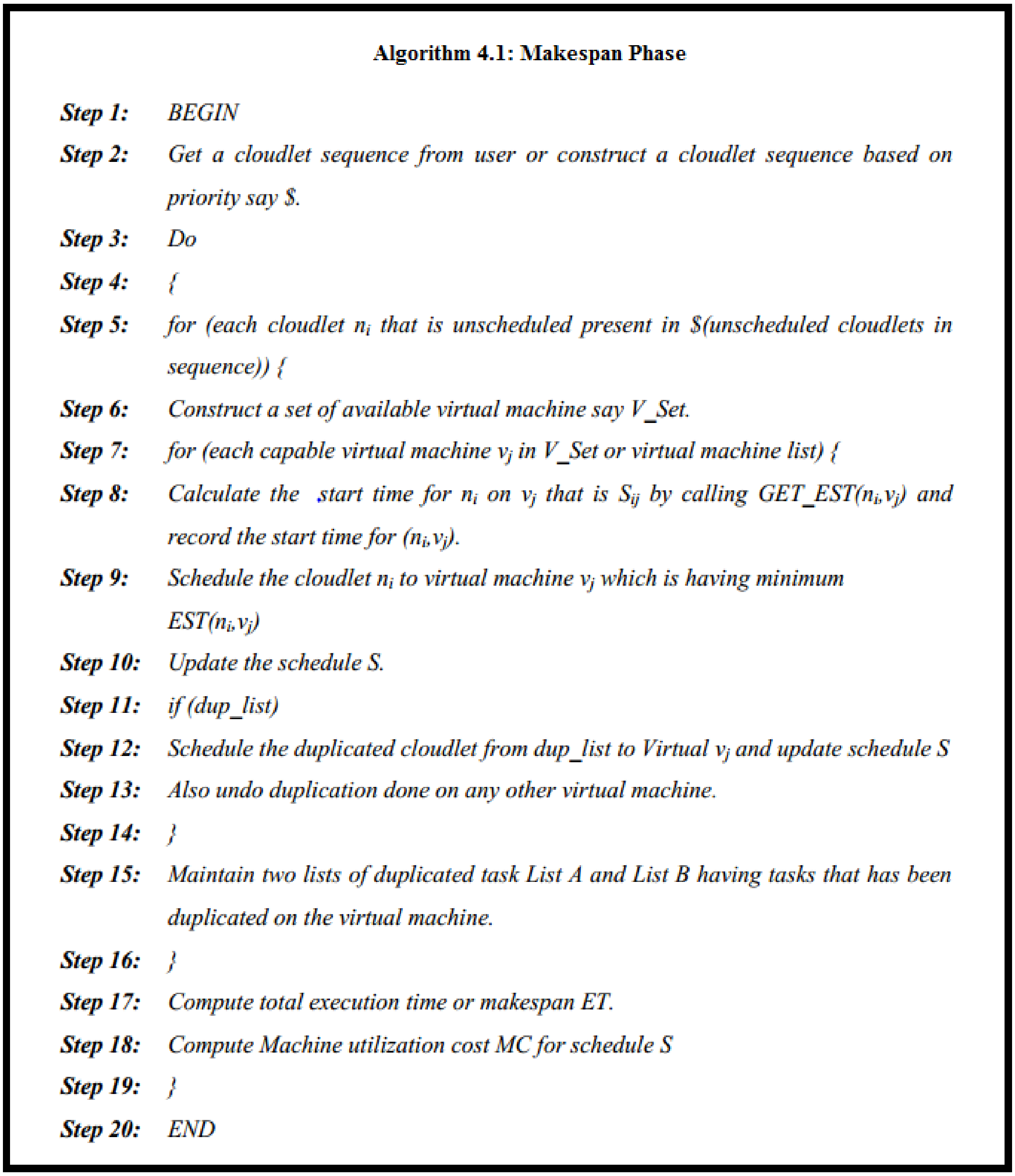
4.2 Algorithm for SODA-M

Inputs:

- A DAG D having cloudlet communication and computation costs.

- A set of resources which is available for location with cost and execution cost per unit time.

4.2.1 Pseudo code for Makespan phase



Algorithm 4.1: Makespan Phase

```
Step 1:  BEGIN
Step 2:  Get a cloudlet sequence from user or construct a cloudlet sequence based on
         priority say $\$$.
Step 3:  Do
Step 4:  {
Step 5:  for (each cloudlet $n_i$ that is unscheduled present in $\$$(unscheduled cloudlets in
         sequence)) {
Step 6:  Construct a set of available virtual machine say V_Set.
Step 7:  for (each capable virtual machine $v_j$ in V_Set or virtual machine list) {
Step 8:  Calculate the start time for $n_i$ on $v_j$ that is $S_{ij}$ by calling GET_EST($n_i$,$v_j$) and
         record the start time for ($n_i$,$v_j$).
Step 9:  Schedule the cloudlet $n_i$ to virtual machine $v_j$ which is having minimum
         EST($n_i$,$v_j$)
Step 10: Update the schedule S.
Step 11: if (dup_list)
Step 12: Schedule the duplicated cloudlet from dup_list to Virtual $v_j$ and update schedule S
Step 13: Also undo duplication done on any other virtual machine.
Step 14: }
Step 15: Maintain two lists of duplicated task List A and List B having tasks that has been
         duplicated on the virtual machine.
Step 16: }
Step 17: Compute total execution time or makespan ET.
Step 18: Compute Machine utilization cost MC for schedule S
Step 19: }
Step 20: END
```

Figure 4.1: Pseudo Code for Makespan Phase

```
Algorithm 4.2 : Get Earliest Starting time

Step 1:   Get_EST($n_i$,$v_j$)
Step 2:   BEGIN
Step 3:   ES= Earliest Starting time $S_{ij}$ of cloudlet $n_i$ on virtual machine $v_j$.
Step 4:   $M_i$= Find most important immediate parent of Cloudlet $n_i$.
Step 5:   If the $M_i$ =0 or parent doesn't exist
              a. Then return ES
Step 6:   Else
              a. Find the first suitable gap slot for $M_i$ and calculate earliest Start time of gap
                 for cloudlet $M_i$ on virtual Machine $v_i$.
Step 7:   If there is gap exist then
Step 8:   Find Earliest finish time(EFT)= EST($M_i$,$v_j$) +w($M_i$).
Step 9:   Else
Step 10:  Return ES
Step 11:  Now if EFT is Less than ES
Step 12:  Then duplicate the $M_i$ on virtual machine $v_j$ and go to step 18
Step 13:  Else
Step 14:  Return ES
Step 15:  END
```

Figure 4.2: Pseudo Code for GET_EST() method

In makespan phase of SODA-M algorithm, demonstrate a generic algorithm which improves makespan time by duplicating the cloudlets on the virtual machines, presumed that it helps in improving the performance. The pseudo code for makespan phase is given in . Now we describe the working of this code. Main steps of this phase are described as:

○ **Cloudlet Sequence Generation Step**

An ordered cloudlet sequence is generated using priority based method of critical path. In this work, a scheme used for prioritizing the tasks was proposed by Ahmad et al. [8], with considerable modifications. In this step a critical path node are calculated by finding the critical path of the DAG for example DAG shown in figure 3.1, the critical path is (n1,n7,n9) and n1, n7 and n9 are said to be the critical path(CP) nodes. The longest directed path in terms of communication and computation costs is known as critical path. Generated cloudlet sequence should satisfy the precedence constraints, i.e. the parent (predecessor) nodes must be executed before the child (successor) nodes. So during the cloudlet sequence generation, nodes are added in the sequence on basis of higher b-level and ties are broken on the basis of lower t-level. So for DAG shown in figure 3.1 the sequence of cloudlets are (n1; n3; n2; n7; n6; n5; n4; n8; n9). The first unscheduled cloudlet is taken as the candidate task n_i.

- **Virtual Machine Selection**

In this step, a set of virtual machine available for the given cloudlet are generated say V_Set. Here in this work we assume all the virtual machine are connected to each other with some hope distance. Now for all available virtual machines available for the tasks, calculate the earliest start time of the task for each machine in V_Set. Then the cloudlet is schedule on the machine having minimum EST (n_i,v_j) (Earliest start time).

- **Cloudlet Assignment and Duplication**

In this step, the cloudlet n_i is assigned to a virtual machine v_j available in V_Set then calculate EST(n_i,v_j), using duplication approach as follow: The cloudlet n_i can start execution on a virtual machine v_j if and only if data arrives from all of its immediate parents(precedence constraints). The data arrives last from parent is termed as the most important immediate parent of n_i. Data arrival time for machine network are calculated as follow:

$$EST\ (n_i,v_j) = max\ \{DAT(M_i,v_j),\ min\ \{V^R_{j,}\ G^S_k\ \}\}$$

Where,

V^R_j is the ready time of the virtual machine j.

G^S_k is the start time of available gap.

$$DAT\ (n_i,v_j) = max\ \{F_{t,k} + c_{ti}\ x\ h_{kj}\}\ for\ all\ n_t \in parent(i)$$

Where,

$F_{t,k}$ is the finish time of cloudlet t scheduled on machine k

c_{ti} is the Data cost between cloudlets t and i.

h_{kj} is the hope distance between machines k and j.

The DAG figure 3.1 have nine cloudlets, which is going to schedule on the four virtual machine connected as a fully connected network. Table 4.1 shows the stepwise trace of this phase. And figure 4.3 shows the schedule S after this makespan phase.

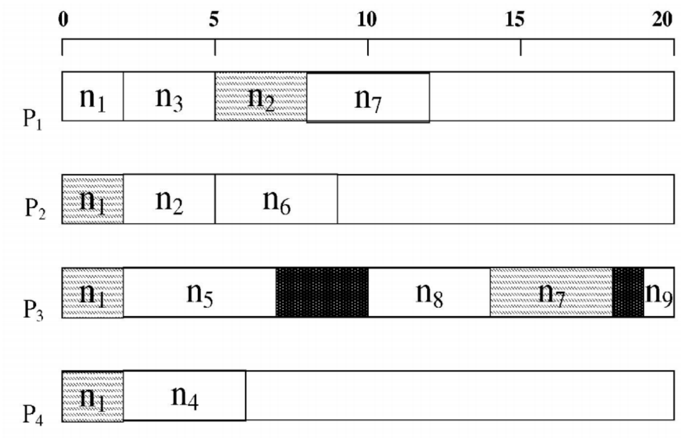


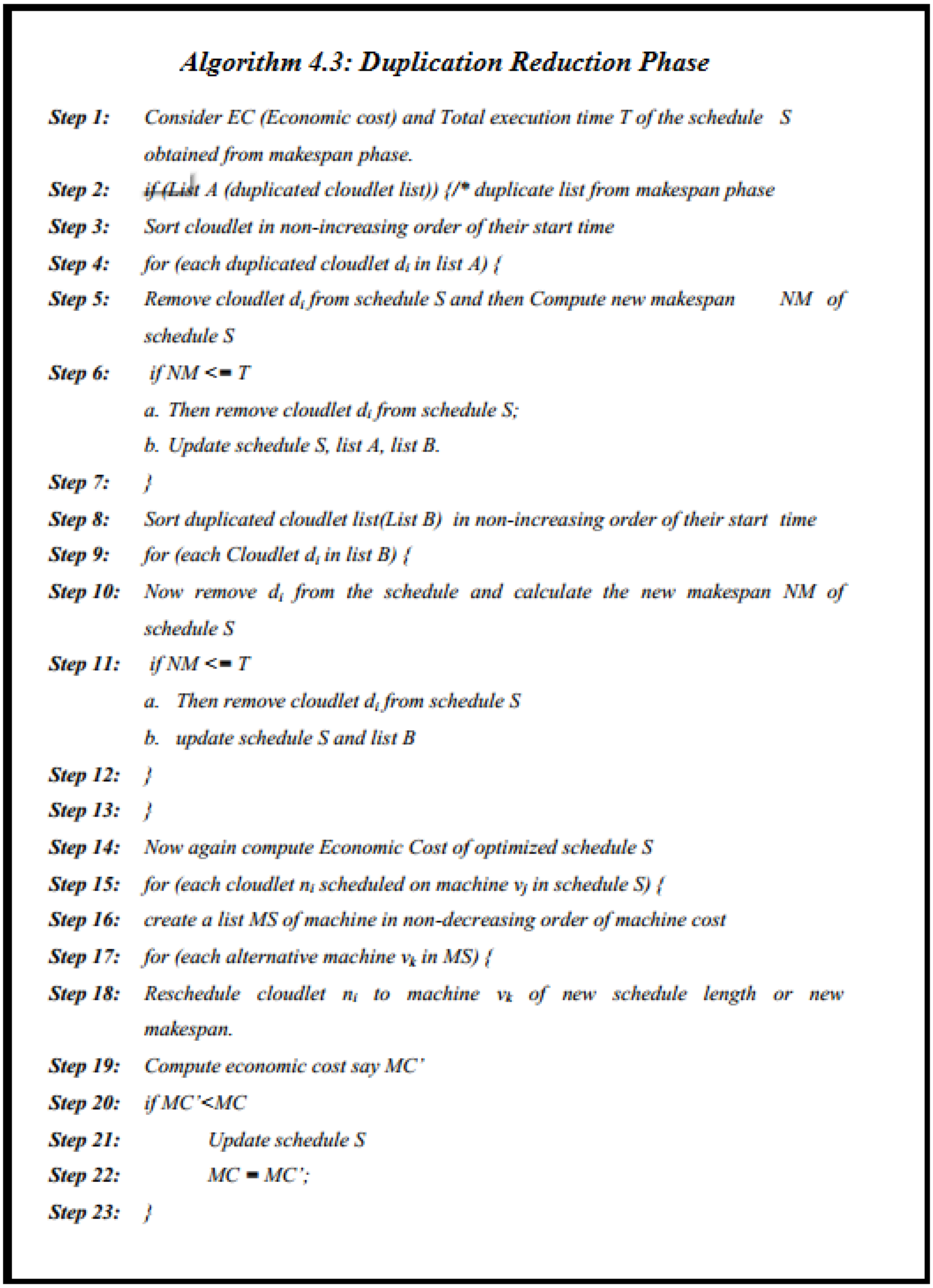
Figure 4.3: Schedule generated by SD algorithm

Nodes	*s-level*	*b-level*	*t-level*	Task Sequence	step	Start times on candidate processors P_1	P_2	P_3	P_4	Node/s duplicated
n_1	12	37	0	n_1	1	0 *	0	0	0	none
n_2	8	23	6	n_3	2	2*	2	2	2	none
n_3	8	23	3	n_2	3	5	2*	2	2	n_1
n_4	9	20	3	n_7	4	8*	8	8	8	n_2
n_5	10	30	3	n_6	5	12	5*	6	6	none
n_6	5	15	10	n_5	6	12	9	2*	2	n_1
n_7	5	15	22	n_4	7	12	9	7	2*	n_1
n_8	5	15	18	n_8	8	17	14	10*	11	none
n_9	1	1	36	n_9	9	21	21	19*	21	n_7

Table 4.1: Stepwise trace of the SD algorithm

4.2.2 Duplication reduction phase

Input: Makespan T and Machine (Economic) cost MC from schedule S



Algorithm 4.3: Duplication Reduction Phase

```
Step 1:   Consider EC (Economic cost) and Total execution time T of the schedule S
          obtained from makespan phase.
Step 2:   if (List A (duplicated cloudlet list)) {/* duplicate list from makespan phase
Step 3:   Sort cloudlet in non-increasing order of their start time
Step 4:   for (each duplicated cloudlet $d_i$ in list A) {
Step 5:   Remove cloudlet $d_i$ from schedule S and then Compute new makespan NM of
          schedule S
Step 6:    if NM <= T
          a. Then remove cloudlet $d_i$ from schedule S;
          b. Update schedule S, list A, list B.
Step 7:   }
Step 8:   Sort duplicated cloudlet list(List B) in non-increasing order of their start time
Step 9:   for (each Cloudlet $d_i$ in list B) {
Step 10:  Now remove $d_i$ from the schedule and calculate the new makespan NM of
          schedule S
Step 11:   if NM <= T
          a.  Then remove cloudlet $d_i$ from schedule S
          b.  update schedule S and list B
Step 12:  }
Step 13:  }
Step 14:  Now again compute Economic Cost of optimized schedule S
Step 15:  for (each cloudlet $n_i$ scheduled on machine $v_j$ in schedule S) {
Step 16:  create a list MS of machine in non-decreasing order of machine cost
Step 17:  for (each alternative machine $v_k$ in MS) {
Step 18:  Reschedule cloudlet $n_i$ to machine $v_k$ of new schedule length or new
          makespan.
Step 19:  Compute economic cost say MC'
Step 20:  if MC'<MC
Step 21:          Update schedule S
Step 22:          MC = MC';
Step 23:  }
```

Figure 4.4: Pseudo Code for Duplication Reduction Phase

After getting the schedule from the makespan phase, this phase try to minimize the duplication by

a. Detecting and removing the useless duplicated cloudlets.

b. Detecting and removing the unproductive cloudlets.
c. Shifting of schedule to the cheaper resources without affecting makespan.

- **Duplication reduction phase:** Firstly removing the duplicated cloudlet then check it affect the makespan or execution time or not. If the removal of duplicated cloudlet does not affect the makespan than it is removed from the schedule permanently. Similarly we remove cloudlets which are duplicated on another processor and check whether it is affecting the makespan or not, if not then the cloud let is said to be unproductive or not usable. So remove the unproductive cloud make the schedule more optimum towards the optimum resource utilization criterion.
- **Economic Cost Reduction phase:** Now as the utilization cost for every machine are not same. So we try to schedule more cloudlet on the machine having less cost.

Figure 4.5 shows the schedule after applying duplication reduction phase

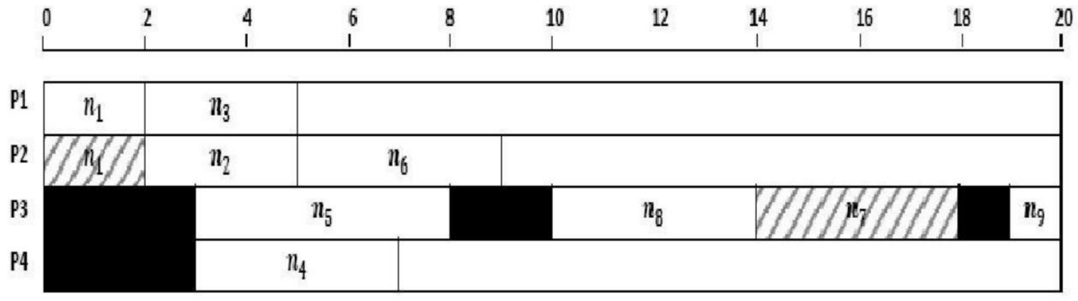


Figure 4.5: Schedule after duplication removal phase

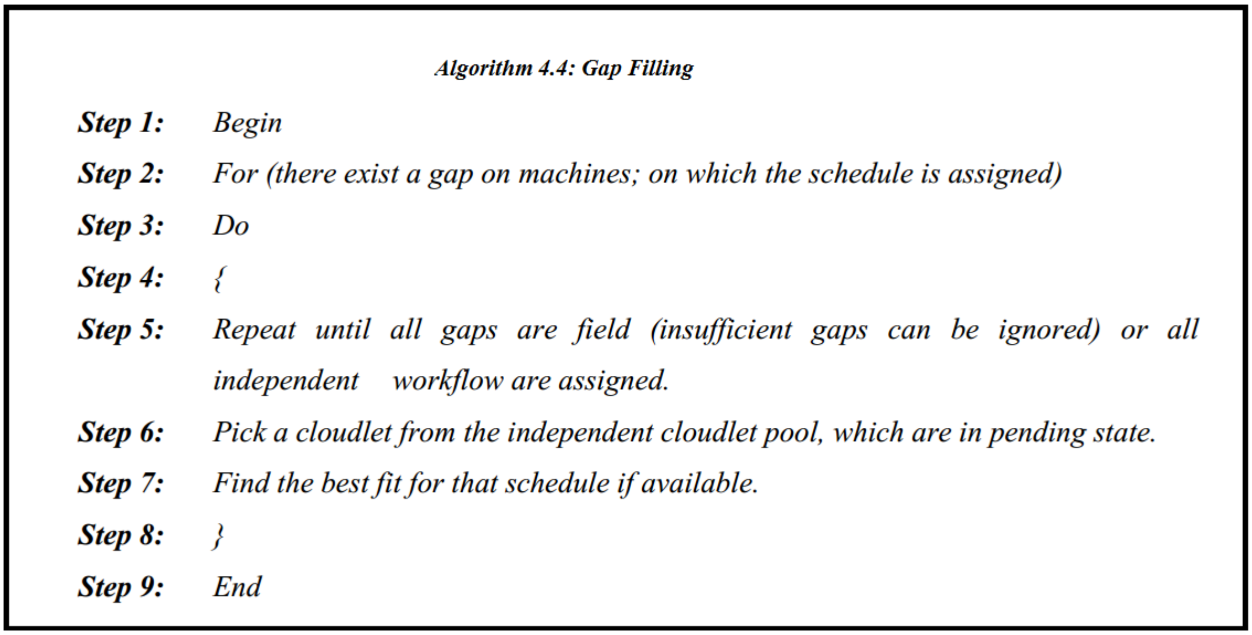
Now for the second part of the algorithm:

There is a need of scheduling independent cloudlets of workflow for maximize the resource utilization. In Cloud there are various QOS users which want machine for a period of time. But sometime what happen when a schedule is created then there are certain times when the machine will remain idle. And idle means no work is done by machine and also the power consumption and other losses are there. So to prevent from those losses we have to utilize the wasted resource by assigning the another workflow in that gap or idle time.

4.2.3 Pseudo Code for assigning independent cloudlets to the gaps of Independent workflows

Input:

Schedule S generated by the part and the ready time of each processor, independent cloudlet pool with their computation cost w[c].



Algorithm 4.4: Gap Filling

```
Step 1:   Begin
Step 2:   For (there exist a gap on machines; on which the schedule is assigned)
Step 3:   Do
Step 4:   {
Step 5:       Repeat until all gaps are field (insufficient gaps can be ignored) or all
              independent    workflow are assigned.
Step 6:       Pick a cloudlet from the independent cloudlet pool, which are in pending state.
Step 7:       Find the best fit for that schedule if available.
Step 8:   }
Step 9:   End
```

Figure 4.6: Pseudo Code for assigning gaps of Independent workflows

The independent workflow i.e. in which the tasks are not depend on other task having independent cloudlets, so these cloudlets will be assigned to the best fitted gap available for the task. These tasks will be served as the first in first out manner. So figure 4.7 shows that the holes are filled by the independent tasks (n_i).

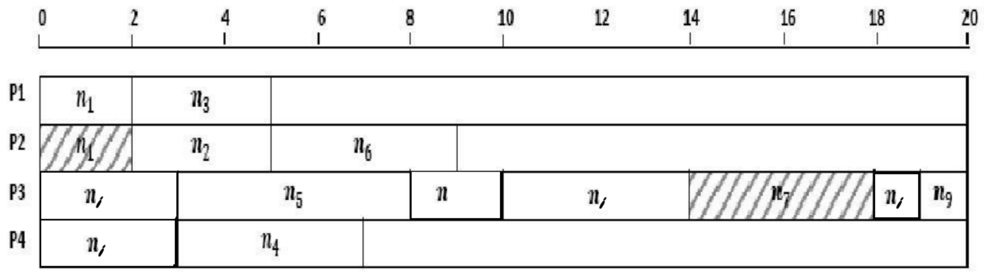


Figure 4.7: schedule generated after applying SODA-M algorithm

CHAPTER 5

IMPLEMENTATION NOTES AND EXPERIMENTAL RESULTS

5.1 Eclipse

It is an IDE(Integrated development environment) [25] which support many languages for the development of the software like C, C++, PHP, HTML, Android, Java etc. As the cloudsim and workflowsim both are developed in java; so it easy to embed them with the java environment using eclipse IDE.

5.2 Implementation Details

The proposed algorithm for scheduling is basically implemented using java. Basically, in this work three workflow scheduling is implemented, every algorithm is improvement of previous one.

So the outputs of these algorithms are shown in the various screenshots:

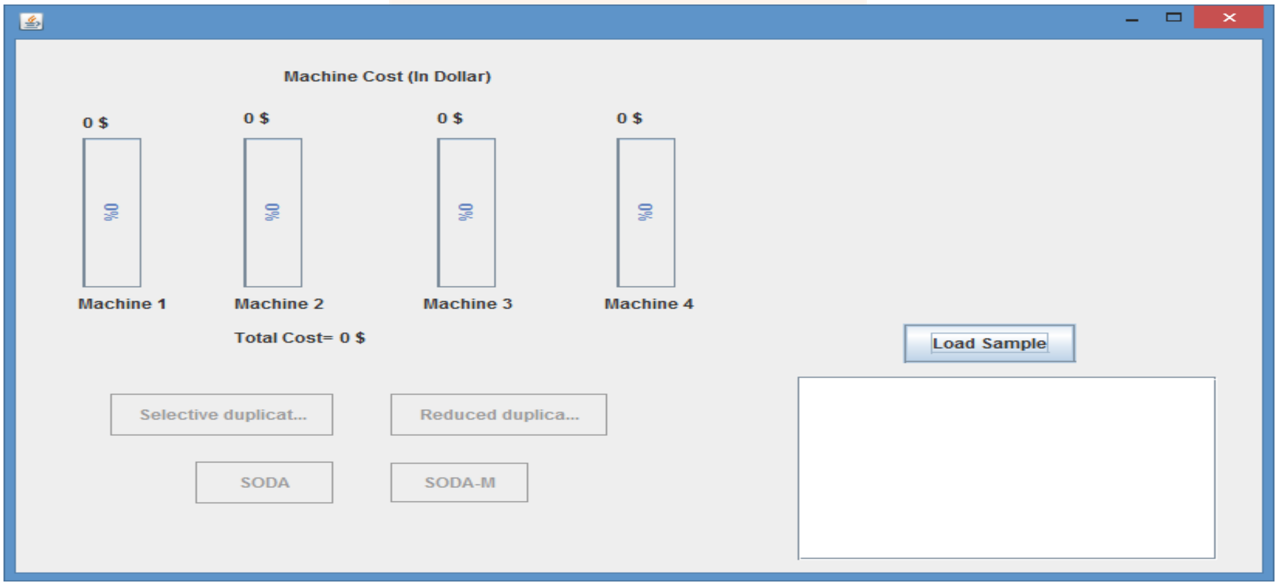


Figure 5.1: Main page to load the sample

This output is generated by using swings, which provide to make an interactive GUI design in java. The first screen shot (figure 5.1) shows that by clicking on load sample the data flow diagram and the task sequence will be assigned. And in figure 5.2 the sample data flow is loaded and ready to execute the algorithms.

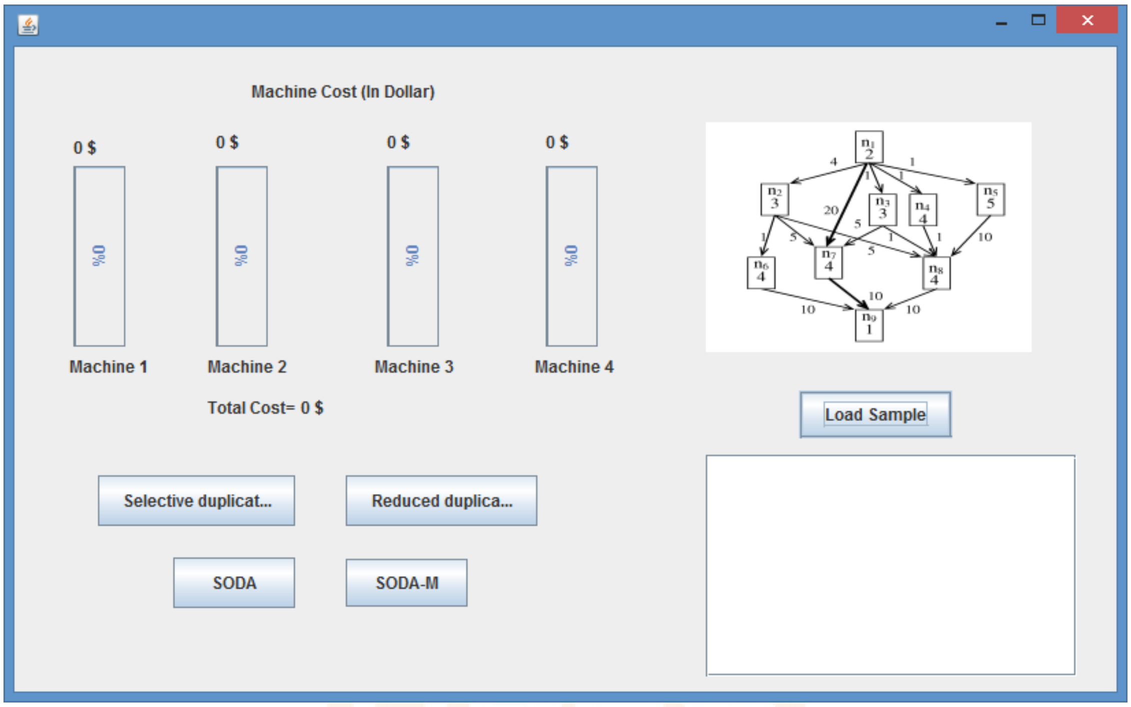
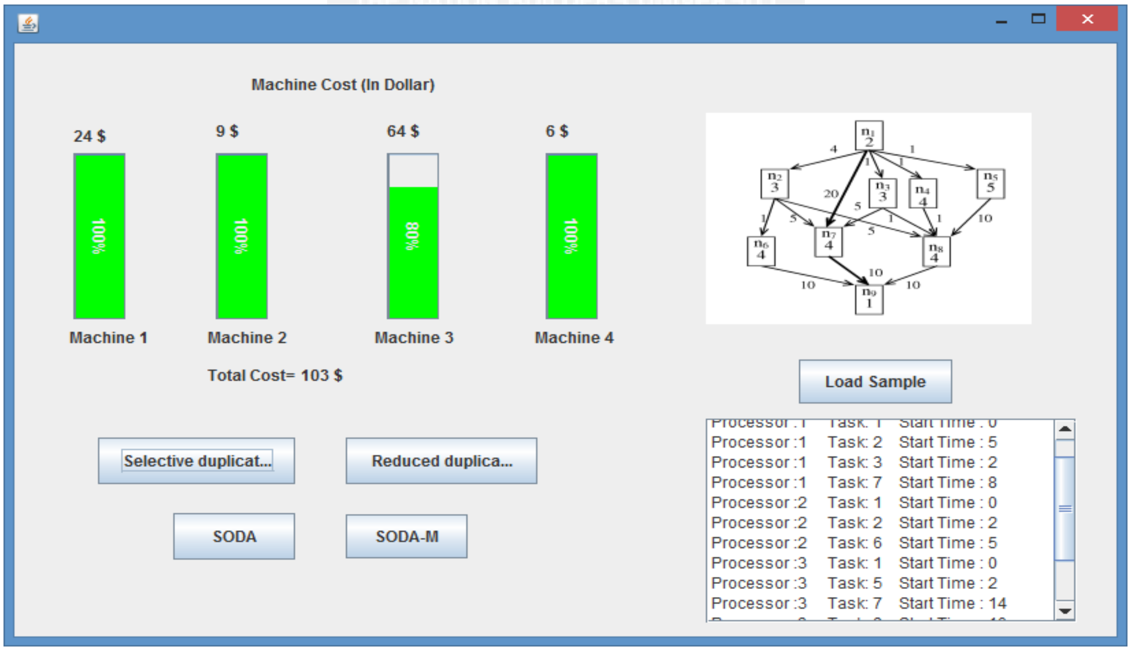


Figure 5.2: Sample is loaded

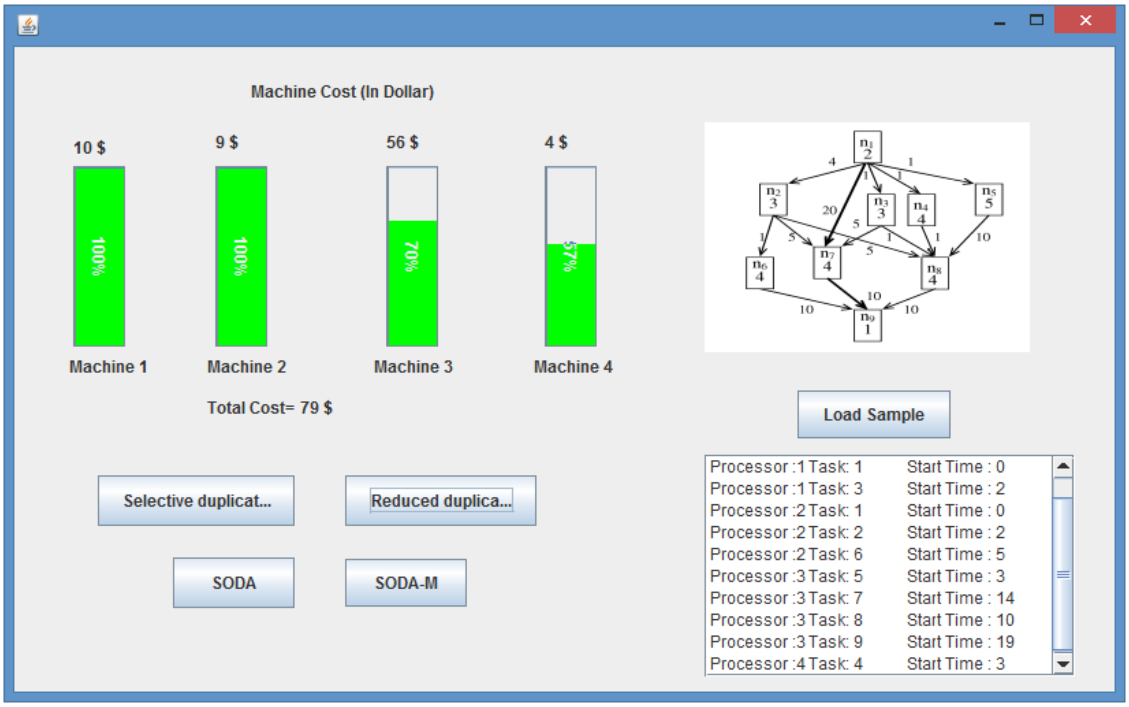


```
Processor :1   Task: 1   Start Time : 0
Processor :1   Task: 2   Start Time : 5
Processor :1   Task: 3   Start Time : 2
Processor :1   Task: 7   Start Time : 8
Processor :2   Task: 1   Start Time : 0
Processor :2   Task: 2   Start Time : 2
Processor :2   Task: 6   Start Time : 5
Processor :3   Task: 1   Start Time : 0
Processor :3   Task: 5   Start Time : 2
Processor :3   Task: 7   Start Time : 14
```

Figure 5.3: Scheduled tasks and utilization of 4 machines after scheduling by SD Algorithm

When the Selective duplication button is clicked it generates a schedule for the workflow (figure 3.1) and the memory or machine utilization shown in figure 5.3.

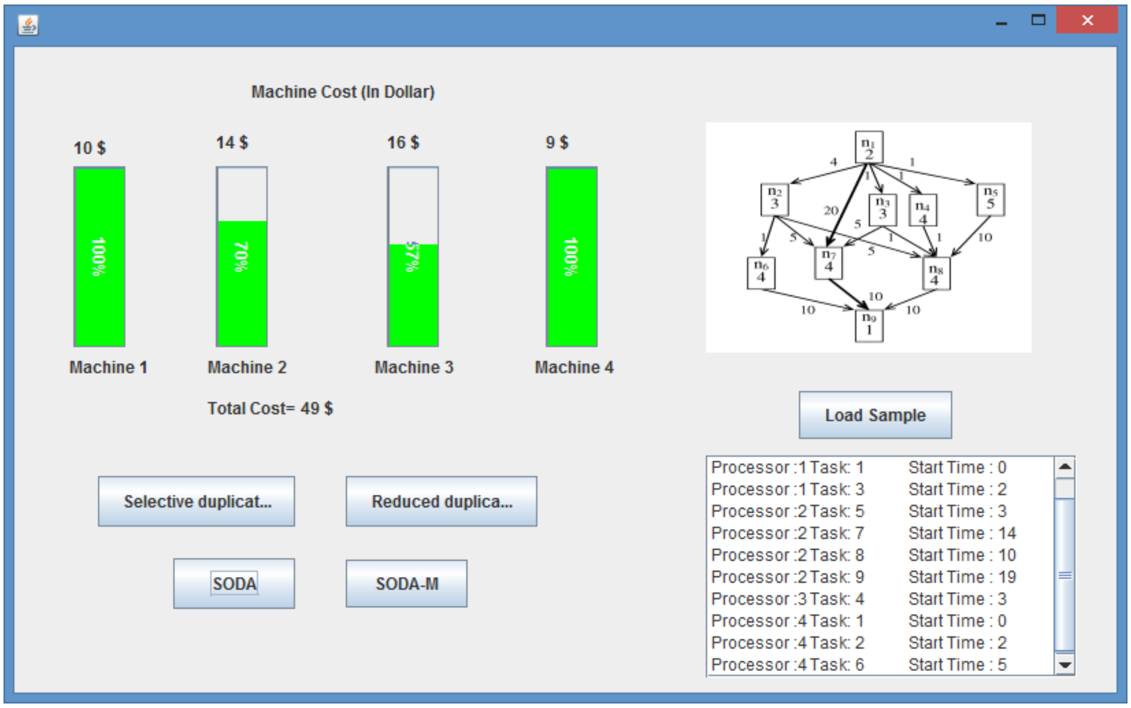
After this next phase is reduced duplication phase, so when clicked on the reduced duplication, the resulting schedule and utilization are shown in figure 5.4.



```
Processor :1 Task: 1    Start Time : 0
Processor :1 Task: 3    Start Time : 2
Processor :2 Task: 1    Start Time : 0
Processor :2 Task: 2    Start Time : 2
Processor :2 Task: 6    Start Time : 5
Processor :3 Task: 5    Start Time : 3
Processor :3 Task: 7    Start Time : 14
Processor :3 Task: 8    Start Time : 10
Processor :3 Task: 9    Start Time : 19
Processor :4 Task: 4    Start Time : 3
```

Figure 5.4: Scheduled tasks and utilization after Reduced Duplication Algorithm

Similarly, when the SODA (schedule optimization with duplication-based bi-criteria scheduling algorithm) button is clicked it generates a schedule for the workflow (figure 3.1) and the memory or machine utilization by this algorithm is shown in figure 5.5.

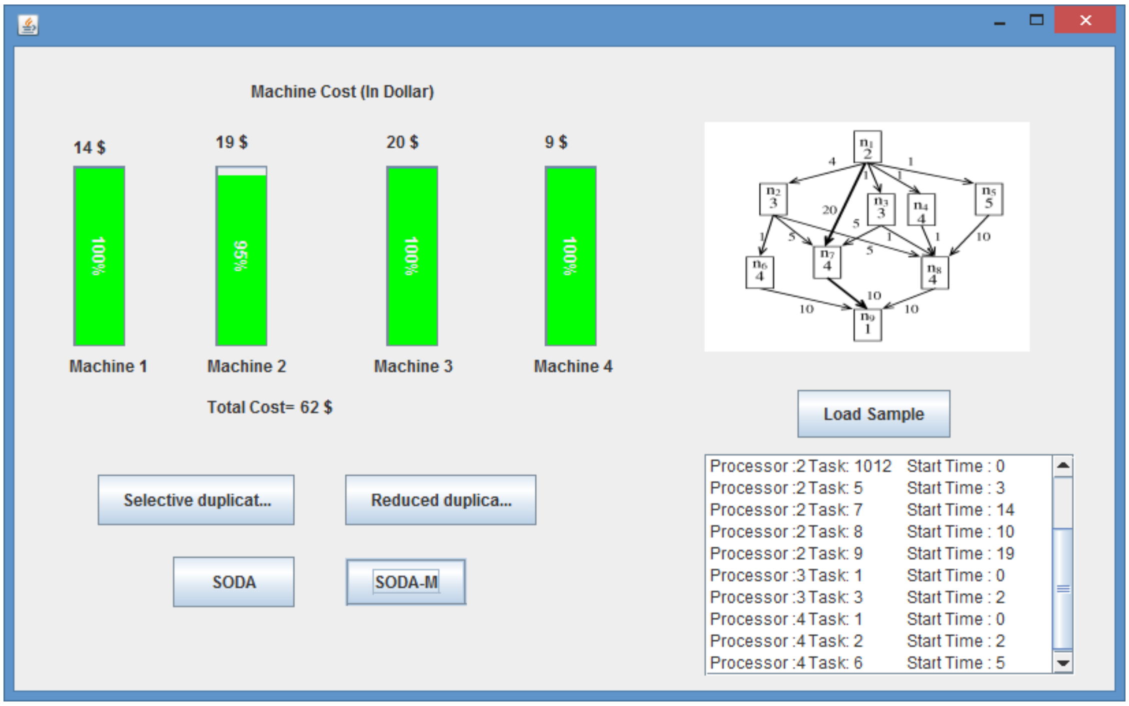


```
Processor :1 Task: 1    Start Time : 0
Processor :1 Task: 3    Start Time : 2
Processor :2 Task: 5    Start Time : 3
Processor :2 Task: 7    Start Time : 14
Processor :2 Task: 8    Start Time : 10
Processor :2 Task: 9    Start Time : 19
Processor :3 Task: 4    Start Time : 3
Processor :4 Task: 1    Start Time : 0
Processor :4 Task: 2    Start Time : 2
Processor :4 Task: 6    Start Time : 5
```

Figure 5.5: Scheduled Tasks and Machine utilization by SODA algorithm

Again, when the SODA-M (schedule optimization with duplication-based algorithm for multiple workflow) button is clicked it generates a schedule for the workflow (figure 3.1) and the memory or machine utilization by this algorithm is shown in figure 5.6.

Here, as there is more duplication in the first algorithm (i.e. Selective duplication), so the memory utilization by this algorithm is high. So there is need of removing the duplicated tasks that are unnecessary and the tasks that are unproductive. This results in less machine utilization. After that, the time at which the machine remains idle is also the waste of power. So again to increase the utilization of the resource, the tasks of which are not depend on the other task i.e. independent workflow can be assigned to improve the utilization. So the gap available i.e. the time which are not assigned to any task will assigned to those tasks to make the machine utilization more.



```
Processor :2 Task: 1012   Start Time : 0
Processor :2 Task: 5      Start Time : 3
Processor :2 Task: 7      Start Time : 14
Processor :2 Task: 8      Start Time : 10
Processor :2 Task: 9      Start Time : 19
Processor :3 Task: 1      Start Time : 0
Processor :3 Task: 3      Start Time : 2
Processor :4 Task: 1      Start Time : 0
Processor :4 Task: 2      Start Time : 2
Processor :4 Task: 6      Start Time : 5
```

Figure 5.6: Schedule Tasks and Machine utilization by SODA-M

5.3 Results:

The graph (figure 5.7) shows the comparison of machine costs between various algorithms. In this, there are various graph (1) denotes machine 1 and the blocks denotes the cost of using that machine by that algorithm. And the total cost comparison among different algorithms is shown in figure 5.8.

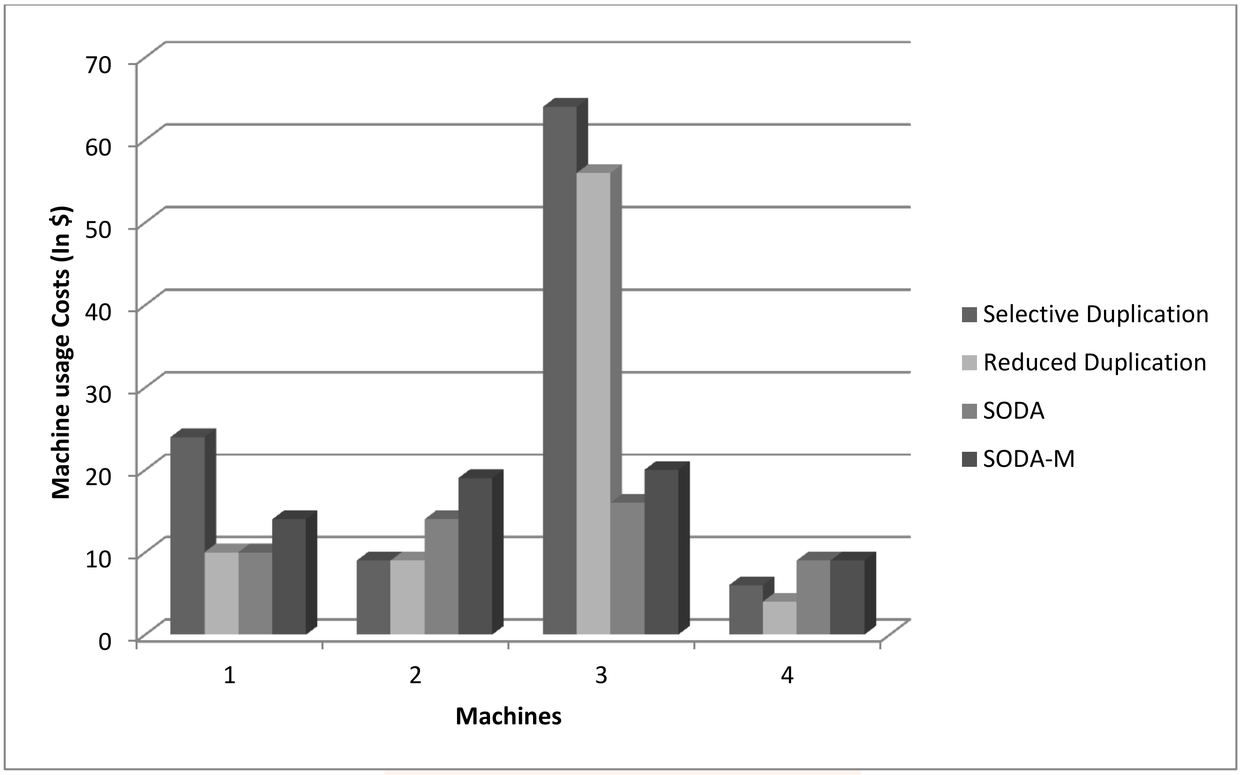


Figure 5.7: Graph showing Costs for different Algorithms

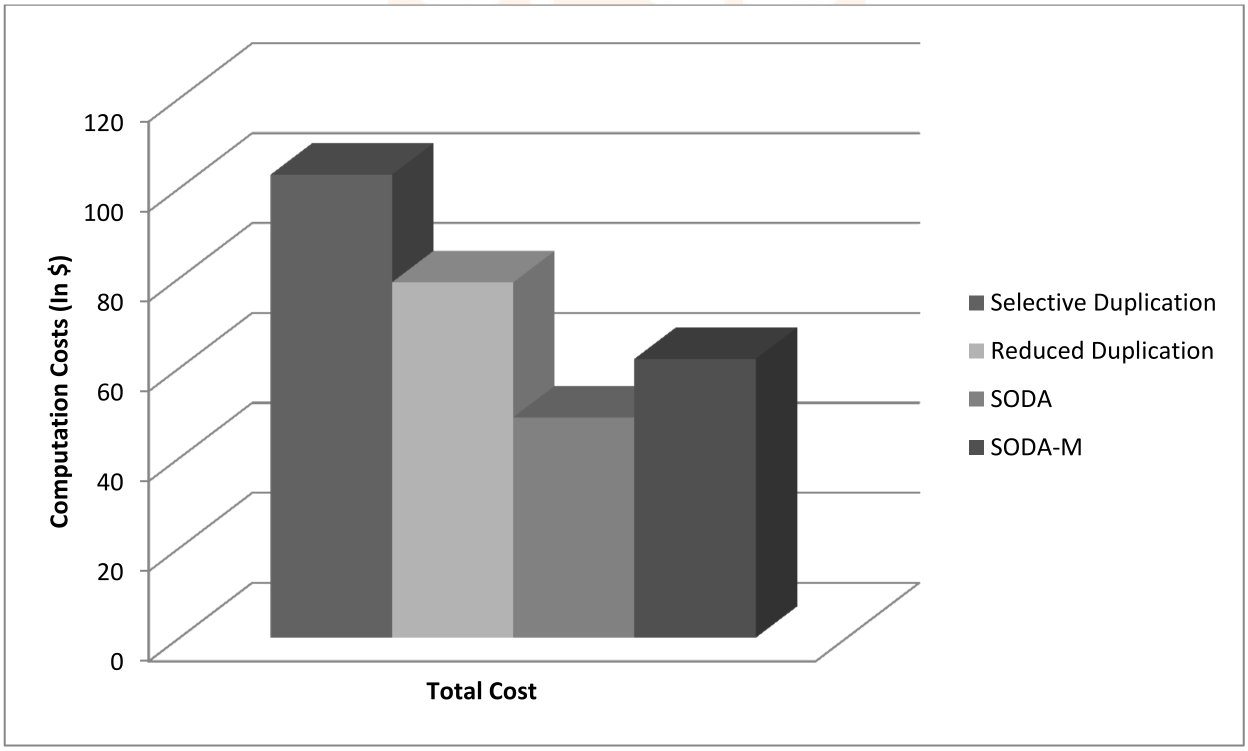


Figure 5.8: Graph showing Total Cost for different algorithms

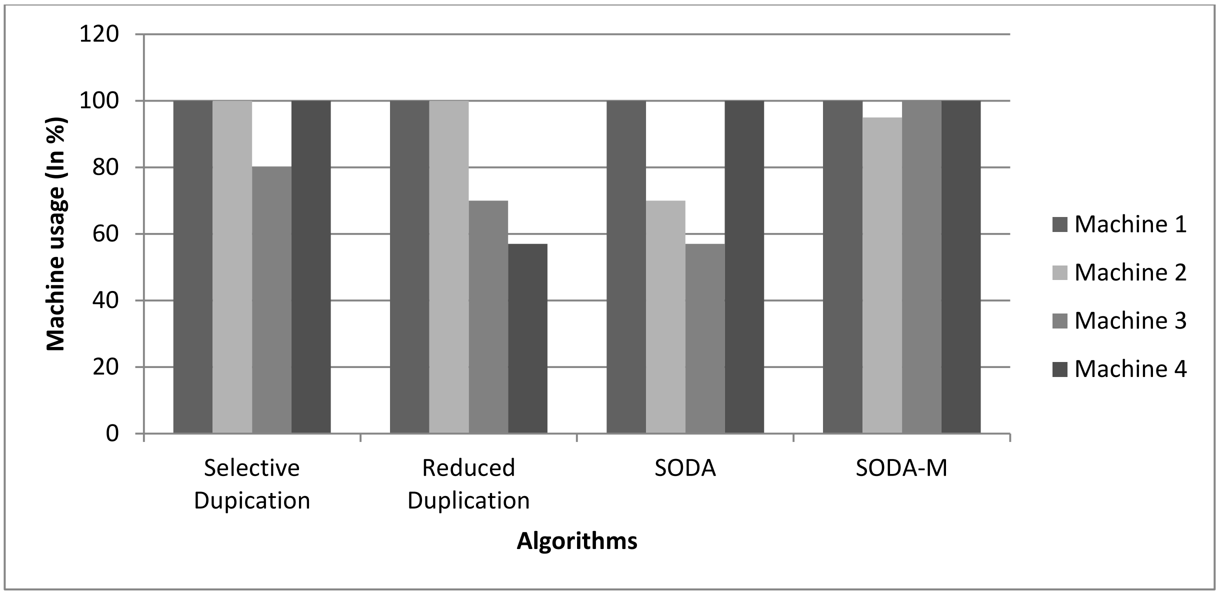


Figure 5.9: Machine usage by different algorithms

Here figure 5.9 shows the machine usage of different algorithms. As shown that the algorithm Selective duplication usage is very high as compare to the reduced duplication and SODA. So it is clear that the algorithm SODA and reduced duplication produces the schedule which will take less cost and usage as compare to the selective duplication. But the optimal utilization of resources is also necessary, so the SODA-M gives the better utilization of resources as compare to the SODA and reduced duplication.

CHAPTER 6

CONCLUSION AND FUTURE WORK

In this paper, a novel workflow scheduling approach has been presented and analysed. The efficient scheduling algorithm called ‘schedule optimisation with duplication-based scheduling algorithm for multiple workflow’ (SODA-M) which optimises the makespan and economic cost of the schedule, minimises the requirements of processors and proper utilization of resources. The algorithms have been implemented to schedule DAGs onto different clouds of homogenous machine of various sizes. The schedule generated by SODA-M algorithm is better than other related algorithms in respect of Makespan, utilization and economic cost. This algorithm decouples processor economisation from schedule length minimization and also by filling the gaps using best fit algorithm. As a future work, we are working to extend our algorithm by applying different QoS constraints posed in heterogeneous cloud environments. Also, we intended to implement and test the performance of our algorithm in real world cloud environment with different sizes of workflows.

REFERENCES

[1] Foster, Ian, and Carl Kesselman, Eds. *The Grid 2: Blueprint for a new computing infrastructure*. Elsevier, 2003.

[2] Mell, Peter, and Tim Grance. "The NIST definition of cloud computing."*National Institute of Standards and Technology* pp.50. 2009.

[3] Brian, O., T. Brunschwiler, and H. Dill. "White Paper Cloud Computing." *Swiss Academy of Engineering*, 2012.

[4] Zhang, Qi, Lu Cheng, and Raouf Boutaba. "Cloud computing: state-of-the-art and research challenges." *Journal of internet services and applications*, pp. 7-18. 2010.

[5] Wyld, David C. *Moving to the cloud: An introduction to cloud computing in government*. IBM Center for the Business of Government, 2009.

[6] Dillon, Tharam, Chen Wu, and Elizabeth Chang. "Cloud computing: issues and challenges." In *Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on*, pp. 27-33. IEEE, 2010.

[7] Senkul, Pinar, and Ismail H. Toroslu. "An architecture for workflow scheduling under resource allocation constraints." *Information Systems*, pp. 399-422. 2005.

[8] Ahmad, Ishfaq, and Yu-Kwong Kwok. "On exploiting task duplication in parallel program scheduling." *Parallel and Distributed Systems, IEEE Transactions*, pp. 872-892. 1998.

[9] Park, Gyung-Leen, Behrooz Shirazi, and Jeff Marquis. "DFRN: A new approach for duplication based scheduling for distributed memory multiprocessor systems." In *Parallel Processing Symposium, 1997. Proceedings., 11th International*, pp. 157-166. IEEE, 1997.

[10] Munier, Alix, and Claire Hanen. "Using duplication for scheduling unitary tasks on m processors with unit communication delays." *Theoretical Computer Science*, pp. 119-127. 1997.

[11] Colin, Jean-Yves, and Philippe Chrétienne. "CPM scheduling with small communication delays and task duplication." *Operations Research*, pp. 680-684. 1991.

[12] Bansal, Savina, Padam Kumar, and Kuldip Singh. "Duplication-based scheduling algorithm for interconnection-constrained distributed memory machines." In *High Performance Computing—HiPC 2002*, pp. 52-62. Springer Berlin Heidelberg, 2002.

[13] Fakhfakh, Fairouz, Hatem Hadj Kacem, and Ahmed Hadj Kacem. "Workflow Scheduling in Cloud Computing: A Survey." In *Enterprise Distributed Object Computing Conference Workshops and Demonstrations (EDOCW), 2014 IEEE 18th International*, pp. 372-378. IEEE, 2014.

[14] Couvares, Peter, Tevfik Kosar, Alain Roy, Jeff Weber, and Kent Wenger. "Workflow management in condor." In *Workflows for e-Science*, pp. 357-375. Springer London, 2007.

[15] Yu, Jia, and Rajkumar Buyya. "A taxonomy of scientific workflow systems for grid computing." *ACM Sigmod Record*, pp. 44-49. 2005.

[16] Annette J, Ruby, Aisha Banu W, and Shriram Shriram. "A Taxonomy and Survey of Scheduling Algorithms in Cloud: Based on task dependency."*International Journal of Computer Applications*, pp. 20-26. 2013.

[17] Ma, Tianchi, and Rajkumar Buyya. "Critical-path and priority based algorithms for scheduling workflows with parameter sweep tasks on global grids." In*Computer Architecture and High Performance Computing, 2005. SBAC-PAD 2005. 17th International Symposium on*, pp. 251-258. IEEE, 2005.

[18] Wieczorek, Marek, Stefan Podlipnig, Radu Prodan, and Thomas Fahringer. "Bi-criteria scheduling of scientific workflows for the grid." In *Cluster Computing and the Grid, 2008. CCGRID'08. 8th IEEE International Symposium on*, pp. 9-16. IEEE, 2008.

[19] Braun, Tracy D., Howard Jay Siegel, Noah Beck, Ladislau L. Bölöni, Muthucumaru Maheswaran, Albert I. Reuther, James P. Robertson et al. "A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems." *Journal of Parallel and Distributed computing*, pp. 810-837. 2001.

[20] Iverson, Michael, and Fusun Ozguner. "Dynamic, competitive scheduling of multiple DAGs in a distributed heterogeneous environment." In *Heterogeneous Computing Workshop, 1998.(HCW 98) Proceedings. 1998 Seventh*, pp. 70-78. IEEE, 1998.

[21] Singh, Lovejit, and Sarbjeet Singh. "A Genetic Algorithm for Scheduling Workflow Applications in Unreliable Cloud Environment." In *Recent Trends in Computer Networks and Distributed Systems Security*, pp. 139-150. Springer Berlin Heidelberg, 2014.

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

[23] Agarwal, Amit, and Padam Kumar. "Economical duplication based task scheduling for heterogeneous and homogeneous computing systems." In*Advance Computing Conference, 2009. IACC 2009. IEEE International*, pp. 87-93. IEEE, 2009.

[24] Agarwal, Amit, and Padam Kumar. "An effective compaction strategy for bi-criteria DAG scheduling in grids." *International Journal of Communication Networks and Distributed Systems*, pp 331-346, 2010.

[25] Eclipse Juno [online] http://eclipse.org/ [Accessed 10 March 2015]