An Efficient Workflow Scheduling Approach in Cloud Computing

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

Master of Engineering
in
Software Engineering

Submitted By
Vijay Prakash
(Roll No. 801231029)

Under the supervision of:
Ms. Anju Bala
Assistant Professor

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, "*An Efficient Workflow Scheduling Approach 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 *Ms. Anju Bala* and refers other researcher's 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.

(Vijay Prakash)

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

(Ms. Anju Bala)
Assistant
Professor,
CSED,
Thaper University,
Patiala.

Countersigned by

(Dr. Deepak Garg)
Head
Computer Science and Engineering Department
Thapar University
Patiala

(Dr. S. K. Mohapatra)
Dean (Academic Affairs)
Thapar University
Patiala

Acknowledgement

First of all, I am very much thankful to the God for his blessings and showing me the right direction to reach here.

No words to express my gratitude and pleasure towards the guidance and help I have received from my guide, Ms. Anju Bala. I am very much thankful for her support, and valuable suggestion time to time. Without her help this work was not possible to complete and she not only provided me help whenever needed, but also the resources required to complete this thesis report on time.

I am also thankful to Dr. Deepak Garg, Head, Computer Science and Engineering Department for his kind help and cooperation. I express my gratitude to all the staff members of Computer Science and Engineering Department for providing me all the facilities required for the completion of my thesis work.

I would also like to say thanks to all my friends for their support. I want to express my thankfulness to every person who contributed with either inspiring me or helping me in actual work to this thesis.

Last but not the least I am highly grateful to all my family members for their inspiration and ever encouraging moral support, which enables me to purse my studies.

Vijay Prakash

(801231029)

Abstract

Cloud Computing is a latest trend in today's world. It provides on demand services like hardware, software, platform, infrastructure and storage etc. dynamically to the user according to the "pay per use" model by using virtualized resources over the internet. Cloud computing is able to host various applications such as business, social networks and scientific applications.

While Cloud computing provides various services like IaaS, PaaS and SaaS etc. to end users but due to novelty of cloud computing, it also suffers from many types of research issues such as security, performance, database management, virtual machine migration, server consolidation, fault tolerance and workflow scheduling etc. Among these workflow scheduling is major issue for scientific applications.

Existing workflow scheduling algorithms in the grid and cloud environment focused on several QoS parameters such as cost, CPU time, makespan and reliability etc. have been surveyed out and some of the time based scheduling algorithms such as First Come First Serve (FCFS), Min-min, Max-min, and Minimum Completion Time (MCT) has also been discussed in this thesis. The time based scheduling algorithms have been used to minimize the execution time only but no awareness has given to utilize resources for reducing execution time. So, a new scheduler named MaxChild has been proposed to increase the resource utilization and to reduce the overall completion time. MaxChild scheduler focuses on the parent-child relationship in the tasks of workflow and schedules the task having the maximum number of Childs. The proposed scheme has been validated by using simulation based analysis though WorkflowSim.

Table of Contents

List of Figure and Tables

Figures:

Tables:

Chapter 1
Introduction

This chapter introduces the basis of Cloud computing, its background and evolution with related technologies such as grid computing and utility computing along with its characteristics and its types. The various research issues related to Cloud computing have also been discussed and structure of the thesis is given at the end of this chapter.

1.1 Background

With the development of computer and internet, new challenges are arising day by day. One of those challenges is increasing in demand of connectivity and resource handling. Due to globalization of every IT infrastructure dynamic resource and access is become mandatory. To overcome these challenges, a very elastic infrastructure is needed that can be managed according to increase or decrease in demand. These factors are responsible for the concept of Cloud. It has been said that "Cloud" is a new technology in computer era but its market performance show a totally different picture. Cloud computing concept came in the early 1960 when John McCarthy depicted that "computation may someday be organized as a public utility" i.e. utility computing came into existence.

1.2 Evolution of Cloud Computing

Cloud computing has been evolved through many phases as shown in Figure 1.1. Grid computing is based on the concept of linking a number of systems to increase the availability and scalability. Utility Computing grows up with the standard web 2.0 [2] for compute services in which resources such as storage, infrastructure, computing power and applications etc. are measured on the pay-per-use basis and utility computing acts as the base of Cloud computing. Therefore, there are various definitions of Cloud computing given by several research scholars and organizations. 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" [3].

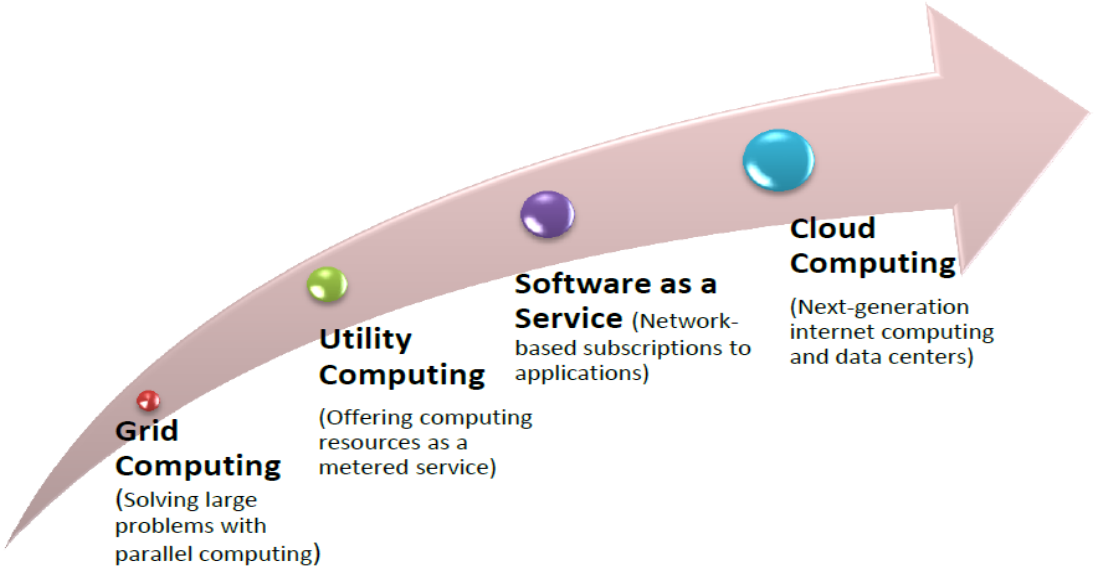


Figure 1.1: Evolution of Cloud Computing [1]

Figure 1.2 describes the main aspects of Cloud computing which includes its benefits, its service types and modes along with its features and stakeholders.

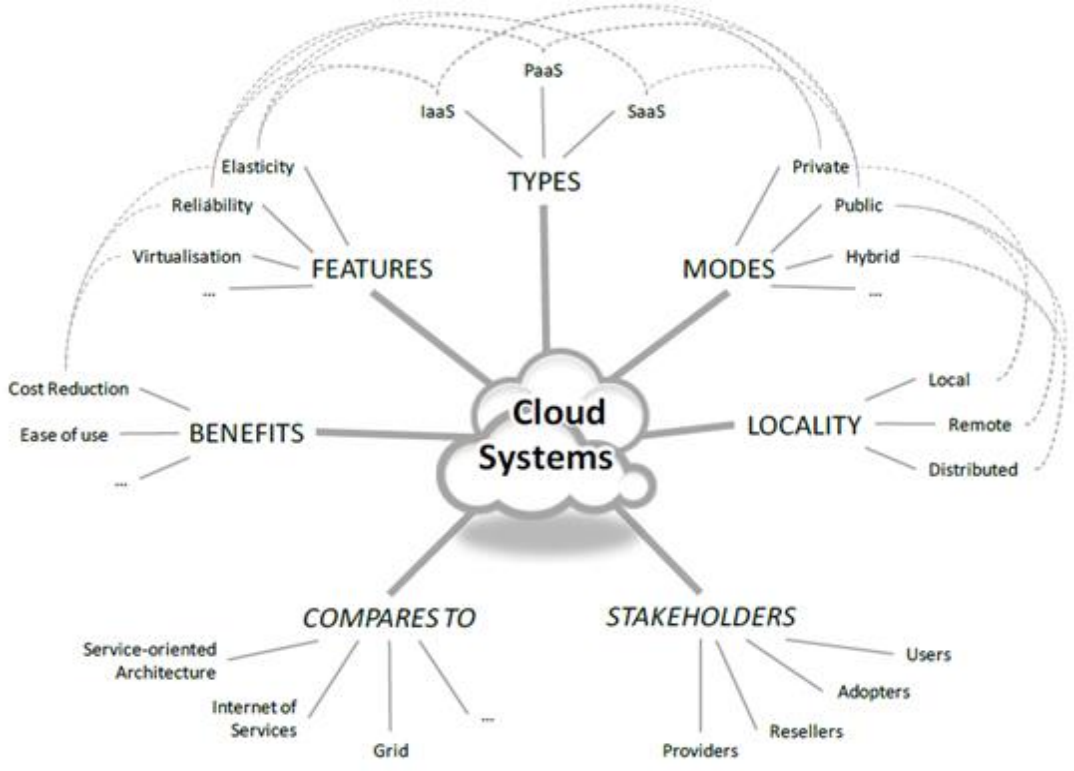


Figure 1.2: NON-Exhaustive View on the main aspect forming a Cloud System [4]

Figure 1.2 shows that Cloud computing is touching most of the aspects of market and every level of consumers. There are several benefits of Cloud computing [1] [5] which are described below:

- **Open Access:** Cloud service provider can be accessed with the help of fair internet connection.
- **Improved economies of scale:** On the user side, by decrease in the investment and running costs and on the provider side, by providing higher productivity in arranging infrastructure services with high survivability and flexibility.
- **Capacity for on-demand infrastructure and computational power:** Users can demand for the computational power, storage and other infrastructure according to their need according to pay per use model.
- **Improved resource utilization:** Resources are utilized properly because whenever users don't need a resource then they return it back to the cloud provider. So in this way elasticity and flexibility can be increased.
- **Reduced information technology (IT) infrastructure needs:** Cloud computing provides infrastructure-as-a-service on demand for user. So there is no permanent need to purchase the infrastructure for the IT. The user can purchase it from cloud provider whenever needed.
- **Resource pooling:** The consumer generally has no information about the locality of the service provider. Thus the provider serves multiple consumers by assigning resources dynamically and virtually.
- **Control systems with abstract policies:** There is no need to provide unnecessary detail about the core business component to the user.
- **Organizations focuses on their core competencies:** Non IT user can contact to IT service provider for their business activity needs.

1.3 General characteristics

Cloud computing offers the following general characteristics [5] [6]:

- **Utility-based pricing:** Lowers the cost because it provides services on pay per use model. The charge for the service changes from one perspective to another e.g. one provider may purchase services from another on the hour basis pay model and it may further provides these services to another customers on the basis of number of client it can serve simultaneously.

- **Broad network access:** Cloud services are made available via internet. Any device having internet connection such as mobile phones, laptops, and tablet etc. can access these services. Also these service provider have their data center location at many places.
- **Shared resource pooling:** The necessary resources are provided dynamically to various consumers by using virtualization and multi-tenancy technologies.
- **Dynamic resource provisioning:** Resources are willingly available for the demand of user in the form what they want without requiring any manual involvement.
- **Measured service:** The resources provided measured in terms of customer demand and consumption of these resources billing is measured by "pay-per-use-model."
- **Self-organizing:** Cloud service provider have to manage the resources according to their own needs. Consumption of these resources are managed according to the demand of the user.

1.4 Service model

There are basically three main service model: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). Different services are provided by cloud providers. These services are described in Figure 1.3. In general these services are known as XaaS (everything as a Service). The service provider, provides these services as "black boxes" and only tells about how can these services be offered and how can these services be accessed but the internal working detail of the service is hidden from the end user. In the following section these services are described in detail with their benefits:

- **Infrastructure as a Service** – IaaS [6] [7] is the delivery of computer infrastructure because of the fluctuations in market demand of hardware and software resources. To fulfill constantly changing demand, infrastructure is provided as a service on cloud. Customer can order as many resources as they need and return them back to the provider when their goal is achieved. One of the examples for IaaS is Amazon EC2 which provides services for computing power and S3 for storage capacity. Benefits of IaaS are :
    - Lowers and match the power consumption according to demand

- Reduces cost by providing hardware
- Higher returns in terms of higher resource utilization

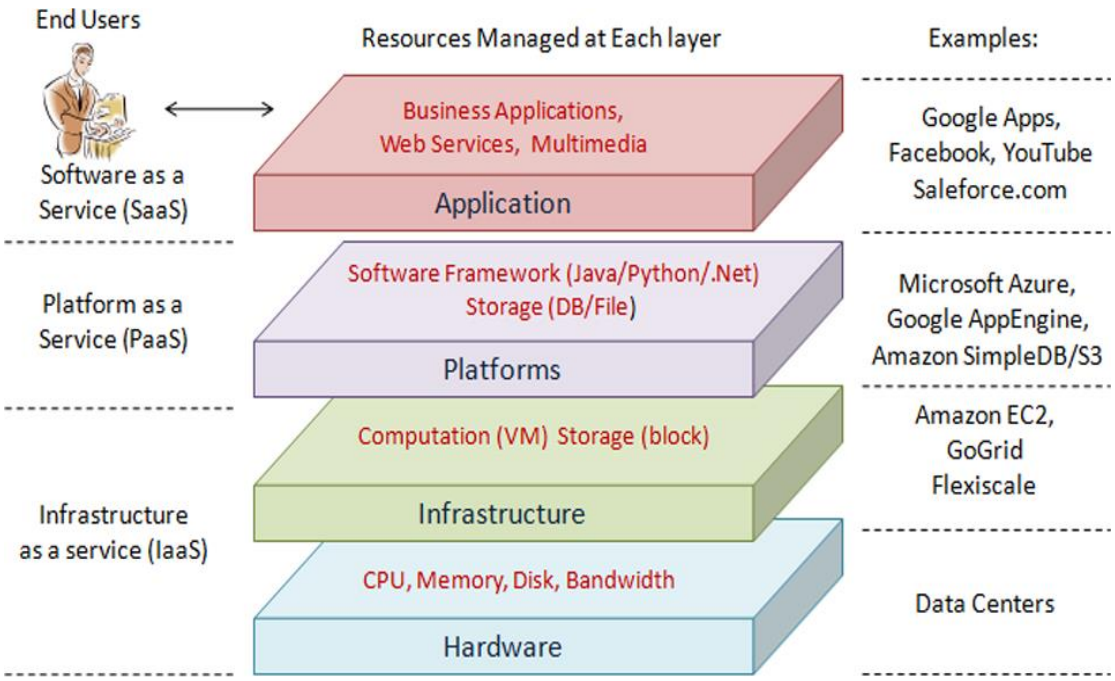


Figure 1.3: Service models describing Level of services inherited [6]

- **Platform as a Service** – PaaS [6] [7] provides a complete set of tools and technologies which are required to develop, test, deploy and operate SaaS applications. This Application Development mostly done by with the help of Web Browsers itself. PaaS examples include Microsoft Azure Services Platform, Google App engine and Amazon S3 etc. The benefits of PaaS are:
  - Pay per use model for development, test and operate SaaS environment
  - Developers just need to focus on application code

- **Software as a Service** –SaaS [6] [7] is a multi-tenant platform. In this, multiple clients get the services from common resources and same infrastructure for both the application and underlying database is provided simultaneously to multiple customers. Cloud computing provides software as a service on internet. Individuals don't need to buy the proprietary rights of software they want to use. They can borrow that software from cloud providers according to their need. Some of the biggest name in market that provide SaaS are salesforce.com providing it's CRM application as service, Google web based calendar

application, and Microsoft online SharePoint etc. These services are charges as per the usage on monthly basis. The benefits of SaaS are:

- Elimination of licensing and version compatibility
- Reduced upfront cost and hardware cost

1.5 Cloud Computing Deployment Model

There are basically four types of cloud deployment models [6] [7] which have been defined in the cloud community as shown in Figure 1.4:

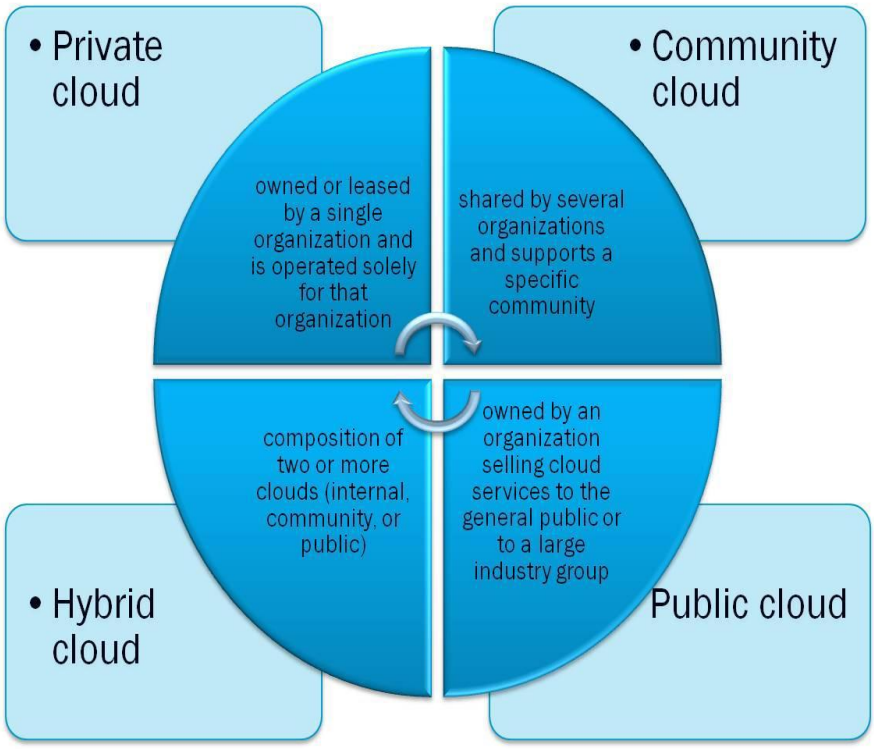


Figure 1.4: Cloud Deployment Model [8]

- **Private cloud:** Private clouds are operated and managed within a single private organization. Sometimes a third party apart from organization manages the private cloud in both on-premise and off-premise. Private cloud is also known as the internal cloud. The benefits of private cloud are:
  - To maximize and optimize the in-house resource utilization within the organization
  - Data privacy for the security concerns
  - Minimum data transfer cost
  - Reliable

- **Public cloud:** Public clouds provides various services to the cloud users and the cloud provider has the full control of these kinds of clouds with its own value, profit, charge model and profits etc. Examples of public cloud include Amazon S3, EC2 and Google AppEngine etc. The benefits of public cloud are:
    - No initial capital investment on infrastructure
    - No data transfer risk

- **Community cloud:** Community clouds are established by sharing the specific resources, infrastructure, requirements and policies of same cloud to a community or a business sector. These clouds are different from public clouds, which serves the different needs of multiple users and also different from private cloud, which serves within a single organization. Benefits of Community clouds are:
    - The cloud infrastructure can be hosted by a third party organization
    - Economically scalable

- **Hybrid cloud –** Two or more type of clouds (public, private or community) are combined together to form this type of cloud known as hybrid cloud. To optimize the resources and core competencies hybrid clouds are used by several organizations as the private cloud and for business purpose they used is as a public cloud. Benefits of hybrid clouds are:
    - More flexible than public and private cloud
    - Provide more security over data than public cloud
    - Provide on demand service expansion and compaction by provisioning and releasing the resources whenever needed

1.6 Research Issues in Cloud Computing

Cloud computing is an emerging technology which has high controls and impact on the IT industry from recent years but it is still in its initial phase and suffers from many challenges. This section briefly describes the research issues [6] [9] in Cloud computing environment.

1.6.1 Automated Service Provisioning

Cloud provider allocate and de-allocate resources dynamically to satisfy its service level objectives. These approach mainly involves (i) an application level model is

constructed to handle the dynamic request of resources at each level; (ii) Future demand for the resources requirement is predicted periodically and (iii) allocating the resources requirement automatically which has been predicted in the previous step [6].

1.6.2 Energy Management

Service provider has pressure to reduce the energy consumption by meeting the government rules and environmental standards. To manage energy properly, service provider have to design efficient data centers, for example server consolidation and energy aware job scheduling are the two approaches to manage the energy effectively by turning off the unused machines [6].

1.6.3 Virtual Machine Migration

Virtual machine (VM) migration is done with the help of virtualization technology by balancing the load across the data centers. VM migration can be achieved with the help of process migration techniques. The difficulties occurred in the VM migration through process migration techniques has been avoided by migrating the whole OS with all its service components and applications as a single unit [6].

1.6.4 Server Consolidation

VMs residing on multiple under-utilized server can be migrated to single server by the live VM migration approach, so that other servers can be in energy saving state. But to control the resource utilization and energy consumption properly, VM migration alone is not sufficient. So server consolidation is used to maximize resource utilization along with minimizing the energy consumption in the cloud environment. Application performance still remains same after the server consolidation because the resource usage by individual VM vary over time [6].

1.6.5 Security Issues

There are mainly two security issues [9] as follows:

- **Availability of Service –** All available SaaS products have a very high standard and users expect high accessibility of these standards from service providers. But it is very difficult for single provider to create and maintain more than one stack of these software to provide independent software stack for different companies. Another availability problem is Distributed Denial of Service (DDoS) attacks.

- **Data security –** Security system of datacenters are not accessible to service providers but they can only specify the security setting and depends upon the infrastructure provider for accessing the control of complete data security. In this process, confidentiality and auditability of data security is achieved. Confidentiality is achieved through cryptography which is necessary for secure data transfer and auditability is necessary to check whether system security settings has been tempered or not.

1.6.6 Data Issues

Data issues [9] mainly include the following:

- **Data lock-in -** Software stacks are more compatible with interoperability between platforms, but it is difficult for customer to extract their data and programs from one location and then switch to another. Customer seems to be more striking to these service providers. Users of Cloud computing are worried about consistency problems and service providers for leaving out of the business. So SaaS providers deploys the services and data on multiple clouds so that there is no fear of change of customer data due to failure of a single company. The only fear is that they are much worried about the cloud pricing and compress the profits. There are two options to relieve this fear. Firstly, the quality is also important in comparison with the price, so that customers will not attract to the lowest cost service. Secondly, it concern of data lock-in justification, which is APIs standardization leads to a new model for private cloud and public cloud with same software infrastructure usage.

- **Data transfer bottlenecks -** The applications which moved across the boundaries of clouds can make problems of data placement and transport. Cloud providers and users have to minimize costs on the idea of the traffic and placement of data at each level of the system. One option is to keep the data on cloud and other option is to transfer this data in form of disks instead of high cost of bandwidth transfer.

1.6.7 Workflow Management Issue

Workflow Management is one of the issue in Cloud computing. Most of the business process can be represented in terms of workflow. So a workflow can be described as the set of tasks which is used to complete some business process. Task invocation, task

synchronization, and information flow are done in a specific order which is described by workflow management. The tasks of workflows are varying in nature. Main issue in workflow management system is workflow scheduling because it is very difficult to identify the available resource from the central pool of resources at the time of execution of workflow. Workflow scheduling is a problem of finding a correct execution sequence for the workflow tasks, i.e., execution that obeys the constraints which represents the business logic of the workflow. Mapping and management of workflow task's execution on shared resources is done with the help of workflow scheduling [10]. Workflow Management System and workflow scheduling has been discussed in detail in the next chapter.

1.7 Structure of the Thesis

The rest of the thesis is organized as follow:

Chapter 2 – This chapter discusses the detailed analysis of literature review of workflow management system, fault tolerance and existing techniques/algorithms for workflow scheduling.

Chapter 3 - This chapter describes the problem definition along with gap analysis and objective of research work.

Chapter 4 - This chapter discusses in detail the solution of the problem with the help of flowchart and Proposed algorithm.

Chapter 5 - This chapter focuses on implementation details and experimental results, with the description of WorkflowSim, Netbeans and snapshots of comparative experimental results of the proposed technique with some of the already existing technique.

Chapter 6 - This chapter describes the conclusion, contributions of work done and future directions possible.

Chapter 2
Literature Review

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

2.1 Workflow Management System

Workflow Management System (WMS) is used for proper management of executing the workflow tasks on the computing resources. The major components of WMS are shown in Figure 2.1.

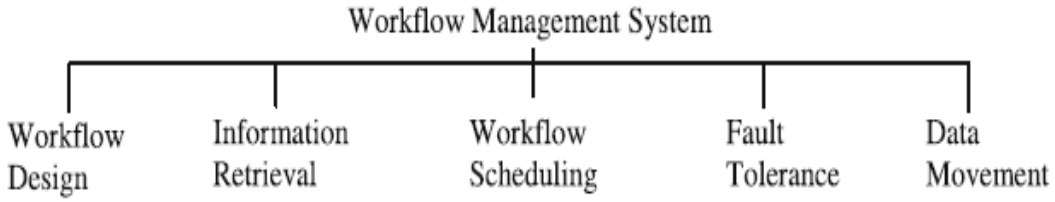


Figure 2.1: Elements of a WMS [11]

Workflow design describes how components of workflow can be defined and composed. The elements of workflow design as shown in Figure 2.2.

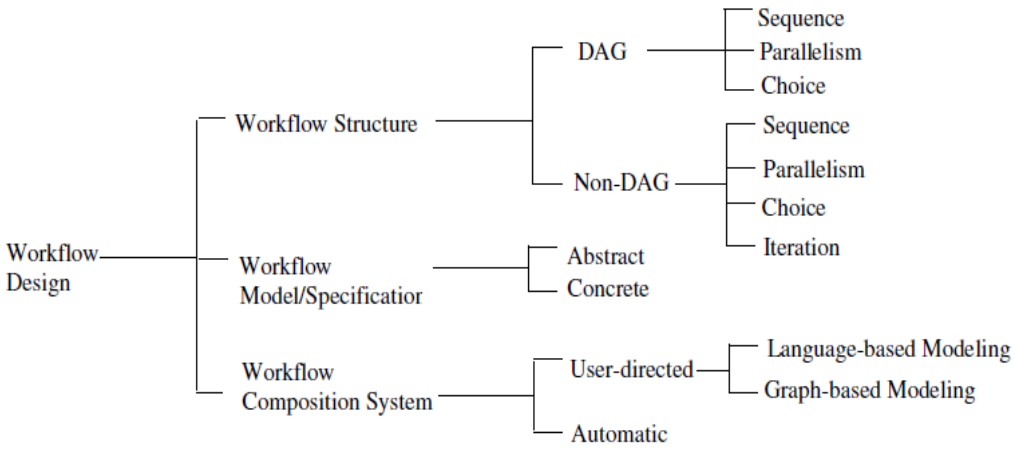


Figure 2.2: Taxonomy of Workflow Design [12]

- A workflow structure describe the relationship between the tasks of the workflow. A workflow structure can be of two type Directed Acyclic Graph (DAG) and Non DAG. Workflow structure can further be categorized as Sequence, Parallelism and Choice in the DAG based structure. In the sequence

structure, the tasks execute in a series. A new task can only be executed if the previous task has been successfully executed. In parallelism structure, tasks of workflow can executes concurrently. In choice structure, 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 model defines the workflow, its tasks and its structure. Workflow models are of two types; abstract and concrete. Abstract workflow describe the workflow in an abstract form which means that specific resources are not referred for task execution. Concrete workflow describes the workflow tasks binded to specific resources.

- Workflow composition system allows users to accumulate component to the workflow. In the user directed system, user can modify the workflow structure directly by editing in the workflow. But in automatic, system generates workflow automatically. User directed system is further categorized in Language based modeling and graph based modeling. In former, user describe the workflow using some markup language such as XML. It is a difficult approach because user have to remember a lot of language syntax. Graph based modeling is simple and easy way to modify the workflow with the help of basic graph elements [11].

2.2 Workflow Scheduling

Mapping and management of workflow task's execution on shared resources is done with the help of workflow scheduling. So workflow scheduling finds a correct sequences of task execution which obeys the business constraint. The elements of workflow scheduling are shown in the Figure 2.3.

- Scheduling Architecture is very important in case of quality, performance and scalability of the system. In the centralized workflow environment, scheduling decisions for all the tasks in the workflow is done by a single central scheduler.

There is no central controller for multiple schedulers in decentralized approach but communication between schedulers is possible. Each scheduler schedules the workflow to the resource having less load. On the other hand in hierarchical scheduling there is a central manager. Not only Workflow execution is controlled by this central manager but also the sub-workflows are assigned to the lower-level schedulers [11].

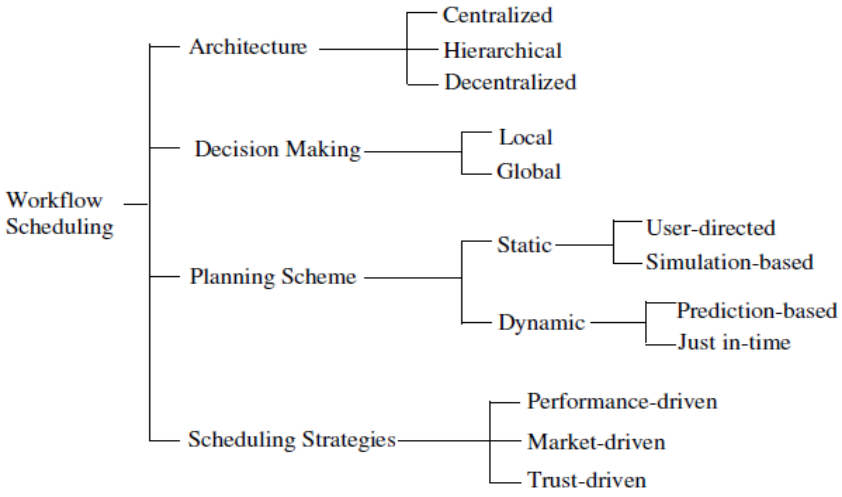


Figure 2.3: Elements of Workflow Scheduling [12]

- Scheduling decisions which are taken on the basis of task or sub workflow are known as local decisions and which are taken by keeping in mind the whole workflow are called global decisions. Global decisions based scheduling gives better overall results because only one task or sub workflow is considered in local decision scheduling.

- Transformation of abstract models to concrete models is done in two ways: static and dynamic. Concrete models are generated before the execution with the help of current available information about the execution environment and the dynamic change of resource is not considered in the static scheme. Static schemes further categorized in two types; user-directed and simulation-based. In the user directed planning scheme, decision about resource mapping and scheduling is done on the basis of user's knowledge, preferences and performance criteria. But in case of simulation based, a best schedule can be achieved by simulating task execution on resources before the workflow execution starts. Both static and dynamic information about resources used for

making scheduling decisions at run-time are consider in case of dynamic scheme. For dynamic scheme, there are two approaches prediction-based and just-in-time scheduling is used. In prediction-based, dynamic information is used along with some results on basis of prediction and in simulation-based static scheduling, in which prediction of performance of execution of tasks using resources is done along with generation of optimal schedule before the execution of task is done. But due to this the initial schedule changed during the execution. Just in-time scheduling makes decision at execution time.

- Scheduling strategies are categorized into performance driven, market driven and trust driven. In case of performance driven strategies, focus is to achieve the highest performance for the user defined QoS parameter. So the workflow's tasks mapped with resources gives the optimal performance. Most of the performance driven scheduling strategies focuses to maximize the makespan of the workflow. Market driven Strategies focuses on resource availability, allocation cost and quality for the budget and deadline. A market model is used to schedule the workflow tasks to the available resource dynamically which results in less cost. Trust driven focuses on security and reputation of the resources and tasks are scheduled based on the trust by considering these two parameters [11].

2.3 Related Work

In this section most of the workflow scheduling strategies exists in the grid and cloud environment has been reviewed briefly with respect to the technique/algorithm description, scheduling parameter considered and tools/platform used for the implementation for result analysis. These existing scheduling algorithms are classified in the following categories:

- **Heterogeneous Earliest Finish Time:** Wieczorek [13] and Durillo [35] focused on makespan and cost. In [13] average communication time and average execution time of each task of workflow is calculated first and then rank is assigned on the basis of these two parameters. The task having highest rank is scheduled first. MOHEFT [35] extends HEFT in the cloud environment by using the SPEA2* algorithm [41] for bi-objective scheduling criteria to optimize both makespan and the economic cost in the cloud environment.

- **Optimizing parameter based Scheduling:** This category mainly include [14] [16] [17] [19] [22] [28] [31] [32] [34] [39], In these algorithms two or more parameters are considered for scheduling in which one parameter is fixed and on the basis of this parameter another parameters are optimized. These algorithms works in both the Grid and Cloud Environment.

- **Market Oriented Scheduling:** In [15] market oriented scheduling approach the resource are available in the market and user have to buy the resource by bidding procedure to schedule their tasks and in [40] on the basis of budget the tasks are scheduled with minimum execution time.

- **Genetic Algorithms:** This category mainly includes [16] [29] [36] in which various different parameters are considered for scheduling such as makespan, time, cost and reliability etc. Genetic algorithms uses the mutations and crossover functions to optimize a parameter by considering the other parameters.

- **Critical Path based Scheduling:** This category mainly includes [18] [20] [21], these algorithms focuses on makespan and schedules the task by dynamically identifying the critical path which gives the lowest execution time for the workflow. Some variation of critical path algorithm focuses on identifying the predecessors of a task for critical path creation.

- **Multiple QoS with Multiple workflow:** Meng Xu [23] focused on makespan and cost. This algorithm calculates the mean execution time and mean execution cost of all the workflows and schedules the task first having minimum covariance of time and cost to optimize both the makespan and cost.

- **Ant Colony Optimization (ACO) based scheduling algorithms:** There are various version based on ACO such as [25] [26]. The basic ACO algorithms is based on the foraging behavior of ants. Whenever any ant searched a food, then it spread the chemical known as pheromone to make the shortest path from the food to their destination. Others ants follow this path with the help of pheromone and reached to the food as soon as possible by following the shortest path. The algorithm based on ACO consider many QoS and objective of these

algorithm is to find a best optimized solution by considering the user preferred QoS parameter.

- **Particle Swarm Optimization (PSO) based Scheduling Algorithms:** Pandey [27] proposed a PSO based heuristic which consider the computational and transmission cost as the scheduling parameters. The algorithm calculates the average computational and average transmission cost of each task and schedules tasks to the resource with minimized cost.

- **Priority based scheduling algorithms:** Ghanbari [33] proposed a priority based job scheduling algorithm by considering makespan as the scheduling parameter in the cloud environment by considering three levels of Analytical Hierarchy Process which are represented with three types of job priority which are objective level, scheduling level and job level.

- **Hierarchical scheduling strategy:** Zhangjun Wu [37] includes GA, ACO and PSO heuristics for the job level and resource level scheduling known as the assignment of task-to-service scheduling by considering makespan, cost and resource utilization as the scheduling parameters.

- **Trust based Scheduling algorithm:** Yuli Yang [38] schedules the task by verifying resource failure probability during the transmission of task with security and reliability constraint.

A total of 30 research papers has been considered and summarized in the Table 2.1 which focuses on the various scheduling strategies.

Table 2.1: workflow scheduling strategies exists in the grid and cloud environment

Author & Year	Algorithm/ Technique	Scheduling Parameter	Description	Tools/ Platform	Environment
Marek Wieczorek (2005) [13]	Heterogeneous Earliest Finish Time	Makespan	(i) Calculates average communication time and average execution time. (ii) Schedules the task having highest rank	ASKALON	Grid
Gurmeet Singh (2005) [14]	Optimizing based grid scheduling	Makespan	(i) Minimizes the various cost related to resource matching, task submission and updating in the ready queue.	Condor and TeraGrid	Grid

			(ii) Restructure the workflow for multiple submission host.		
Chia Hung (2005) [15]	Market Oriented Scheduling	Resource utilization	(i) Schedules the tasks by bidding procedure.	Real testbed consists of 15 CPU, 10 network connection and 15 units of memory and 30 discs.	Grid
Jia Yu (2006) [16]	Genetic algorithm	Time and Cost	(i) Time and cost is optimized.	GridSim	Grid
Yongcai Tao (2007) [17]	Reliability Cost Grid Scheduling	Reliability and Cost	(i) Assigns higher ranks to big task. (ii) Tasks are grouped in decreasing order of rank and finally scheduled using max-min or min-min heuristics.	Real testbed at Cluster and Grid Computing Lab	Grid
Bogdan Simion (2007) [18]	Improved Critical path using Descendant Prediction	Makespan and Load balancing	(i) As Late As Possible (ALAP) time for each task is calculated. (ii) Scheduled the task having minimum ALAP.	Mon-Alisa farms and ApMon	Grid
Rizos Sakellariou (2007) [19]	Budget based scheduling technique	Makespan and Budget	(i) Minimizes the makespan and cost by using the LOSS and GAIN approach.	GridSim	Grid
Mustafizur Rahman (2007) [20]	Dynamic Critical Path	Makespan	(i) Schedules the task by dynamically identifying the critical path which gives the lowest execution time for the workflow.	GridSim	Grid
Marek Wieczorek (2008) [21]	Dynamic Constraint Algorithm	Bi-criteria	(i) Uses variable parameter pair and slicing constraint to optimize one parameter by slicing the second parameter dynamically.	ASKALON	Grid
Rajiv Ranjan (2008) [22]	Decentralized and Co-operative Scheduling	Makespan and Scalability	(i) Calculates the average execution time, average response time and average Coordination delay. (ii) Schedules the task with the resource co-operation to decrease the probability of failure.	GridSim and PlanetSim	Grid

Meng Xu (2009) [23]	Multiple QoS with Multiple Workflow	Makespan and Cost	(i) Schedules first the task having minimum covariance of time and cost.	Simulation environment with 20 services and 5-25 users.	Cloud
Sanchez Santiago (2009) [24]	Dynamic Balanced Scheduler	Response time	(i) Splits the whole workflow into balanced partitions so that all partitions are executed within the same time.	GridSim	Grid
Wei Neng Chen (2009) [25]	ACO approach with various QoS	Time, Cost and Reliability	(i) Optimizes one parameter according to user constraint given for remaining two parameters.	GridSim	Grid
Ruay shiung Chang(2009) [26]	Balanced Ant Colony optimization (BACO)	Makespan	(i) Selects and schedules the job on the basis of MCT heuristic.	UniGrid	Grid
Suraj Pandey (2010) [27]	Best Resource Selection(BRS) using PSO	Computational and Transmission cost	(i) Calculates the average computational and average transmission cost of each task. (ii) Schedules tasks to the resource with minimized cost.	JSwarm	Cloud
Wang yong (2011) [28]	Deadline and budget constraint scheduling algorithm	Makespan, budget and relative cost	(i) Sorts the tasks in increasing order of size. Then equal sized chunks are created and theses chunks are scheduled according to increasing order of relative cost.	GridSim	Grid
Xiaofeng Wang (2011) [29]	Look ahead genetic algorithm	Makespan and Reliability	(i) Works in two steps namely evolution and evaluation. During the evaluation step algorithm itself provides the order of execution of workflow instead of evolution step.	GridSim	Cloud
El-Sayed (2012) [30]	Extended Max-min Algorithm	Execution time	(i) Schedules the smaller jobs on slower resources and bigger jobs on faster resources. (ii) Therefore overall waiting time of jobs is reduced and due to which there is improvement in the execution time of workflow.	Java 6	Cloud
Hamid Mohammadi Fard (2012) [31]	Multi-objective workflow scheduling	Makespan, Reliability, Energy consumption and Cost	(i) Considers any three parameter and optimizes the fourth parameter. (ii) Works in two ways either minimizing the	ASKALON	Grid/ Cloud

			optimal parameter by maximizing the remaining parameters or vice versa.		
George Amalarethinam (2012) [32]	Minimum Makespan Grid Workflow Scheduling	Makespan	(i) Reserves the resources in advance. (ii) Does resource preference statically and resource allocation dynamically.	Simulation environment consist of 4-8 resources with speed 1, 1.25, 1.5 and 1.75.	Grid
Shamsollah Ghanbari (2012) [33]	Priority Job Scheduling Creteria	Makespan	(i) Based on three levels of Analytical Hierarchy Process which are represented with three types of job priority which are objective level, scheduling level and job level.	Cloud environment consists of 3 resources.	Cloud
Arash Ghorbannia (2012) [34]	Reliable Scheduling Distributed Technique	Makespan	(i) Calculates the request time of task and acknowledgement time of resource indepen-dently and in the shared mode. (ii) Calculates the difference between above two approaches to increase the efficiency.	Java	Cloud
Juan J. Durillo (2012) [35]	MOHEFT	Makespan and Cost	(i) Extends HEFT [13] and merge with SPEA2* algorithm [41] for bi-objective scheduling criteria to optimize both makespan and the economic cost.	GridSim	Grid/ Cloud
Somayeh Kianpisheh (2012) [36]	Genetic Algorithm	Makespan	(i) Minimizes the overall completion time of the workflow by considering the communication and computational cost.	GridFlow	Grid
Zhangjun Wu (2013) [37]	Hierarchical scheduling strategy	Makespan, Cost and Resource utilization	(i) Includes GA, ACO and PSO heuristics for the job level and resource level scheduling known as the assignment of task-to-service scheduling.	SwinDeW-C	Cloud
Yuli Yang(2013) [38]	Trust based Scheduling algorithm	Reliability and Security	(i) Schedules the task by verifying resource failure probability during the transmission of task with security and reliability constraint.	CloudSim	Cloud

Dong-ki kang (2014) [39]	Cost based heuristic scheduling scheme	Cost	(i) Tasks assigned to resources are allocated in sequence to the virtual machine instance. (ii) Tasks assigned in the first step to different VM are combined in a single VM instance and executed in parallel. (iii) Less resources are used and there is nearly 30% of reduction in the cost.	Openstack	Cloud
Hamid Arabnejad (2014) [40]	Heterogeneous Budget Constrained Scheduling Algorithm	Execution time and Cost	(i) Calculates the minimum execution time with highest cost. (ii) Calculates the minimum cost with the corresponding deadline. (iii) Reduction in 30% of execution time with same budget level.	Real cloud simulation by sharing bandwidth	Cloud

2.4 Fault Tolerance

Fault tolerance is related to handling the failure which can occur during the scheduling process due to several reasons such as resource unavailability/failure, task failure, overloaded resource, network fault, and lack of memory etc. Fault Tolerant WMS should be able to control these failures. The basic elements of Fault tolerance is shown in Figure 2.4.

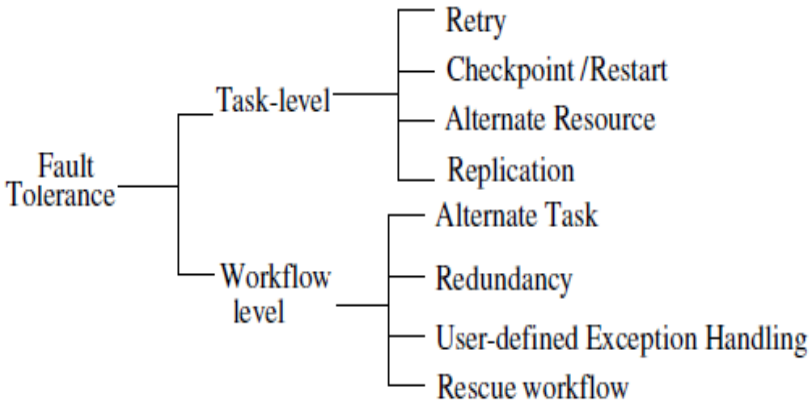


Figure 2.4: Element of fault tolerance [12]

Fault tolerance is categorized into two main categories as task level and workflow level. In case of task level, failures occurred due to workflow task are handled and in case of

workflow level, failure occurred due to the workflow structure is handled. Task level failure handling is further categorized into four aspects [11]:

- Retry technique is simplest method to handle the failure. In case of failure of a task or resource, Retry method simply tries to execute the same task on the same resource again.
- Checkpoint technique stores the state of the system when the failure occurred and then tries to execute the same task on some other resource and execution starts from the saved state after the failure occurrence.
- Alternate resource technique tries to execute the failed task on some other resource when there is a failure happened.
- Replication technique tries to execute the same task on the same resource again and again so that any one of the trial handles the failure.

Workflow level failure handling mechanism consider the whole workflow instead of the subtask or sub workflow. Workflow level failure handling is further categorized into four aspects [11]:

- Alternate task technique tries to execute another task of the workflow in place of previous failed task.
- The redundancy technique tries to execute multiple alternate tasks simultaneously instead of single alternate task in place of previous failed task.
- User defined exception handling provides a specific mechanism to handle the task failure in the workflow.
- The rescue technique ignores the failed task once and tries to complete the execution of other task in the workflow until the execution is possible without handling the failed task till the completion of the workflow. If the whole workflow executes except the failed task then after the workflow execution completed, the rescue technique identify the failed tasks and allowed them to submit again for execution.

2.5 Workflow Task Clustering

Task clustering is basically used to combine the small sized tasks of the workflow into a comparatively large sized task to minimize the makespan of the workflow by reducing the impact of queue wait time. So task clustering restructure the workflow by combining the small sized tasks into a cluster and execute this cluster as a single task.

Due to clustering the number of task in the workflow is reduced and there is less queue wait time as compared to small sized tasks. Task clustering is categorized as level- and label based clustering [42], vertical, blocked and balanced clustering [43]. Figure 2.5 represents a simple workflow. Figure 2.6 describes horizontal clustering in level 2 and level 4 with each cluster contains two tasks.

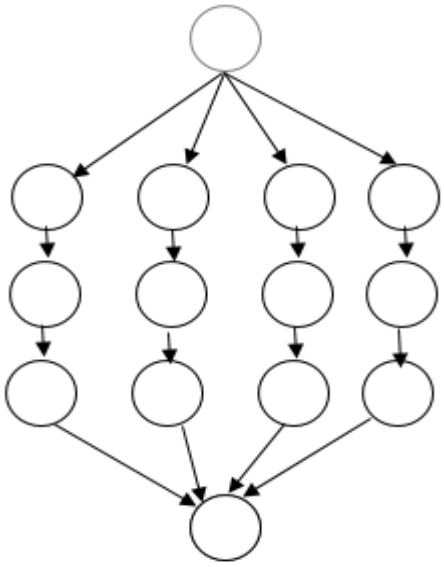


Figure 2.5: A simple workflow

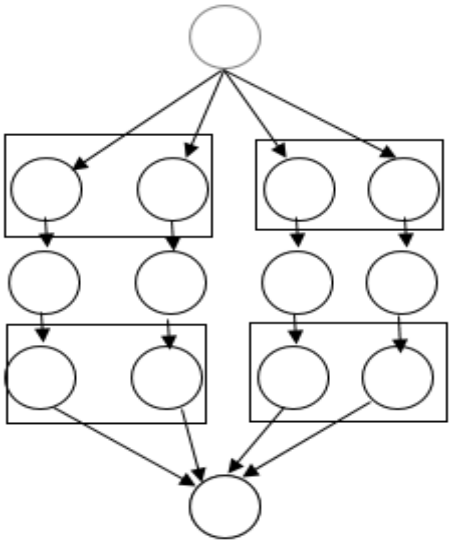


Figure 2.6: Horizontal clustering

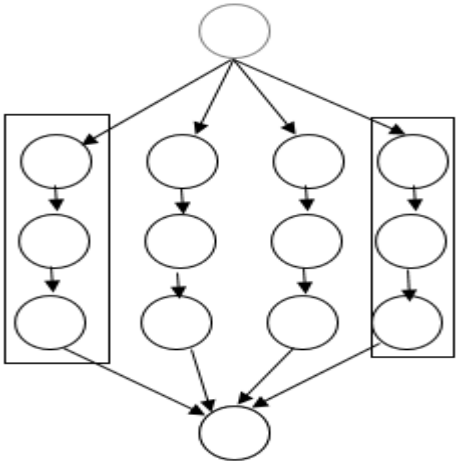


Figure 2.7: Vertical clustering

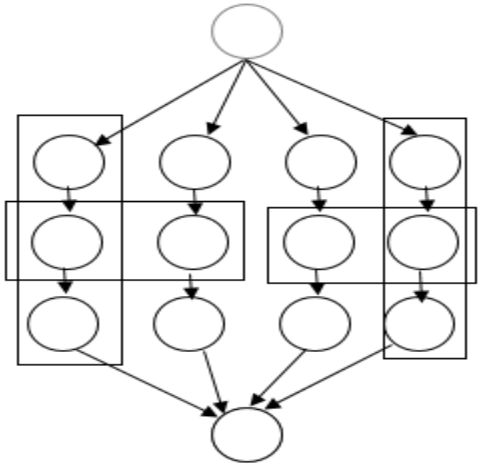


Figure 2.8: Blocked Clustering

- Level-based clustering is also called horizontal clustering. In this type of clustering the tasks are independent and the tasks of the same level of the workflow can be combined together.
- In vertical clustering, tasks of the same pipeline can be combined together.

- Blocked clustering represents the combination of both horizontal and vertical clustering in the same workflow.

Figure 2.7 represents vertical clustering, in which tasks in the same pipeline can be combined together to form a cluster. Each cluster of vertical clustering contains three tasks. Figure 2.8 shows the blocked clustering, in which horizontal and vertical clustering can be combined together.

Tasks in the same level of workflow can have different execution time and whenever these tasks are combined without the consideration of their runtime variance then it causes the problem of load imbalance, i.e., some cluster may contain the smaller tasks and other may have larger tasks. Due to this inappropriate clustering, workflow overhead is produced. Another problem is data dependency between the tasks of workflow in case of level-based clustering which is called dependency imbalance. So to overcome these two problem, balanced clustering comes into existence in 2013[43].

Balanced clustering consider two metrics, Horizontal Runtime Variance (HRV) and Impact Factor Variance (IFV) to solve runtime imbalance and dependency imbalance problem respectively. HRV can be defined as "the ratio of the standard deviation in task runtime to the average runtime of tasks/jobs at the same horizontal level of a workflow" [43]. IFV of tasks can be defined as the standard deviation of their impact factor. The Impact Factor (IF) of a task t_u is defined as shown in Eq. (1)

$$IF(t_u) = \sum_{t_v \in Child(t_u)} \frac{IF(t_v)}{L(t_v)} \qquad (1)$$

Where Child(t_u) denotes the set of child tasks of t_u, and L(t_v) the number of parent tasks of t_v [43].

2.6 Simulators

To evaluate the performance of workflows in a real grid and Cloud environment is a challenging tasks. So instead of real environment, various simulators have been modeled and Scientifics workflows are most popularly evaluated by these simulators. There are various workflow simulators available such as GridSim, CloudSim, CloudAnalyst and EMUSIM in the grid and cloud environment.

Deelman et al. [44] have discussed workflow systems and categories of its components, such as composition of components, their mapping, and their execution etc. These all factors are considered by simulators.

GridSim [45] was proposed by R. Buyya because it was not possible to evaluate scheduler performance repeatedly and in a controlled way in the grid environment as the users have their own policies and resources are distributed across the multiple organizations. This simulator performs the simulation of heterogeneous resources and other functionality of grid computing such as new application task creation, management and mapping of tasks and resources. Scheduling algorithm that keep track of time and cost constraints can be simulated in this as well as their performance can be evaluated.

CloudSim [46] was also proposed by R. Buyya which extends the features of GridSim that allows modeling and simulating the environment of cloud, datacenters, virtual machines, and cloudlets etc. CloudSim deals only with single work load but it is not suitable for workflow scheduling as multiple tasks need to be scheduled together. CloudSim deals with task execution, but it doesn't consider dependencies between tasks and task clustering. It also doesn't deal with overhead due to failures.

CloudAnalyst [47] was also proposed by R. Buyya to help the developers to estimate the maximum requirements of applications in case of graphic distribution of both server and workload. CloudAnalyst basically simulate the applications in order to check their behavior in different environment for different requirements and inputs. It checks the behavior of application under different deployment environments. Using CloudAnalyst, developers get better idea about where they can deploy the application so that it gives better performance result and can manage the peak requests.

EmuSim [48] was also proposed by R. Buyya because in all the other simulators, developers itself provide the application behavior and information regarding its input. EmuSim is an integrated architecture that automatically gathers information from application behavior and then input that information to generate simulation model.

However, these existing workflow simulators fail to provide a framework which considers heterogeneous systems overhead and failures. They have also lack of workflow optimization techniques such as task clustering and job scheduling. In this

section, WorkflowSim is introduced, which extends the existing CloudSim simulator by providing a higher layer of workflow management.

2.7 Time Comparison Based Scheduling Heuristics

The main objective of scheduling algorithm is to achieve the best system throughput with proper resource utilization and high performance by obeying the user's specified QoS parameter. There are various time comparison based scheduling heuristics exists in the grid and Cloud computing environment, which schedules the tasks by comparing the arrival time or execution time of the tasks. In the following section these scheduling heuristics are described.

2.7.1 First Come First Serve (FCFS)

In this algorithm, tasks are compared on the basis of their arrival time and the task which comes first in the ready queue is served first. Advantage of this algorithm is its simplicity and fast execution behavior. But the main disadvantage of this algorithm is that sometimes due to the execution of a longer job, which comes in the queue first, small jobs have to wait for its completion. Due to this problem the waiting time of tasks increased and overall performance of the workflow execution decreases.

2.7.2 Min-min

In this algorithm, small task is executed first so that large task delays for long time. Algorithm begins with by sorting the set of all unmapped tasks in increasing order of their completion time. Then the tasks having the minimum completion is scheduled from the unmapped task set and the mapped task has been removed from unmapped task list, and the process repeats until all the tasks of unmapped list is mapped to the corresponding available resources [49].

2.7.3 Max-min

In this algorithm, large task is executed first so that small task delays for long time. This algorithm is very similar to Min-min algorithm, instead of sorting the task in the increasing order of completion time. This algorithms sorts the tasks in decreasing order of their completion time. Then the task with the overall maximum completion time is selected from this task list and scheduled to the corresponding available resource. Then the scheduled task has been removed from unmapped task set and the process repeats until all tasks of unmapped list is mapped [49].

2.7.4 Minimum Completion Time (MCT)

In this algorithm, task that takes least time to complete is allocated to a machine randomly. So MCT behaves somewhat like Min-min. However, Min-min algorithm considers all the unmapped tasks during each mapping decision but on the other hand MCT considers only one task at a time [50].

Chapter 3
Problem Description

Previous chapter discusses about the existing workflow scheduling techniques in both Grid and Cloud environment. This chapter has been focused on analysis of gaps during the literature review of existing workflow scheduling algorithms.

3.1 Gap Analysis

Based on the literature review in the previous chapter, various authors identifies several problems in the existing workflow scheduling techniques. These problems basically due to the ignored QoS parameters selected for the scheduling decisions which are listed in the Table 3.1.

Table 3.1: Workflow Scheduling problem identified in the existing techniques

Author and Year	**Ignorance of QoS parameter in existing scheduling algorithms**
Chia Hung (2005)	Dynamic environment and combinational resource requirement
Mustafizur Rahman (2007)	Independent on size and resource availability
Marek Wieczorek (2008)	Variable Bi-criteria scheme
Rajiv Ranjan (2008)	Co-operation between the resources
Meng Xu (2009)	Multiple QoS with multiple workflow which are different in structure
Quin Tao (2009)	stability, scalability and flexibility with load balancing
Sanchez Santiago (2009)	Ignorance of response time
Wang yong (2011)	Resource heterogeneity inconsistency in computational grid environment
Xiaofeng Wang (2011)	Random solution given by genetic algorithm which may be invalid
George Amalaret-hinam (2012)	keeping the resources in advance
Shamsollah Ghanba ri (2012)	Priority of jobs before scheduling
Arash Ghorbannia (2012)	Request time and acknowledgement time
Juan J. Durillo (2012)	Bi-objective scheduling criteria for combination of makespan and economic cost
Somayeh Kianpisheh (2012)	Data transmission time
Yuli YANG (2013)	Identified trust scheduling by considering failure probability with reliability
Dong-ki kang (2014)	Unnecessary extra cost due to waste of resources allotted whenever there is no job available for scheduling.

3.2 Problem Statement

As there are four algorithms related to direct comparison of time for scheduling discussed in the previous chapter but if the execution time of all the tasks is same then there is no concept of Max-min, Min-min and MCT scheduler and scheduling is done by FCFS scheduler. So, there is need of an algorithm which suited best for such situation by considering a new QoS parameter.

3.3 Problem Description

Consider the Figure 3.1 in which all the tasks having the same execution time X.

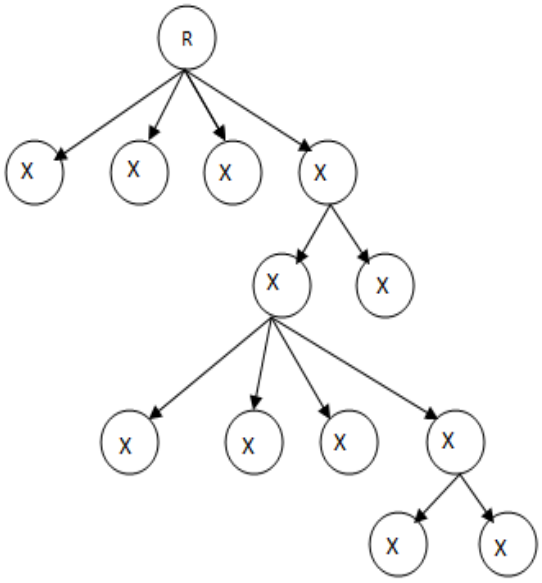


Figure 3.1 : Workflow having same execution time for all tasks

When FCFS scheduling is implemented as shown in Figure 3.2 by consideration of three virtual machines at a time only. Now for first schedule, FCFS scheduler will schedule task 0, 1 and 2 at three virtual machines paralelly, so it will take X time to execute these three tasks. But for second schedule, only task 3 is available for execution because task 4 and task 5 can't be executed at this time since there parent has to be executed first. So it will take another X time to complete task 3 only. During this schedule, two virtual machines remains idle. Similarly for third and sixth schedule, one virtual machine remains idle in each case and for each schedule X time is consumed. Fourth schedule also behaves in the same way as first schedule does and this will also consume X time. Fifth schedule behaves like second schedule and it will also take X time to complete the execution of the single task 9. So in this manner resources are not

utilized properly and a total of 6X time is required to complete the whole workflow execution.

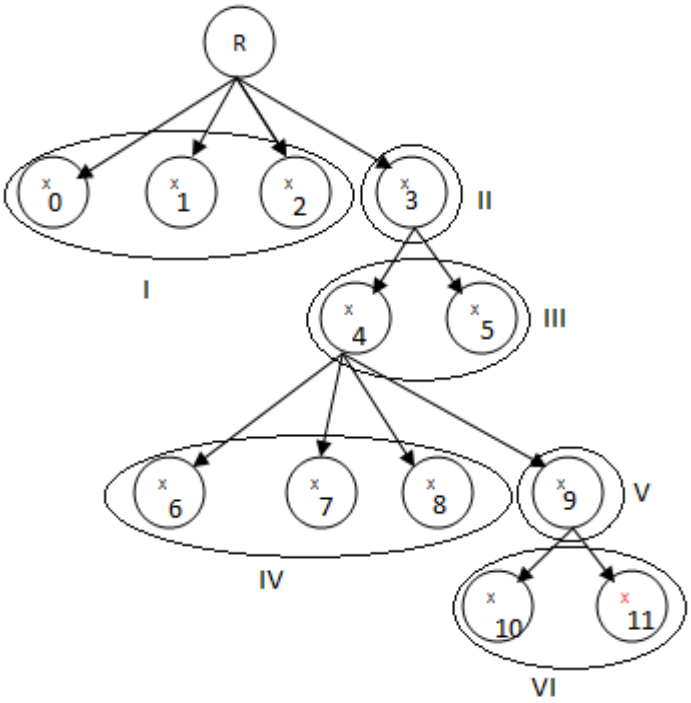


Figure 3.2: Scheduling of similar task with FCFS scheduler

3.3.1 Objective

The main objective of this research work is to schedule different workflows in efficient way so that overall completion time would be minimized. In this work, we propose a new heuristic for dynamic allocation of workflow tasks to current available resources. The objective of this scheduling heuristic is to reduce the available resource which are not utilized properly whenever there is no tasks available for scheduling.

Chapter 4
Proposed Solution

This chapter discusses about how the problem stated in previous chapter can be solved with the help of the proposed workflow scheduling technique. This chapter also describes the design of the proposed technique.

4.1 Proposed Scheduler Description

A new scheduling approach is proposed which is shown in Figure 5.1 named as MaxChild. In this approach, the task which has maximum number of Childs will be scheduled first, so that maximum number of tasks can be available for the next schedules and resource are utilized properly.

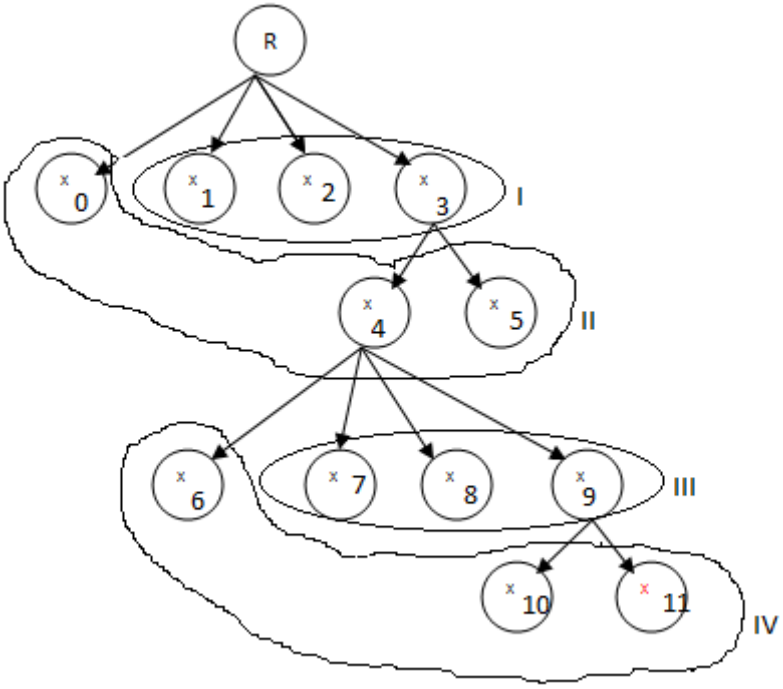


Figure 4.1: Scheduling with MaxChild approach

MaxChild Scheduler will schedule task 3 prior to all the other tasks at level 1on any of three virtual machines along with any 2 tasks from the remaining task of that level (let's consider its considers task 1 and task 2). So X time is consumed for these three tasks of level one. Now at the second schedule, MaxChild scheduler will schedule task 4 and task 5 of level 2, along with the remaining one task (task 0) of level 1. To execute these

three tasks another X time is consumed. Similarly at the third schedule MaxChild scheduler will schedule task 9 prior to all the other tasks of level 3 along with task 8 and task 9. And in the final schedule, it will schedule task 10 and task 11 of level 4 along with the remaining one task of level 3. So in this way resource are utilized properly and the given workflow will be executed completely in 4X time. This execution process is shown in Table 4.1.

Table 4.1: Virtual machine allocation by using MaxChild scheduler

Scheduler Process	Virtual Machine 1	Virtual Machine 2	Virtual Machine 3	Time Consumed
Schedule 1	Task 3	Task 1	Task 2	X
Schedule 2	Task 0	Task 4	Task 5	X
Schedule 3	Task 9	Task 7	Task 8	X
Schedule 4	Task 6	Task 10	Task 11	X

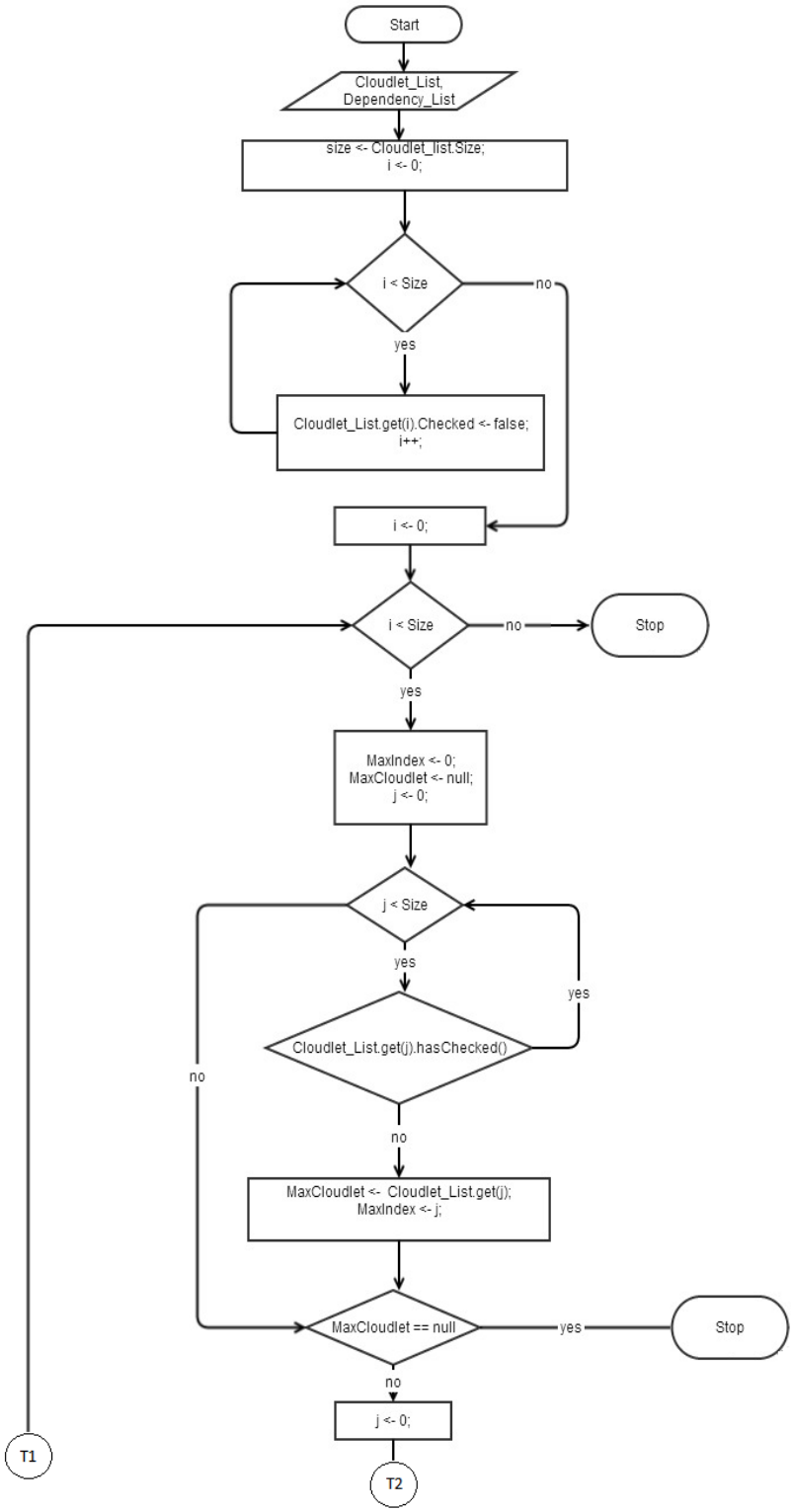
4.2 Design of Proposed Approach

In the proposed scheduling algorithms referred as MaxChild (presented in section 4.3) several notations have been used which are described in the following section:

- Cloudlet_List is defined as the list of all the tasks in workflow.
- Size is a variable which holds the number of tasks in the Cloudlet_List.
- MaxCloudlet is defined as the task having maximum number of Childs
- MaxIndex is defined as the index of MaxCloudlet
- vmList is a defined as the list of all virtual machine.
- VM_ID is defined as virtual machine number.

First of all, the user will enter the Cloudlet_List, Cloudlet dependency list and vmList as input to the scheduler. Then the scheduler stores the size of Cloudlet_List in the Size variable and all the Cloudlets are initially checked as false. Initially MaxIndex is set as 0 and MaxCloudlet is set as null. Now the Cloudlet which is still unchecked is set as the MaxCloudlet and index of this Cloudlet is set as the MaxIndex. Then it will iterate the whole Cloudlet_List to get that Cloudlet which has maximum number of Childs by comparing number of Cloudlet Childs with number of current MaxCloudlet Childs. If MaxCloudlet is null then it break from that pass (means no cloudlet left to schedule). When it gets the Cloudlet with maximum number of Childs then checked value of that Cloudlet is set to true and search for the Idle Virtual Machine. By iterating

over the vmList it gets the id of the Idle VM. if first_IDLE_VM is null then it break from that pass and finally when it gets the idle VM, it schedules MaxCloudlet to that VM. Figure 4.2 represents the flow of activities.



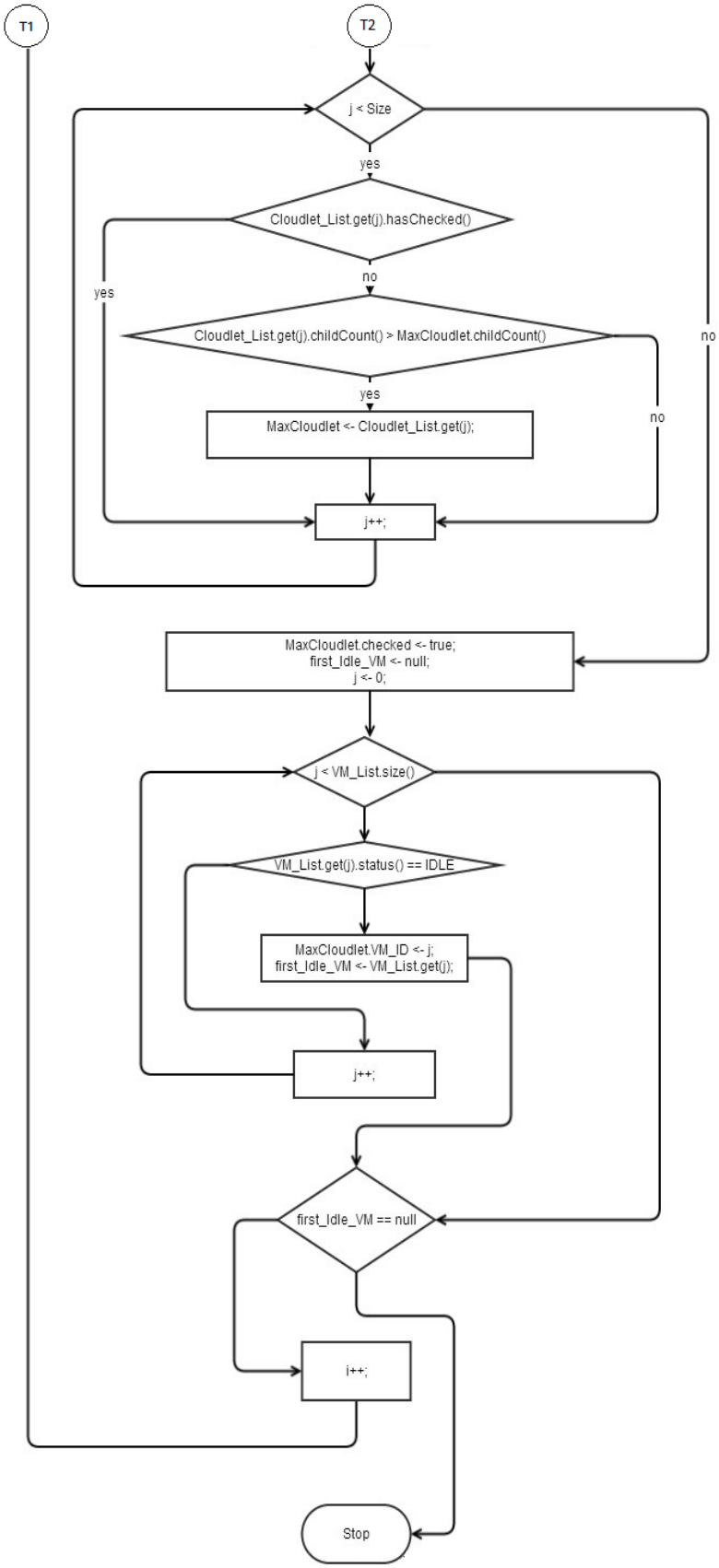


Figure 4.2: Flowchart of the proposed approach

4.3 Proposed Scheduling Algorithm

MaxChild Algorithm
Input : Cloudlet_List, Dependency_List
Output : Scheduled_List

```
1. Set Size ← Cloudlet_List.Size

2. for i ← 0 to Size -1
       Set Coudlet_List.get (i).Checked to false
   end for

3. for i ← 0 to Size -1
        a. Set MaxIndex to Zero
        b. Set MaxCloudlet to null
        c. for j ← 0 to Size -1
              if Cloudlet_List.get(j) is not checked
                 Set MaxCloudlet to Cloudlet_List.get(j)
                 Set MaxIndex to j
                 break
              end if
           end for

        d. if MaxCloudlet is null
              break
           end if

        e. for j ← 0 to Size – 1
                 if Cloudlet_List.get(j) is checked
                    continue
                 end if
                 if Cloudlet_List.get(j).ChildCount >
                 MaxCloudlet.ChildCount
                     Set MaxCloudlet ← Cloudlet_List.get(j)
                 end if
           end for

        f. Set MaxCloudlet_Checked to true
        g. Set first_IDLE_VM to null
        h. for j ← 0 to vmList.Size – 1
                 if vmList.get(j) status is IDLE
                    Set MaxCloudlet.VM_ID to j
                    Set first IDLE_VM to vmList.get(j)
                   break
                 end if
           end for
           if first IDLE_VM is null
                break
end for
```

Chapter 5
Implementation and Experimental Results

This chapter discusses various tools required for Cloud environment setup. Implementation of the proposed workflow scheduling technique has been done on the CloudSim and WorkflowSim simulator by setting up it on Netbeans IDE.

5.1 Tools for Setting the Cloud Environment

Cloud applications have different requirement of configurations and deployment. Scheduling and evaluating the performance of these different applications on real cloud environment is an extremely challenging task. The use of the different service models with the real time infrastructures such as Amazon EC2 or some other IaaS for these application is very challenging. So an alternative is to create a simulation environment that helps to evaluate the performance and scheduling of these applications in an easy way. The major advantages of these simulation environment is

- No cost of infrastructure and services needed to test these applications in a repeatable and controlled environment.
- Evaluating the performance bottleneck of these services in the virtual environment before deploying and performing these on real cloud environment.

These simulation environment evaluates the different kind of resource leasing, on the provider side under the different conditions with different load distribution. This kind of study can't be done on the real cloud, because it require real resources and services for the evaluation of these faults which incurs a major amount of cost. So this study help providers to optimize the cost of resources by focusing on the leas of resource which is identified by these simulators. If such simulation environment is not available then cloud providers and customers have to depend on the theoretical evaluations, or have to depend on some other approaches that leads to errors and provides inefficient results. So these simulation based approach reduces the cost and complexity of the experimental set-up and helps to evaluate the performance before deploying the application on the real cloud infrastructure. Tools required to set up a simulation environment for evaluating the performance of proposed workflow scheduling technique are described as:

5.1.1 WorkflowSim

WorkflowSim [51] extends the existing CloudSim simulator that allows modeling and simulating the environment of cloud, datacenters, virtual machines, and cloudlets by dealing only with single work load but it is not suitable for workflow scheduling as multiple tasks need to be scheduled together. So WorkflowSim provides a higher layer of workflow management over CloudSim layer. WorkflowSim provides the following novel features:

- Considers heterogeneous overheads and failures
- Provides task clustering with the help of workflow clustering engine
- Provides workflow partitioning
- Support for simulation of networks between the system elements

- **WorkflowSim Architecture**

Figure 5.1 represents the WorkflowSim architecture which contains a workflow engine, clustering engine, workflow mapper and workflow scheduler. The components of the workflow can be described as follows:

- **Workflow Mapper**

A workflow mapper, maps the abstract workflow into concrete workflow by importing the DAG files and related information such as size of the workflow tasks and dependency between these tasks, formatted in the XML file. From this information, Workflow mapper creates a workflow tasks lists and maps it with available executable resources.

- **Clustering Engine**

A clustering engine combines the small tasks into large task with the help of clustering to reduce the scheduling overheads. A task can be defined as a program which user want to execute and a job which can contain multiple tasks which can be executed in sequence as well as in parallel.

- **Workflow Engine**

A workflow engine handles the data dependencies between the workflow tasks. A job can only be executed when all of its parent job have successfully completed. So workflow engine schedules the free job only. The example of workflow engine which is used in real cloud environment is DAGMan [52].

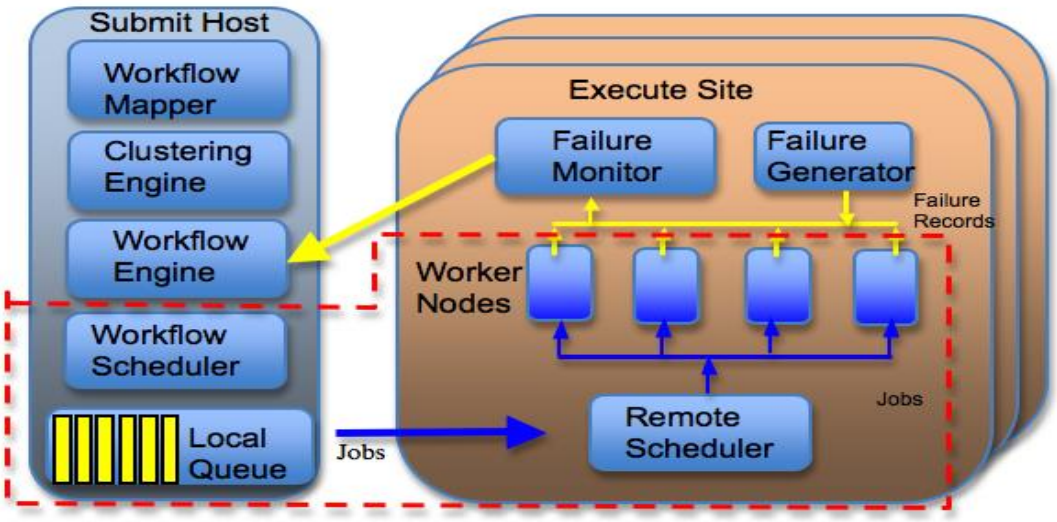


Figure 5.1: WorkflowSim architecture [51]

- **Workflow Scheduler**

Workflow scheduler is used to schedule the jobs to available resource according to the scheduling criteria defined by the user. In case of CloudSim, static scheduling is done during the planning phase, i.e., jobs are scheduled statically before the execution of workflow starts. But WorkflowSim also support the dynamic scheduling in which jobs are scheduled to the remote scheduler when the corresponding resource is idle. The overall working of the WorkflowSim is event-based approach. Figure 5.2 shows how these different components interact with each other.

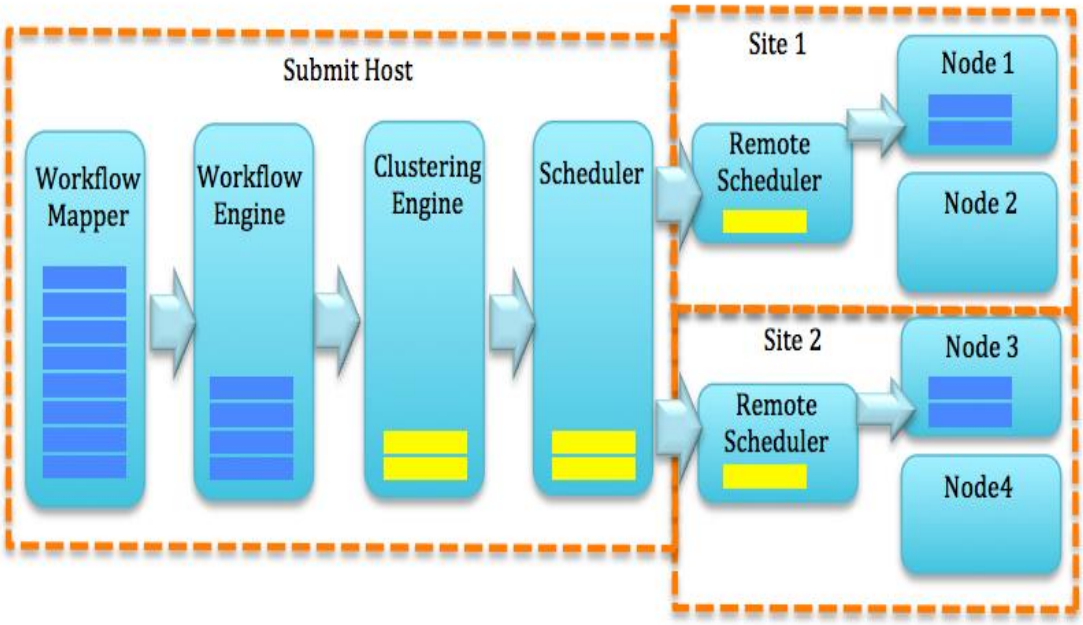


Figure 5.2: Interaction between different components [51]

WorkflowSim handles different layers overheads and failures. These workflow overheads are classified into five categories as:

- **Workflow Engine Delay**

This is the time difference between the completion time of a parent job and submission time of its child job. In case of failures, it is the time difference between the last retry done for the any of parent or child job. Workflow Engine's efficiency is reflected by this delay. Less the delay more is efficiency and vice versa.

- **Queue Delay**

This is the time difference when a job is submitted by the workflow engine to the local queue and the time when it starts execution on remote source. Workflow scheduler's efficiency and availability of resources is reflected by queue delay.

- **Data Transfer Delay**

This is the time delay when the data is transferred between the jobs. It includes Staging data in, clean-up and staging data out. Staging data in, is the time required to transfer the job from the source workflow to the destination resource. Cleanup is the time required to delete the intermediate data which is no longer required by the workflow. Staging data out is the time required to transfer workflow output for storage and analysis.

- **Clustering Delay**

It is the time difference between the individual tasks execution time and the sum of execution time of the tasks combined in the job after clustering. This delay occurs due to consumption of time to group small jobs into a bigger job by using the clustering techniques.

- **Postscript Delay and Prescript Delay**

Prescript delay is the time required to create directories for job execution. Postscript delay is the difference in execution time of a light weight script before and after the completion of a job. It identifies the exit code of the job after the computation has finished.

Other components of the WorkflowSim are workflow partitioner, provenance collector, Layered failures and Job retry. Workflow partitioner is used to divide a large workflow into small sub-workflows. Provenance collector which tracks the history of the execution of workflow tasks. Failures are classified into two categories named as the task failure and job failure. In case of task failure, it is not necessary that others tasks within the jobs also fails because task failure has no effect on the execution of other tasks of the job. But in case of job failure all the tasks within the job fails. In WorkflowSim, two components responses the failure occurred:

- **Failure Generator**

  Failures of task/jobs in the workflow at the time of execution are generated with the help of failure generator component. Failure generator randomly includes tasks/job failure with the help of different kinds of distributions such as Weibull, Gamma, Normal and LogNormal distributions parameters and average failure rate specified by the user.

- **Failure Monitor**

  Failure monitor, identifies the tasks and resource failed with its attributes such as task id, resource id and job id etc. This information is then returned to WMS to handle the fault dynamically by applying the job/task retry fault tolerance method.

There are some other options also which are used to optimize fault tolerance and workflow performance such as reclustering and retry technique. In case of reclustering, task clustering technique is adjusted on the basis of size of the cluster and detected failure rate. Cluster size implies the number of tasks within a job. Reclustering is classified into three categories such as Dynamic Clustering (DC), Selective Reclustering (SR) and Dynamic Reclustering (DR). In case of DC, if the job failure rate is high then the size of the cluster is decreased. In case of SR, failed task is selected and combined with a new job for retry. DR is combination of both DC and SR and in this case, failed task is selected as well as the size of the cluster is decreased. Workflow Engine supports the functionality of reclustering.

Retry technique have further two options which are supported by Workflow Scheduler. User can either retry the whole job or retry only the failed task.

5.1.2 Netbeans

Netbeans [53] is a multi-language software development environment tool which supports many languages such as C/C++, Java, HTML, Java Script, JSP and PHP etc. The Netbeans Standard Document Kit (SDK) which include the java development tool for the developer of java. Its editor is extensible so we can plug-in support for many other languages. The Netbeans platform provide various built in library classes and users don't need to code from the scratch. WorkflowSim Simulation toolkit is developed on Java Platform, so Netbeans SDK is used to create simulated Cloud Environment.

5.2 Implementation Details

The proposed workflow scheduling approach has been implemented in WorkflowSim toolkit by programmatically extending the core framework provided in the WorkflowSim. The basic configurations of the virtual machines and parameters considered are given shown in Figure 5.3.

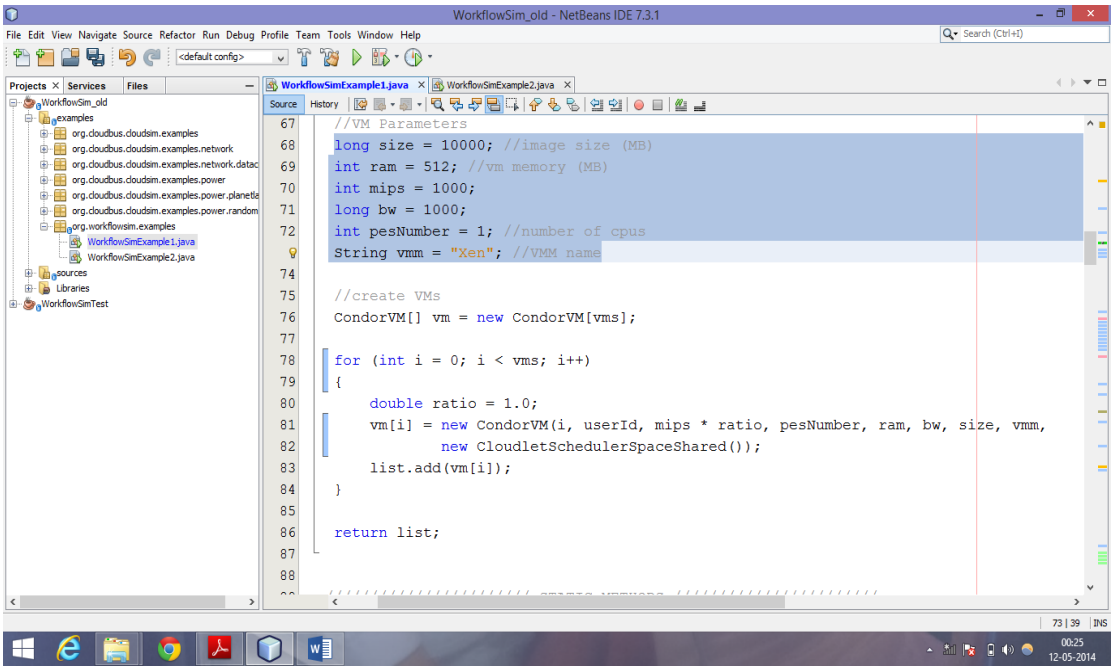


Figure 5.3: Creation of the Virtual Machines

Virtual machines can be created in the space shared mode and time shared mode. Initially less number of virtual machines are created for the execution of small workflow and number of virtual machine can be changed dynamically according to the execution of large workflows. According to the proposed solution described, only three

virtual machines are created along with the creation of the workflow planner, workflow engine and data center as shown in the Figure 5.4.

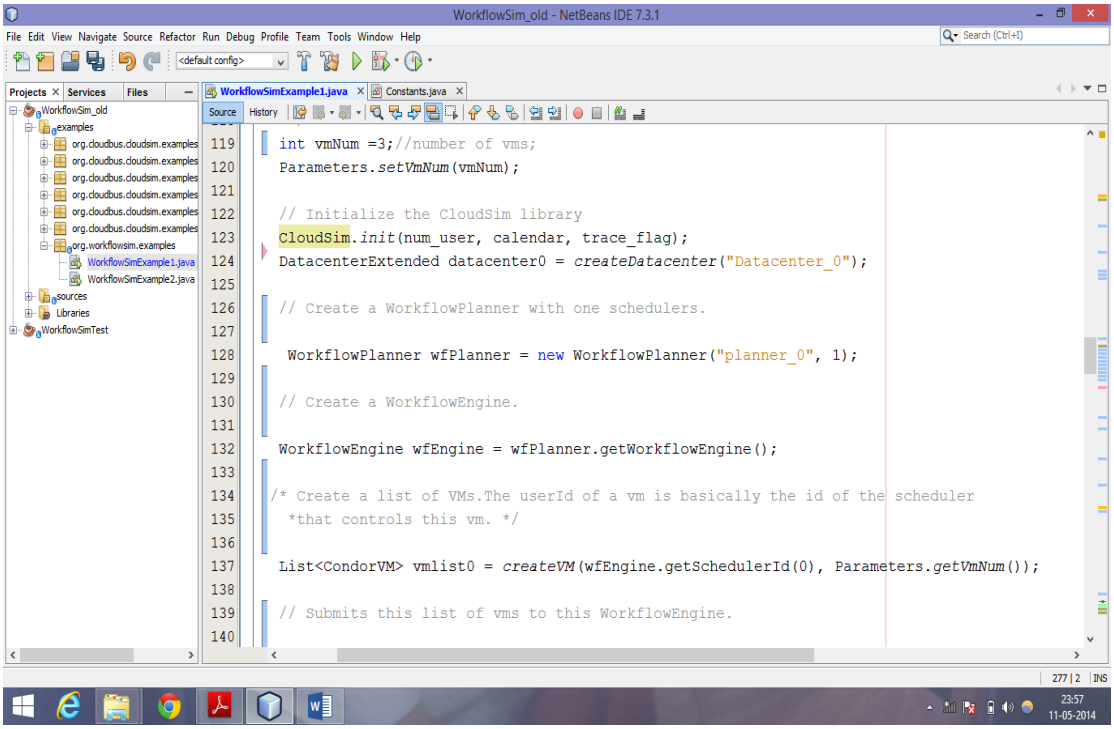


Figure 5.4: Creation of workflow planner and workflow engine

All of these three virtual machines are binded with the workflow engine and workflow scheduler as shown in the Figure 5.5.

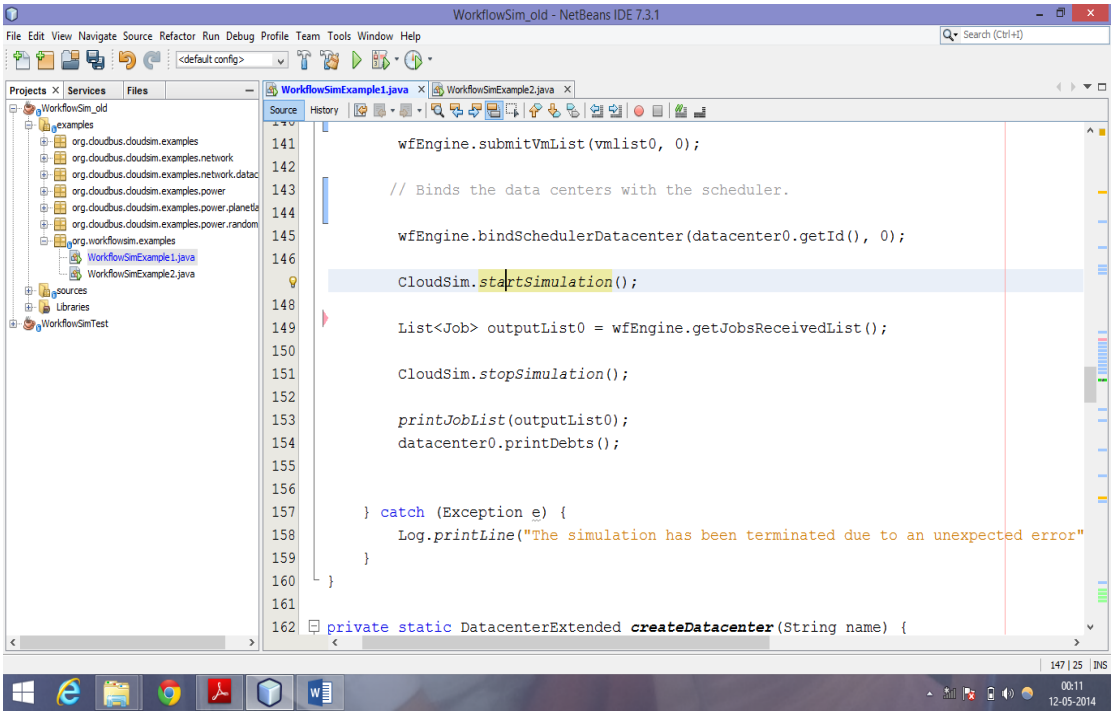


Figure 5.5: Binding of virtual machine to workflow engine and workflow scheduler

A basic machine contains one or more than one CPUs. Next step is to create a data center which contains 4 cores/CPUs for the quad core machine. The configurations of the machines and the characteristics of data center along with its architecture, machines details and allocations policies is shown in the Figure 5.6 and Figure 5.7 respectively.

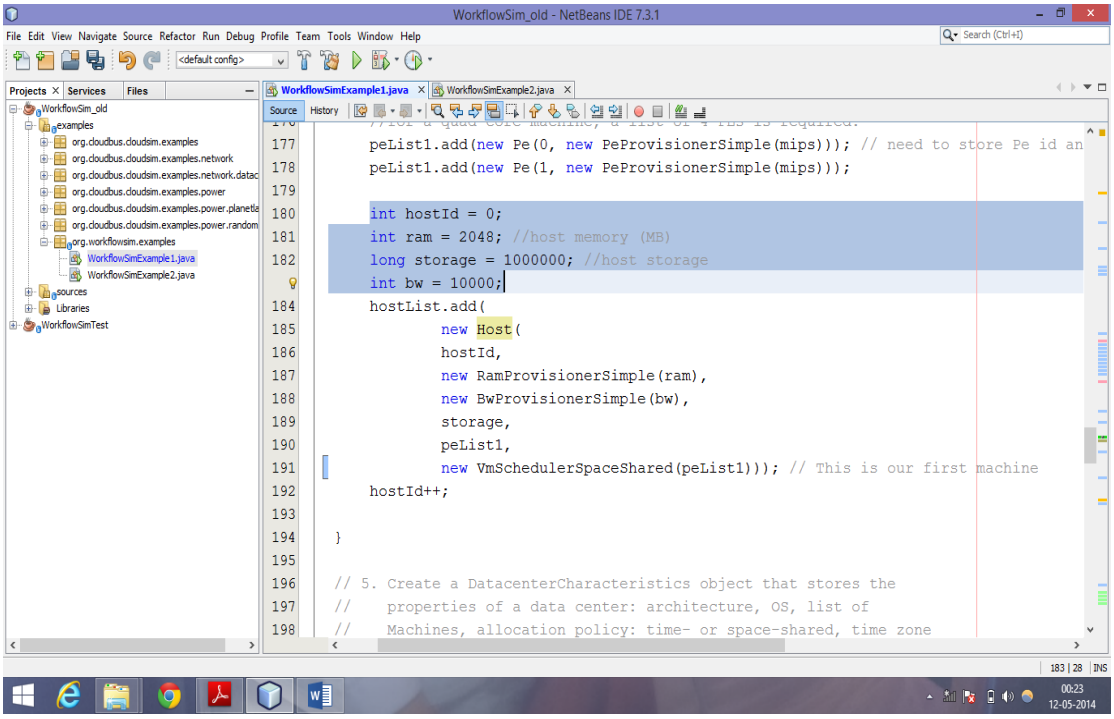


Figure 5.6: Virtual machine configuration

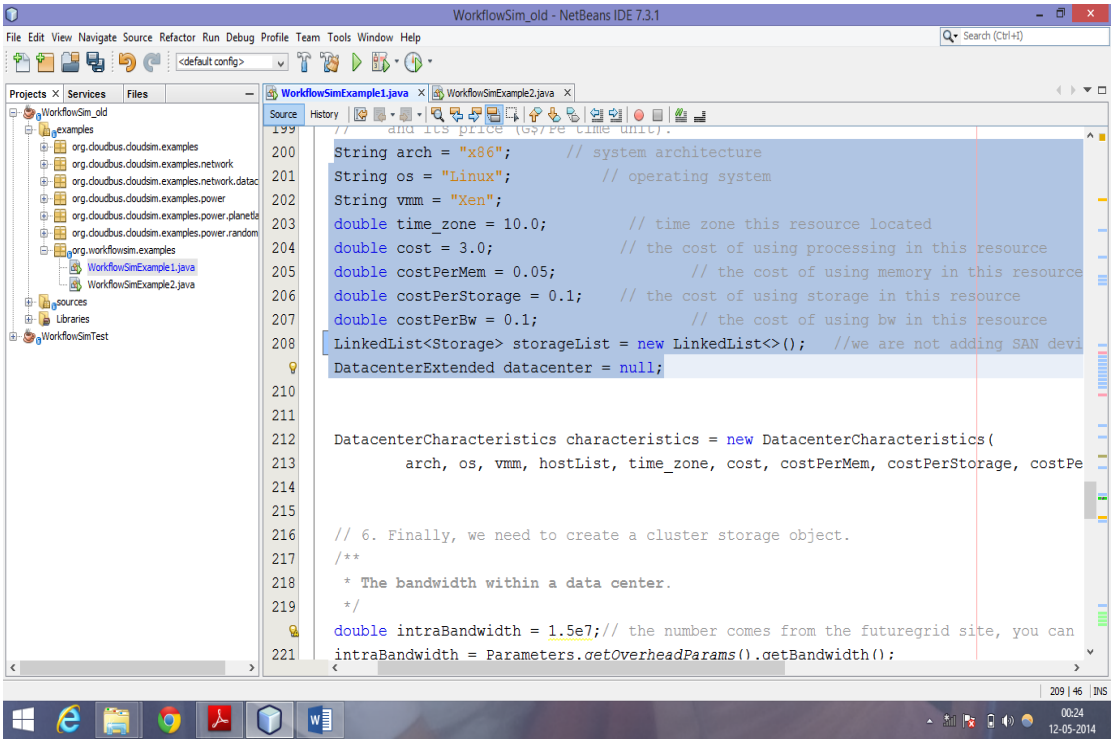


Figure 5.7: Data center architecture and machine details

Next step is to create the objects of the scheduler and objects of clustering technique selected. The desired scheduler and clustering technique can be selected with the help of switch case as shown in the Figure 5.8 and Figure 5.9 respectively.

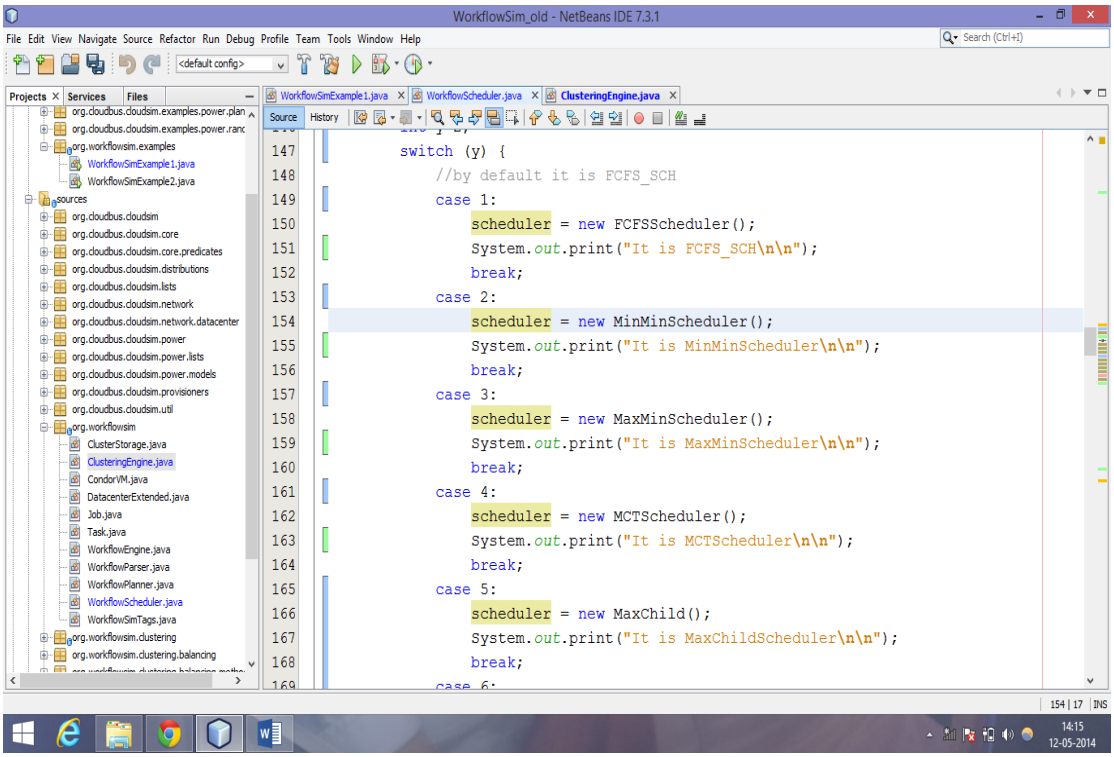


Figure 5.8: Creation of the objects of all the scheduler

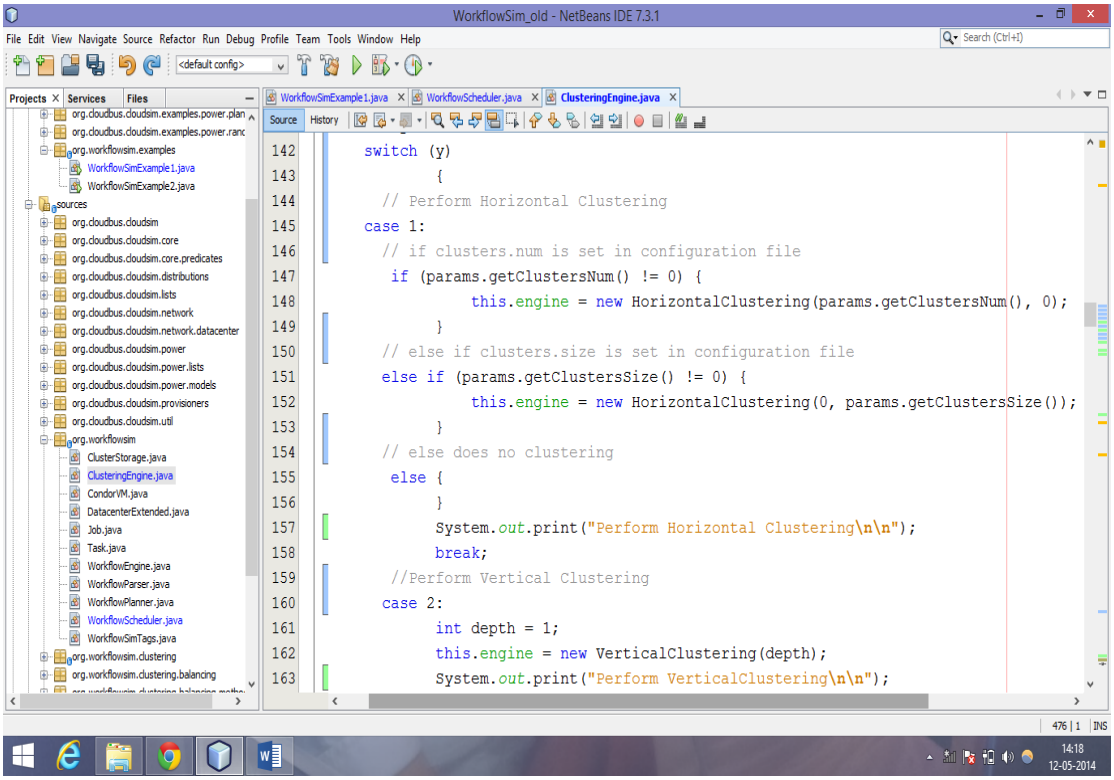


Figure 5.9: Creation of objects of clustering techniques

5.3 Experimental Results

The experimental results have been compared using existing scheduling algotithms such as FCFS, Min-min, Max-min and MCT with the proposed MaxChild algorithm for workflows of different size and shape.

Case 1: Workflow of 12 tasks and 3 virtual machines

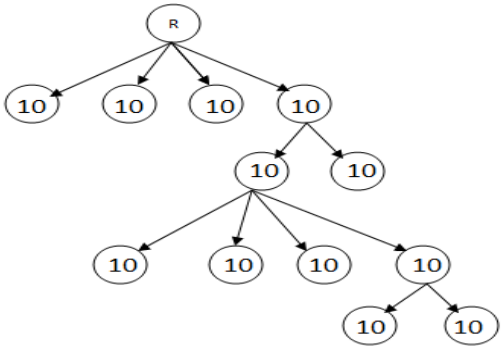


Figure 5.10: Workflow with all the tasks having execution time 10 units

- **FCFS**

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID        Time    Start Time    Finish Time    Depth
    12         SUCCESS         2             0         0.11       0.1          0.21          0
    0          SUCCESS         2             0          10        0.21         10.21          1
    9          SUCCESS         2             1          10        0.21         10.21          1
    10          SUCCESS         2             2          10        0.21         10.21          1
    11          SUCCESS         2             0          10        10.21         20.21          1
    1          SUCCESS         2             0          10        20.21         30.21          2
    2          SUCCESS         2             1          10        20.21         30.21          2
    3          SUCCESS         2             0          10        30.21         40.21          3
    6          SUCCESS         2             1          10        30.21         40.21          3
    7          SUCCESS         2             2          10        30.21         40.21          3
    8          SUCCESS         2             0          10        40.21         50.21          3
    4          SUCCESS         2             0          10        50.21         60.21          4
    5          SUCCESS         2             1          10        50.21         60.21          4
*****Datacenter: Datacenter_0*****
User id         Debt
6               3076.8
**********************************
```

Figure 5.11: FCFS scheduler output

- **Min-min**

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID        Time    Start Time    Finish Time    Depth
    12         SUCCESS         2             0         0.11       0.1          0.21          0
    0          SUCCESS         2             0          10        0.21         10.21          1
    9          SUCCESS         2             1          10        0.21         10.21          1
    10          SUCCESS         2             2          10        0.21         10.21          1
    11          SUCCESS         2             0          10        10.21         20.21          1
    1          SUCCESS         2             0          10        20.21         30.21          2
    2          SUCCESS         2             1          10        20.21         30.21          2
    3          SUCCESS         2             0          10        30.21         40.21          3
    6          SUCCESS         2             1          10        30.21         40.21          3
    7          SUCCESS         2             2          10        30.21         40.21          3
    8          SUCCESS         2             0          10        40.21         50.21          3
    4          SUCCESS         2             0          10        50.21         60.21          4
    5          SUCCESS         2             1          10        50.21         60.21          4
*****Datacenter: Datacenter_0*****
User id         Debt
6               3076.8
**********************************
```

Figure 5.12: Min-min scheduler output

- **Max-min**

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID        Time    Start Time    Finish Time    Depth
    12         SUCCESS        2             0           0.11       0.1           0.21           0
    0         SUCCESS         2             0           10         0.21          10.21          1
    9         SUCCESS         2             1           10         0.21          10.21          1
    10         SUCCESS         2             2           10         0.21          10.21          1
    11         SUCCESS         2             0           10         10.21         20.21          1
    1         SUCCESS         2             0           10         20.21         30.21          2
    2         SUCCESS         2             1           10         20.21         30.21          2
    3         SUCCESS         2             0           10         30.21         40.21          3
    6         SUCCESS         2             1           10         30.21         40.21          3
    7         SUCCESS         2             2           10         30.21         40.21          3
    8         SUCCESS         2             0           10         40.21         50.21          3
    4         SUCCESS         2             0           10         50.21         60.21          4
    5         SUCCESS         2             1           10         50.21         60.21          4
*****Datacenter: Datacenter_0*****
User id         Debt
6               3076.8
***********************************
```

Figure 5.13: Max-min scheduler output

- **MCT**

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID        Time    Start Time    Finish Time    Depth
    12         SUCCESS        2             0           0.11       0.1           0.21           0
    0         SUCCESS         2             0           10         0.21          10.21          1
    9         SUCCESS         2             1           10         0.21          10.21          1
    10         SUCCESS         2             2           10         0.21          10.21          1
    11         SUCCESS         2             0           10         10.21         20.21          1
    1         SUCCESS         2             0           10         20.21         30.21          2
    2         SUCCESS         2             1           10         20.21         30.21          2
    3         SUCCESS         2             0           10         30.21         40.21          3
    6         SUCCESS         2             1           10         30.21         40.21          3
    7         SUCCESS         2             2           10         30.21         40.21          3
    8         SUCCESS         2             0           10         40.21         50.21          3
    4         SUCCESS         2             0           10         50.21         60.21          4
    5         SUCCESS         2             1           10         50.21         60.21          4
*****Datacenter: Datacenter_0*****
User id         Debt
6               3076.8
***********************************
```

Figure 5.14: MCT scheduler output

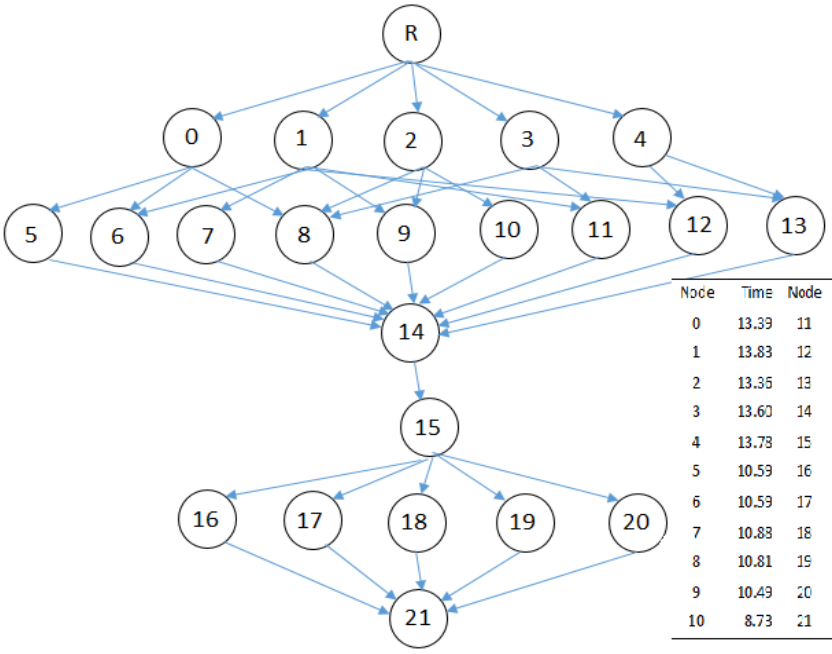
- **MaxChild**

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID        Time    Start Time    Finish Time    Depth
    12         SUCCESS        2             0           0.11       0.1           0.21           0
    0         SUCCESS         2             0           10         0.21          10.21          1
    9         SUCCESS         2             1           10         0.21          10.21          1
    10         SUCCESS         2             2           10         0.21          10.21          1
    11         SUCCESS         2             0           10         10.21         20.21          1
    1         SUCCESS         2             1           10         10.21         20.21          2
    2         SUCCESS         2             2           10         10.21         20.21          2
    3         SUCCESS         2             0           10         20.21         30.21          3
    6         SUCCESS         2             1           10         20.21         30.21          3
    7         SUCCESS         2             2           10         20.21         30.21          3
    8         SUCCESS         2             0           10         30.21         40.21          3
    4         SUCCESS         2             1           10         30.21         40.21          4
    5         SUCCESS         2             2           10         30.21         40.21          4
*****Datacenter: Datacenter_0*****
User id         Debt
6               3076.8
***********************************
```

Figure 5.15: MaxChild scheduler output

Case 2: Workflow of 21 tasks and 3 virtual machines

The nodes in the workflow are represented with node number. Execution time of corresponding node is as shown in the attached index.



Node	Time	Node	Time
0	13.39	11	10.51
1	13.83	12	10.62
2	13.35	13	10.37
3	13.60	14	1.42
4	13.78	15	10.39
5	10.59	16	10.64
6	10.59	17	10.64
7	10.88	18	10.83
8	10.81	19	10.93
9	10.49	20	10.76
10	8.73	21	8.73

Figure 5.16: Workflow with 21 tasks

- **FCFS**

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID        Time    Start Time    Finish Time    Depth
    21        SUCCESS        2            0           0.11        0.1           0.21           0
    13        SUCCESS        2            2           13.38       0.21          13.59          1
    10        SUCCESS        2            1           13.62       0.21          13.83          1
    0         SUCCESS        2            0           13.8        0.21          14.01          1
    14        SUCCESS        2            2           10.51       14.01         24.52          2
    20        SUCCESS        2            1           13.41       14.01         27.42          1
    17        SUCCESS        2            0           13.85       14.01         27.86          1
    3         SUCCESS        2            0           10.41       27.86         38.27          2
    16        SUCCESS        2            1           10.85       27.86         38.71          2
    11        SUCCESS        2            1           10.55       38.71         49.26          2
    9         SUCCESS        2            0           10.62       38.71         49.33          2
    12        SUCCESS        2            2           10.96       38.71         49.68          2
    15        SUCCESS        2            0           10.53       49.68         60.21          2
    18        SUCCESS        2            1           10.59       49.68         60.27          2
    19        SUCCESS        2            2           10.63       49.68         60.31          2
    4         SUCCESS        2            0           2.14        60.31         62.45          4
    1         SUCCESS        2            0           10.76       62.45         73.21          5
    6         SUCCESS        2            2           10.83       62.45         73.28          5
    5         SUCCESS        2            1           10.93       62.45         73.38          5
    8         SUCCESS        2            1           10.39       73.38         83.77          5
    7         SUCCESS        2            0           10.64       73.38         84.02          5
    2         SUCCESS        2            0           8.86        84.02         92.87          9
*****Datacenter: Datacenter_0*****
User id         Debt
6               3076.8
**********************************
```

Figure 5.17: FCFS scheduler Output

- **Min-min**

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID        Time    Start Time    Finish Time    Depth
    21         SUCCESS        2             0          0.11        0.1           0.21           0
    13         SUCCESS        2             0          13.38       0.21          13.59          1
    20         SUCCESS        2             1          13.41       0.21          13.62          1
    10         SUCCESS        2             2          13.62       0.21          13.83          1
    14         SUCCESS        2             0          10.51       13.83         24.34          2
    16         SUCCESS        2             1          10.85       13.83         24.68          2
    0         SUCCESS        2             2          13.8        13.83         27.63          1
    17         SUCCESS        2             0          13.85       27.63         41.48          1
    3         SUCCESS        2             0          10.45       41.48         51.94          2
    15         SUCCESS        2             0          10.49       51.94         62.43          2
    11         SUCCESS        2             1          10.59       51.94         62.53          2
    18         SUCCESS        2             2          10.67       51.94         62.61          2
    19         SUCCESS        2             0          10.63       62.61         73.24          2
    9         SUCCESS        2             1          10.66       62.61         73.27          2
    12         SUCCESS        2             2          10.88       62.61         73.49          2
    4         SUCCESS        2             0          2.14        73.49         75.63          4
    8         SUCCESS        2             0          10.39       75.63         86.02          5
    7         SUCCESS        2             1          10.64       75.63         86.27          5
    1         SUCCESS        2             2          10.76       75.63         86.39          5
    6         SUCCESS        2             0          10.83       86.39         97.22          5
    5         SUCCESS        2             1          10.93       86.39         97.32          5
    2         SUCCESS        2             0          8.85        97.32         106.17         9
*****Datacenter: Datacenter_0*****
User id         Debt
6               3076.8
*********************************
```

Figure 5.18: Min-min scheduler Output

- **Max-min**

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID        Time    Start Time    Finish Time    Depth
    21         SUCCESS        2             0          0.11        0.1           0.21           0
    10         SUCCESS        2             2          13.62       0.21          13.83          1
    0         SUCCESS        2             1          13.8        0.21          14.01          1
    17         SUCCESS        2             0          13.85       0.21          14.06          1
    3         SUCCESS        2             2          10.41       14.06         24.47          2
    13         SUCCESS        2             1          13.38       14.06         27.44          1
    20         SUCCESS        2             0          13.41       14.06         27.47          1
    11         SUCCESS        2             2          10.55       27.47         38.02          2
    9         SUCCESS        2             1          10.66       27.47         38.13          2
    12         SUCCESS        2             0          10.92       27.47         38.39          2
    18         SUCCESS        2             1          10.63       38.39         49.02          2
    19         SUCCESS        2             2          10.63       38.39         49.02          2
    16         SUCCESS        2             0          10.85       38.39         49.24          2
    15         SUCCESS        2             1          10.49       49.24         59.73          2
    14         SUCCESS        2             0          10.51       49.24         59.75          2
    4         SUCCESS        2             0          2.14        59.75         61.89          4
    1         SUCCESS        2             2          10.76       61.89         72.65          5
    6         SUCCESS        2             1          10.83       61.89         72.72          5
    5         SUCCESS        2             0          10.93       61.89         72.82          5
    8         SUCCESS        2             1          10.39       72.82         83.21          5
    7         SUCCESS        2             0          10.64       72.82         83.46          5
    2         SUCCESS        2             0          8.86        83.46         92.32          9
*****Datacenter: Datacenter_0*****
User id         Debt
6               3076.8
*********************************
```

Figure 5.19: Max-min scheduler Output

- **MCT**

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID        Time    Start Time    Finish Time    Depth
    21        SUCCESS        2             0          0.11        0.1           0.21           0
    13        SUCCESS        2             2          13.38       0.21          13.59          1
    10        SUCCESS        2             1          13.62       0.21          13.83          1
    0         SUCCESS        2             0          13.8        0.21          14.01          1
    14        SUCCESS        2             2          10.51       14.01         24.52          2
    20        SUCCESS        2             1          13.41       14.01         27.42          1
    17        SUCCESS        2             0          13.85       14.01         27.86          1
    3         SUCCESS        2             0          10.41       27.86         38.27          2
    16        SUCCESS        2             1          10.85       27.86         38.71          2
    11        SUCCESS        2             1          10.55       38.71         49.26          2
    9         SUCCESS        2             0          10.62       38.71         49.33          2
    12        SUCCESS        2             2          10.96       38.71         49.68          2
    15        SUCCESS        2             0          10.53       49.68         60.21          2
    18        SUCCESS        2             1          10.59       49.68         60.27          2
    19        SUCCESS        2             2          10.63       49.68         60.31          2
    4         SUCCESS        2             0          2.14        60.31         62.45          4
    1         SUCCESS        2             0          10.76       62.45         73.21          5
    6         SUCCESS        2             2          10.83       62.45         73.28          5
    5         SUCCESS        2             1          10.93       62.45         73.38          5
    8         SUCCESS        2             1          10.39       73.38         83.77          5
    7         SUCCESS        2             0          10.64       73.38         84.02          5
    2         SUCCESS        2             0          8.86        84.02         92.87          9
*****Datacenter: Datacenter_0*****
User id         Debt
6               3076.8
************************************
```

Figure 5.20: MCT scheduler Output

- **MaxChild**

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID        Time    Start Time    Finish Time    Depth
    21        SUCCESS        2             0          0.11        0.1           0.21           0
    13        SUCCESS        2             2          13.38       0.21          13.59          1
    10        SUCCESS        2             1          13.62       0.21          13.83          1
    0         SUCCESS        2             0          13.8        0.21          14.01          1
    14        SUCCESS        2             0          10.55       14.01         24.56          2
    20        SUCCESS        2             1          13.41       13.83         27.24          1
    17        SUCCESS        2             2          13.85       13.59         27.44          1
    3         SUCCESS        2             0          10.41       24.56         34.97          2
    16        SUCCESS        2             1          10.85       27.24         38.09          2
    9         SUCCESS        2             2          10.66       27.44         38.1           2
    11        SUCCESS        2             0          10.55       34.97         45.52          2
    15        SUCCESS        2             2          10.49       38.1          48.59          2
    12        SUCCESS        2             1          10.92       38.09         49.01          2
    18        SUCCESS        2             0          10.63       45.52         56.15          2
    19        SUCCESS        2             2          10.63       48.59         59.22          2
    4         SUCCESS        2             0          2.14        59.22         61.36          4
    1         SUCCESS        2             0          10.76       61.36         72.12          5
    6         SUCCESS        2             2          10.83       61.36         72.19          5
    5         SUCCESS        2             1          10.93       61.36         72.29          5
    8         SUCCESS        2             2          10.39       72.19         82.58          5
    7         SUCCESS        2             0          10.64       72.12         82.76          5
    2         SUCCESS        2             0          8.86        82.76         91.62          9
*****Datacenter: Datacenter_0*****
User id         Debt
6               3076.8
************************************
```

Figure 5.21: MaxChild scheduler Output

Case 3: Workflow of 21 tasks and 3 virtual machines with fault tolerance

The previous example workflow discussed in Case 2 has also been implemented using retry fault tolerance technique. It can be infer from experimental results that MaxChild has maximum resource utilization.

- FCFS

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID        Time    Start Time    Finish Time    Depth
    21         SUCCESS         2             0         0.11        0.1           0.21          0
    13         SUCCESS         2             2        13.36       0.21          13.57          1
    10         SUCCESS         2             1         13.6       0.21          13.81          1
    0          SUCCESS         2             0        13.78       0.21          13.99          1
    14         SUCCESS         2             2        10.51       13.99          24.5          2
    20         FAILED          2             1        13.39       13.99         27.38          1
    17         FAILED          2             0        13.83       13.99         27.82          1
    3          SUCCESS         2             0        10.37       27.82         38.19          2
    22         SUCCESS         2             1        13.39       27.82         41.21          1
    23         SUCCESS         2             0        13.83       41.21         55.04          1
    16         SUCCESS         2             0        10.81       55.04         65.85          2
    11         SUCCESS         2             1        10.51       65.85         76.36          2
    9          SUCCESS         2             0        10.62       65.85         76.47          2
    12         SUCCESS         2             2        10.88       65.85         76.73          2
    15         SUCCESS         2             0        10.49       76.73         87.22          2
    18         SUCCESS         2             1        10.59       76.73         87.32          2
    19         SUCCESS         2             2        10.59       76.73         87.32          2
    4          SUCCESS         2             0         2.14       87.32         89.46          4
    1          FAILED          2             0        10.76       89.46        100.22          5
    6          FAILED          2             2        10.83       89.46        100.29          5
    5          FAILED          2             1        10.93       89.46        100.39          5
    8          SUCCESS         2             1        10.39      100.39        110.78          5
    7          SUCCESS         2             0        10.64      100.39        111.03          5
    24         SUCCESS         2             2        10.76      100.39        111.15          5
    25         SUCCESS         2             0        10.83      111.15        121.98          5
    26         SUCCESS         2             1        10.93      111.15        122.08          5
    2          SUCCESS         2             0         8.73      122.08        130.81          9
BUILD SUCCESSFUL (total time: 0 seconds)
```

Figure 5.22: Output FCFS scheduler with failure

- Min-min

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID        Time    Start Time    Finish Time    Depth
    21         SUCCESS         2             0         0.11        0.1           0.21          0
    13         SUCCESS         2             0        13.36       0.21          13.57          1
    20         SUCCESS         2             1        13.39       0.21          13.6           1
    10         SUCCESS         2             2         13.6       0.21          13.81          1
    14         SUCCESS         2             0        10.51       13.81         24.32          2
    16         SUCCESS         2             1        10.81       13.81         24.62          2
    0          SUCCESS         2             2        13.78       13.81         27.59          1
    17         SUCCESS         2             0        13.83       27.59         41.42          1
    3          SUCCESS         2             0        10.37       41.42         51.79          2
    15         FAILED          2             0        10.49       51.79         62.28          2
    11         FAILED          2             1        10.51       51.79          62.3          2
    18         FAILED          2             2        10.59       51.79         62.38          2
    22         SUCCESS         2             0        10.49       62.38         72.87          2
    23         SUCCESS         2             1        10.51       62.38         72.89          2
    19         SUCCESS         2             2        10.59       62.38         72.97          2
    24         SUCCESS         2             0        10.59       72.97         83.56          2
    9          SUCCESS         2             1        10.62       72.97         83.59          2
    12         SUCCESS         2             2        10.88       72.97         83.85          2
    4          SUCCESS         2             0         2.14       83.85         85.99          4
    8          SUCCESS         2             0        10.39       85.99         96.38          5
    7          SUCCESS         2             1        10.64       85.99         96.63          5
    1          SUCCESS         2             2        10.76       85.99         96.75          5
    6          SUCCESS         2             0        10.83       96.75        107.58          5
    5          SUCCESS         2             1        10.93       96.75        107.68          5
    2          FAILED          2             0         8.73      107.68        116.41          9
    25         SUCCESS         2             0         8.73      116.41        125.14          9
```

Figure 5.23: Output Min-min scheduler with failure

- **Max-min**

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID    Time    Start Time    Finish Time    Depth
    21         SUCCESS         2             0      0.11       0.1          0.21          0
    10         SUCCESS         2             2      13.6       0.21         13.81         1
    0          SUCCESS         2             1      13.78      0.21         13.99         1
    17         SUCCESS         2             0      13.83      0.21         14.04         1
    3          SUCCESS         2             2      10.37      14.04        24.41         2
    13         SUCCESS         2             1      13.36      14.04        27.4          1
    20         SUCCESS         2             0      13.39      14.04        27.43         1
    11         SUCCESS         2             2      10.51      27.43        37.94         2
    9          SUCCESS         2             1      10.62      27.43        38.05         2
    12         SUCCESS         2             0      10.88      27.43        38.31         2
    18         SUCCESS         2             1      10.59      38.31        48.9          2
    19         SUCCESS         2             2      10.59      38.31        48.9          2
    16         SUCCESS         2             0      10.81      38.31        49.12         2
    15         FAILED          2             1      10.49      49.12        59.61         2
    14         FAILED          2             0      10.51      49.12        59.63         2
    22         FAILED          2             0      10.49      59.63        70.12         2
    23         SUCCESS         2             0      10.51      70.12        80.63         2
    24         FAILED          2             0      10.49      80.63        91.12         2
    25         SUCCESS         2             0      10.49      91.12        101.61        2
    4          SUCCESS         2             0      2.14       101.61       103.75        4
    1          SUCCESS         2             2      10.76      103.75       114.51        5
    6          SUCCESS         2             1      10.83      103.75       114.58        5
    5          SUCCESS         2             0      10.93      103.75       114.68        5
    8          SUCCESS         2             1      10.39      114.68       125.07        5
    7          SUCCESS         2             0      10.64      114.68       125.32        5
    2          SUCCESS         2             0      8.73       125.32       134.05        9
BUILD SUCCESSFUL (total time: 0 seconds)
```

Figure 5.24: Output Max-min scheduler with failure

- **MCT**

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID    Time    Start Time    Finish Time    Depth
    21         SUCCESS         2             0      0.11       0.1          0.21          0
    13         SUCCESS         2             2      13.36      0.21         13.57         1
    10         SUCCESS         2             1      13.6       0.21         13.81         1
    0          SUCCESS         2             0      13.78      0.21         13.99         1
    14         SUCCESS         2             2      10.51      13.99        24.5          2
    20         SUCCESS         2             1      13.39      13.99        27.38         1
    17         SUCCESS         2             0      13.83      13.99        27.82         1
    3          SUCCESS         2             0      10.37      27.82        38.19         2
    16         SUCCESS         2             1      10.81      27.82        38.63         2
    11         FAILED          2             1      10.51      38.63        49.14         2
    9          FAILED          2             0      10.62      38.63        49.25         2
    12         FAILED          2             2      10.88      38.63        49.51         2
    15         FAILED          2             0      10.49      49.51        60            2
    18         FAILED          2             1      10.59      49.51        60.1          2
    19         FAILED          2             2      10.59      49.51        60.1          2
    22         SUCCESS         2             0      10.51      60.1         70.61         2
    23         SUCCESS         2             1      10.62      60.1         70.72         2
    24         SUCCESS         2             2      10.88      60.1         70.98         2
    25         SUCCESS         2             0      10.49      70.98        81.47         2
    26         SUCCESS         2             1      10.59      70.98        81.57         2
    27         SUCCESS         2             2      10.59      70.98        81.57         2
    4          SUCCESS         2             0      2.14       81.57        83.71         4
    1          SUCCESS         2             0      10.76      83.71        94.47         5
    6          SUCCESS         2             2      10.83      83.71        94.54         5
    5          SUCCESS         2             1      10.93      83.71        94.64         5
    8          SUCCESS         2             1      10.39      94.64        105.03        5
    7          SUCCESS         2             0      10.64      94.64        105.28        5
    2          SUCCESS         2             0      8.73       105.28       114.01        9
BUILD SUCCESSFUL (total time: 0 seconds)
```

Figure 5.25: Output MCT scheduler with failure

- **MaxChild**

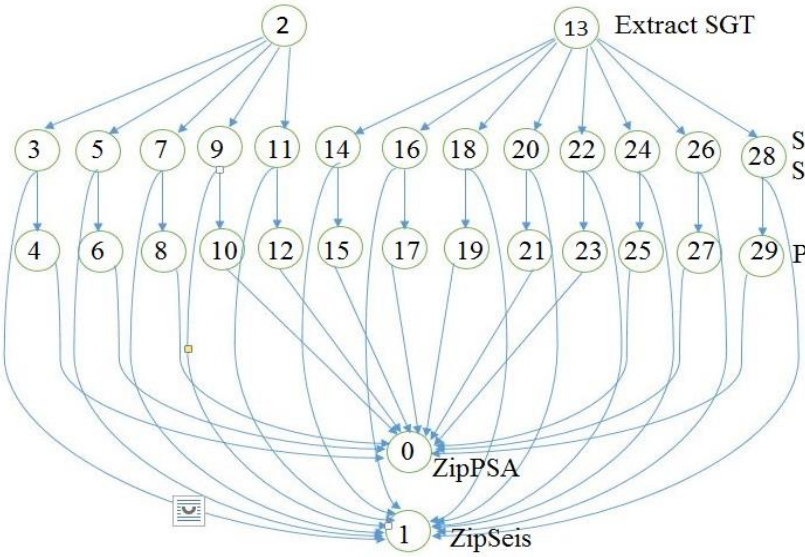
```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID    Time    Start Time    Finish Time    Depth
    21         SUCCESS        2             0       0.11       0.1          0.21          0
    13         FAILED         2             2      13.36      0.21         13.57          1
    10         FAILED         2             1       13.6      0.21         13.81          1
    0          FAILED         2             0      13.78      0.21         13.99          1
    20         SUCCESS        2             1      13.39     13.81          27.2          1
    22         SUCCESS        2             0      13.36     13.99         27.35          1
    17         SUCCESS        2             2      13.83     13.57          27.4          1
    14         SUCCESS        2             2      10.51      27.4         37.91          2
    23         SUCCESS        2             1       13.6      27.2          40.8          1
    24         SUCCESS        2             0      13.78     27.35         41.13          1
    16         SUCCESS        2             2      10.81     37.91         48.72          2
    15         SUCCESS        2             1      10.49      40.8         51.29          2
    18         SUCCESS        2             0      10.59     41.13         51.72          2
    19         SUCCESS        2             2      10.59     48.72         59.31          2
    11         SUCCESS        2             1      10.51     51.29          61.8          2
    12         SUCCESS        2             0      10.88     51.72          62.6          2
    3          SUCCESS        2             2      10.37     59.31         69.68          2
    9          SUCCESS        2             1      10.62      61.8         72.42          2
    4          SUCCESS        2             0       2.14     72.42         74.56          4
    1          SUCCESS        2             0      10.76     74.56         85.32          5
    6          SUCCESS        2             2      10.83     74.56         85.39          5
    5          SUCCESS        2             1      10.93     74.56         85.49          5
    8          SUCCESS        2             2      10.39     85.39         95.78          5
    7          SUCCESS        2             0      10.64     85.32         95.96          5
    2          SUCCESS        2             0       8.73     95.96        104.69          9
BUILD SUCCESSFUL (total time: 0 seconds)
```

Figure 5.26: Output MaxChild scheduler with failure

Case 4: Cybershake workflow of 30 tasks and 3 virtual machines

Cybershake workflow [54] has been developed by Southern California Earthquake Center (SECE) which is used to identify earthquake hazards by identifying the earthquake raptures having moment magnitude value greater than 6. It uses Probabilistic Seismic Hazard Analysis (PSHA) technique to characterize the earthquake hazards. Firstly, it calculates the faults through SGTs (Strain Green Tensors) function, then it generates the ground motion of each rapture variations on the faults. Cybershake workflow is paralllel in nature and has 5 levels as shown in Figure 5.27. The execution time of each job is as shown in the attached index. In collaboration with the SGT data, estimated future faults rapture is also calculated along with the variations in these raptures. ExtractSGT at level 1, extract the SGTs files and partitions the data.

SeismorgramSynthesis at level 2, generates seismorgram synthesis for each rapture variation which is calculated at level 1. PeakValCalcokaya at level 3, calculates peak value of intensity for each job. ZipPSA at level 4 and ZipSeis at level 5, Collects the values of peak intensity and synthetic seismorgram respectively and compressed in the archival form. Due to large number of files ExtractSGT and ZipPSA consumes a lot of time to extract and compress the data respectively [55].



Node	Time	Node	Time
0	0.07	15	1.53
1	0.19	16	45.60
2	158.10	17	1.53
3	39.06	18	28.67
4	0.63	19	1.36
5	38.49	20	24.56
6	0.80	21	1.36
7	36.27	22	31.05
8	0.73	23	1.43
9	32.29	24	54.87
10	0.74	25	0.78
11	62.25	26	23.99
12	1.42	27	1.10
13	96.91	28	26.46
14	47.44	29	0.85

Figure 5.27: Cybershake Workflow [55].

Case 5: Cybershake workflow of 50 tasks and 5 virtual machines

Case 6: Cybershake workflow of 100 tasks and 10 virtual machines

Case 7: Cybershake workflow of 1000 tasks and 20 virtual machines

The experimental resuts of all the four scheduling algorithms has been compared with the proposed algorithms for the workflow described above and summarized in Table 5.1. A simulation overhead of 0.21 is added by the WorkflowSim simulator for the execution of root node.

Table 5.1: Comparison of the completion time of all the scheduler

Scheduler	FCFS	Min-min	Max-min	MCT	MaxChild	Workflow
Completion Time	60.21	60.21	60.21	60.21	**40.21**	Case 1
	92.87	106.17	92.32	92.87	**91.62**	Case 2
	130.81	125.14	134.05	114.01	**104.69**	Case 3
	416.63	513.52	393.08	416.63	**320.78**	Case 4
	485.61	584.92	453.46	485.61	**357.05**	Case 5
	550.59	525.77	470.11	550.59	**383.68**	Case 6
	1924.63	2166.60	1426.42	1924.63	**1263.47**	Case 7

Chapter 6
Conclusion and Future Work

This chapter concludes the work presented in this thesis and discusses the future directions to extend the present work.

6.1 Conclusion

Existing scheduling algorithms focused on the time. The main aim of these schedulers is to minimize the overall completion time of the workflow but no consideration was given on the resource utilization. In this thesis, gaps in existing workflow scheduling techniques in grid and cloud environment have been analyzed and on the basis of gap analysis an efficient scheduling approach for workflow management in cloud environment has been proposed. It has been analyzed that proposed scheme is effective enough to optimally use the resources. The proposed approach has been implemented in the simulation environment by using the WorkflowSim simulator. The experimental results have shown that proposed scheduling approach minimizes the overall execution time of workflow.

6.2 Thesis Contribution

In order to maximize the resource utilization, MaxChild scheduling algorithm has been proposed which deals with the parent child relationship between the workflow tasks. MaxChild scheduler is successfully implemented in WorkflowSim and overall execution time has reduced compared to existing scheduling algorithms.

6.3 Future Scope

At this time the MaxChild scheduler only considers the parent-child relationship as the scheduling parameter. But in the future, time constraint can also be considered in parallel with this parent-child approach for scheduling. So along with number of child as a parameter for this, time taken by other tasks will also be considered as another parameter in parallel with MaxChild scheduler. However there are still many constraints that are needed to be overcome for more effective and apprehensive results. At present proposed approach is integrated with Retry fault tolerance technique but in the future it can be integrated with other fault tolerance techniques such as Checkpoint, Alternate Resource and Replication at the task level and Alternate task, Redundancy

and Rescue workflow at the workflow level. In future, the work can be extended over heterogeneous environment and the validation can be tested over real time cloud environment.

References

[1] D. C. Wyld, "Moving to the Cloud: An Introduction to Cloud Computing in Government," IBM center for The Business of Government e- Government Series, 2009.

[2] B. Alexander, "Web 2.0: A New Wave of Innovation for Teaching and Learning?" Learning, vol. 41, no. 2, pp. 32-44, 2006.

[3] P. Mell, and T. Grance, "The NIST Definition of Cloud computing," National Institute of Standards and Technology, 2009.

[4] K. Jeffery, B. Neidecker-Lutz, "The Future of Cloud Computing. Opportunities for European Cloud Computing Beyond 2010", Expert Group Report, European Commission, January 2010.

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

[6] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud Computing: State-of-the-Art and Research Challenges," *Journal of Internet Services and Application*, Springer, pp. 7–18, 2010.

[7] T. Dillon, C. Wu, and E. Chang, "Cloud Computing: Issues and Challenges," *24th IEEE Int'l Conference on Advanced Information Networking and Applications (AINA), 2010*, IEEE, 2010.

[8] "Cloud Deployment Model," [online] Available: http://www.google.com, [Accessed: 20 December, 2013].

[9] V. Reddy, B. Rao, L. Reddy, P. Kiran, "Research Issues in Cloud Computing," Global Journal of Computer Science and Technology, vol. 11, no. 11, July 2011.

[10] P. Senkul and I. Toroslu, "An Architecture for Workflow Scheduling under Resource Allocation Constraints," *Information Systems*, vol. 30, no. 5, pp. 399-422, 2005.

[11] J. Yu and R. Buyya, "A Taxonomy of Workflow Management Systems for Grid Computing," *journel of Grid Computing*, Springer, pp. 171-200, 2005.

[12] J. Yu and R. Buyya, "A taxonomy of scientific workflow systems for grid computing," *SIGMOD Record*, Vol. 34, no. 3, pp. 44-49, 2005.

[13] M. Wieczorek, R. Prodan and T. Fahringer, "Scheduling of scientific workflows in the ASKALON grid environment," *SIGMOD Record*, vol. 34, no. 3, pp. 56–62, 2005.

[14] G. Singh, C. Kesselman, and E.Deelman, "Optimizing grid-based workflow execution," *Journal of Grid Computing*, vol. 3, no. 3-4, pp. 201-219, 2005.

[15] C. Chien, P. H. Chang, and V. Soo, "Market-oriented multiple resource scheduling in grid computing environments," *Proceedings in 19th International Conference on Advanced Information Networking and Applications, AINA 2005*, vol. 1, pp. 867-872. IEEE, 2005.

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

[17] Y. Tao, H. Jin, and X. Shi, "Grid workflow scheduling based on reliability cost," In *Proceedings of the 2nd international conference on Scalable information systems*, pp. 12. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007.

[18] B. Simion, C. Leordeanu, F. Pop and V. Cristea, "A Hybrid Algorithm for Scheduling Workflow Applications in Grid Environments (ICPDP)," OTM Confederated International Conferences, Springer Berlin Heidelberg, pp. 1331-1348, 2007.

[19] R. Sakellariou, H. Zhao, E. Tsiakkouri, and M.D Dikaiakos, "Scheduling workflows with budget constraints," *Integrated Research in GRID Computing*, *Springer US*, pp. 189–202, 2007.

[20] M. Rahman, S. Venugopal and R. Buyya, "A Dynamic Critical Path Algorithm for Scheduling Scientific Workflow Applications on Global Grids," *E- Science and Grid Computing, IEEE International Conference*, pp. 35-42, 2007.

[21] M. Wieczorek, S. Podlipnig, R. Prodan and T. Fahringer, "Bi-criteria scheduling of scientific workflows for the grid," *In 8th IEEE International Symposium on Cluster Computing and the Grid, CCGRID'08*, pp. 9-16. IEEE, 2008.

[22] R. Ranjan, M. Rahman, and R. Buyya, "A decentralized and cooperative workflow scheduling algorithm," *In 8th IEEE International Symposium on Cluster Computing and the Grid, CCGRID'08*, pp. 1-8. IEEE, 2008.

[23] M. Xu, L. Cui, H. Wang, and Y. Bi, "A multiple QoS constrained scheduling strategy of multiple workflows for cloud computing," *IEEE 11th Int'l Symposium*

on Parallel and distributed Processing with Applications, Chengdu, China, pp. 629–634,2009.

[24] A. Santiago, A. Yuste, J. Expósito, S. Galán, J Marín, and S. Bruque, "A dynamic-balanced scheduler for Genetic Algorithms for Grid Computing," *WSEAS Transactions on Computers,* vol. 8, no. 1, pp. 11-20, 2009.

[25] W. Chen and J. Zhang, "An ant colony optimization approach to grid workflow scheduling problem with various QoS requirements," *IEEE Transaction on Systems, Man, and Cybernetics,* vol. 39, no. 1, pp. 29–43, 2009.

[26] R. Chang, J. Chang, and P. Lin, "An ant algorithm for balanced job scheduling in grids," *Future Generation Computer Systems,* vol. 25, no. 1, pp. 20-27, 2009.

[27] S. Pandey, L. Wu, S. Guru and R. Buyya "A particle swarm optimization based heuristic for scheduling workflow applications in cloud computing environments," *24th IEEE Int'l Conference on Advanced Information Networking and Applications (AINA), Perth, Australia,* pp. 400–407,2010.

[28] Y. Wang, R. Bhati and M. Bauer, "A Novel Deadline and Budget Constrained Scheduling Heuristic for Computation Grids," *Journal of Central South University of Technology,* vol. 18, no. 2, pp. 465-472,2011.

[29] X. Wang, C. Shin, J. Su and R. Buyya, "Optimizing the makespan and reliability for workflow applications with reputation and a look-ahead genetic algorithm," *Future Generation Computer Systems,* vol 27, no. 8, pp. 1124-1134,2011.

[30] El. kenawy, El. Desoky F. Al-rahamawy, "Extended Max-MIn Scheduling using Petri Net and Load Balancing," *International Journal of Soft Computing and Engineering,* vol. 2, no. 4, 2012.

[31] H. Fard, R. Prodan, J. Barrionuevo, and T. Fahringer, "A multi-objective approach for workflow scheduling in heterogeneous environments," In *Proceedings of the 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (ccgrid 2012),* pp. 300-309. IEEE Computer Society, 2012.

[32] D. Amalarethinam and F. Selvi, "A minimum makespan grid workflow scheduling algorithm," In *International Conference on Computer Communication and Informatics (ICCCI),* pp. 1-6, IEEE, 2012.

[33] S. Ghanbari and M. Othman, "A priority based job scheduling algorithm in cloud computing," *Proceeding In International Conference on Advances Science and Contemporary Engineering 2012 (ICASCE 2012),* pp. 778-785, 2012.

[34] A. Delavar, M. Javanmard, M. Shabestari and M. Talebi, "RSDC (RELIABLE SCHEDULING DISTRIBUTED IN CLOUD COMPUTING)," *International Journal of Computer Science, Engineering & Applications*, vol. 2, no. 3, 2012.

[35] J. Durillo, H. Fard and R. Prodan, "MOHEFT: A multi-objective list-based method for workflow scheduling," In *4th International Conference on Cloud Computing Technology and Science (CloudCom), 2012*, pp. 185-192. IEEE, 2012.

[36] S. Kianpisheh and N. Charkari, "A genetic based workflow scheduling considering data transmission time," In *Sixth International Symposium on Telecommunications (IST)*, pp. 571-576, IEEE, 2012.

[37] Z. Wu, X. Liu, Z. Ni, D. Yuan and Y. Yang, "A Market Oriented Hierarchical Scheduling Strategy in Cloud Workflow Systems," *The Journal of Super Computing*, vol. 63, no. 1, pp. 256-293, Springer US.

[38] Y. Yang and X. Peng, "Trust-Based Scheduling Strategy for Workflow Applications in Cloud Environment," In *Eighth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC)*, pp. 316-320. IEEE, 2013.

[39] D. Kang, C. Youn and M. Chen, "Cost adaptive workflow scheduling in cloud computing," In *Proceedings of the 8th International Conference on Ubiquitous Information Management and Communication*, pp. 65, ACM, 2014.

[40] H. Arabnejad and J. Barbosa, "A Budget Constrained Scheduling Algorithm for Workflow Applications," *Journal of Grid Computing*, pp. 1-15, 2014.

[41] J. Yu, M. Kirley and R. Buyya, "Multi-objective planning for workflow execution on grids," In *Proceedings of the 8th IEEE/ACM International Conference on Grid Computing*, GRID '07, IEEE Computer Society, pages 10–17,Washington, DC, USA, 2007.

[42] G. Singh et al. "Workflow task clustering for best effort systems with Pegasus," In *Proceedings of the 15th ACM Mardi Gras conference: From lightweight mash-ups to lambda grids: Understanding the spectrum of distributed computing requirements, applications, tools, infrastructures, interoperability, and the incremental adoption of key capabilities*. ACM, 2008.

[43] W. Chen, R. Silva, E. Deelman, and R. Sakellariou, "Balanced Task Clustering in Scientific Workflows," In *proceedings IEEE 9th International Conference on eScience*, pp. 188-195. 2013.

[44] E. Deelman, D. Gannon, M. Shields and I. Taylor, "Workflows and e-Science: An overview of workflow system features and capabilities," *Future Generation Computer Systems*, July 10th, 2008.

[45] R. Buyya and M. Murshed, "GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing," *Concurrency and Computation: Practice and Experience* 14 no.13-15, pp. 1175-1220, 2002.

[46] R. Calheiros, R. Ranjan, A. Beloglazov, C. DeRose and R. Buyya, "CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms," *Software: Practice and Experience*, vol. 41, no. 1, Wiley Press, New York, USA, Pages: 23-50, January 2011.

[47] B. Wickremasinghe, R. Calheiros, and R. Buyya, "CloudAnalyst: A CloudSim-based Visual Modeller for Analysing Cloud Computing Environments and Applications." *Proceedings of the 24th IEEE Int'l Conference on Advanced Information Networking and Applications (AINA 2010)*, Perth, Australia, April 20-23, 2010.

[48] R. Calheiros, M. Netto, C. DeRose and R. Buyya, "EMUSIM: An Integrated Emulation and Simulation Environment for Modeling, Evaluation, and Validation of Performance of Cloud Computing Applications," *Software: Practice and Experience*, 2012.

[49] D. Tracy 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*, vol.61, no. 6, pp. 810-837, 2001.

[50] F. Dong and S.G. Akl, "Scheduling Algorithms for Grid Computing: State of the Art and Open Problems," Technical Report of the Open Issues in Grid Scheduling Workshop, School of Computing, University Kingston, Ontario, January 2006.

[51] W. Chen and E. Deelman, "WorkflowSim: A toolkit for simulating scientific workflows in distributed environments," *E-Science (e-Science), 2012 IEEE 8th International Conference on*. IEEE, 2012.

[52] P. Couvares, T. Kosar, A. Roy, J. Weber and K. Wenger, "Workflow in Condor," *In Workflows for e-Science*, Springer Press, January 2007.

[53] "Netbeans," [online] Available: https://netbeans.org/features/index.html, [Accessed: 7th May, 2014].

[54] E. Deelman et. al., “Cybershake: A physics-based seismic hazard model for southern California,” *Pure and Applied Geophysics*, vol. 168, no. 3-4, pp. 367-381, 2011.

[55] G. Juve, A. Chervenak, E. Deelman, S. Bharathi, G. Mehta, and K. Vahi. “Characterizing and profiling scientific workflows,” *Future Generation Computer Systems,* vol. 29, no. 3, pp. 682-692, 2013.

List of Papers

Accepted

[1] Vijay Prakash and Anju Bala, "A Novel Scheduling Approach for Workflow Management in Cloud Computing," in proceedings International Conference on Signal Propagation and Computer Technology (ICSPCT-2014), IEEE, 2014.

Communicated

[1] Vijay Prakash and Anju Bala, "A Survey on Workflow Scheduling Algorithms in Grid and Cloud environment," in 7th International Conference on Contemporary Computing (IC3-2014), IEEE, 2014.