A Hybrid Cloud Scheduling Model Using AHP Approach

Nilofar Parveen R.[1], Sophia Josephine A.[2], Mr. A. Sandana Karuppan [3]

[1]IV year, Department of Information Technology, SSN College of Engineering, Chennai.
[2]IV year, Department of Information Technology, SSN College of Engineering, Chennai.
[3]Department of Information Technology, SSN College of Engineering, Chennai.

ABSTRACT

The goal of cloud scheduling is to achieve high system performance and in-turn high throughput. [1] Scheduling of jobs in distributed and heterogeneous environments require decision making approach for fair utilization of the various available resources. In this proposal, A Hybrid scheduling model based on Analytic Hierarchy Process (AHP) is discussed for the cloud computing environment. We propose a Task-oriented scheduling model, which tries to optimally assign priorities to the task-set based on multiple criteria. It is suitable for scheduling tasks of different nature depending upon the available resources and varying application requirements. Analysing the performance of this model in the real Cloud Environment is extremely difficult, as the evaluation is constrained by cost, rigid infrastructure and time. Hence, a practical solution is to use a cloud simulator to conduct modelling and simulation of our algorithm, through the CloudSim platform.

Keywords - *Cloud Scheduling, Analytic Hierarchy Process, Multi-Criteria Decision Making, Task-oriented scheduling. CloudSim Toolkit.*

I. INTRODUCTION

A) Cloud Computing:

Cloud has become the most pervasive technology in today's world with rapid advancements in web based computing. Cloud is now regarded as the most common utility service analogous to electricity, water, gas etc. Cloud can be defined as a parallel and distributed system consisting of a collection of interconnected and virtualized computers. They are dynamically provisioned and presented as one or more unified computing resources, based on service-level agreements (SLAs), established through negotiations between the service provider and consumer. Cloud computing is a promising technology as long as the internet era is in existence. Investing in cloud based solutions can be highly profitable for the organisations. For example, Lilly, the first company to mass produce penicillin, used cloud computing services to conduct high performance biological sequential analysis without the help of supercomputer. This reduced drug deployment time and lowered the cost to a great extent. In another example, Sensata Technologies, a big supplier of electrical equipment and power solution, used cloud computing in order to improve the company's operation. A majority of the company employee's e-mail accounts were moved to the cloud by the IT department. This decision reduced Sensata's interaction tool cost by $500,000, after only 4 months. Thus, Cloud Computing has many advantages and benefits to offer. But a major problem faced by cloud service providers is to provide service to each user within the deadline and to maximize the utilization of resources.

B) Scheduling In Cloud:

In cloud, we have an execution environment that is elastic and dynamically scaling. It requires users to pay for the amount of resourcing power and services used based on time. Therefore, scheduling has become a crucial aspect in the cloud environment. A good scheduling algorithm increases overall cloud performance by minimising the task completion time. It mainly focuses on efficient utilisation of resources and organisation of tasks.

Task Scheduling is a NP Hard problem that involves assigning of tasks to the available resources in an optimal way. Resource scheduling involves allocation of the computing resources present, appropriately to tasks depending on the user requirements and task configurations. Scheduling of tasks in cloud environment is dynamic. The word dynamic implies that the arrival of task sets, their paths of execution, and available resources can never be predetermined.

C) Analytical Hierarchical Process:

Analytic Hierarchy Process (AHP) is a structured, decision-making technique for organizing and analysing complex decisions, based on mathematics. It was developed by **Thomas L. Saaty** in the 1970s and has been extensively studied and refined since then. The Theory of AHP is a descriptive psychological process that deals with the measurement of intangibles by taking into account the human judgement. It uses pairwise comparisons to arrive at a priority value that can be used to make judgements in complex scenarios. To make comparisons the criteria are given relative importance based on Saaty's fundamental scaling of numbers (shown in table 3.1). It also provides a way to identify inconsistent judgements by means of consistency ratio (CR).

The decisions made from AHP have proven to be correct and useful in solving various problems in different fields. Some of the areas where AHP proved to be successful are cited below.

(1) AHP is used by military organisations for selection of eligible candidates, promoting personnel etc.

(2) AHP is used by the American government to locate the best sites for effective disaster recovery.

(3) AHP was applied to analyse South African Conflict in 1986. The judgements arrived through AHP analysis from the release of Nelson Mandela to removal of apartheid were proven to resolve the conflict when implemented.

(4) Recently, AHP resolved the Israeli Palestine conflict, in 2011, by helping the officials to draft an effective agreement known as the Pittsburgh Principles.

II. LITERATURE SURVEY

A) AHP in Cloud:

Scheduling is a method to effectively distribute the available computing resources to incoming jobs. Raja Manish Singh, Sanchita Paul, Abhishek Kumar review some of the cloud scheduling techniques in reference paper [9]. Their viewpoint is that efficient scheduling of cloud is not achievable by taking only single criteria. Considering a variety of factors and rules improves the overall performance of the cloud. They conduct a comparative study of different algorithms and comment on their feasibility, adaptability and sustainability. They conclude that there is always a chance of modification and coalescing of ideas as task scheduling in cloud requires varied approaches to accommodate its dynamic diverse needs. Resource allocation is a complex process since varied effects and influences must be taken into account. The idea of using Analytic Hierarchy Process (AHP) for serving resources is presented in reference paper [15]. The proposed framework addresses a variety of application factors that need to be judged for making critical decisions for allocating resources. The authors use AHP to choose the best application design- Desktop App (DA), Cloud App (CA) or Web App (WA). They consider multiple attributes such as completion time, cost, bandwidth, and reliability to identify the effective scheme. They calculate priorities to rank the design schemes accordingly. Their experiment portrays that Cloud App (CA) is better when reliability, cost and completion time is considered and WA (Web App) is better in terms of scalability. Saaty is recognised internationally for framing out the hierarchical approach of goal, criteria and alternative candidates for decision making. The paper mentioned in reference [3] gives a detailed explanation on the AHP (Analytical Hierarchical Process) Model, measurement of intangibles through absolute numbers scaling system, steps involved in calculating the principal Eigen vectors and obtaining of priority ranking from multiple criteria. The elimination of inconsistencies in the judgements with the aid of consistency ratio (CR) calculation and verification of the validity of weights obtained for the alternatives are also explained. A clear real time case study, demanding multiple criteria to be accounted when a decision has to be made, is illustrated through mathematical equations and formulas. Reference Paper [14] proposes a Task oriented resource allocation strategy for cloud using AHP as a priority scheduling technique. The tasks are ranked by the pairwise matrix comparison method and resources were allocated based on the ranking of the tasks. It explores the issue of inconsistency through induced bias matrix and illustrates that improving the consistency ratio beyond the 0.1 threshold further helps in allocating better resources. This framework is successful in achieving accuracy on calculating weights for the tasks. The paper directs the research line to allocate resources to the tasks in the dynamic cloud environment accurately.

C) Comparing Simulation Tools:

Evaluation of algorithms in real time cloud may not allow free change in parameters and will incur high cost. Also, lack of success in experimenting might be an expensive mistake. Therefore, it is wise to use simulation tools. Each simulator needs its own runtime environment and requires code to be written in different languages. As every simulator has its own pros and cons, it is essential to choose the appropriate simulator to meet user requirements. The papers referenced in [4] and [10] discuss the various simulation tools and their features. It reviews and compares the various cloud modelling and simulation tools such as CloudSim, GreenCloud, Network CloudSim, Open Cirrus and few others. Based on the study the authors conclude that CloudSim is a more sophisticated and convenient simulation tool. In reference [5] and [6] CloudSim is proposed as a solution to model the complex cloud environment. The papers explain the behaviour of various classes used in CloudSim library. They also give in depth explanation about the CloudSim components such as data centers, cloudlets, virtual machines etc. federation policies, allocation techniques and other custom interfaces of CloudSim are also elaborated. A brief description on the development of CloudSim architecture is also presented. To illustrate the usefulness of the tool various case studies involving federation and dynamic provisioning have been presented and their results were documented.

III. EXISTING SYSTEM

A. Cloud Scheduling Environment:

There are three major components that make up any cloud environment. A detailed description of the various components illustrated in Figure 1, is given as follows:

The **CIS** (Cloud Information Service) component is like a registry. It holds the identities of the all data centers (DCs), hosts and virtual machines (VMs) that make up the cloud environment. Any new broker or DC that joins the cloud must first register with the CIS. It is also used by the broker to get details on various DCs, VMs or hosts during recourse allocation or task scheduling processes.

The **Broker** component responds to the task requests, in a timely manner. It collects and maintains the various tasks, also known as cloudlets. It also allocates the tasks to various resources (DCs, hosts, VMs), keeps track of the task's progress and provides the user with the status updates. The broker is basically the User Interface (UI) between the cloud and the client. There can be many Brokers within a cloud, but a minimum of one is required in-order to have a proper working cloud environment.

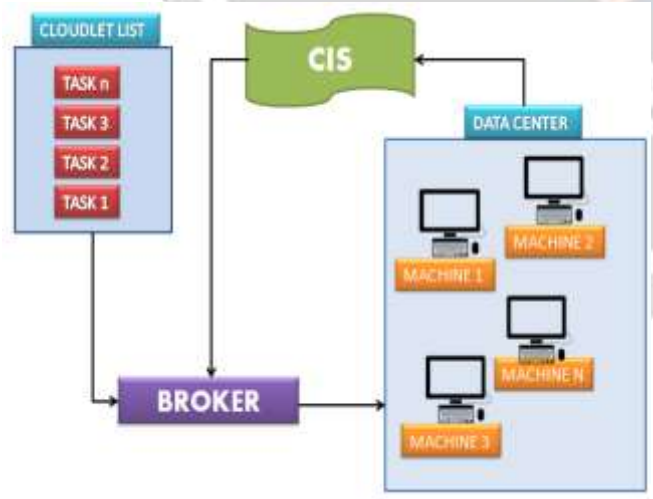

Figure 1: Major Components in Cloud Environments

Finally, the **Data Center** component is a distributed, heterogeneous resource pool that makes up the real-time cloud. It includes all the hosts and VMs that are connected to the cloud. There can be many DCs within a cloud. But there must always be a minimum of one, similar to Broker.

B. Problems in the Existing System:

By default a straight forward First Come First Serve (FCFS) policy is used in scheduling tasks. It provides Response/Service to a particular Request/Task, based on First-Come basis. This method is very easy to implement and understand, but when used in the Cloud computing environment it has many shortcomings. Consider the following scenario, suppose a low priority task is given first followed by a high priority, urgent task. This urgent task now has to wait for the low priority task to complete before it can be allocated a resource. Thus, the urgent task may fail to meet a particular deadline due to this delay. Hence, such conventional methods may not be a suitable, due to high degree of heterogeneity and distribution in the cloud platform. Also, the private scheduling policies used in commercial cloud like (Amazon EC2, Microsoft Azure) are not available to the public.

As mentioned in Section I, scheduling is a NP Hard problem. Therefore, heuristic methods such as Ant Colony, Particle Swarm Optimisation, etc. can be used. But, Heuristic methods operate in a restricted manner. Search Space is narrowed so the result may not be consistent all the time. It also takes higher cost and is sometimes less efficient.

C. Problem Statement:

Classical scheduling methods are not sufficient to meet the increased demand and needs of the customer within the available time. Cloud scheduling requires complex decisions to be made for optimal task assignment as there are multiple resources with varying configurations and capabilities. This problem can be handled by taking other aspects of the task into account as well rather than a single fixed parameter. For example, a task's expected completion time could also be taken into consideration along with the task's submission time.

In this paper, a hybrid scheduling algorithm based on Analytical Hierarchical Process is presented. It is a **task-oriented scheduling model** suitable for prioritizing tasks of different nature depending on application and resources requirements (processing power, main memory, bandwidth etc.). The priority for each task is calculated using Saaty's fundamental scale of absolute numbers, ranging from 1 to 9 (given in Table 1), to perform one-to-one comparisons between the various criteria and generate the pairwise comparison matrices. As the tasks keep dynamically arriving we calculate the priority values for each static subset of tasks. Tasks from the sorted priority pool are then assigned to the ideal resources.

IV. PROPOSED SYSTEM:

A. The AHP Modified Approach:

The modified AHP method is used to calculate a priority value, called weight, for each task based on its attributes and the user requirements. From the Figure 2, we can see the various steps involved in deriving the priority value or weight used for sorting the various tasks. To make a decision in an organized way and to generate the pairwise matrices, one should have relevant and sufficient knowledge about the application and the cloud environment setup.

The AHP modified method is decomposed into the following steps:

Select the Criteria:

1. Define the various Criteria to be considered by taking into account the user requirements and the capabilities of the available data centers. The Designer must choose the criteria that are important in their viewpoint. In this example we have takes into account the application requirements (Completion time, Storage size, Computational capacity of the cloud system) and user requirement (user's task ranking).

Pairwise Comparison Matrix:

2. Construct the pairwise Criteria Matrix by arranging and assigning importance to each criterion based on the Saaty's fundamental scale of absolute numbers given in Table 1, below. (i.e.) comparative judgements are made by deciding how much more a criterion is important over the other. In the relative criteria matrix shown in Table 2, the value of each cell denotes the importance of row criteria over the column criteria.

Intensity Of Importance	Definition	Explanation
1	Equal importance	Two activities contribute equally to the objective
2	Weak	
3	Moderate importance	Experience and Judgment slightly favor one activity over another
4	Moderate plus	
5	Strong importance	Experience and Judgment strongly favor one activity over another
6	Strong plus	
7	Very strong or demonstrated importance	Judgment strongly favor one activity over another, Dominance is demonstrated in Practice.
8	Very, very strong	
9	Extreme importance	Evidence favoring one activity over another is the highest possible order of affirmation
Reciprocals of Above	If activity i has one of the above non zero numbers assigned to it when compared with activity j, then j has the reciprocal value when compared with i.	A reasonable assumption
Values between 1.1 to 1.9	If the activities are very close	May be difficult to assign the best value but when compared with other contrasting activities the size of the small numbers would not be too noticeable, yet they can still indicate the relative importance of activities

Table 1- The fundamental scale of absolute numbers

	Rank	Completion Time	Storage	Computational Capacity
Rank	1	3	7	4
Completion Time	1/3	1	6	3
Storage	1/7	1/6	1	1/7
Computational Capacity	1/4	1/3	7	1

Table 2- The Relative Criteria Matrix - C

Calculate Weights:

3. Normalisation is performed, on the matrix, to scale the weights of the numbers from 0 to 1. It is done by calculating the sum all the elements in each column (Table 3) and dividing each element with the corresponding column sum, to get the normalized matrix (Table 4).
4. The principal Eigen vector is calculated by finding the average of rows of the normalised matrix. (Table 4)

	Rank	Completion Time	Storage	Computational Capacity
Rank	1	3	7	4
Completion Time	1/3	1	6	3
Storage	1/7	1/6	1	1/7
Computational Capacity	1/4	1/3	7	1
Sum	1.7262	4.5000	21.0000	8.1429

Table 3- The Column sum of each column

	Rank	Completion Time	Storage	Computational Capacity	Priorities
Rank	0.5793	0.6666	0.3333	0.4912	0.5176
Completion Time	0.1931	0.2222	0.2857	0.3684	0.2674
Storage	0.0827	0.0370	0.0476	0.0175	0.0462
Computational Capacity	0.1448	0.0740	0.3333	0.1228	0.1688
Sum	1	1	1	1	

Table 4 - The Normalised Criteria Matrix with the Calculated Priority Vector

5. Repeat steps 2 to 4; build the table with respect to all criteria. Normalise and obtain the individual priority vector by considering each of the criteria individually. In our example we have four tasks to be executed in the cloud. Table 5 shows the comparison matrix of all the tasks with each other with respect to the user assigned ranking alone is.

User Ranking	T1	T2	T3	T4
T1	1.00000	0.75000	6.00000	2.00000
T2	1.33333	1.00000	8.00000	2.66667
T3	0.16667	0.12500	1.00000	0.33333
T4	0.50000	0.37500	3.00000	1.00000

Table 5- Comparative scaling for Tasks based only on Criteria 1 (User Ranking).

6. Combine the priority vectors into a matrix [W_{CT}]. Multiplying the transpose of [W_{CT}] matrix with the [W] vector gives the overall global priority weights (Table 6) for all the tasks considering multiple criteria. This priority value is used to sort the various tasks and submit them accordingly to the cloud broker.

$[W_{CT}]^T$	Rank	Completion Time	Storage	Computational Capacity		Priorities		Global Priority
T1	0.3333	0.15625	0.1814	0.1250		0.5176		0.24380
T2	0.4444	0.2187	0.2271	0.3750	x	0.2674	=	0.36233
T3	0.0556	0.2812	0.2728	0.1250		0.0462		0.13766
T4	0.1666	0.3437	0.3185	0.3750		0.1688		0.25619

Table 6 - Global Priority Calculation

To Calculate And Check Consistency Of A Matrix:

When there are many criteria involved in decision making it is difficult to keep track of their transitive properties. In such cases, Consistency Ratio (CR) helps to identify whether the matrix is made of reliable scaling. If the CR value of a matrix is 0.00, the matrix is perfectly consistent. A maximum threshold of 0.10 can be tolerated. The matrix is recalculated for CR values above the threshold. The range 0.00 to 0.1 is allowed in order to make continuous revision in understanding the effect of judgement in arrived decisions. The CR threshold can be made even smaller in order to get more accurate weights. Thus, this process makes sure that all the internal dependencies are also considered before computing the global weights.

7. [W] is the priority vector of the relative criteria matrix C. [Ws] is calculated by multiplying the criteria matrix [C] with its principal Eigen vector [W].

8. Consistency vector is computed by calculating the dot product (scalar multiplication) of Ws and 1/W. The average of the elements of the consistency vector is denoted by λ which is also the Eigen value of the system.

W	W_s	1/W	Consistency vector
0.5176	2.3184	1.9319	4.4789
0.2674	1.2236	3.7402	4.5766
0.0462	0.1889	21.6265	4.0843
0.1688	0.7110	5.9256	4.2129
			+
		Eigen Value λ	4.3382
		CI = (λ-n)/(n-1)	0.1127
		RI	0.8900
		CR = CI/RI	0.1267

Table 7 - The Consistency Ratio Calculation

The Consistency Index is calculated using the formula:

$$CI = (\lambda - n)/(n-1)$$

Where,

λ - Eigen value

n - Number of criteria.

The Consistency Ratio is given by

$$CR = CI/RI$$

Where,

CI – Consistency Index

RI – Random Index

RI (Random index) is the consistency index for a (n x n) matrix if the pairwise comparisons were completely random. The random values obtained from one set of such simulations are taken for reference from Saaty's random index table.[8] The random index for n=4 is 0.89. Table 5 depicts the calculation of CI for the matrix in Table 2.

It can be seen that the weights of T1 and T4 vary slightly. The consistency ratio can be reduced further to improve variations of weights accurately.

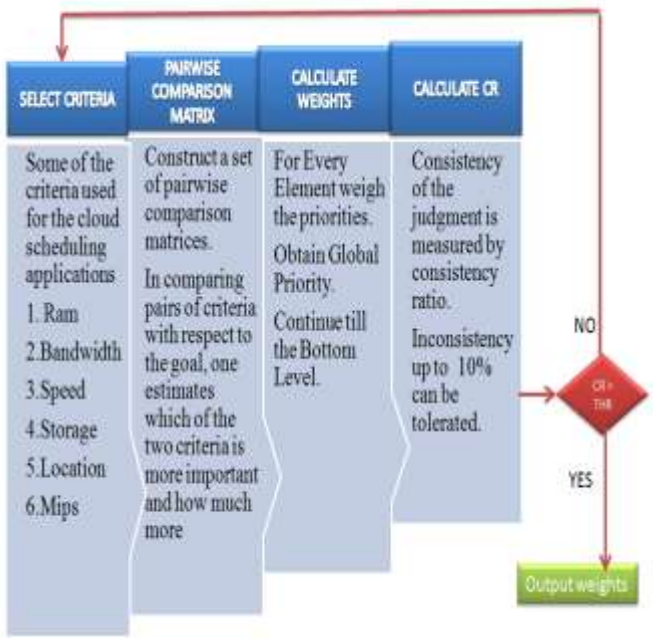

Figure 2: Steps Involved in the AHP Model

The steps elaborated above can be visually depicted through Figure 2, for better understanding.

B. Working Methodology:

Figure 3, given below, portrays the basic workflow process that takes place during the real-time cloud scheduling. It makes use of the components, discussed in Part A of the literature review section (section II), with some slight modifications.

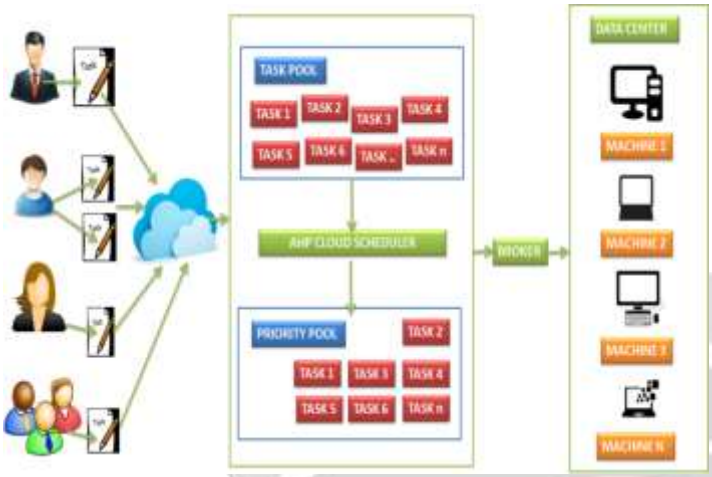

Figure 3: Architecture – Cloud Scheduling Using AHP Approach

The CIS component is modified to create the **AHP Scheduler** component. It has same functionality as mentioned before (Section III - Part A) except, apart from acting as a registry (by holding all the cloudlet IDs) it also now performs the AHP algorithmic procedure detailed in the previous section (Section III – Part D) to calculate the global weights and assigns these values as each cloudlet's priority.

The **Broker**, being similar to what has been previously discussed (Section III - Part A), additionally sorts the cloudlets in descending order of the priority value computed.

Assumptions:

For the simulation to work effectively and to calculate the performance, we have to make sure that it adheres to some base conditions.

(1) There must be at least one Data Center (DC) so that proper scheduling of tasks can be simulated and realistic performance values can be obtained for analysis.

(2) There must be a minimum of one Broker for simulation for real-time task scheduling and resource allocation.

(3) To effectively analyse the system performance there must be at least, but not limited to, a thousand tasks or cloudlets.

V. SIMULATION TOOL

A) Tool Introduction:

It is unwise to directly analyse the behaviour of the AHP Scheduling Model in the cloud environment using real-time hardware and software resources, as it could entail very high cost and/or loss. To analyse and evaluate the proposed model we use CloudSim - a modelling and simulation toolkit for cloud environments. CloudSim was invented as CloudBus Project at the University Of Melbourne, Australia. It supports system and behaviour modelling of cloud components such as data centers, virtual machines (VMs) and resource provisioning policies. [10]

CloudSim is an Open Source application that is free to be downloaded from the web. CloudSim was developed based on GridSim. GridSim was used to model distributed grid environments. The main advantages of using CloudSim include: (i) time effectiveness (ii) provisioning test environment (iii) flexibility and applicability. Some features that are uniquely present in CloudSim are (i) virtualization engine (ii) flexibility to switch between space-shared and time-shared allocation of processing cores to virtualized services. CloudSim is not a ready to use solution as it does not provide a predefined environment where you can set parameters to achieve results. It is a library where the users have to model a specific cloud scenario by writing code in Java, define the output and input parameters, collect the values and analyse the results. The Basic Classes such as data

centers, virtual machines, applications, users, computational resources, and policies for management can be used to build a cloud environment for testing the various scheduling and allocation policies. It is an Event Based simulator, communication between CloudSim entities is achieved by message passing.

B) Tool Architecture:

Before the advent or domination of cloud computing, Grid had been the most popular distributed computing model. Therefore, various grid simulators were developed for modelling, developing and testing the grid environment. But the sophisticated multi layered abstraction (SaaS – Software as Service, PaaS – Platform as a Service, IaaS - Infrastructure as a Service) and virtualization which was unique to cloud environment could not be modelled using the grid simulators. Hence, rather than developing a library from scratch, the bottom layer of the grid simulation framework was used and additional CloudSim components were built on top of it (Figure 4). CloudSim emulator uses a hierarchical organization, which consists of four levels. From the bottom up, they are the SimJava, GridSim, CloudSim, and user code.

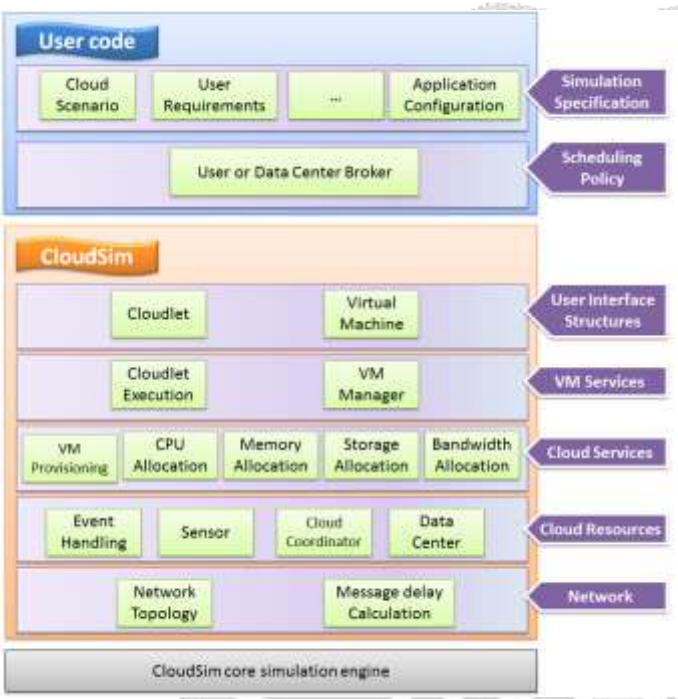

Figure 4: CloudSim Architectural Layers

The bottommost is a discrete event simulation engine SimJava, which is responsible for the implementation of the core functions of high-level simulation framework, such as: query and processing events, build system components (services, clients, data centers, agents and virtual machines), the communication between different components and the analog clock management.

The CloudSim simulation layer supports modelling of the DataCenter environments in cloud, (i.e. managing virtual machines and their configurations)

The user code layer is used to implement the user's scheduling/allocation policy by making use of the classes provided in the CloudSim library.

C) AHP Simulation:

The scheduling and prioritisation of the tasks in the cloud environment setup, using modified AHP approach, was simulated using CloudSim version 3.0.3.

The Cloudlet is an abstract class modelling the various services offered by the cloud. Every cloudlet has parameters such as Id, length, file size, number of processing elements that are pre assigned. Apart from this, additional attributes (Cloudlet Priority and Cloudlet ID) are included in the pre-existing Cloudlet java class. The data members of the Cloudlet class is altered and used to calculate the comparative scaling in the pairwise comparison matrix for all the criteria.

The DataCenter class is used to configure the resources in the Data centre such as number of hosts in a data centre, ram, bandwidth, storage etc.

DataCenterBroker class is responsible for negotiating between the tasks and cloud providers, discovery of resources etc.

In this simulation we have imported various classes available in the CloudSim library to create our custom java scheduler class - AhpScheduler, where the actual computational steps for calculating the global priorities is carried out

The Cloudlet along with the computed priority weights are sent from our custom AHP scheduler class to the DataCenterBroker. After sorting the tasks by priority (calculated with multiple criteria) it allocates resource to them.

VI. RESULTS AND DISCUSSION

The simulation is run for various scenarios and the results are recorded. Similarly, the same set of scenarios is also simulated for the default FCFS scheduling class. Using these results, performance of AHP scheduler is analysed.

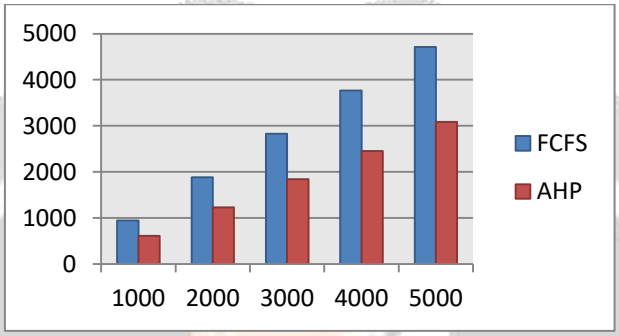

Figure 5: Comparison of Total Execution Time for AHP and FCFS with Space shared VM allocation

VII. CONCLUSION

The cloud environment requires users to pay for the services used, for the time and amount of resourcing power; therefore, the order of execution of the tasks has vital effect on the user expenses. Scheduling mechanisms are important to improve the server and resource utilization and increase the performance. In this project we have used a modified AHP model as a task-oriented priority scheduling technique. The main goal of this approach is to find the proper sequence of jobs to be executed by considering multiple parameters. The well-defined hierarchical structure of the AHP is used to obtain the priorities for the tasks, in the cloud environment where the alternatives are in multitude and very much tangled. The tasks are pairwise compared by considering both the user's requirement and application requirement to obtain the priority vector. They are then sorted based on the computed global priority value and submitted for execution.

Checking the fitness of the scheduling algorithm is good practice. Hence, this algorithm's performance is tested by comparing its priority task ordering and completion time with other algorithms. The experimental results shows that this scheduling mechanism is more beneficial when compared to FCFS algorithm, due to its maximum utilization and also it decreased latency period. This method can be adopted in the existing cloud computing systems for faster execution and response for important tasks.

In our study, attributes such as user ranking, completion time, storage and computing capacity are considered for prioritization but there are still plenty of other attributes that can be considered. The consistency ratio can be reduced to obtain more accurate weights and ordering of the tasks. The priority values obtained are only used to frame the execution order. This method does not deal with dynamic allocation of the computing resources to tasks. For future improvement we can combine this modified AHP task scheduler with any resource scheduling algorithm to further reduce the response time and improve the make-span.

VIII. REFERENCES

[1] Morgan Kaufmann Publications Hwang and Geoffrey Fox, *"Distributed and Cloud Computing"*

[2] Aditya Makwe and Priyesh Kanungo, *"Scheduling in Cloud Computing Environment Using Analytic Hierarchy Process Model"* in *IEEE International Conference on Computer, Communication and Contro*l on 2014

[3] Thomas L. Saaty, "*Decision making with the analytic hierarchy process*" in 2008

[4] Utkal Sinha and Mayank Shekhar, *"Comparison of Various Cloud Simulation tools available in Cloud Computing"*

[5] Rodrigo, Rajiv, Anton, Cesar A. F.,and Rajkumar Buyya,*"CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms"*, Published online on 24th August 2010 in Wiley Online Library

[6] Rajkumar Buyya, Rajiv and Rodrigo, *"Modeling and Simulation of Scalable Cloud Computing Environments and the CloudSim Toolkit: Challenges and Opportunities"* in 2009.

[7] Rodrigo N. Calheiros, Rajiv Ranjany, and Rajkumar Buyya*"Virtual Machine Provisioning Based on Analytical Performance and QoS in Cloud Computing Environments"* in *International Conference on Parallel Processing* on 2011.

[8] Thomas L. Saaty, "*The Measurement of Intangibles. A Principal Eigenvector Approach to Relative Measurement Derived from Paired Comparisons".*

[9] Raja Manish Singh, Sanchita Paul, Abhishek Kumar, "*Task Scheduling in Cloud Computing: Review"*

[10] Dr. Rahul Malhotra and Prince Jain, "*Study and Comparison of Various Cloud Simulators Available in the Cloud Computing*"

[11] Azeem Ahmad, Magnus Goransson, Aamir Shahzad, " *Limitations of the Analytic Hierarchy Process Technique with Respect to Geographically Distributed Stakeholders"* in International Journal of Computer, Electrical, Automation, Control and Information Engineering Vol:4, No:9, on 2010.

[12] Alessio Ishizaka and Ashraf Labib, *"Analytical Hierarchical Process- Benifits and Limitations"* in 2009.