ISSN: 2250–3676
[IJESAT] INTERNATIONAL JOURNAL OF ENGINEERING SCIENCE & ADVANCED TECHNOLOGY Volume-2, Issue-4, 1144 – 1151

DYNAMIC RESOURCE ALLOCATION IN THE CLOUD COMPUTING USING NEPHELE'S ARCHITECTURE

Koti Reddy S[1], Ch. Subba Rao[2]

[1]*M.Tech, Dept. of C.S.E, Quba College of Engineering, Venkatachalam, Nellore, A.P, India*
[2]*Associate Professor, Dept. of C.S.E, Quba College of Engineering, Venkatachalam, Nellore, A.P, India*
kotimcp@gmail.com

Abstract

In recent years ad-hoc parallel data processing has emerged to be one of the most important applications for Infrastructure-as-a-Service (IaaS). Major Cloud computing companies have started to integrate frameworks for parallel data processing in their product portfolio, making it easy for customers to access these services and to deploy their programs. However, the processing frameworks which are currently used have been designed for static, homogeneous cluster setups and disregard the particular nature of a cloud.

Consequently, the allocated compute resources may be inadequate for big parts of the submitted job and unnecessarily increase processing time and cost. Nephel's architecture offers for efficient parallel data processing in clouds. It is the first data processing framework for the dynamic resource allocation offered by today's IaaS clouds for both, task scheduling and execution. Particular tasks of a processing job can be assigned to different types of virtual machines which are automatically instantiated and terminated during the job execution.

Index Terms: *Cloud Computing, Parallel Data Processing, Dynamic resource allocation*

--- *** ---

1. INTRODUCTION

Cloud computing is the delivery of computing and storage capacity as a service to a community of end-recipients. The name comes from the use of a cloud-shaped symbol as an abstraction for the complex infrastructure it contains in system diagrams. Cloud computing entrusts services with a user's data, software and computation over a network.

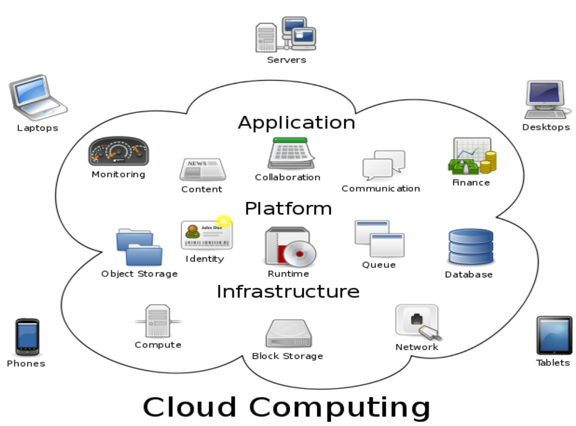


Fig-1: Cloud Computing

There are three types of cloud computing:

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

Using Software as a Service, users also rent application software and databases. The cloud providers manage the infrastructure and platforms on which the applications run.

End users access cloud-based applications through a web browser or a light-weight desktop or mobile app while the business software and user's data are stored on servers at a remote location. Proponents claim that cloud computing allows enterprises to get their applications up and running faster, with improved manageability and less maintenance, and enables IT to more rapidly adjust resources to meet fluctuating and unpredictable business demand.

Cloud computing relies on sharing of resources to achieve coherence and economies of scale similar to a utility (like the electricity grid) over a network (typically the Internet). At the foundation of cloud computing is the broader concept of converged infrastructure and shared services.

ISSN: 2250–3676
[IJESAT] INTERNATIONAL JOURNAL OF ENGINEERING SCIENCE & ADVANCED TECHNOLOGY Volume-2, Issue-4, 1144 – 1151

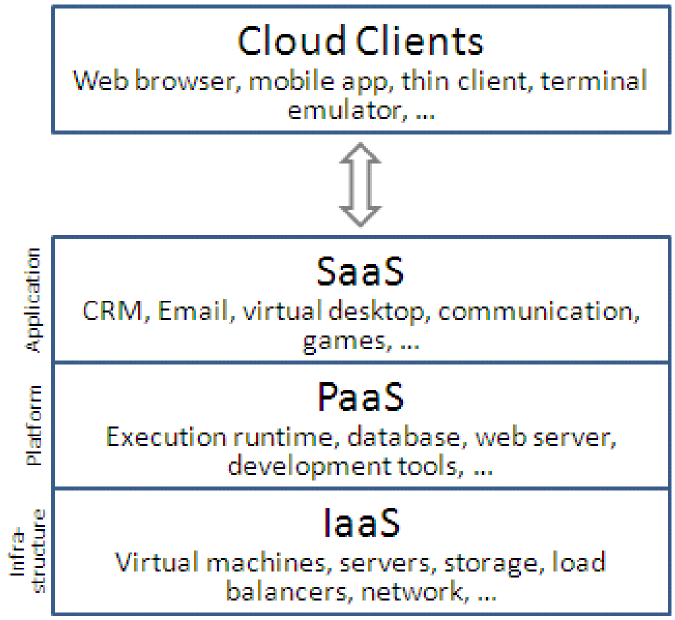


Fig-2: Cloud computing layers

1.1 Problem Definition

The efficient parallel data processing is achieved by using Nephele's framework by dynamic resource allocation offered by today's IaaS clouds for both, task scheduling and execution. Particular tasks of a processing job can be assigned to different types of virtual machines which are automatically instantiated and terminated during the job execution. Based on this, we perform extended evaluations of MapReduce-inspired processing jobs on an IaaS cloud system.

The processing framework then takes care of distributing the program among the available nodes and executes each instance of the program on the appropriate fragment of data. Most notably, Nephele is the first data processing framework to include the possibility of dynamically allocating/ de-allocating different compute resources from a cloud in its scheduling and during job execution.

1.2 Disadvantages of Existing System

The disadvantages of existing systems are as follows:

- Expensive
- Complex
- Increases data base organization

1.3 Proposed System

In recent years a variety of systems to facilitate MTC has been developed. Although these systems typically share common goals (e.g. to hide issues of parallelism or fault tolerance), they aim at different fields of application. MapReduce is designed to run data analysis jobs on a large amount of data, which is expected to be stored across a large set of share-nothing commodity servers.

Once a user has fit his program into the required map and reduce pattern, the execution framework takes care of splitting the job into subtasks, distributing and executing them. A single Map Reduce job always consists of a distinct map and reduce program.

1.4 Advantages of Proposed System

The advantages of proposed systems are as follows:

- Dynamic resource allocation
- Parallelism is implemented
- Designed to run data analysis jobs on a large amount of data
- Many Task Computing (MTC) has been developed
- Less expensive
- More effective
- More Faster

2. SYSTEM ANALYSIS

2.1 Economical Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

2.2 Social Feasibility

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

2.3 Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

3. SYSTEM REQUIREMENTS

3.1 User Requirements

- Operating System : Windows NT
- Web Browsers : IE 6.0+, Firefox 10.0+
- Processor : Pentium IV 2.4 GHz
- Disk Space : 512 MB
- RAM : 1 GB
- Monitor : 15 VGA Colour
- Mouse : Logitech

3.2 Software Requirements

- Operating System : Windows NT
- Prog. Language : ASP.Net, C#
- Web Programming : HTML, CSS
- Database : SQL Server 2008
- Tools : Visual Studio 2010

3.3 Hardware Requirements

- Cloud Computing : Cloud Provider
- Processor : 8 x 1.6GHz CPU
- Hard Disk : 320 GB
- RAM : 14GB RAM

4. SYSTEM DESIGN

4.1 Architecture

Nephele's architecture follows a classic master-worker pattern as illustrated below:

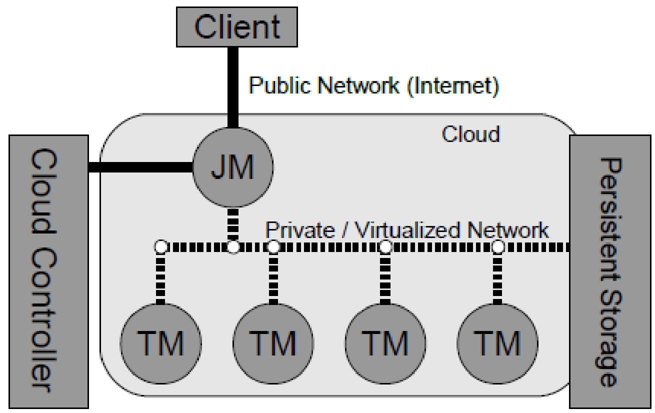


Fig-3: Nephele's Architecture

4.2 Job Graph

Defining a Nephele job comprises following mandatory steps:

- The user must connect to virtual machine and start his task
- The task program must be assigned to a vertex
- Finally, the vertices must be connected by edges to define the communication paths of the job

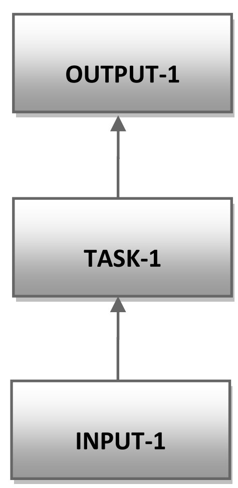


Fig-4: An example of Job Graph in Nephele

4.2 Execution Graph

Once the valid Job Graph is received from the user, Nephele's Job Manager transforms it into Execution Graph. An Execution Graph is Nephele's primary data structure for scheduling and monitoring the execution of a Nephele job. The Execution Graph contains all the information required to schedule and executes the received job on the cloud. It explicitly models task parallelization and the mapping of tasks to instances. Depending on the level of annotations the user has provided with his Job Graph, Nephele may have different degrees of freedom in constructing the Execution Graph. Task 1 is e.g. split into two parallel subtasks which are both connected to the task Output 1 via file channels and are all scheduled to run on the same instance. The exact structure of the Execution Graph is explained in the following:

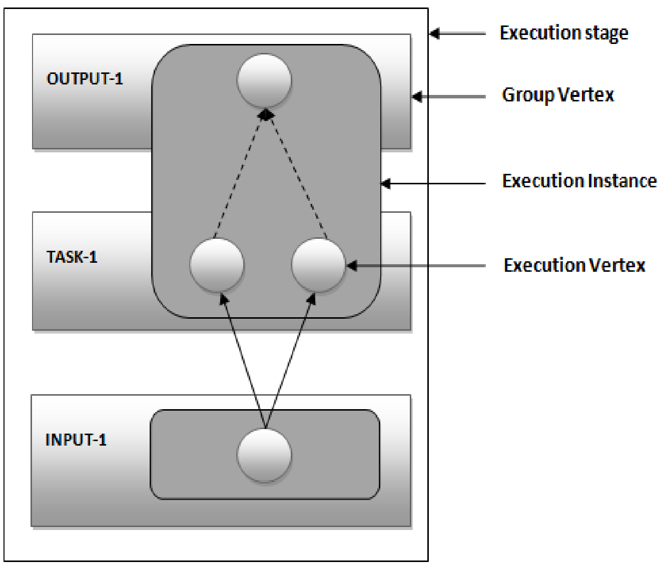


Fig-5: Execution Graph created from the original Job Graph

4.3 Input Design

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

Objectives

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

4.4 Output Design

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2. Select methods for presenting information.

3. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, current status or projections of the Future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action.

4.5 System Flow

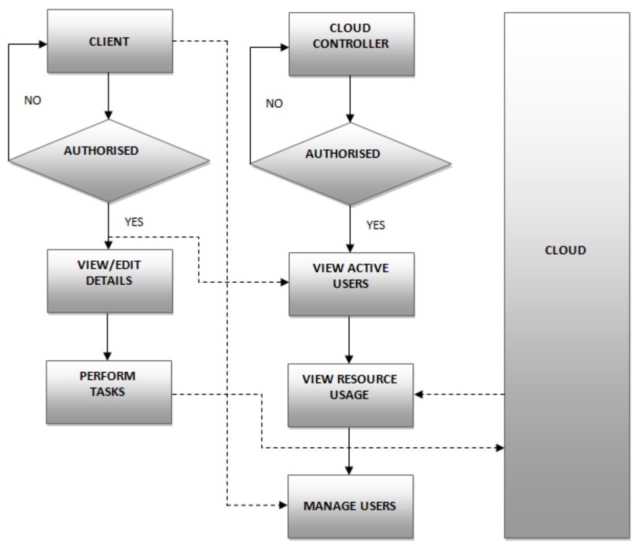


Fig-6: System Flow

5. IMPLEMENTATION

5.1 Network Module

Server - Client computing or networking is a distributed application architecture that partitions tasks or workloads between service providers (servers) and service requesters, called clients. Often clients and servers operate over a computer network on separate hardware. A server machine is a high-performance host that is running one or more server programs which share its resources with clients. A client also shares any of its resources; Clients therefore initiate communication sessions with servers which await (listen to) incoming requests.

5.2 LBS Services

In particular, users are reluctant to use LBSs, since revealing their position may link to their identity. Even though a user may create a fake ID to access the service, her location alone may disclose her actual identity. Linking a position to an individual is possible by various means, such as publicly available information city maps. When a user wishes to pose a query, she sends her location to a trusted server, the anonymizer through a secure connection (SSL). The latter obfuscates her location, replacing it with an anonymizing spatial region (ASR) that encloses you. The ASR is then forwarded to the LS. Ignoring where exactly u is, the LS retrieves (and reports to the AZ) a candidate set (CS) that is guaranteed to contain the query results for any possible user location inside the ASR. The AZ receives the CS and reports to u the subset of candidates that corresponds to her original query.

5.3 System Model

The ASR construction at the anonymization process abides by the user's privacy requirements. Particularly, specified an anonymity degree K by u, the ASR satisfies two properties: (i) it contains u and at least another K * 1 users, and (ii) even if the LS knew the exact locations of all users in the system.

- We propose an edge ordering anonymization approach for users in road networks, which guarantees K-anonymity under the strict reciprocity requirement (described later).

- We identify the crucial concept of border nodes, an important indicator of the CS size and of the query processing cost at the LS.

- We consider various edge orderings, and qualitatively assess their query performance based on border nodes.

- We design efficient query processing mechanisms that exploit existing network database infrastructure, and guarantee CS inclusiveness and minimality. Furthermore, they apply to various network storage schemes.

- We devise batch execution techniques for anonymous queries that significantly reduce the overhead of the LS by computation sharing.

5.4 Scheduled Task

Recently, considerable research interest has focused on preventing identity inference in location-based services.

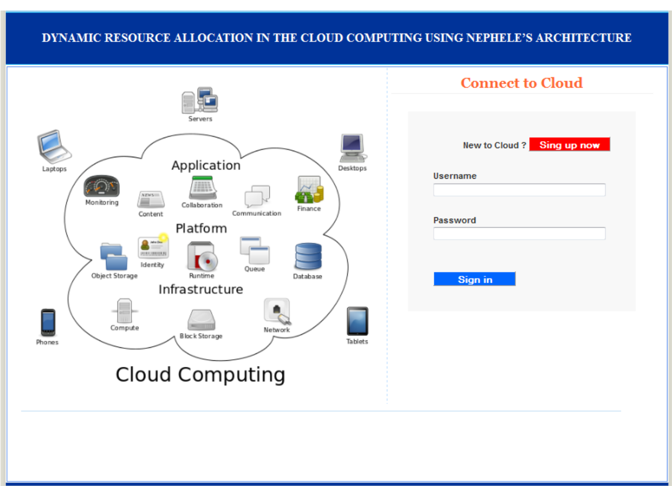
Proposing spatial cloaking techniques. In the following, we describe existing techniques for ASR computation (at the AZ) and query processing (at the LS). At the end, we cover alternative location privacy approaches and discuss why they are inappropriate to our problem setting. This offers privacy protection in the sense that the actual user position u cannot be distinguished from others in the ASR, even when malicious LS is equipped/advanced enough to possess all user locations. This spatial K-anonymity model is most widely used in location privacy research/applications, even though alternative models are emerging.

5.5 Query Processing

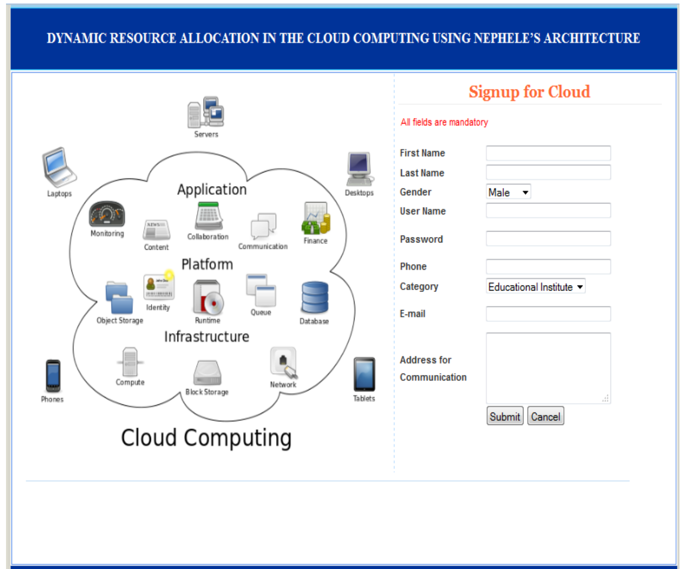
Processing is based on implementation of the theorem uses (network-based) search operations as off the shelf building blocks. Thus, the NAP query evaluation methodology is readily deployable on existing systems, and can be easily adapted to different network storage schemes. In this case, the queries are evaluated in a batch. we propose the network-based anonymization and processing (NAP) framework, the first system for K- anonymous query processing in road networks. NAP relies on a global user ordering and bucketization that satisfies reciprocity and guarantees K-anonymity. We identify the ordering characteristics that affect subsequent processing, and qualitatively compare alternatives. Then, we propose query evaluation techniques that exploit these characteristics. In addition to user privacy, NAP achieves low computational and communication costs, and quick responses overall. It is readily deployable, requiring only basic network operations.

6. FORMS & SCREENS

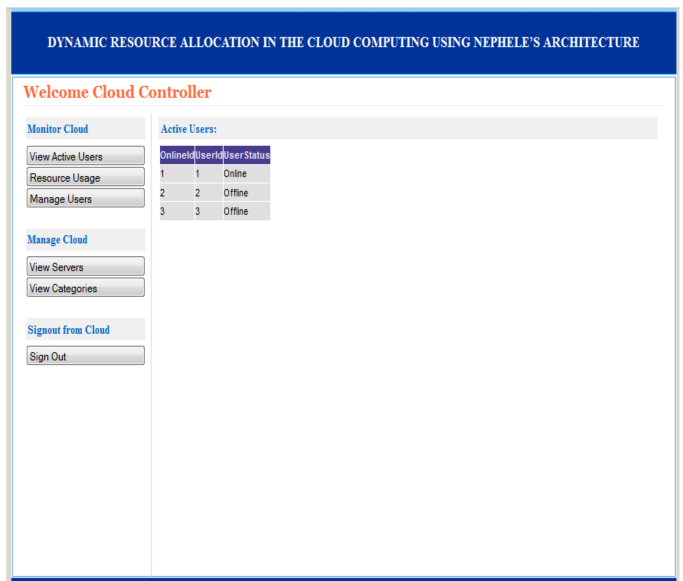
6.1 Client - Login Form



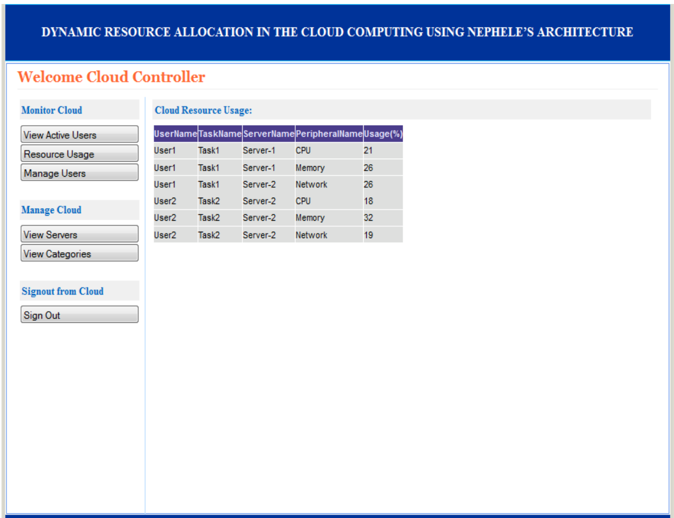
6.2 Client - Registration Form



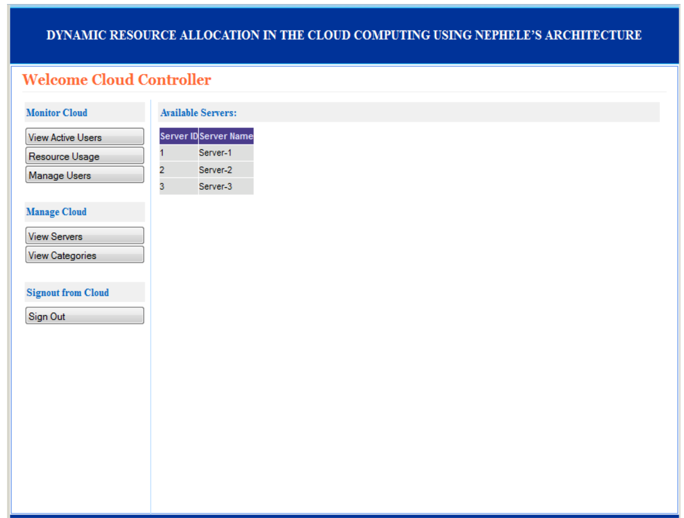
6.3 Controller – View Active Users Screen



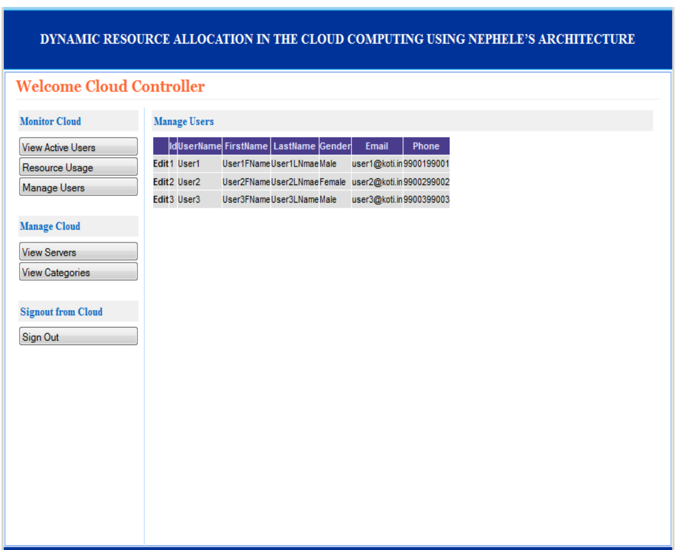
6.4 Controller – View Resource Usage Screen



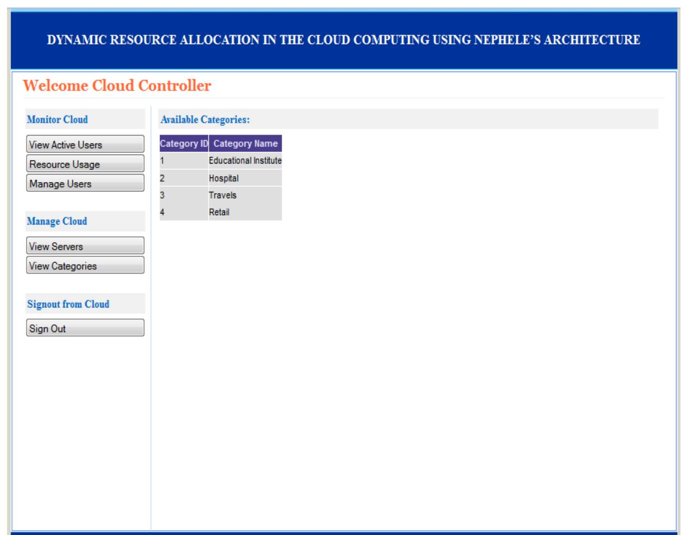
6.6 Controller – View Servers Screen



6.5 Controller – Manage Users Screen



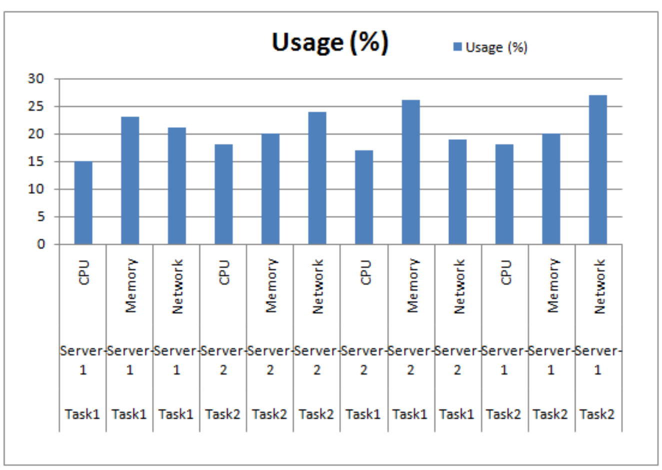
6.7 Controller – View Categories Screen

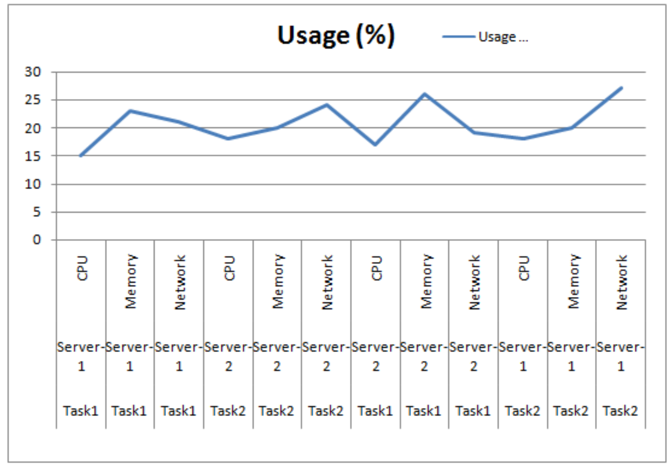


6.8 Cloud Client Screen

7. RESULT ANALYSIS

User Name	Task Name	Server Name	Peripheral Name	Usage (%)
User1	Task1	Server-1	CPU	15
User1	Task1	Server-1	Memory	23
User1	Task1	Server-1	Network	21
User1	Task2	Server-2	CPU	18
User1	Task2	Server-2	Memory	20
User1	Task2	Server-2	Network	24
User2	Task1	Server-2	CPU	17
User2	Task1	Server-2	Memory	26
User2	Task1	Server-2	Network	19
User2	Task2	Server-1	CPU	18
User2	Task2	Server-1	Memory	20
User2	Task2	Server-1	Network	27





8. CONCLUSION

The conclusion is that, clients can connect to virtual machines of the cloud and they can perform their business tasks in parallel by dynamic resource allocation mechanism of Nephele's architecture.

REFERENCES

[1]. Daniel Warneke, Member, IEEE, and Odej Kao, Member,IEEE," Exploiting Dynamic Resource Allocation for Efficient Parallel Data Processing in the Cloud", IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 22, NO. 6, JUNE 2011.

[2] Amazon Web Services LLC. Amazon Elastic Compute Cloud (Amazon EC2). http://aws.amazon.com/ec2/, 2009.

[3] Amazon Web Services LLC. Amazon Elastic MapReduce. http://aws.amazon.com/elasticmapreduce/, 2009.

[4] AmazonWeb Services LLC. Amazon Simple Storage Service. http://aws.amazon.com/s3/, 2009.

[5] D. Battr´e, S. Ewen, F. Hueske, O. Kao, V. Markl, and D. Warneke.Nephele/PACTs: A Programming Model and Execution Framework for Web-Scale Analytical Processing. In SoCC '10: Proceedings of the ACM Symposium on Cloud Computing 2010, pages 119– 130, New York, NY, USA, 2010. ACM.

[6] R. Chaiken, B. Jenkins, P.-A. Larson, B. Ramsey, D. Shakib, S. Weaver, and J. Zhou. SCOPE: Easy and Efficient Parallel Processing of Massive Data Sets. Proc. VLDB Endow., 1(2):1265– 1276, 2008.