ISSN(Online): 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 6, June 2016

COST EFFECTIVE RESOURCE ALLOCATION BASED on PG CAN-TOF

R.Sowndharya, C.Gayathri

PG Student M.E (Computer Science Engineering), Department of CSE, Mahendra Institute of Technology, Mallasamudram. Salem, India

Assistant Professor, Department of CSE, Mahendra Institute of Technology, Mallasamudram, Salem, India

ABSTRACT: The rank of cloud computing is rapidly growing and many more cloud providers are emerging, cost efficiency and source cost maximization become two major concerns of cloud providers to remain competitive while making profit. In the first part of the thesis, we investigated the profit maximization problem in federated cloud environments where cloud providers cooperate to increase the degree of multiplexing. In this part, we outline novel economics inspired source allocation mechanisms to tackle the profit maximization problem from the perspective of a cloud provider acting solely. we propose admission control mechanisms adapted within a Profit management framework to maximize source cost where various pricing plans in multiple market places are sustained by the provider and an auction-based dynamic pricing mechanism suitable for selling the spare capacity of the data center. The realization of the proposed dynamic pricing mechanism with in a pricing as a service framework.

I. INTRODUCTION

Cloud computing is a technology that make use of the Internet to obtain the software or other IT services on demand.This on-demand computing power and storage capacity makes cloud computing as an efficient computing platform. Cloud computing provides the shared sources needed by the users at any given time by pay as they go and keeping cost to the user down. Cloud computing is a technology and business model as well where the providers are providing the software, hardware, display place, or storage providers, deliver their hand-outs over the Internet. The data application of the providers is maintained by the central remote servers by using internet. The applications are used without installation and the personal files which can be located at different location form user can be access through internet. Cloud computing is not a new technology but it leverages the advantages of existing and increases the efficiency. When compared to grid computing which coordinates the networked sources as the distributed computing paradigm, the cloud computing satisfies this distributed sources objectives like computing power, software, storage services, and platforms by virtualization technologies helps customer or user to complete the jobs in less time and cost.

When compared to utility computing which provides on demand sources and charge based on the usage, the cloud computing perceived this utility based scheme by increasing the source utilization and decrease the operating cost. Virtualization is the key feature that acts as the foundation for providing the sources to users on demand. The commonly used virtualized server called virtual machines (VM) helps for assigning and reassigning the sources on demand. The cloud offering as services can be mainly categorized as three namely software as a service (SaaS), infrastructure as a service (IaaS) and platform as a service (PaaS). These services can be provides by cloud from everywhere and at any time through internet.

The cloud computing technologies and virtualization growth requires the virtual machines for number of jobs. Since cloud computing becomes the emerging and attractive technology, many user deploy their application in the cloud or extend their home clusters during high demand results in workflow management system. The workflow management system should balance both performance and cost. Different types of scheduling optimization approaches have been developed to reduce the cost and increase the performance. In this paper the cost based task scheduling and dynamically optimized source allocation strategies has been surveyed.

ISSN(Online): 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 6, June 2016

II. EXISTING SYSTEM

In existing system we have witnessed many scientific applications partially or wholly shifting from traditional computing raised area (e.g., grid) to the cloud. Due to the pay-as-you-go computational behavior, performance and (monetary) cost optimizations enclose recently become a blistering research topic for workflows in the cloud. To address the limitations of current approaches, we propose Profit Maximization, a transformation-based optimization framework for optimizing the performance and cost of workflows in the cloud. Profit Maximization models the cost and performance optimizations of workflows as transformations It performance and monetary cost optimizations for workflows from various applications in the cloud have become a blistering research topic. Yet, we find that most existing studies adopt ad hoc optimization approaches, which fail to capture the key optimization opportunities for different work source cost s and cloud offerings (e.g., virtual machines with different prices).

Drawbacks of Existing system:

- ✓ This TOF Planning has tendency to make administration inflexible.
- ✓ There is no range for character freedom on performance and cost of workflows in the cloud.
- ✓ Detailed planning may create a fake sense of security to the effect that everything is taken for granted.
- ✓ Consequently they cloud service may be stop working to take up timely actions and an opportunity is lost.

PROPOSED SYSTEM

The source allocation approach is based on locate the many risk in Profit Maximization on multiple clouds. At rest there are many practical and exigent issues for current multi-cloud environments. Individual's issues include relatively partial cross-cloud network bandwidth and lacking of cloud standards among cloud providers relies on the assumption that all qualified nodes must satisfy Inequalities in existing system. To meet this requirement, we design a source discovery protocol, namely pointer-gossiping PG-TOF, to find these qualified nodes. We choose PG-TOF to adapt to the multidimensional feature. Like traditional PG-TOF, each node (a.k.a., duty node) under PG-TOF is responsible for a distinctive multidimensional range zone arbitrarily selected when it joins the overlie. It calls them neighbors to each other. Some of them are inherit in the process of planning like severity and other occur due to shortcoming of the techniques on multi cloud by themselves in this proposed work. A Profit Maximization, a general transformation-based optimization framework for workflows in the cloud. Specifically, Profit Maximization formulates six basic workflow transformation operations. A random performance and cost optimization process PG-TOF be represented as a transformation plan (i.e., a sequence of basic transformation operations including Amazon EC2 and Rack space. Our experimental results demonstrate the effectiveness of Profit Maximization in optimizing the performance and cost in comparison with other existing approaches.

III. MODULE DESCRIPTION

HANDLING TASK COST SCHEDULING MODULE:

In this module, the merge operation performs when two vertices are allocated to the occasion of same type. The vertices are allocated to one after another. The request node of the instance DAG is united to form the super node and maintain the hierarchical relationship and structural enslavement among the nodes in DAG. The demote operation achieves the execution of single vertex by assigning it to the cheaper instance which causes the longer execution time. The dependencies of the relegate vertex also belated by the demoted vertex. The moving operation is used for moving one task after the end of another task to condense the task. The dependencies of the moved vertex were also belated by the moved vertex.

CO-SCHEDULING MODULE COST OPTIMIZATION MODULE:

The co-scheduling process is performed when multiple tasks running at the equivalent time. The multiple tasks which have comparable start time and end time with comparable leftover time for deadline can be run at the same instance type. Optimization Sequence, ToF Planner This ToF planner aids to find the sequence of operation to be performed for outlay and performance optimization. The planner has three designs. First the planner scuttles periodically. Second the main schemes and assisting schemes are performed alternatively and cost model avoids the discarded transformation operations.

ISSN(Online): 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 6, June 2016

COST BASED OPERATION HANDLING:

Merge operation: The merge operation performs when two vertices are allocated to the instances of similar type. The vertices are allocated to one after another. The instance nodes of the instance DAG are mutual to form the super node and maintain the hierarchical relationship and structural dependencies among the nodes in DAG.

Demote operation: The demote operation performs the execution of single vertex by conveying it to the cheaper instance which causes the longer execution time. The dependency of the demote vertex also belated by the demoted vertex also delayed by the demoted vertex.

Move operation: The moving operation is used for moving one task after the end of one more task to reduce the task. The dependencies of the moved vertex were also delayed by the moved vertex.

Split Operation: It perform when more urgent task need to run on the same type instance by recessing the current task for an exacting time. The balanced technique can be resumed by the checkpoint technique after the completion of the urgent task.

Promote operation. The promote operation is performed during the execution of the task to a superior or costlier occasion for decreasing the execution time. The promote operation are essentially performed to satisfy the deadlines. The promote operation maintain with the merge operation to exploit the instances.

System	Pentium IV 2.4 GHz
Hard Disk	40 GB
RAM	256 MB
Key Board	LG
Mouse	Logitech
Floppy Drive	1.44 MB
Monitor	15 inch VGA Color monitor

Table1: Software Description

SOFTWARE DESCRIPTION

The software requirement specification is created at the end of the analysis task. The function and performance allocated to software as part of system engineering are developed by establishing a complete information report as functional representation, a representation of system behavior, an indication of performance requirements and design constraints, appropriate validation criteria.

CLIENT/SERVER:

A server is anything that has some source that can be shared. There are compute servers, which provide computing power; print servers, which manage a collection of printers; disk servers, which provide networked disk space; and web servers, which store web pages. A client is simply any other entity that wants to gain access to a particular server. A server process is said to “listen” to a port until a client connects to it. A server is allowed to accept multiple clients connected to the same port number, although each session is unique. To manage multiple client connections, a server process must be multithreaded or have some other means of multiplexing the simultaneous I/O.

RESERVED SOCKETS:

Once connected, a higher-level protocol ensues, which is dependent on which port user are using. TCP/IP reserves the lower, 1,024 ports for specific protocols. Port number 21 is for FTP, 23 is for Telnet, 25 is for e-mail, 79 is for finger, 80 is for HTTP, 119 is for Netnews-and the list goes on. It is up to each protocol to determine how a client should interact with the port.

JAVA AND THE NET:

Java supports TCP/IP both by extending the already established stream I/O interface. Java supports both the TCP and UDP protocol families. TCP is used for reliable stream-based I/O across the network. UDP supports a simpler, hence faster, point-to-point datagram-oriented model.

INETADDRESS:

The Inet Address class is used to encapsulate both the numerical IP address and the domain name for that address. User interacts with this class by using the name of an IP host, which is more convenient and understandable

ISSN(Online): 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 6, June 2016

than its IP address. The Inet Address class hides the number inside. As of Java 2, version 1.4, Inet Address can handle both IPv4 and IPv6 addresses.

FACTORY METHODS:

The Inet Address class has no visible constructors. To create an Inet Address object, user use one of the available factory methods. Factory methods are merely a convention whereby static methods in a class return an instance of that class. This is done in lieu of over source costing a constructor with various parameter lists when having unique method names makes the results much clearer.

Three commonly used Inet Address factory methods are:

1. Static Inet Address get Local Host ( ) throws Un known Host Exception
2. Static Inet Address get By Name (String host Name) Throws Unknows Host Exception
3. Static Inet Address [ ] get All By Name (String host Name) throws Unknown Host Exception

The get Local Host ( ) method simply returns the Inet Address object that represents the local host. The get By Name ( ) method returns an Inet Address for a host name passed to it. If these methods are unable to resolve the host name, they throw an Unknown Host Exception.

On the internet, it is common for a single name to be used to represent several machines. In the world of web servers, this is one way to provide some degree of scaling. The get All By Name ( ) factory method returns an array of Inet Addresses that represent all of the addresses that a particular name resolves to. It will also throw an Unknown Host Exception if it can't resolve the name to at least one address. Java 2, version 1.4 also includes the factory method get By Address ( ), which takes an IP address and returns an Inet Address object. Either an IPv4 or an IPv6 address can be used.

INSTANCE METHODS:

The Inet Address class also has several other methods, which can be used on the objects returned by the methods just discussed. Here are some of the most commonly used.

Boolean equals (Object other)-Returns true if this object has the same Internet address as other.

1. Byte [ ] get Address ( )- Returns a byte array that represents the object's Internet address in network byte order.
2. String get Host Address ( ) - Returns a string that represents the host address associated with the Inet Address object.
3. String get Hostname ( ) - Returns a string that represents the host name associated with the Inet Address object.
4. Boolean is Multicast Address ( )- Returns true if this Internet address is a multicast address. Otherwise, it returns false.
5. String to String ( ) - Returns a string that lists the host name and the IP address for convenience.

TCP/IP CLIENT SOCKETS:

TCP/IP sockets are used to implement reliable, bidirectional, persistent, point-to-point and stream-based connections between hosts on the Internet. A socket can be used to connect Java's I/O system to other programs that may reside either on the local machine or on any other machine on the Internet. There are two kinds of TCP sockets in Java. One is for servers, and the other is for clients. The Server Socket class is designed to be a "listener," which waits for clients to connect before doing anything. The Socket class is designed to connect to server sockets and initiate protocol exchanges.

The creation of a Socket object implicitly establishes a connection between the client and server. There are no methods or constructors that explicitly expose the details of establishing that connection. Here are two constructors used to create client sockets:

- Socket (String host Name, int port) - Creates a socket connecting the local host to the named host and port; can throw an Unknown Host Exception or an IO Exception.
- Socket (Inet Address ip Address, int port) - Creates a socket using a preexisting Inet Address object and a port; can throw an IO Exception.

A socket can be examined at any time for the address and port information associated with it, by use of the following methods:

- Inet Address get Inet Address ( ) - Returns the Inet Address associated with the Socket object.
- Int get Port ( ) - Returns the remote port to which this Socket object is connected.

ISSN(Online): 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 6, June 2016

- Int get Local Port ( ) - Returns the local port to which this Socket object is connected.

Once the Socket object has been created, it can also be examined to gain access to the input and output streams associated with it. Each of these methods can throw an IO Exception if the sockets have been invalidated by a loss of connection on the Net.

- Input Stream get Input Stream ( ) - Returns the Input Stream associated with the invoking socket.
- Output Stream get Output Stream ( ) - Returns the Output Stream associated with the invoking socket.

TCP/IP SERVER SOCKETS:

Java has a different socket class that must be used for creating server applications. The ServerSocket class is used to create servers that listen for either local or remote client programs to connect to them on published ports. ServerSockets are quite different form normal Sockets.

When the user create a ServerSocket, it will register itself with the system as having an interest in client connections.

- ServerSocket(int port) - Creates server socket on the specified port with a queue length of 50.
- Serversocket(int port, int maxQueue) - Creates a server socket on the specified port with a maximum queue length of maxQueue.
- ServerSocket(int port, int maxQueue, InetAddress localAddress)-Creates a server socket on the specified port with a maximum queue length of maxQueue. On a multihomed host, localAddress specifies the IP address to which this socket binds.
- ServerSocket has a method called accept( ) - which is a blocking call that will wait for a client to initiate communications, and then return with a normal Socket that is then used for communication with the client.

URL:

The Web is a loose collection of higher-level protocols and file formats, all unified in a web browser. One of the most important aspects of the Web is that Tim Berners-Lee devised a scalable way to locate all of the sources of the Net. The Uniform Source Locator (URL) is used to name anything and everything reliably. The URL provides a reasonably intelligible form to uniquely identify or address information on the Internet. URLs are ubiquitous; every browser uses them to identify information on the Web.

SYSTEM DESIGN

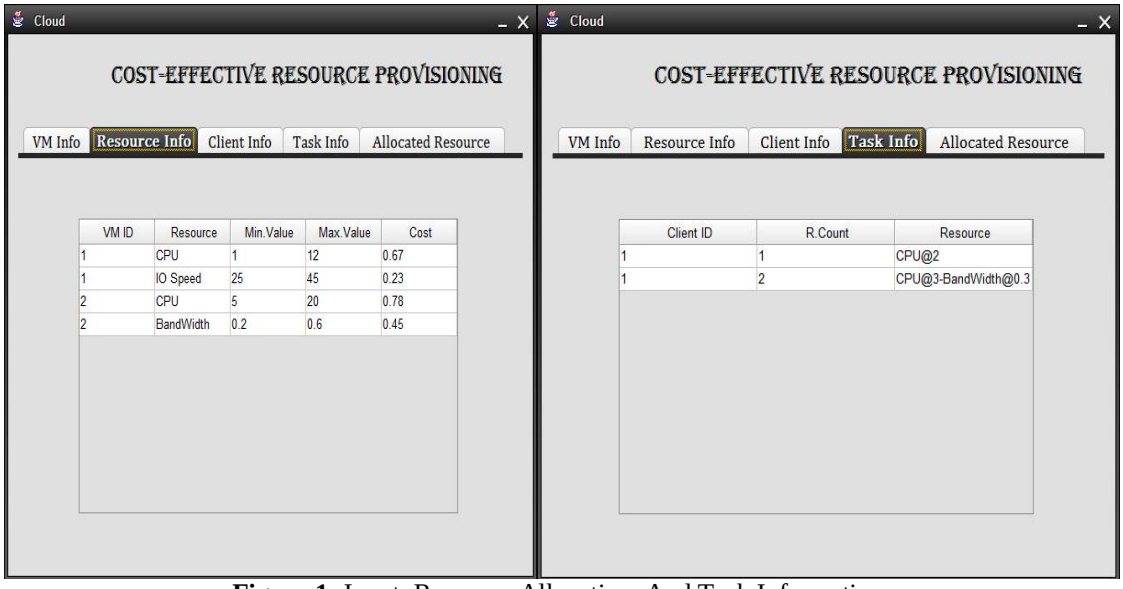

Figure 1: Input. Resource Allocation. And Task Information

ISSN(Online): 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 6, June 2016

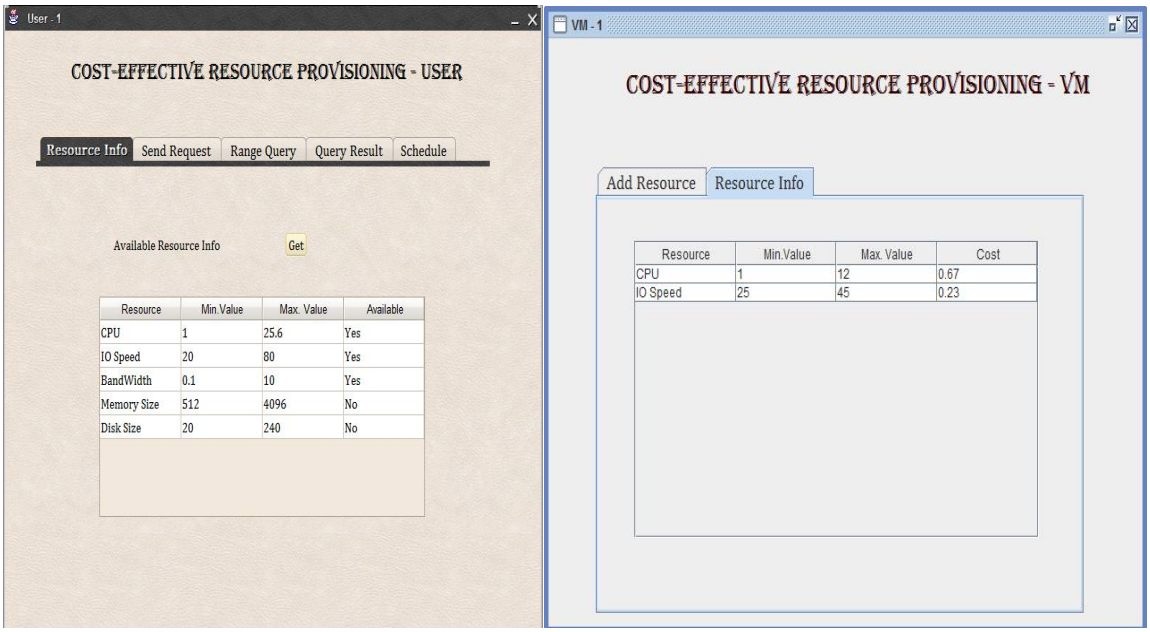

Figure 2: Checking Availability of Resource In User1 And Resource Information In Virtual Machine1

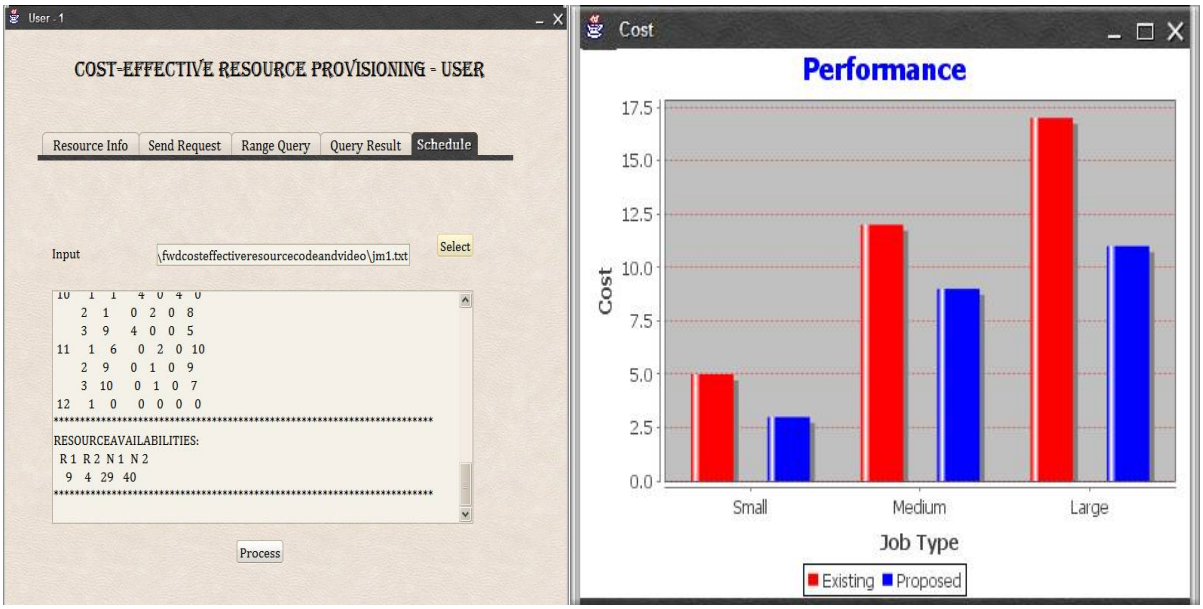

Figure 4 : Process Scheduling In User1 And Output Cost Estimation Graph

IV. CONCLUSION

The proposed work, resource utilization is highly optimized through making sure that every computing resource is distributed efficiently by using Pointer Gossip Content Addressable Network algorithm. The solution is formulated which can optimize the task execution performance based on its assigned resources under the user budget by using Trade of Planner algorithm. Resources contention problem is minimized. The cost and performance

ISSN(Online): 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 6, June 2016

optimization become the most important issue during scheduling in cloud computing. In this paper various workflow management techniques have been surveyed.

REFERENCES

[1] H.M. Fard, R. Prodan, and T. Fahringer, "A Truthful Dynamic Workflow Scheduling Mechanism for Commercial Multicloud Environments," IEEE Trans. Parallel and Distributed Systems, vol. 24, no. 6, pp. 1203-1212, June 2013.
[2] J.-S. V€ockler, G. Juve, E. Deelman, M. Rynge, and B. Berriman,"Experiences Using Cloud Computing for a Scientific Workflow Application," Proc. Second Int'l Workshop Scientific Cloud Computing (ScienceCloud, pp. 15-24, 2011.
[3] S. Abrishami, M. Naghibzadeh, and D.H.J. Epema, "Cost-Driven Scheduling of Grid Workflows Using Partial Critical Paths," IEEE Trans. Parallel and Distributed Systems, vol. 23, no. 8, pp. 1400-1414, Aug. 2012.
[4] A. Ben Othman, J. Nicod, L. Philippe, and V. Rehn Sonigo, "Optimal Energy Consumption and Throughput for Workflow Applications on Distributed Architectures," Proc. IEEE 21st Int'l
Workshop Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), pp. 115-120, 2012.
[5] X. Tang, C. Chen, and B. He, "Green-Aware Workload Scheduling in Geographically Distributed Data Centers," Proc. IEEE Fourth Int'l Conf. Cloud Computing Technology and Science (CLOUDCOM), pp. 82-89, 2012.
[6] Windows Azure Case Studies, "http//www.microsoft.com/ azure/casestudies.mspx." 2014.
[7] E.-K. Byun, Y.-S. Kee, J.-S. Kim, and S. Maeng, "Cost Optimized Provisioning of Elastic Resources for Application Workflows," Future Generation Computer Systems, vol. 27, no. 8, pp. 1011-1026,2011.
[8]. J. Lucas Simarro, R. Moreno Vozmediano, R. Montero, and I. Llorente, "Dynamic Placement of Virtual Machines for Cost Optimization in Multi-Cloud Environments," Proc. Int'l Conf. High Performance Computing and Simulation (HPCS), pp. 1-7, 2011.\
[9] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud Computing and Grid Computing 360-Degree Compared," Proc. Grd Computing Environments Workshop (GCE), pp. 1-10, 2008.
[10] X. Tang, C. Chen, and B. He, "Green-Aware Workload Scheduling in Geographically Distributed Data Centers," Proc. IEEE Fourth Int'l Conf. Cloud Computing Technology and Science (CLOUDCOM), pp. 82-89, 2012.