A VIRTUALIZED DISTRIBUTED COMPUTING INFRASTRUCTURE OF CLOUD ENVIRONMENT

K.Thamaraiselvi[1] K.Dinesh Kumar[2] G.T.Rajaganapathy[3]

ABSTRACT

Along with the development of grid computing and cloud is finite computing resources supplied by own server usually become the bottleneck in the process of system implementation of online trading platform and it is difficult to reuse algorithm module and the system also becomes more complicated to achieve. Traditional online trading framework cannot preferably solve above the problem . so we introduce a framework of online trading platform based on cloud environment. It is sufficient to set up a a high complexity online trading platform using infinite virtual computing capability provided by cloud and at the same time this framework can recognize the sharing of data sets after the integration of heterogeneous and distributed data based on layered thought way, and supply theoretical foundation and framework support for setting up a combined,parallel and distributed online trading platform.

Keywords - resource; heterameworkheterogeneous; distributed.

I. INTRODUCTION

Cloud computing is a new way of delivering computing resources, not a *new technology*. Computing services ranging from data storage and processing to software, such as email handling, are now available instantly, commitment-free and on-demand. Since we are in a time of belt-tightening, this new economic model for computing has found fertile ground and is seeing massive global investment. [1] There are three categories of cloud computing, **Software as a service (SaaS)** is software offered by a third party provider, available on demand, usually via the Internet configurable remotely. Examples include online word processing and spreadsheet tools, CRM services and web content delivery services (Salesforce CRM, Google Docs, etc). **Platform as a service (PaaS)** allows customers to develop new applications using APIs deployed and configurable remotely. The platforms offered include development tools, configuration management, and deployment platforms. Examples are Microsoft Azure, Force and Google App engine. **Infrastructure as service (IaaS)** provides virtual machines and other abstracted hardware and operating systems which may be controlled through a service API. Examples include Amazon EC2 and S3, Terremark Enterprise Cloud, Windows Live Skydrive and Rackspace Cloud. [3]

It is hardly necessary to repeat the many rain-forests' worth of material which has been written on the economic,

[1] Assistant Professor, Department computer science and engineering K.S.R College of engineering Namakkal D.t-Tamilnadu Thamaraikrishna.210@gmail.com

[2] Assistant Professor, Department Cse, K.S.R College of engineering, Namakkal D.t- Tamilnadu, dinesh.mecse@gmail.com

[3] Assistant Professor, Department Cse, K.S.R College of engineering, Namakkal D.t- Tamilnadu, dinesh.mecse@gmail.com

technical and architectural and ecological benefits of cloud computing. However, in the direct experience of the members of our expert group, as well as according to recent news from the 'real world', an examination of the security risks of cloud computing must be balanced by a review of its specific security benefits. [1]Cloud computing has significant potential to improve security and resilience. What follows is a description of the key ways in which it can contribute.

Put simply, all kinds of security measures are cheaper when implemented on a larger scale. Therefore the same amount of investment in security buys better protection. This includes all kinds of defensive measures such as filtering, patch management, hardening of virtual machine instances and hypervisors,[4] human resources and their management and vetting, hardware and software redundancy, strong authentication, efficient role-based access control and federated identity management solutions by default, which also improves the network effects of collaboration among various partners involved in defense.[1]

II. Problem And Solution

Later on, some experts proposed the distributed parallel computing platform of the Knowledge Grid based on Globus Toolkit, which makes full use of grid computing provided by the Globus Toolkit, and it solves the problem of low capability in traditional stand-alone computing. Therefore, this framework has become cornerstone of studies on such issue in recent years, and lots of corrective methods based on **Glob us Toolkit have been proposed for this platform. [2]**

However, it is difficult to realize commercial applications by using grid computing technologies mentioned above. What's more, systems based on the traditional grid computing are weak in computing. Therefore, computing capability has become the bottleneck of systems using traditional grid computing, which demands higher hardware requirements. In our proposed approach is a kind of computing platform distributed in large-scale data center, which meets the needs of scientific research and e-commerce by dynamically providing several kinds of server resources. [1]

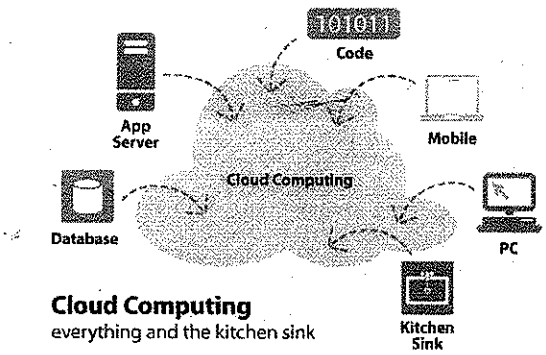


Figure 1. Overview Representation of cloud computing

It can be viewed as the evolution of distributed computing, parallel computing and grid computing and so on. Cloud computing platform use the virtualization technology to dynamically and transparently supply virtual computing and storage resources to satisfy user's different requirements according to the relative scheduling strategy. [7]

Moreover, it can also dynamically reclaim resources unused by current user for other user and provide users with low-cost computing and storage resources just like power plant supplying enough power so that normal user can carry on large-scale parallel computing and operations on mass data.[6]

It is obvious that this computing method supplies underlying support for setting up a unified and open knowledge grid system. [7]This paper proposes a kind of scalable and open online trading platform framework based on cloud computing. Furthermore, components of this framework can be extended and reused conveniently.

Example: Large-scale processing of structured Web data

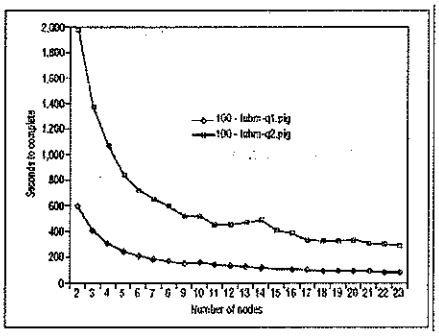


Figure 2. Performance Measurement (No of nodes Vs seconds)

Table 1

Name	Description
IGGs	InterGrid Gateways
VM	Virtual Machine
DVE	Distributed Virtual Environment
SSH	Secure Shell
VMM	Virtual Machine Manager
VIEs	Virtual Infrastructure Engines
EMO	Evolutionary Multi - Criterion Optimization

III. System Perfomance

The IGG works with a repository of VM templates that is, the gate-way administrator can register templates to the repository to let users find and request instances of specific VMs.[6] In addition, the gateway administrator must upload the images to Ama-zon if the gateway uses the cloud as a resource provider. Users currently can't submit their own templates or disk images to the gateway.

A. Distributed VE Manager

A DVE manager interacts with the IGG by making requests for VMs and querying their status. The DVE manager requests VMs from the gateway on behalf of the user application it represents.

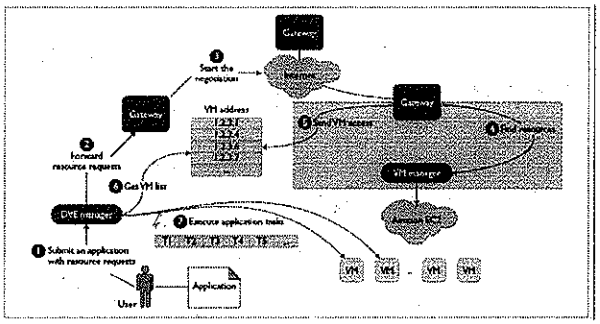


Figure 3. The main interactions among InterGrid components.

When the reservation starts, the DVE man-ager obtains the list of requested VMs from the gateway. This list contains a tuple of public IP/private IP for each VM, which the DVE man-ager uses to access the VMs (with Secure Shell [SSH] tunnels). With EC2, VMs have a public IP, so the DVE can access the VMs directly with-out tunnels. Then, the DVE manager deploys the user's application. [4]

B. Intergrid gateway at runtime

Figure 3 shows the main interactions between InterGrid's components. When the user first requests a VM, a command-line interface handles the request. Users must specify which VM template they want to use; they can also specify the number of VM instances, the ready time for the reservation, the deadline, the wall time (that

is, the time the user esti-mates the job will take), and the address for an alternative gateway. The client returns an identifier for the submitted request from the gateway. Next, the user starts a DVE man-ager with the returned identifier (or a list of identifiers) and its application as parameters. The application is described via a text file in which each line is one task to execute on a remote VM. (The task is indeed the command line that runs with SSH.) The DVE manager waits until InterGrid has scheduled or refused the request. The local gateway tries to obtain resources from the underlying VIEs.[9] When this isn't possible, the local gateway starts a negotiation with any remote gateways to ful-fill the request. When a gateway can fulfill the request — that is, can schedule the VMs — it sends the access information for connect-ing to the assigned VM to the requester gate-way. Once this gateway has collected all the VM access information, it makes it available for the DVE manager. Finally, the DVE man-ager configures the VM, sets up SSH tunnels, and executes the tasks on the VM. In future work, we want to improve the description of applications to allow file transfer, dependen-cies between tasks, and VM configuration. Under the peering policy we consider in this work, each gateway's scheduler uses conser-vative backfilling to schedule requests. When the scheduler can't start a request immediately using local resources, then a redirection algo-rithm will take the following steps:[8]

1. Contact the remote gateways and ask for offers containing the earliest start time at which they would be able to serve the request, if it were redirected.
2. For each offer received, check whether the request start time a peering gateway pro-poses is that given by local resources. This being the case, the algorithm redirects the request; otherwise, the algorithm will check the next offer.
3. If the request start time that local resources have given is better than those the remote gateways have proposed, then the algorithm will schedule the request locally.

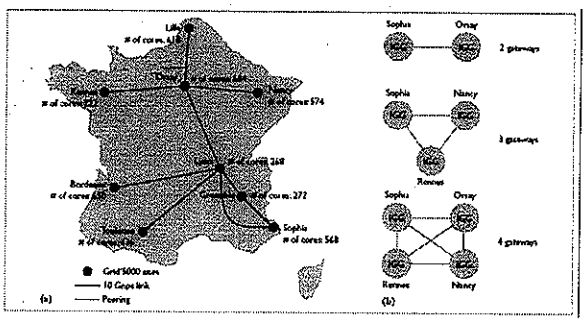


Figure 4. InterGrid testbed over Grid'5000. We can see the Grid'5000 sites as well as the gateway configurations we evaluated.

Table 2

Table 1. Job slowdown improvement under different gateway configurations.

Site	Two gateways	Three gateways	Four gateways
Orsay	-0.09006	N/A	0.00010
Nancy	N/A	3.35780	4.30864
Rennes	N/A	7.84212	12.81716
Sophia	0.20168	-6.12047	-3.12708

C. Peering Arrangements

For our testing, we used the French experimen-tal grid platform, Grid'5000, as both a scenario and a test bed. Grid'5000 comprises nine sites geographically distributed across France, and currently features 4,792 cores. Each gateway created in this experiment represents one Grid'5000 site; the gateway runs on that site. To prevent gateways from inter-fering with real Grid'5000 users, we

used the emulated VMM, which instantiates fictitious VMs. The number of emulated hosts is the num-ber of real cores available on each site. Figure 9 illustrates the Grid'5000 sites and the evaluated gateway configurations. We generated the site's workloads using Uri Lublin and Dror G. Feitelson's model, 8 which we refer to here as Lublin99. We configured Lublin99 to generate one-day-long workloads; the maximum number of VMs that generated requests require is the number of cores in the site. To generate different workloads, we set the mean number of VMs that a request requires (specified in log2) to log2m – *umed*, where m is the maximum number of VMs allowed in the system. We randomly varied *umed* from 1.5 to 3.5. In addition, to simulate a burst of request arrivals and heavy loads, thus stretching the system, we multiplied the inter arrival time by 0.1.[8],[10]

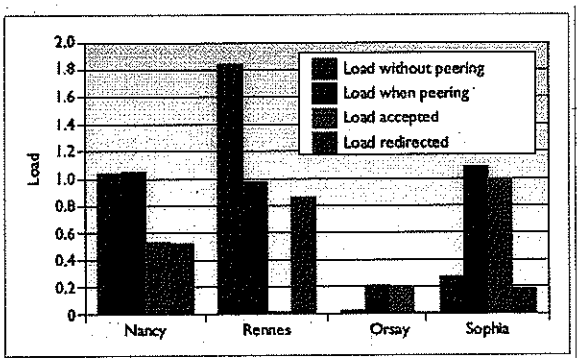


Figure 5. Load characteristics under the four-gateway scenario.

Figure 5 shows the load characteristics under the four-gateway scenario. The teal bars indicate each site's load when they aren't inter-connected;[5] the magenta bars show the load when gateways redirect requests to one another; the green bars correspond to the amount of load each gateway accepts from other gateways; and the brown bars represent the amount of load redirected. The results show that the policy the gateways use balances the load across sites, making it tend to 1.[6] Rennes, a site with heavy load, benefits from peering with other gateways as the gateway redirects a great share of its load to other sites. Table 2 presents the job slowdown improve-ment resulting from gateway interconnection.

Overall, the interconnection improves job slow Nancy down — for example, sites with the heaviest loads (that is, Rennes and Nancy) have better improvements. However, the job slowdown of sites with lower loads gets worse; as the number of gateways increases, though, this impact is minimized, which leads to the conclusion that sites with light loads suffer a smaller impact when more interconnected gateways are pres-ent. This experiment demonstrates that peering is overall beneficial to interconnected sites — these benefits derive from load balancing and overall job slowdown improvement.[6]&[7]

D. Deploying a Bag-of-Tasks Application

For our second experiment, we considered *Evo-lutionary Multi-Criterion Optimization* (EMO), a bag-of-tasks application for solving optimi-zation problems using a multi-objective evolu-tionary algorithm. Evolutionary algorithms are a class of population-based metaheuristics that exploit the concept of population evolution to find solutions to optimization problems. They can find the optimal solution using an iterative process that evolves the collection of individu-als to improve the solution's quality. Each task is an EMO process that explores a different set of populations.[10]

Figure 6 shows the test bed for running the experiment. We carried out each test in two steps. First, we evaluated EMO's execution time using a single gateway, and then we forced InterGrid to provide resources from two gate-ways. In this case, we limited the number of available cores for running VMs, and the DVE manager submitted two requests. For VMs, we limited both gateways to five cores, and the DVE manager sent two requests for five VMs each. Next, for 20 VMs, we set the limit to 10 cores, and the DVE manager requested 10 VMs twice.[3] The two gateways used resources from Amazon EC2 — the requests demanded a small EC2 instance running Windows Server 2003. Table 3 reports both steps' results. The execu-tion time of the bag-of-tasks application doesn't suffer important performance degradations with one or two gateways.

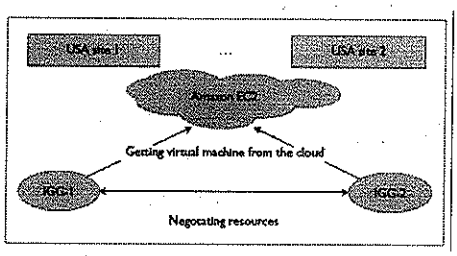


Figure 6. Test bed used to run Evolutionary Multi-Criterion Optimization on a cloud computing provider. The test bed is composed of two InterGrid gateways (IGGs), each using resources from Amazon EC2

Table 3

Table 2. Experimental results with one and two gateways using resources from Amazon EC2

Number of virtual machines (VMs)	One gateway (seconds)	Two gateways* (seconds)
5	4,780	–
10	3,017	3,177
15	2,407	–
20	2,108	2,070

*Each gateway provides half of the VMs

Our experiments with InterGrid have shown that it can balance load between distributed sites and have validated that a bag-of-tasks application can run on distributed sites using VMs.[8] We currently provide a minimal gateway that lets resource providers interconnect sites and deploy VMs on different kinds of infra-structures, such as local clusters, Amazon EC2, and Grid'5000.

In future work, we plan to improve the VM template directory to let users submit their own VMs and synchronize the available VMs between gateways. In addition, although we haven't addressed security aspects in this work because they're handled at the operating system and network levels, it would be interesting to address those concerns at the InterGrid level.[10]

IV. Conclusion

In this paper, we propose a new online trading platform based on cloud computing, which possesses not only high flexibility, high reliability, low-level transparency, security and other features in cloud computing, but also openness, reusability, scalability as three major characteristics. It provides a theoretical basis for achieving safe, reliable online transactions and framework support for implementing an algorithm reusing, knowledge sharing, unified and open online trading platform.

Acknowledgment

We also express our thanks to our institution, parents, and friends, well wishers for their encouragement and best wishes in the successful completion of this dissertation.

References

[1] Foster I, Kesselman C. Globus: *"a metacomputing infrastructure toolkit."* International Journal of Supercomputing Applications 11, 1997, pp. 115-128.

[2] Foster I., Kesselman C. (eds.) The Grid: *"Blueprint for a Future Computing"* Inf., Morgan Kaufmann Publishers, 1999, pp.105-129.

[3] GA Jan S Rellermeyer. *"Services everywhere: Osgi in distributed environments."* In EclipseCon 2007, Santa Clara, CA, 2007.

[4] L Richardson, S Ruby. *"Restful web services."* O'Reilly. 2007

[5] F. Cappello et al., *"Grid'5000: A Large-Scale and Highly Reconfigurable Grid Experimental Testbed,"* Int'l J. High Performance Computing Applications, vol. 20, no. 4, 2006, pp. 481–494.

[6] X. Zhu et al., *"1000 Islands: Integrated Capacity and Workload Management for the Next Generation Data Center,"* Proc. Int'l Conf. Autonomic Computing (ICAC 08), IEEE CS Press, 2008, pp. 172–181.

[7] M. Armbrust et al., Above the Clouds: *"A Berkeley View of Cloud Computing, tech."* report UCB/EECS-2009-28, EECS Dept., Univ. of California, Berkeley, Feb. 2009.

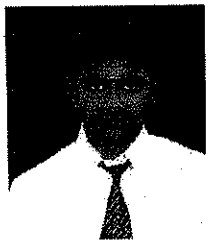
[8] M. Dias de Assunção, R. Buyya, and S. Venugopal, *"InterGrid: A Case for Internetworking Islands of Grids,"* Concurrency and Computation: Practice and Experience, vol. 20, no. 8, 2008, pp. 997–1024.

[9] K. Keahey et al., *"Virtual Workspaces: Achieving Quality of Service and Quality of Life in the Grids,"* Scientific Programming, vol. 13, no. 4, 2006, pp. 265–275.

[10] J.S. Chase et al., *"Dynamic Virtual Clusters in a Grid Site Manager,"* Proc. 12th IEEE Int'l Symp. High Performance Distributed Computing (HPDC 03), IEEE CS Press, 2003, p. 90.

AUTHORS' BIOGRAPHY

Ms.K.Thamaraiselvi, received her B.Tech Degree in 2010, from Anna University-Chennai and ME Degree in 2012 from Karpagam University –Coimbatore. Now she is working has Assistant Professor at K.S.R College of Engineering-Tiruchengode



Mr.**K.Dinesh Kumar** received his B.Tech Degree in 2007, from Anna University-Chennai and ME in 2011from Anna University-Chennai. Now he is working as Assistant Professor at K.S.R College of engineering-Tiruchengode

Mr.G.T.Rajaganapathy received his B.E Degree in 2010, from Anna University-Chennai and ME in 2014from Anna University-Chennai. Now he is working as Assistant Professor at K.S.R College of Engineering-Tiruchengode