RMORM: A framework of Multi-objective Optimization Resource Management in Clouds

Wenyun Dai, Haopeng Chen, Wenting Wang, Xi Chen
REINS Group, School of Software
Shanghai Jiao Tong University
Shanghai, P.R.China
{scorpiodwy, chen-hp, wwtvanessa, chenxi_sjtu} @ sjtu.edu.cn

***Abstract*—Cloud computing has attracted increasing attention in recent years. With the growth in the number and frequency of applications being deployed into clouds, the burden of resource management of cloud providers is becoming heavier. The resource deployment must satisfy the need about the performance, availability and reliability of applications from the view of clients, but also ensure the high resource utilization of the cloud providers. In this paper, we design a multi-objective serial optimization with priorities approach, named RMORM, to find the resource deployment in clouds rapidly. This approach is of great practical significance and engineering value and scalable to add new constraints.**

***Keywords*—Cloud-computing, SLA, Resource Management, Multi-objective**

I. Introduction

Cloud computing is the realization of the idea of large-scale computing as a utility or service. Recent years, cloud computing becomes one of the biggest concerns in modern internet technologies and concepts. Many IT-companies, such as Amazon, Google, IBM and Microsoft, launch their products about cloud services including infrastructure-as-a-service (IaaS), platform-as-a-service (PaaS) and software-as-a-service (SaaS). These services are usually described by service level agreement (SLA), which is a part of a service contract where a service is formally defined. In the service contract, customers propose their needs about the performance, the availability, the reliability and others of their applications which will be deployed in cloud. Managing resources at large scale while satisfying the needs of both clients and providers is a key challenge for any cloud service provider.

As states in [1], providing a performance guarantee becomes necessary if cloud providers oversubscribe the resources of physical servers to decrease the number of physical servers used and increase their utilization. Further, enterprises purchasing cloud based services demand a minimal level of performance guarantee. Unfortunately none of cloud providers offer the performance guarantees for their cloud service.

Mike Lees, business manager for Hardware PT, says: For me the key benefit is that cloud is a useful tag that ties together several technologies that are all about the holy grail of manufacturing: availability. This concept has to be at the root of the decisions we make about cloud and, in my opinion, at the heart of all the decisions we make about industrial computing. [2]. Most public cloud providers, such as Amazon, Microsoft and Rackspace promise that they can offer client cloud services with guarantee about availability.

Amazon provides block level storage volumes, named Elastic Block Store (EBS) [3], which provides highly available, highly reliable, predictable storage volumes that can be attached to a running Amazon EC2 instance and exposed as a device within the instance. Azure ensures 99.95% availability, and Windows Azure Traffic Manager and SQL Azure Data Sync can be used to improve the availability to 99.999%. [4] Rackspace guarantee that their service, including Block Storage, Databases, Files, Load-Balancers and Servers, will be available 99.9% or higher of the time in any given monthly billing period. [5]

It is clear that the cloud providers can only promise the performance of their physical servers, since the performance of applications is application-dependent, and if a provider wants to assure the performance of the whole system, he must collect, manage and analyze a huge amount of applications, and this task may be even harder than operating a public cloud. As a result, the cloud providers always ignore offering the performance of applications and leave it to the users who own the applications. Meanwhile, the availability of applications is similar to the performance, and the availability that the cloud providers offered is the availability of their infrastructure, instead of the availability of the whole system. It is possible that the performance and the availability of the physical servers are high, but the application ran on them has a poor performance and availability. On one hand, the cloud providers can't ensure these indexes. On the other hand, the cloud users usually can't manage these indexes due to the lack of professional ability. Therefore it is significant that our system helps cloud providers to satisfy users' needs and generates a deployment approach which is of great engineering value.

In this paper, our goal is designing a multi-objective optimal resource management system to help providers to estimate and satisfy the users' requirements, including the performance, the availability, the cost, the reliability. Meanwhile, this system also improve the resource utilization to save cost for providers.

The rest of this paper is organized as follows. Section 2 discusses the related work about the resource management in cloud. Section 3 analyzes the design principle of the system. Section 4 describes the architecture of the system and flow of how to reaching a resource deployment approach. Simulation results are given is the Section 5. Conclusion and future work are presented in Section 6.

II. Related Work

Salman A. Baset [1] broke down a cloud SLA into several components, and used it for comparing SLAs of well known public IaaS providers. He indicated that none of the IaaS providers offered any performance based SLAs for compute services. Moreover, all cloud providers left the burden of providing evidence for SLA violation on the customer. He then discussed how SLAs should be defined for future cloud services.

Hadi G. et al. [6] [7] considered VM placement to minimize the power and migration cost in a cloud system. They proposed an algorithm based on convex optimization method and dynamic programming to meet the constraints on response time for the clients. However, the power and migration cost are just the constraints from the view of providers. For clients, this paper only considered the response time, which can be attributed to the performance, and besides performance there are some other important requirements from clients, such as the availability and the reliability. Jonathan L. et al. [8] also proposed similar algorithms to improve the performance and reduce SLA violation.

Astrid U. et al. [9] discussed the SLAs about the availability in cloud and put forward three improvements to be made. First, the SLAs must become more detailed with respect to actual KPIs used to define availability. Next, in order to deploy also important enterprise services in clouds, different levels of availability should be offered. Finally, the SLAs should be available on demand, that means that they should be adjustable on demand. They designed an overall availability model for a cloud system, including the network. However, the service they used to model was too simple to describe the web service in clouds, such as a tiered web service. Moreover, the performance should be taken into consideration.

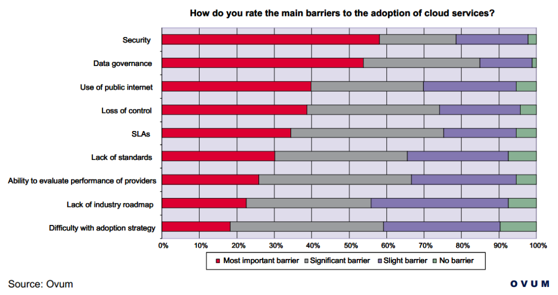

Fig. 1. Barriers to cloud adoption

III. Design Principle

From Fig. 1, we can find that the security, the data, the control, the SLAs and the performance are the main constraints which the users measure their applications in clouds. We can classify the data, the control, the SLAs and the performance together as the resource management. Meanwhile, the resource management approach should consider the profits of providers, such as the resource utilization and the cost.

The performance of an application is mainly affected by the amount of virtual machines and the distance between them. It is obvious that the more virtual machines uses the higher the performance will be, because more virtual machines can provide more computing power. Meanwhile, the distance between virtual machines also impacts the performance.

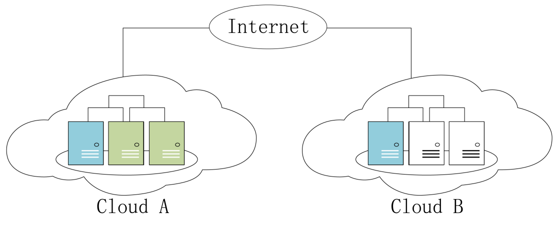

Fig. 2. Two Deployment Approaches

Fig. 2 shows two deployment approaches of some application both of which consist of two physical servers. The two physical hosts of the blue application are respectively in the cloud A and the cloud B, while both of the physical hosts of the green application are in the cloud A. Basically there are two methods of the communication between the virtual machines ran at the physical hosts, as shown in Fig. 3. The first is that the virtual machines communicate with each other, wherever they are running at, to complete some tasks together. The second is that all the virtual machines in the same zone only communicate with a Load-Balancer, then each Load-Balancer belongs to various zones communicates with each other. However, using both of the two patterns, the cost of communication of the blue application is bigger than the green one.

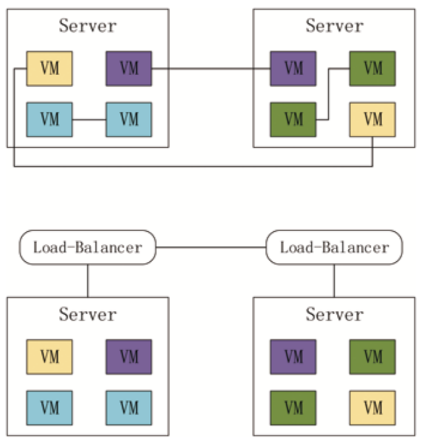

Fig. 3. Two methods of communication between VMs

In the meantime, the distance between virtual machines affects the availability. It is obvious that the availability of the green application in Fig. 2 is lower than the blue one while the performance of the green application is higher than the blue one.

To increase the performance and availability, the amount of virtual machines is the more the better. However, the cost of the application limits the amount of virtual machines. As a result, the resource management is multi-objective, including the performance, the availability and the cost. Unfortunately, the solution, which achieves all the objectives, usually is not exist in practice or spends too much time to find it out. Thus we use the multi-objective optimization model to model our problem and try to find the solution that is of the most important practical significance.

In this paper, we consider the cost, the performance, the availability, the reliability from the view of users and the resource utilization from the view of providers. The security of the application also is an important constraint must be meet, but it is complicated to model and compute. So, we ignore the security and will consider it in the future. Besides the security, there are some other constraints of applications in cloud, and they will be added into consideration afterwards. Thus the architecture we used to model the resource management must be scalable to be convenient for expanding new constraints.

There are three basic means to deal with the problem of resource management in clouds. The first is Multi-objective Parallel Optimization in which the problem is solved by several models in parallel. This type of approach usually uses algorithms like bin-packing, flow network, mixed integer programming and hill climbing or the combination of them. In theory, the deployment found by this approach is optimal, but it may not meet any one specified request, because it possibly is the minimum weighted value from all the targets rather than meeting them all. Furthermore, what even worse, the time-cost of this approach is too high to satisfy the variability, and the worst is that it may cost extremely expensive to add a new constraint.

Jim ZW Li et al. designed Heuristic Packing with MIP (H-MIP) Algorithm to obtain the solution of optimal deployment for clouds. [10] From the angle of arith, the solution computed by CloudOpt is the optimal. However, in the environment of running 500 tasks of 50 applications in 81 hosts, the time cost of computing the solution is 54.547 seconds. Furthermore if adding iterating to include contention, the time for optimization will be 61.62 seconds, and time for LQNS will be 121.86 second for running 20 applications under high stress. The run time is too long to satisfy the mutability of the system with far more than 20 even 50 applications.

The second schema is with feedback loop which sends feedbacks about new evaluation of the whole deployment from the latter model to the former one, as shown in Fig. 4. The former model compares the feedbacks with the requirements, and sends back the validated information to the latter model, if the feedbacks meet the requirements, and sends invalidated sign to the latter model, if the feedbacks fail to satisfy the needs. Suppose that all the feedbacks meet the needs, the final deployment found by this approach will be considerable. However, it obviously cant be ensure that the feedback loop will be convergent, which means this approach may obtain none solution. In addition, the process of finding the optimal deployment may be fluctuate which spends huge time.

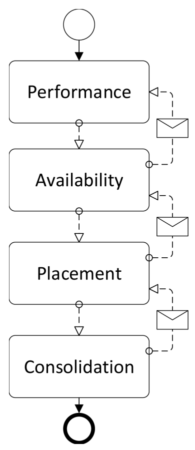

Fig. 4. The flow chart of the deployment schema with feedback loop

The third is Serial Optimization in which the constraints are analyzed and solved serially in each model. The solution obtained by this approach is usually not the optimal one, in that some specified requirement may not be satisfied. To offset the miss of accuracy, priority level can be added into each constraint, and the same to the relevant model. The higher the priority level is, the earlier the constraint is solved, and the more former the model locates in the architecture. This approach has low time cost, and it is convenient for expanding more constraints. Therefore, we choose this approach to manage the resource in cloud.

In the serial process, the amount of virtual machines needed to meet the requirement of users is determined to be calculated firstly because it is the most basic attribute in the whole solution, and it is the precondition of the relative locations of virtual machines. The amount of virtual machines is closely related to the performance of the application. Thus, the performance owns the highest priority. Obviously the Relative Locations of virtual machines are the prerequisite of the Absolute Locations of virtual machines, so the priority of the availability is higher than the placements. Until the virtual machines are placed into physical hosts, the consolidation can be done. For this reason, the priority of the placement is higher than the consolidations. Overall the priorities of these models from high to low are the performance, the availability, the placement and the consolidation.

IV. Architecture and Implement

Our management system architecture, shown in Fig. 5, is composed of the following components:

1) Performance Model: It is the first model in architecture. This model accepts the user's SLAs and satisfies the users' requirements about the performance.

2) Availability Model: This model is responsible for accepting the data from the performance model and obtains the topological structure of virtual machines, which is called the Relative Locations. This Relative Locations meets the clients' need about the availability.

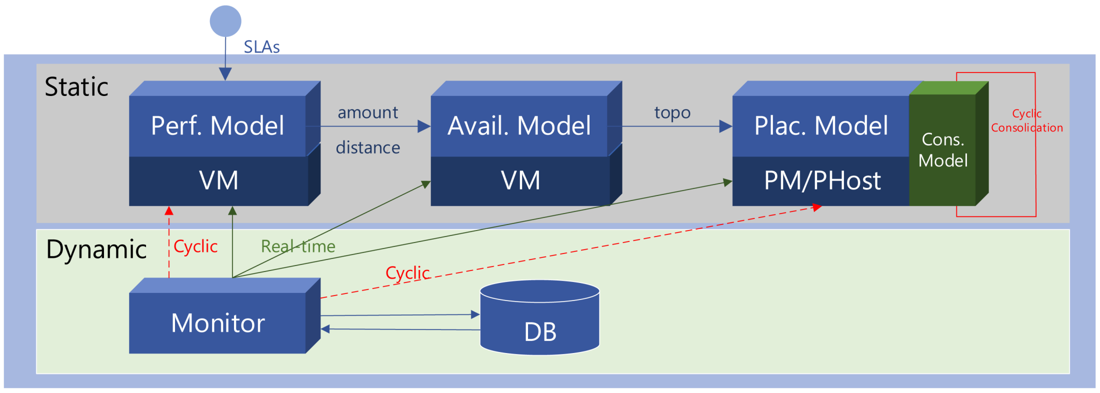

Fig. 5. The architecture

3) Placement Model: It transfers the Relative Locations to the Absolute Locations, which means the real locations of virtual machines.

4) Consolidation Model: This model is responsible for the consolidation, which helps providers to save cost and improve the resources utilization of their physical servers.

5) Monitor: It monitors all the models mentioned above and is responsible for the operation of reading and writing with the database.

At the first time, the system obtains one deployment approach rapidly. In the process of computing this approach, the monitor may not affect the result, so we call this part of system the Static Part. Meanwhile, at the running time, the monitor is collecting and preprocessing data, as a result it can accept and change the deployment approach based on the information of virtual and physical machines, and we call this part the Dynamic Part.

A. Performance Model

The client puts forward his needs of application which will be deployed in cloud. The Performance Model processes the data and obtains the result including the amount of virtual machines that needed to ensure the performance of the application and the max distance between virtual machines can be divided.

For example, Amazon EC2 provides customers three different purchasing models that are On-Demand Instances, Reserved Instances and Spot Instances. [11] On-Demand Instances allow users to pay for computing capacity by the hour with no long-term commitments or upfront payments. Users can increase or decrease their compute capacity depending on the demands of the application and only pay the specified hourly rate for the instances their using; Reserved Instances allow clients to make a low, one-time, upfront payment for a instance, reserve it for a one or three year term, and pay a significantly lower hourly rate for that instance. The clients are assured that their Reserved Instance will always be available for the operating system, choosing by the clients themselves; Spot Instances provide the ability for customers to purchase compute capacity with no upfront commitment and at hourly rates usually lower that the On-Demand rate.

Obviously, these three models are at different prices and appropriate for diverse situations. For purpose of meeting the requirement about performance and saving clients cost, the performance model should work out that how many virtual machines are needed and how to allot these virtual machines to the three kinds of models.

The Fig. 6 shows the page views of worldcupblog.org during recent three months from quantcast. [12] We can roughly get an approach of purchasing resources from Amazon to show the combination usage of the three types of instance purchasing options. Primarily, we split the data by three lines, which are red, orange and purple respectively. Most of time the page views are less than or equal to the red line, so the organization just need to rent the Reserved Instances lasting one or three years. However, sometimes the page views of the website are above the red line but under the purple, in these circumstances, the organization should purchase some On-Demand Instances additionally to ensure the performance of their website. Moreover, in some precious situations, the page views are above the orange line, and these situations usually are transient. As a result, the organization can purchase some Spot Instances, besides the Reserved Instances, to pull through.

The precondition of solving the problem about allocation of virtual resources is the performance prediction. It is necessary for the system to know the throughout at next time node because it will spend some time to decide whether to change the scheme of purchasing resources. We have already had relevant research and design in performance modeling and prediction in [13]. We assumed that the change of users request is relatively periodically during one certain time, e.g. one month, and it has specific amount of request during certain time, e.g. off-peak, sub-peak, and peak time during a single period. Hence we use a time-series method to forecast the future workload based on past workload history. Then we used the algorithm designed in [13], named NumberOfVMsNeeded, to compute the amount of virtual machines needed with the lowest cost.

Moreover, because performance is affected by the communication cost between virtual machines, the performance model

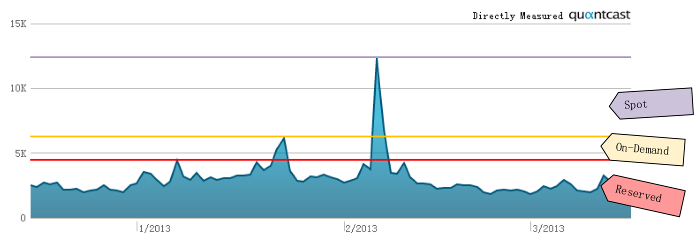

Fig. 6. The Page Views of worldcupblog.org during recent 3 months

should compute the max distance between virtual machines that maintains the performance which meets the clients need, that means the distances between virtual machines should be limited into a reasonable range.

B. Availability Model

Firstly, the Availability Model obtains the data from the performance model including the amount of virtual machines and the maximum distance between them. Then we compute the topological structure of the virtual machines. We already have related work about the virtual machine placement approach. We model the cloud infrastructure as a tree structure. In the tree structure of cloud, from boot to leaf, they are the cloud, the regions, the zones, and the physical hosts, and the virtual machines are placed in the physical hosts. The availability of a virtual machine on a physical server depends on the availability of physical host as well as its parent nodes including the zone and the region. Thus, we should take all the nodes above into consideration to compute the availability of the virtual machines which run at the specific node.

As discussed above, the distance between virtual machines also affects the availability of an application. We designed an availability-aware scaling approach in [14] to improve overall system availability while maintain the communication costs. Firstly, we modeled the cloud structure and the availability of applications must be done in effective and scalable way to support heterogeneous environment. Then we used the four algorithms, including VerticalResizeUp, HorizontalResizeUp, VerticalResizeDown and HorizontalResizeDown, to resize up/down the virtual machines to meet the requirement about availability. Finally, the availability model sends the Relative Locations to the next model.

C. Placement Model

The placement model is responsible for placing the virtual machines into physical hosts based on the Relative Locations, while meeting the users requirement about the reliability of their application deployed in cloud. For data storage, the distributed multi-replica data bases with main/auxiliary function are usually used to increase the reliability. For application, it is obvious that launching more instances on various physical hosts which are far apart can raise the reliability. However, it is also clear that the physical servers that providers have are not unlimited. The placement model reached, considering the reliability, the deployment approach based on the Relative Locations and the real information about physical servers the providers own.

D. Consolidation Model

The consolidation model is mainly design from the view of providers, and the aim of resource consolidation is to save power and reduce the operating cost. The provider hopes that all the occupied physical servers are running at the appropriate status which means the resource utilization of each physical server maintains within a rational range. After running for a while, the deployment scheme computed from the placement model may break the status, and the utilization of resources varies with the runtime workload of the application. Therefore, consolidation should be done to increase the resources utilization and saving cost. When gets notice about the information of physical hosts, the consolidation model will decide to migrate or split these physical hosts. [15] We do these operations of virtual machines limited in the same zone, because that will not affect the availability and the reliability.

E. Monitor

The monitor is watching the states of all the models mentioned above during runtime, and pretreat the information to decide whether write them into the database. The two different methods in which the monitor works are real-time and periodic. Meanwhile, the monitor collects data from the virtual machines and physical servers for different purposes.

1) For the virtual machines: The monitor collects the data of the virtual machines used for the application in cloud. The metrics watched include CPU Utilization (%), Memory Utilization (%), Swap Utilization (%), Disk Space Utilization (%) and so on. We set the thresholds of these metrics, such as the lower bound of CPU Utilization is 20% while the upper bound of it is 75%. If the data collected from one virtual machine is out of the scope, this virtual machine is identified as being over-load or under-loaded. Under this circumstance, the monitor writes the information of this virtual machine into the database for the record, and informs the models that due to this improper virtual machine the performance, the availability may be changed and consolidation may need to be done.

2) For the physical servers: The monitor gathers the data of the resource utilization of physical servers for the consolidation model. The resource utilization of the physical host includes utilizations of CPU, Memory, I/O and Disk Space, and we set the lower and upper bounds of it as well. If the resource utilization of certain physical host is smaller or greater than the bound, the monitor will notice the consolidation model and record the information into database.

F. OSGi

Considering more constraints will be added into the resource deployment in cloud, we use OSGi framework, [16] a module system and service platform, to develop this system. OSGi reduces complexity by providing a modular architecture for large-scale distributed system. All the data transferring among models are the deployment solution with partly or whole data.

Firstly, the performance model figures out the amounts and the maximum distance between virtual machines and fill these data into the solution which will be sent to next model. Secondly, the availability model fill the solution with the topological structure of virtual machines and sent the solution to next model. Thirdly, the placement model reaches the Absolute Locations, which is the whole solution. Lastly, the input and the output of the consolidation model both are the whole solution. In this way, we define the uniform interface between models, so it is convenient for any one who wants to add a new model into the system.

V. Experiment Evaluation

Firstly, we designed and implemented a prototype of our framework using CloudSim [17], a toolkit for modeling and simulation of cloud computing infrastructures and services. [13] We simulated dynamic workloads that act as a typical load of 1998 world cup web site of a period [18]. The simulation data, as described in [13], consisted of 20 virtual machines on homogenous physical nodes, which had one CPU core with performance of 3000 MIPS, 8 GB of RAM and 1 TB storage, and each virtual machine had 512 MB of RAM. The Fig. 7 shows that our algorithm can forecast the amount of virtual machines needed.

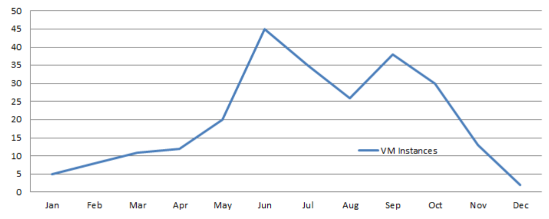

Fig. 7. Amount of virtual machines needed

Then, we created a prototype of the proposed framework in Java and simulated a cloud in a hypothesized environment consisting of over 200 homogenous physical machines. [14] We set up 5 regions, each of which was composed of 5 availability zones with an even distribution of all hosts. Regions and availability zones had 95% and 93% availability respectively, and the availability of physical machines is 90%. We assessed the resizing up and resizing down in detail in [14]. The Fig. 8 shows the availability and the communication cost when the amount of virtual machines increases from 3 to 9. It indicates that using our approach can accurately compute the availabilities on the application with various virtual machines, and under the specific maximum distance between virtual machines, which represents the maximum acceptable communication cost, our approach can obtain the appropriate topological structure of virtual machines.

VI. Conclusion and Future Work

In this paper, we analyzed the SLAs in cloud including clients care most and providers can offer. We picked the performance, the availability and the reliability of the applications as the clients requirement, while the resource utilization and

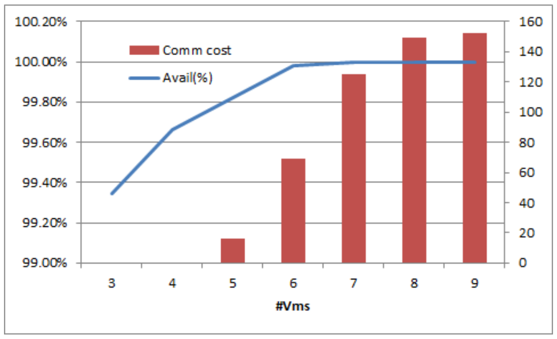

Fig. 8. Availability and communication cost change when scaling up

cost as the providers need. To help to reduce the pressure of providers promising these indicators of applications, we designed a multi-objective serial optimization with priorities approach to manage the resources in cloud. The deployments found by using this approach may be not the optimal one in theory, but they are low time-cost and of great practical significance and engineering value.

The algorithms we used in the models above are primal and needed to be improved. Moreover, with the development of cloud federation, one application may be deployed in more than one cloud, and the metrics measuring applications, such as the performance, the availability and the reliability, will be more complicated to compute. Thus we must modify the algorithms to support the cloud federation.

In the Performance Model, we just consider the ways of purchasing resources in one cloud. [13] To extend to the cloud federation, we need to take the means of purchasing resources in multiple clouds into consideration.

In the Availability Model, we ignored the diversity of virtual machines when modeling and computing the availability of an application [14]. However, in practice the virtual machines are not identical in one cloud let alone clouds. As a result, we need to improve the algorithm in the Availability Model by considering the difference of virtual machines. In single cloud, we can classify virtual machines based on the information about them offered by the provider, and then set them relevant weights. It is similar to the cloud federation that we can enlarge the range of weight to support the cloud federation.

In the Consolidation Model, the consolidation was limited in one zone to cause no effect on the availability and reliability. In the cloud federation, we can expand the 'zone' to the 'cloud' in the first stage. The next step is migrating the virtual machines among clouds.

References

[1] S. A. Baset, "Cloud slas: present and future," *SIGOPS Oper. Syst. Rev.*, vol. 46, no. 2, pp. 57–66, Jul. 2012.
[2] A. in the cloud, 2011. [Online]. Available: http://www.controlengeurope.com/article/44307/Availability-in-the-cloud.aspx
[3] A. E. B. Store, http://aws.amazon.com/cn/ebs/, 2013.
[4] W. A. C. SLA, http://www.microsoft.com/download/en/details.aspx, 2013.
[5] R. C. S. SLA, http://www.rackspace.com/cloud/legal/sla, 2013.

[6] H. Goudarzi and M. Pedram, "Multi-dimensional sla-based resource allocation for multi-tier cloud computing systems," in *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, 2011, pp. 324–331.

[7] H. Goudarzi, M. Ghasemazar, and M. Pedram, "Sla-based optimization of power and migration cost in cloud computing," in *Cluster, Cloud and Grid Computing (CCGrid), 2012 12th IEEE/ACM International Symposium on*, May, pp. 172–179.

[8] J. Lejeune, L. Arantes, J. Sopena, and P. Sens, "Service level agreement for distributed mutual exclusion in cloud computing," in *Cluster, Cloud and Grid Computing (CCGrid), 2012 12th IEEE/ACM International Symposium on*, 2012, pp. 180–187.

[9] A. Undheim, A. Chilwan, and P. Heegaard, "Differentiated availability in cloud computing slas," in *Grid Computing (GRID), 2011 12th IEEE/ACM International Conference on*, 2011, pp. 129–136.

[10] J. Z. Li, M. Woodside, J. Chinneck, and M. Litoiu, "Cloudopt: multi-goal optimization of application deployments across a cloud," in *Proceedings of the 7th International Conference on Network and Services Management*, ser. CNSM '11. Laxenburg, Austria, Austria: International Federation for Information Processing, 2011, pp. 162–170. [Online]. Available: http://dl.acm.org/citation.cfm?id=2147671.2147697

[11] A. E. I. P. Options, http://aws.amazon.com/cn/ec2/purchasing-options/, 2013.

[12] Q. worldcupblog.org Page Views(Global), http://www.quantcast.com/worldcupblog.org, 2013.

[13] X. Chen, H. Chen, Q. Zheng, W. Wang, and G. Liu, "Characterizing web application performance for maximizing service providers' profits in clouds," in *Cloud and Service Computing (CSC), 2011 International Conference on*, Dec., pp. 191–198.

[14] W. Wang, H. Chen, and X. Chen, "An availability-aware virtual machine placement approach for dynamic scaling of cloud applications," in *Ubiquitous Intelligence Computing and 9th International Conference on Autonomic Trusted Computing (UIC/ATC), 2012 9th International Conference on*, Sept., pp. 509–516.

[15] C. Zhang, H. Chen, and S. Gao, "Alarm: Autonomic load-aware resource management for p2p key-value stores in cloud," in *Dependable, Autonomic and Secure Computing (DASC), 2011 IEEE Ninth International Conference on*, Dec., pp. 17 404–410.

[16] O. Alliance, http://www.osgi.org/Main/HomePage, 2013.

[17] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Softw. Pract. Exper.*, vol. 41, no. 1, pp. 23–50, Jan. 2011. [Online]. Available: http://dx.doi.org/10.1002/spe.995

[18] M. Arlitt and T. Jin, "A workload characterization study of the 1998 world cup web site," *Network, IEEE*, vol. 14, no. 3, pp. 30–37, 2000.

[19] A. E. S. L. Agreement, http://aws.amazon.com/cn/ec2-sla/, 2013.

[20] G. C. Platform, https://cloud.google.com/, 2013.

[21] I. S. Cloud, http://www.ibm.com/cloud-computing/us/en/, 2013.

[22] S. I. Zone, http://www.sla-zone.co.uk/, 2013.

[23] C. Tang, M. Steinder, M. Spreitzer, and G. Pacifici, "A scalable application placement controller for enterprise data centers," in *Proceedings of the 16th international conference on World Wide Web*, ser. WWW '07. New York, NY, USA: ACM, 2007, pp. 331–340. [Online]. Available: http://doi.acm.org/10.1145/1242572.1242618

[24] Q. Zhang, Q. Zhu, and R. Boutaba, "Dynamic resource allocation for spot markets in cloud computing environments," in *Utility and Cloud Computing (UCC), 2011 Fourth IEEE International Conference on*, Dec., pp. 178–185.

[25] M. Hajjat, X. Sun, Y.-W. E. Sung, D. Maltz, S. Rao, K. Sripanidkulchai, and M. Tawarmalani, "Cloudward bound: planning for beneficial migration of enterprise applications to the cloud," *SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 4, pp. 243–254, Aug. 2010. [Online]. Available: http://doi.acm.org/10.1145/1851275.1851212