Towards Win-win: Multi-objective Constrained Resource Management in Cloud Federation

Haopeng Chen, Wenting Wang, Wenyun Dai, Xi Chen, Yisheng Wang
School of Software
Shanghai Jiao Tong University
Shanghai, P.R.China
chen-hp@sjtu.edu.cn, wwtvanessa@gmail.com, scorpiodwy@163.com, {april-622, easonyq}@hotmail.com

***Abstract*—Cloud computing has been regarded as a preferred technology for many developers to build cloud applications due to its rapid provisioning and elastic scaling. With the increase in the number of cloud providers, the owners of cloud applications have more options to deploy their applications. For example, considering the availability and performance of the cloud applications, they would deploy the applications into a cloud federation which is a cloud of clouds. For cloud providers, it is also attractive to join in a cloud federation because the utilization of their computing resource will be improved and their computing power will be extended in cloud federation. This paper analyzes the motivation of building cloud federation and the models of cloud federation, and proposes a design of the framework of multi-objective constrained resource management for cloud federation, which is composed of the cloud federation center and the extended cloud federation enabling components of cloud providers. The key technique of resource management in cloud federation is also discussed in this paper, including dynamic profit-driven resource provisioning, availability-aware placement and power-saved consolidation. The proposed framework could satisfy various requirements of the different roles in cloud federation and reach a win-win target.**

Keywords-cloud federation; multi-objective constrained; vertical and horizontal federation; resource management

I. INTRODUCTION

Nowadays, the cloud computing paradigm is regarded as a revolution to the conventional information technology. The features of flexible pricing, rapid provisioning and infinite scaling enable make cloud computing appealing for the applications with massive data or large-scale concurrent clients. With cloud computing, the developers can rent the software, platform and infrastructure as services to facilitate rapid application development and reduce the cost of operation and maintenance of their applications. Consequently, more and more applications have been or will be migrated and deployed into clouds.

The cloud providers also can enhance the utilization of their computing resource and obtain extra profit by leasing their idle resource as service in clouds. As a result, almost all of the IT giants build their own public clouds in forms of Software as a Service, Storage as a Service, Infrastructure as a Service and Platform as a Service. For example, Amazon EC2 and S3 respectively provide resizable computing capability and storage space in order to make web-scale computing easier for developers [1] [2]; Google App Engine, in form of Platform as a Service, enables enterprises to build web applications on the same scalable systems that power Google applications [3]; Microsoft Azure enables users to quickly build, deploy and manage applications across a global network of Microsoft-managed data centers [4]; IBM SmartCloud is the IBM vision for cloud computing, and it is used to accelerate business transformation with capabilities from IBM cloud offerings [5]. Besides these enterprises mentioned above, many companies such as Salesforce, AT&T, GoGrid, NetSuite, Rackspace and RightScale also provide cloud computing service in a variety of different manners.

The diversity of public clouds provides the providers of applications with more choices to deploy their own applications. On the one hand, an application can obtain the independence of cloud providers and improve its availability by deployed into an integration of resources from multiple clouds. For example, two instances of an application can be respectively deployed into Amazon EC2 and Windows Azure to improve its availability. On the other hand, the multi-tier architecture of web applications allows each of the tiers to be deployed into different clouds to ensure all the rented services are with best quality. For example, the web tier, application tier and database tier of an application can be respectively deployed into Amazon EC2, Google App Engine and Amazon S3 to ensure each tier is deployed into the cloud with best quality. Both of the cases involve ‘Cloud Federation’ of public clouds which is the cloud of multiple public clouds. As a consequence of cloud collaboration, the cloud federation of public clouds is an inevitable development of cloud computing.

With the advancement of virtualization, it is feasible for an enterprise to make use of virtualization to effectively integrate its heterogeneous computing resource into a private cloud. Thus, more and more enterprises have built or are building their own private clouds. The computing resource of a private cloud is limited, so it is necessary for a private cloud to cooperate with other private or public clouds in order to scale up its computing power when the utilization of its computing resource is saturated. Consequently, the ‘Cloud Federation’ of private and public clouds, sometimes called as ‘hybrid cloud’, becomes very important for cloud owners and providers.

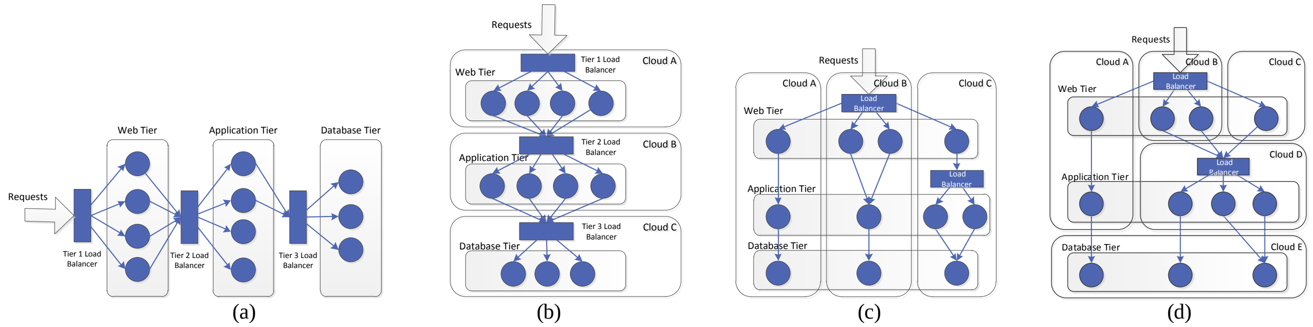

Figure 1. Types of cloud federations: (a) architecture of typical multi-tier applications, (b) a vertical cloud federation, (c) a horizontal cloud federation, (d) a hybrid cloud federation

Actually, for the customers, performance and availability are concerned, while power saving is an important issue for cloud providers. But existing research on cloud federation or hybrid cloud mainly focuses on how to scale up the computing power of a single cloud by building cloud federation. Meanwhile, the research on the cloud federation built for archiving high availability, independence of cloud provider and high quality of services is not adequate yet. Aware of this situation, we analyze the objectives for building cloud federation and put forward a design of multi-objective constrained resource management in cloud federation in order to archive the goal of win-win for both cloud customers and providers.

The remainder of the paper is structured as follows. Section II analyzes the forms of cloud federations and recognized the main objectives of resource management in cloud federations; Section III describes the resource management in cloud federation in details; Section IV gives the results of some simulations; Section V briefly summaries the related works; and conclusion is in Section VI.

II. Objectives of Resource Management

According to the features of various cloud federations, we can classify them into three types: vertical, horizontal and hybrid cloud federation. No matter in which type, the common objectives of resource management are to improve performance and availability and save power.

A. *Types of Cloud Federations*

Most cloud applications inherently have the multi-tier architecture which at least includes web tier, application tier and database tier. Each of the three tiers can be individually deployed in order to meet the customized demand of applications. Each tier can be refined into more fine-grained tiers according to the requirement of applications. For example, the web tier can be divided into presentation tier and controller tier; the application tier can be divided into service tier, domain tier and data access tier. However, the fine-grained tiers of a coarse-grained tier are not deployed separately otherwise the performance will be damaged drastically.

Fig.1(a) is an example of multi-tier application in which the clusters of web tier, application tier and database tier are respectively composed of four, four and three VMs (Virtual Machines). There is a load balancer in front of each cluster to dispatch the requests and balance the workload among the VMs in the same cluster. The multi-tier architecture and the flexible deployment mode of cloud applications enrich the diversity of cloud federations by enabling the vertical cooperation between clouds.

The first type of cloud federation is "*Vertical Cloud Federation*", shown as Fig.1(b), in which the clouds vertically collaborate with each other to provide the application with all the necessary services. The "vertical collaboration" means that each cloud provides hosting environment for only one tier and any request dispatch involving multiple tiers needs to be accomplished by the collaboration between clouds. For example, in Fig.1(b), the web tier, application tier and database tier are respectively deployed into cloud A, B and C. The resource from cloud A, B and C allocated to the target application forms a vertical cloud federation. Such a deployment solution is totally determined by quality of services.

The second type of cloud federation is "*Horizontal Cloud Federation*", shown as Fig.1(c), by which a cloud application can obtain the independence of cloud providers and improve its availability by deployed multiple instances into an integration of resources from multiple clouds. As the saying — 'Don't put all your eggs into one basket', the instances of an application can be horizontally deployed into different clouds to reduce the failure probability. For example, in Fig.1(c), each of cloud A, B and C has a complete instance of the application. Such a deployment solution is also effective to solve the problem of saturation of computing resource.

The third type of cloud federation is "*Hybrid Cloud Federation*", shown as Fig.1(d), which is a combination of vertical and horizontal cloud federations. For example, in Fig.1(d), the collaborations between cloud B and cloud D, cloud C and cloud D, and cloud D and cloud E are vertical while the ones between cloud A and cloud B, cloud B and cloud C, and cloud A and cloud D are horizontal.

B. *Common Objectives of Resource Management*

Either the consumers or the providers of cloud federation want to optimize the resource management in order to improve the utilization of computing power. The common objectives of such resource management includes profit-driven resource provisioning, availability-aware resource placement, and power-saved resource consolidation.

Consequently, the resource management in cloud federation is a multi-objective constrained one.

C. *Profit-driven Resource Provisioning*

From the viewpoint of consumers of cloud federation, the computing power of their cloud federation instances had better be able to be dynamically scaled up and down according to the real-time workload of their cloud applications, since their profit will be maximized by renting the computing power in an economical way.

The profit of a cloud application is from the difference between its revenue and cost. The revenue is determined by the SLA assigned by the application and its consumers. In general, it is in proportion to the performance level achieved. Monotonic non-increasing utility functions are quite realistic to model the relationship between the revenue and the achieved performance, since the better the achieved performance is, the higher the revenues gained per request are. The cost is determined by the amount and price of computing power in the hosting cloud federation. The more the rented computing power is, the more the cost is. Given the unit price of computing power is fixed, there is a linear dependency between the cost and the amount of rented computing power. It is common that more computing power will result in better performance. But it is possible that the cost increased by the more rented computing power is greater than the revenue increased by improved performance. Thus, the better performance would probably result in less profit. So profit-driven resource management needs to find the point in Fig. 2 at which the difference between *Revenue(R)* and *Cost(C)* is the maximum.

In Fig. 2, the horizontal coordinate presents the number of VMs the cloud application rented, while the vertical coordinate presents the sum of profit. The *Cost(C)* linearly depends on the number of VMs (Virtual Machines). The *Revenue(R)* also increases with the increase of the number of VMs, but the relationship is not linear. Given the number of VMs is specified, for example B, we can find point A at which MR equals to MC and the profit is maximal. Given the number of VMs is changed, for example, it is changed to B′ or B″, the point A will be changed to A′ or A″ correspondingly. The *Revenue(R)* does not only depend on the number of VMs, it also depends on the real-time workload of the cloud application, since the achieved performance is determined by these two factors. The global maximal profit is the maximum of all of the maximal profit under each number of VMs.

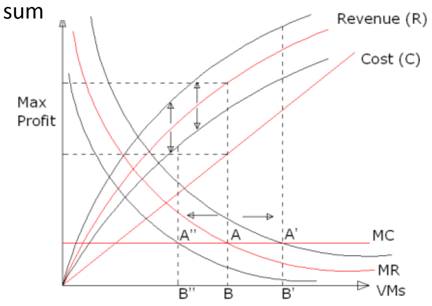

Figure 2. Profit-driven Resource Management

D. *Availability-aware Resource Placement*

Besides of performance, the availability is also a concerned objective for the owner of a cloud application. The profit-driven resource management only focuses on the appropriate amount of resource to be rented, while the availability-aware resource placement aims to determine the places of the rented resource. When some computing resource is rented by a provider to build the infrastructure of hosting environment of its application, the availability of such an infrastructure can be calculated with the availability and the topological structure of the physical hosts.

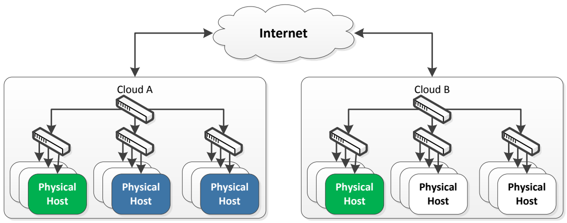

Figure 3. Two Deployment Solutions

For example, Fig. 3 shows two infrastructures of an application both of which are composed of two physical hosts. Both of the physical hosts of the blue one are in the cloud A while the two physical hosts of the green one are respectively in cloud A and cloud B. It is obvious that the availability of the green one is higher than that of the blue one, since the blue one is not available when both the clusters of Cloud A the hosts belong to are simultaneously not available while the green one is not available when both the cluster of Cloud A and the cluster of B the hosts belong to are simultaneously not available. In many clouds, for example, in Amazon EC2, the hosts are clustered into availability zones, and the zones are grouped into availability regions. In such a structure, the further the hosts of a cluster are away from each other, the more available the cluster is. But we should be aware that the high availability is obtained at the cost of performance. As a result, the geographic distribution of the hosts of a cluster should be limited into an acceptable range.

E. *Power-saved Resource Consolidation*

From the view of cloud provider, the aim of resource consolidation is to save power and then cut down the operating cost. On the one hand, the cloud providers want to satisfy the resource requirements of cloud applications with minimal number of running physical hosts. On the other hand, they also hope that all the running physical hosts are running at appropriate status which means the utilization of computing resource on each physical host is greater than the lower bound and smaller than the upper bound.

The input of power-saved resource consolidation is the output of availability-aware resource placement in which the places of VMs are logical ones. The physical places of VMs will be located according to the runtime status of physical hosts. Furthermore, they are not fixed since the utilization of computing resource of physical hosts varies with the runtime workload of cloud applications. As a result, the periodical check for overloaded and underloaded nodes is executed and then the dynamical balancing is accomplished by VM migration.

In a cloud federation, it is possible for cloud providers to lease computing resource from each other which will result in a leasing loop. In such situation, the performance of cloud applications will be impacted since it incurs unnecessary remote communication cost. As a result, the resource consolidation needs to eliminate the leasing loop by VM migration too.

III. Design of the Frame work for Resource Management in Cloud Federation

This section gives a design of the framework for resource management in cloud federation.

A. The Architecture of Cloud Federation

As shown in Fig. 4, the core of the framework of cloud federation we proposed is the *Cloud Federation Center* which contains the lightweight kernel, the infrastructure and the extensions. The Cloud Federation Center acts as an agent for cloud customers and cloud providers to facilitate the construction and deconstruction of cloud federation, coordinate the services provided by clouds and distribute revenue among clouds.

For cloud providers, the *CF(Cloud Federation) Manager* is used to communicate with other clouds. The *App Manager* and the *Cloud Manager* are the extension to existing cloud. The physical layer is the shared virtual computing resources where cloud applications are deployed.

Each cloud involved in cloud federation communicates with Cloud Federation Center and other clouds to provide necessary services to cloud applications. The details of these components will be given in following parts.

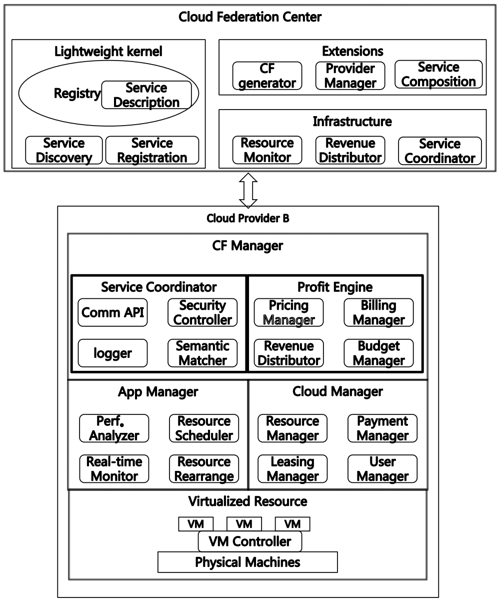

Figure 4. Enabling Components in Cloud Federation

B. Autonomous Federation vs. Centralized Controlled Federation

The models of cloud federation can be classified into the *Autonomous Model* and *Centralized Controlled Model* according to the way of management of cloud federation.

Autonomous Cloud Federation is autonomously managed by cloud providers. The initiators of Autonomous Cloud Federation are cloud providers, especially private cloud providers. They just query available resources through the lightweight core of the Cloud Federation Center and autonomously complete the process of building cloud federation which is hidden from the cloud consumers. This model is used to build horizontal cloud federation in order to extend the computing power of the initiators' cloud.

Centralized Controlled Cloud Federation is managed by the Cloud Federation Center. The initiators of Centralized Controlled Cloud Federation are usually the cloud consumers. Their request of building cloud federation can be divided into two types. In the first case, the cloud consumers retrieve the available resources, lease the resources from cloud providers and deploy their applications. But obviously, it requires the cloud consumers must be very professional. Thus, in the second case, more common than the first one, cloud consumers send their functional requirements of rental resources and SLA (Service-Level Agreement) constraints to the cloud federation center, and then the cloud federation center generate the solution of building cloud federation in form of service composition based on the registered service. This model is used to build both horizontal and vertical cloud federation.

C. Cloud Federation Center

The Cloud Federation Center is the core of the framework we proposed. As shown in Fig. 4, the Cloud Federation Center is composed of three parts.

The *Lightweight Kernel* is a service registry in which various service descriptions are published by the cloud providers. The cloud customers and providers query the kernel to discover desired services. We have designed and implemented a service registry which can discover the alternative services that meet the demand according to the specified functional and QoS requirements[16]. Therefore, the lightweight kernel can be implemented by reusing and extending the existing service registry to support the cloud-specific semantic descriptions. This kernel is necessary for either the Autonomous or the Centralized Controlled cloud federations.

The *Extensions* is used in the Centralized Controlled Cloud Federations but not in the Autonomous ones. The CF(Cloud Federation) Generator generates a solution for deploying Cloud applications according to the SLAs. The generated solution can be a single cloud, or a vertical, a horizontal or a combined cloud federation. We have designed mechanisms for determining the required quantity of computing resource based on the predicted performance[17] and allocating the computing resource dynamically based on the required availability[18]. The Provider Manager is used by the administrator of the Cloud Federation Center to ensure that only qualified providers can register their services into the Lightweight Kernel. The Service Composition component is called by CF Generator to obtain a composite service from multiple clouds when the latter fails to find a single cloud as the hosting environment.

The *Infrastructure* is also used in the Centralized Controlled Cloud Federation but not in the Autonomous ones. The resource involved in cloud federation is monitored by the Resource Monitor at runtime to get their real-time status. The Revenue Distributor can reasonably distribute the revenue of cloud federation to all the involved clouds. The Service Coordinator can do the API and protocol transformation in order to coordinate the service cooperation across clouds.

D. *Dynamic Resource Management*

The dynamic resource management in our framework is realized by the collaboration of App Manager, Cloud Manager and Resource Monitor in the Cloud Federation Center. There are three parts of dynamic resource management shown in Fig. 4.

The *Resource Monitoring Component* is located in the Infrastructure part of the Cloud Federation Center. As we mentioned, in this component, the statistics and analysis of real-time monitoring data collected from the clouds in the cloud federation assure the cloud federation center grasping the global real-time state of computing resources.

The *App Manager* is one module of cloud providers. The Real-time Monitor component monitors the real-time status of Cloud applications deployed in the cloud, including the response time, throughput, failures and so on. It can be implemented in the manners of packet filtering and proxy. The Performance Analyzer will model and predict the performance of cloud applications based on the monitored data. The queuing network and autoregressive model are supposed to be utilized to analyze the performance of cloud applications[17]. The data obtained from the analysis can be used by the Resource Scheduler to allocate or reclaim computing resource for the Cloud applications. The Resource Arranger periodically rearranges the allocated computing resource by live migration of VMs in order to minimize the resource fragmentation generated at runtime.

The *Cloud Manager* is an existing module of cloud providers. We need to add some new functions to its existing components. The Resource Manager determines how and when to construct and deconstruct the Autonomous Cloud Federation based on the global utilization of its computing resource. The Payment Manager discriminates the revenue from cloud federation from that totally from its own cloud since the former needs to be distributed among the clouds involved into the cloud federation. The Leasing Manager doesn’t only manage the leasing contracts signed with the Cloud applications, but also manages the ones signed with other clouds in cloud federation. Meanwhile, the User Manager manages all the registered users and trusted cooperative cloud providers.

E. *Service Cooperation*

As shown in Fig. 4, the service cooperation is implemented through the Service Coordinators in Cloud Federation Center and clouds. The Service Coordinator in cloud is a part of CF Manager, and comprises of the following modules.

The *Comm(Communication) API* should be consistent with the Service Coordinator in the Cloud Federation Center. Meanwhile, various cloud providers should provide adapters for the Comm API, and map it to their proprietary implementation.

The *Security Controller* realizes the strict control access and encryption of sensitive data which are necessary for all cloud providers.

The *Logger* realizes the log management. The configured log system is designed to assure the effective log management.

The *Semantic Matcher* provides a mechanism of semantics extending in which the ontologies and other formal methods are utilized to describe the semantics of collaborative behavior in cloud federation.

F. *Revenue Distribution*

As shown in Fig. 4, the revenue distribution of the framework we proposed is implemented through the Revenue Distributor in Cloud Federation Center and the Profit Engines in clouds. The Profit Engine in cloud is a part of CF Manager, and comprises of the following modules.

The *Pricing Manager* is supported by the dynamic pricing mechanism of existing price management module of cloud providers. Meanwhile, the scheme of constructing cloud federation generated by the cloud federation center is used as an additional factor to determine the dynamic prices of computing resource.

The *Revenue Distributor* should be consistent with the Revenue Distributor in the Cloud Federation Center. The multi-objective optimization algorithm is utilized to design and implement the strategy of revenue distribution for multi-win.

The *Budget Manager* determines the leasing policies according to the SLAs of the Cloud application, including the quantity, location and VM types.

The *Billing Manager* is an existing component of cloud providers to compute the charge of cloud consumers which are either the Cloud applications or the other clouds.

G. *Multi-objective Constrained Resource Management*

Since the real-time workload is varying from time to time, the profit-driven resource management needs to dynamically find the global maximal profit point in order to determine the number of VMs needs to be rented. Furthermore, the predicted workload is more suitable than monitored real-time workload, so a self-learning predictor is desirable for resource management.

In [17], we have proposed an approach to do the profit-driven resource management. In this approach, we proposed a performance model for analyzing and predicting the real-time workload of cloud applications, a predictive and reactive method that determine when to scale up or down the computing power rented by cloud applications, and a profit-driven provisioning technique to maximize profits of SaaS provider. The result of experiments has demonstrated that the approach is effective for profit maximization.

The cloud federation center will determine the construction and separation of cloud federation instances according to the provisioning result. The resource provisioning just determines the amount of computing power needed but not specifies the source of the computing power. The cloud federation center will determine how to locate the computing power according to the requirements of consumers. For instance, if the consumer wants to maintain the status that multiple instances of the cloud application should be deployed into multiple cloud providers, when the computing power needs to be scaled down, the cloud federation center will just scale down the computing power of some instances but not remove any instance. Either the profit of cloud applications or the one of cloud providers will be guaranteed as high as possible in such a dynamic scaling mechanism.

In [18], we proposed an availability-aware approach to place VMs for dynamic scaling of cloud applications. In this approach, we used Bayesian formula to evaluate the availability of the infrastructure of a cloud application. For instance, the green solution in Fig. 3 will be unavailable under the following situations: both the hosts are failure, one host is failure and the other is normal while the master host of its cluster or cloud is failure, both the hosts are normal while both the master hosts of their clusters or clouds are failure. With Bayesian formula, we can calculate the conditional probability that both the hosts of green solution are unavailable and then derive its availability.

In this approach, the maximum distance between any two VMs was also taken into account to prevent the violation of SLA on performance. However, we just put a single upper limitation on this distance which means all the VMs are equivalent to each other. In fact, if a multiple layered application is deployed as Fig.1 (a), (b) or (c), such a single upper limitation is not suitable any more. For example, the VMs in web layer can be distributed far away from each other since they needn't to communicate with each other, while a VM of web layer should be close to a VM of application layer in order to reduce the communication cost between layers.

In this approach, we simplified the cloud environment as a homogeneous one in which all the hosts has same configuration, including CPU, memory, hard disks and bandwidth. It definitely needs to be extended to support heterogeneous environment when apply it into a cloud federation since it is hardly to build a homogeneous environment with the resource from different cloud providers.

During the availability-aware resource placement, a VM placement plan is generated in which the places of VMs are logic ones but not physical ones. On the one hand, from the view of cloud application, the logic places, such as the relative distances between VMs are more important than the physical places because the physical hosts in a cluster are equivalent to each other. For example, for the blue solution in Fig. 3, the cloud application concerns that its two VMs must be deployed into two clusters of a cloud while doesn't care the VMs are deployed into which physical hosts of the two clusters. On the other hand, the physical places should be determined by cloud provider according to the runtime load of physical hosts.

We proposed an approach to dynamic workload balancing in [19], which periodically checks the overloaded and underloaded nodes and then the dynamically balances the workload by VM migration. We also proposed a method for eliminating leasing loop in cloud federation in [20]. Both the two methods can facilitate the dynamic resource consolidation.

IV. Simulation

In [17], we verified the effectiveness of performance prediction and profit-driven resource provisioning we proposed in a single cloud. The aim of this phase is to determine the appropriate amount of resource, so when the profit-driven resource provisioning applying to cloud federation, it needn't to be made any modification.

In [18], we verified the availability-aware resource placement and power-saved resource consolidation in a single cloud. The assumption of placement is that all the physical hosts are identical which is not true in cloud federations. For the resource consolidation, since our method do the local consolidation within regions by VM relocation in order to reduce the complexity and cost of VM migrations, when it is applied into cloud federation, it also needn't to be modified. Furthermore, we proposed an approach to eliminate leasing loops in cloud federation in [20].

Thus, we just need to do simulation to verify the effectiveness of availability-aware resource placement in cloud federation.

A. *Setup*

Suppose we have 3 candidate clouds, A, B and C. All of the three clouds have their own average availabilities of regions, zones and hosts, shown as Table.1.

TABLE I. Availabilities of Candidate Clouds

Cloud	**Availabilities**		
	Region	***Zone***	***Host***
A	99%	98%	97%
B	98%	99%	97%
C	99%	99%	98%

Suppose the communication costs between two VMs on single host, on different hosts in single zone, in different zones of single region, in different regions of single cloud, and in different cloud are respectively 0, 1, 2, 3 and 6. The acceptable maximum distance between two VMs depends on these communication costs.

We designed two scenarios for the experiment, scaling up and scaling down. In scaling up scenario, we suppose the provisioning of an application needs to be scaled up from the initial 3 VMs to 9 VMs and the availability requirement of this application is greater than 99.998%. In scaling down scenario, we suppose the provisioning of this application needs to be scaled down from the initial 9 VMs to 3 VMs and the availability requirement of this application is not less than 99.99%.

We applied 6 policies into the scaling up scenario and 2 policies into the scaling down one. Vertical only policy[18] scales up the resource of existing VMs, while horizontal only policy[18] add resource by creating new VMs . Since in horizontal only policy, the distances among new VMs and existing VMs possibly have negative impact on the performance, we designed horizontal-1, horizontal-2 and horizontal-4 policies which respectively refer to the horizontal only policy with the acceptable maximum communication cost 1, 2 and 4. The last two policies are availability-aware policies without and with relocation. The former means no resource consolidation is executed while the latter means resource consolidation is periodically executed. Both the two policies combine the vertical and horizontal policies into a single vertical preferred policy in which vertical scaling up is prior to horizontal scaling up.

B. *Result and Analysis*

The Fig.5 shows the average availabilities under different policies when scaling up. In this figure, we can find that the vertical only policy doesn't change the availability since it doesn't introduce any new hosts. Meanwhile, the larger the accepted maximum communication cost is, the higher the availability obtained. However, the availability-aware with relocation policy is the best one which satisfies the availability requirement with only 4 VMs. Relocation is important since availability-aware without relocation policy needs 6 VMs to satisfy the availability requirement, which is even worse than horizontal-4 policy.

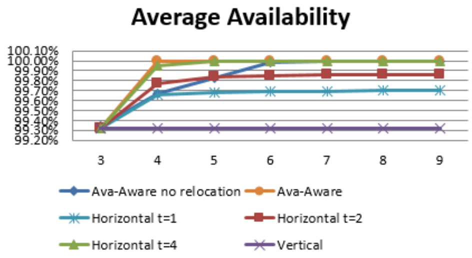

Figure 5. Average Availability When Scaling up

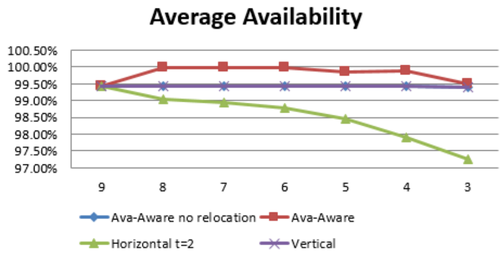

Figure 6. Average Availability When Scaling down

The Fig.6 shows the average availabilities under different policies when scaling down. For simplicity, we only applied 4 policies into this scenario, horizontal-1 and horizontal-4 are ignored. Similarly, the availability-aware with relocation policy is the best one in which the availability is higher than the requirement until the number of VMs less than 4. The availability-aware without relocation policy is same as vertical only policy when scaling down. The horizontal-2 policy is the worst one since it always removes VMs when scaling down.

Summarily, the availability-aware resource placement is also suitable for cloud federation. As a result, the multi-objective constrained resource management, including profit-driven resource provisioning, availability-aware resource placement and power-saved resource consolidation, is effective for cloud federation.

V. Related Work

The concept of cloud federation was first mentioned as Intercloud by Kevin Kelly in 2007, and he said 'eventually we'll have the Intercloud, the cloud of clouds.' [6] Sam Johnton further expatiated that 'the Intercloud is a global cloud of clouds as the Internet is a global network of networks' [7]. However, the concept of Intercloud didn't receive enough concerns, because there was little consensus on how to define the Cloud and many people considered that cloud computing was just a redefinition of the commercial by existing technology.

With the continuous development of cloud computing, more and more people have profoundly understood the essence of cloud computing and realized the importance of cloud federation. During 2009, some researchers used cloud federation to describe the future data center. One of the most important papers was the 'Blueprint for the Intercloud' [8]. This blueprint concerned protocols and formats for cloud computing interoperability but didn't put forth the scheme of many other problems such as when and how to build intercloud, when to deconstruct intercloud and how to distribute profits among all the providers.

In 2009, Global Inter-Cloud Technology Forum [9] was established in Japan and attempted to promote development of intercloud. In 2010, this forum published a White Book [10] about use cases and functional requirements for intercloud computing.

Research on the architecture of cloud federation is the most fundamental among all researches about cloud federation. One of the two main architectures is using an independent third-party heavyweight cloud federation center as the core which takes charge of the dynamic combination and resolution of cloud federation, such as an architecture proposed in [11]. This architecture is convenient to use and don't have drastic changes to the existing cloud architectures, but cloud consumers must change the mode of using cloud resources and the center is the single point of failure of this architecture.

Considering of the autonomy of cloud federation, more researcher prefer the other architecture — the lightweight cloud federation center. In this architecture, the center takes charge of the registration and query of resource information from every cloud; meanwhile, all the clouds combine into and split from the cloud federation dynamically autonomously. Two typical representatives of this architecture are a cloud federation mode proposed by Antonio Celesti [12] and RESERVIOR mode [13] proposed by IBM. In this architecture, the pressure of cloud federation center is largely reduced. This architecture is suitable for active collaboration between cloud providers, but not so helpful for Cloud applications of which all the tiers are not deployed into a single cloud. The two modes mentioned above both are based on performance, not considering other factors, such as availability and power saving, so they can't fully meet the actual multi-objective demand.

Rodrigo N. et al described Aneka, a platform for developing scalable applications on the Cloud, supports such a vision by provisioning resources from different sources and supporting different application models in [14]. They mentioned that the key concepts and features of Aneka support the integration between Desktop Grids and Clouds. Like almost all existing research on cloud federation, Aneka aims at how to scale up the computing power by integrating the computing resource from multiple providers. They ignored the cloud federation which is built for improving the availability and obtaining independence of providers and best quality of services.

The key of cloud federation is that the clouds can communicate with each other by a unified API and specific adaptors. Apache Deltacloud is right such a project that gives customers an opportunity to manage cloud instances in the way they want[15]. This project facilitates the construction of cloud federation and makes it feasible. But it doesn't provide customers any functions to automatically request computing resource according to their constraints.

In conclusion, the most existing research on cloud federation focuses on how to scale up the computing power by cloud federation but not how to improve the quality of services by cloud federation. Both of the two aspects are important for cloud applications. So this paper tried to give a more comprehensive analysis and design of resource management of cloud federation.

VI. Conclusion

This paper proposed a reference framework for multi-objective constrained resource management of cloud federation, which is composed of the cloud federation center and the extended cloud federation enabling components of cloud providers. The proposed framework could satisfy the construction of autonomous and centralized controlled cloud federations, and support profit-driven resource provisioning, availability-aware resource placement, and power-saved resource consolidation in order to improve the utilization of computing power and cut down the rental of consumers and the operational cost of providers at the same time.

With an implementation of the framework proposed in this paper, the independent cloud federation center would be able to schedule computing power for the providers of cloud applications in a transparent way, which would greatly lower the technical threshold of application of clouds.

References

[1] Amazon, *Amazon Elastic Compute Cloud (Amazon EC2)*, Available at: http://aws.amazon.com/ec2/

[2] Amazon, *Amazon Simple Storage Service (Amazon S3)*, Available at: http://aws.amazon.com/s3/

[3] Google, *Google App Engine*, Available at : http://code.google.com/intl/en/appengine/

[4] Microsoft, *Microsoft Azure*, Available at: http://www.windowsazure.com/zh-cn/home/tour/overview/

[5] IBM, *IBM Smart Business Cloud Computing*, Available at: http://www.ibm.com/ibm/cloud/

[6] K. Kelly, *A Cloudbook for the Cloud*, Available at: http://www.kk.org/thetechnium/archives/2007/11/a_cloudbook_for.php

[7] S. Johnston, *The Intercloud is a global cloud of clouds*, Available at: http://samj.net/2009/06/intercloud-is-global-cloud-of-clouds.html

[8] D. Bernstein, E. Ludvigson, K. Sankar, S. Diamond and M. Morrow, "Blueprint for the Intercloud - Protocols and Formats for Cloud Computing Interoperability," in *Conf. 2009 IEEE Fourth International Conference on Internet and Web Applications and Services*, May 24-28, 2009, pp.328-336

[9] GICTF, *Global Inter-Cloud Technology Forum*, http://www.gictf.jp/index_e.html

[10] GICTF, *Use Cases and Functional Requirements for Inter-Cloud Computing*, August 9, 2010, http://www.gictf.jp/doc/GICTF_Whitepaper_20100809.pdf

[11] R. Buyya, R. Ranjan, and R. N. Calheiros, "InterCloud: Utility-Oriented Federation of Cloud Computing Environments for Scaling of Application Services," in *Proc. 10th International Conference on Algorithms and Architectures for Parallel Processing, Busan*, South Korea, May 21-23, 2010, pp.13-31

[12] A. Celesti, F. Tusa, M. Villari, and A. Puliafito, "How to Enhance Cloud Architectures to Enable Cross-Federation," in *Conf. 2010 IEEE 3rd International Conference on Cloud Computing*, Miami, Florida, USA, July 5-10, 2010, pp. 337-345

[13] B. Rochwerger, D. Breitgand, E. Levy, A. Galis, K. Nagin, I. M. Llorente, R. Montero, Y. Wolfsthal, E. Elmroth, J. Caceres, M. Ben-Yehuda, W. Emmerich and F. Galan, "The reservoir model and architecture for open federated Cloud computing," *IBM Journal of Research & Development*, vol. 53, no. 4, pp. 4:1–17, Nov 2009.

[14] R. N. Calheiros, C. Vecchiola, D. Karunamoorthy and R. Buyya, "The Aneka Platform and QoS-Driven Resource Provisioning for Elastic Applications on Hybrid Clouds," *Future Generation Computer Systems*, Volume 28, No. 6, Pages: 861-870, Amsterdam, The Netherlands, June 2012.

[15] Apache, *Deltacloud*, Available at: http://deltacloud.apache.org/

[16] S. Xiong, H. Chen, "QMC: A Service Registry Extension Providing QoS Support," in *Conf. 2009 IEEE New Trends in Information and Service Science*, Beijing, China, Juen 30- July 2, 2009, pp.145-151

[17] X. Chen, H. Chen, Q. Zheng, W. Wang, G. Liu, "Characterizing Web Application Performance for Maximizing Service Porvider's Profits in Clouds," in *Conf. 2011 IEEE International Conference on Cloud and Service Computing*, HongKong, China, Dec 12-14, 2011, pp.200-207

[18] W. Wang, H. Chen, X. Chen, "An Availability-aware Virtual Machine Placement Approach for Dynamic Scaling of Cloud Applications," in *Conf. 2012 IEEE 9th International Conference on Autonomic and Trusted Computing*, Fukuoka, Japan, Sept 4-7, 2012, pp. 509-516

[19] C. Zhang, H. Chen, S. Gao, "ALARM: Autonomic Load-Aware Resource Management for P2P Key-value Stores in Cloud," in *Conf. 2011 IEEE Ninth IEEE International Conference on Dependable, Autonomic and Secure Computing (DASC 2011)*, Sydney, Australia, Dec 12-14, 2011, Pages:404-410

[20] Y. Wang, H. Chen, "Dynamic Resource Arrangement in Cloud Federation," in *the 2012 IEEE Asia-Pacific Services Computing Conference*, Guilin, China, to be appeared.