International Journal of Automation and Computing 10(6), December 2013, 597-606
DOI: 10.1007/s11633-013-0758-1

Resource Virtualization Model Using Hybrid-graph Representation and Converging Algorithm for Cloud Computing

Quan Liang[1] Yuan-Zhuo Wang[2] Yong-Hui Zhang[1]

[1]School of Information Sciences and Engineering, Fujian University of Technology, Fuzhou 350108, China

[2]Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China

Abstract: Cloud computing can provide a great capacity for massive computing, storage as well as processing. The capacity comes from the cloud computing system itself, which can be likened to a virtualized resource pool that supports virtualization applications as well as load migration. Based on the existing technologies, the paper proposes a resource virtualization model (RVM) utilizing a hybrid-graph structure. The hybrid-graph structure can formally represent the critical entities such as private clouds, nodes within the private clouds, and resource including its type and quantity. It also provides a clear description of the logical relationship and the dynamic expansion among them as well. Moreover, based on the RVM, a resource converging algorithm and a maintaining algorithm of the resource pool which can timely reflect the dynamic variation of the private cloud and resource are presented. The algorithms collect resources and put them into the private cloud resource pools and global resource pools, and enable a real-time maintenance for the dynamic variation of resource to ensure the continuity and reliability. Both of the algorithms use a queue structure to accomplish functions of resource converging. Finally, a simulation platform of cloud computing is designed to test the algorithms proposed in the paper. The results show the correctness and the reliability of the algorithms.

Keywords: Cloud computing, resource virtualization, hybrid graph, resource converging, simulation.

1 Introduction

Cloud computing allocates computing tasks to a resource pool composed of numerous computers, and it enables end-users to have an access to the computing capacity, storage and information service according to their various demands. Cloud computing has also brought great support in terms of technology and resource for other network applications. For example, cloud computing enables the realization of the internet of things (IOT) with a support of necessary infrastructure and techniques. The IOT cannot come into being without the aid of cloud computing in terms of storage, data retrieval, etc.

From the perspective of techniques, cloud computing which generates itself from distributed computing, parallel computing, and grid computing, is a further development of all the computing models mentioned above. Compared to others, cloud computing has a further focus on applicability. In recent years, cloud computing is combined with advantages of other technologies like virtualization, Web 2.0, etc. Essentially, cloud computing in fact is mainly a combination of resource and server virtualization technology with infrastructure as a service (IaaS) along with other technologies[1, 2]. Cloud computing is an innovation in business model, whose core is to provide users with service under a use-payment model after resources are virtualized in one or some data centers. As an innovative model, cloud computing is still confronted with many new challenges which require a clear description on the relationships and the activities of all entities so as to facilitate the technique development and application of cloud computing[3, 4].

The main contributions of this paper are as follows: 1) A cloud resource virtualization model based on hybrid-graph structure and detailed descriptions of logical relationship of the entities are presented. 2) Resource converging algorithm and the maintaining algorithm of the resource pools are put forward. They collect all resources and put them into the resource pools, and enable a real-time maintenance for the dynamic variation of resource to ensure the continuity and reliability; 3) A simulation platform of cloud computing is designed to test the proposed algorithms.

The rest of the paper is organized as follows. Section 2 introduces some related works. Section 3 presents a resource virtualization model. A resource converging algorithm and a maintaining algorithm of the resource pools are put forward in Section 4. Section 5 expresses the simulation platform and experiments with the analysis of results. Finally, the paper ends up with brief conclusive remarks and discussions on future research directions.

2 Related works

Distributed computing enables the communication and coordination among numerous computers by means of network. It allocates the idle resource and computing power to accomplish a large scale computing task so as to real-

Manuscript received October 18, 2012; January 4, 2013
This work was supported by National Natural Science Foundation of China (No. 61101139), Natural Science Foundation of Fujian Province (Nos. 2012J01244 and 2012J01243), and Hunan Provincial Project of Science and Technology (No. 2013FJ3090).

ize a thorough share of computing, software, information, communication, etc. There are a lot of descriptive models for distributed environments. Zhu et al.[5] presented a design of dynamic trusted degree evaluation model, made a relatively comprehensive analysis of distributed computing environments, and pointed out some particular features of reliability such as dynamics, continuity, uncertainty, etc. Simultaneously, quite a lot of models have come into being for researching on fault-tolerant distributed computing whose problems have received great attention relatively. Bernadette and Andr[6] presented a model for fault-tolerant distributed computing environment which is only a computing model based on the concept of transmission error. Among many applications in distributed computing environment, web server obviously plays the role of the end terminal of the dynamic server. Paul and Rema[7] not only presented an introduction of a performance evaluation of web server based on the model under the distributed computing environments, but also presented a corresponding queue analysis.

The fundamental principle of the parallel computing is to employ a multitude of processors in a coordinated and collaborative way to solve a relatively large computing task. The parallel computing aims to solve the massive but complex tasks, whose prominent feature is parallelism in the computing process. A large number of researches have been conducted to describe parallel computing environments. For example, Lastovetsky[8] put forward multiple parallel C language (MPC), a higher version of parallel language, which can not only describe parallel computing environment in a convenient way, but also simulate a large scale parallel computing process. Clematis and Corana[9] analyzed and built a performance model of the heterogeneous parallel computing system, in which every node differs from the others in computing power. The paper put forward a simple but efficient way to analyze and further model the performance of the system. Chang et al.[10] put forward a three-stage distributed constructing protocol which is aimed to organize a super cubic computing environment composed of bluetooth devices. In the constructed scattered network, bluetooth devices have an easy access to a fault-tolerant route without intersection with others, which allows the realization of both parallel computing and distributed computing under the bluetooth wireless environment.

Grid computing is relatively more complex, whose detailed description is quite a challenge. The actual grid application requires heterogeneous, various resource as well as coordination among tasks, which complicates the scheduling and allocation of resource. Grid model shows that the representation, description along with computation, is a systematic project. How to expand the relationship among all entities in a grid has been a great concern[11−13]. Based on the grid computing environment, Liang and Wang[14] put forward a method to cope with the reflection, converging and computing of users′ demand. The method gives a full account of tasks of each stage in the grid computing environment. The method captures the essence of grid computing. Tong and Miao[15] designed a performance prediction model for service bulk-synchronous parallelism (BSP) to support the high quality application. The model can help users to optimize those factors which affect performance, it also analyze and improve the application. Then it further draws the activity principles of entities and presents a detailed description of the systematic features.

Cloud computing has multi-functions which are the extension of what has already existed in information industry. According to the comparison and contrast mentioned above, some essential features can be concluded such as distributed computing, storage, resource virtualization, high expansion, users-friendly, better manageable and good security[16−18]. So far, there are only a limited number of researches on the model description of the cloud computing, including our preliminary works[19, 20]. Ergu et al.[21] demonstrated the features of cloud computing in task scheduling and resource allocation. They put forward a task-oriented resource allocation mode. Resource allocation tasks are graded according to a pair-wise matrix analytical hierarchy process (AHP), in which the efficient resource and user preference can serve as a criteria of the allocation in accordance with the various grades of tasks. Subashini and Kavitha[22] described the deployment of cloud environment and gave an account of the danger that cloud computing configuration may bring to the security issue. The paper lists out a comprehensive depiction of various dangers for security, whose discussion demonstrates the features of cloud computing environment better. For the reliability of dynamical resources, Tian et al.[23] put forward a strategy to cope with node resources dynamically based on invalidity rule under a heterogeneously integrated cloud platform. After a synthetical consideration of resource demands as well as the temporal and spatial invalidity rule of resource, the paper designs a strategy to guarantee the reliability of node resource and further verifies its practicability. The design consists of a simulation platform which integrates heterogeneous load and a multi-dimension model frame of the system resource. In terms of the application of specific resource data, Zheng et al.[24] put forward a data placement strategy for data-intensive applications to cope with problems like how to reduce data transmission across data centers, how to maintain the reliability, and how to achieve the win-win situation of high efficiency and the overall load equilibrium, which will serve as a good inspiration to the establishment of resource virtualization model for cloud computing.

3 Resource virtualization model of cloud computing environment

3.1 Features of cloud computing environment

So far, cloud computing does not have a unified definition. Through the comparative analysis mentioned above, we can conclude some essential features of cloud computing, e.g, distributed computing, distributed storage, high expansion, users-friendly and better manageable. To be specific, there are the following features[25]: 1) It supports virtual-

ization. Cloud computing can be seen as a virtual resource pool. With deployment of service virtual machine (VM) and application on a single server, there will be a rise in utilization of resource. Then when a server is over-loaded, it supports the migration of workload to other servers. 2) Quality of service (QoS) is guaranteed. Cloud computing system can provide users with qualified services which meet the demand of QoS and further make an adjustment to the system in correspondence to users′ demands, e.g., hardware configuration, network bandwidth, storage capacity, etc. 3) It is high reliable, available and scalable. Cloud computing must make a guarantee to provide users with reliable services and ensure the availability regardless of time and space. Furthermore, when the system scale changes, it can be a flexible and elastic system in accordance with users′ demand. 4) It is autonomous. Cloud computing system is an autonomous system, in which the administration is transparent to users, whose autonomy demonstrates itself in the fact that not only different administration tasks are completed automatically, but also the hardware, software and storage can be allocated automatically. It also allows the allocation to be adjusted to the very need of the users.

3.2 Description of resource virtualization model

The model must reflect the features of cloud computing as well as its inherent mechanism. Taking advantage of the processing ability of grid distributed computing along with other highly-sophisticated technologies such as sever virtualization and storage virtualization, cloud computing aggregates all sorts of resources into resource pools to facilitate a real-time supervision and allocation for users. All resources which can be considered as a massive virtualized resource pool with a great power are transparent to users. The resource may be deployed in any place, and can be gathered together to serve for the users through the technology of resource converging and virtualization. The resource in cloud computers is in support of the task migration to maintain the load equilibrium[26, 27]. The logical relationships in the resource converging and organization can be demonstrated in Fig. 1, which makes it possible to establish a resource virtualization model (RVM) based on cloud computing.

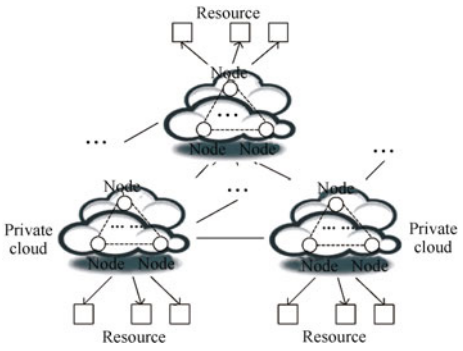


Fig. 1 Logical relationships in virtualization environment of cloud computing

There might be variation in the internal structures of private clouds, which may possess different node quantities and different heterogeneous networks whose nodes have a variation of resources in type, quantity, etc. Private cloud, node and resource can join and exit in a dynamic way. Attentions should be paid that several private clouds can be combined into one private cloud of a larger scale in accordance with the demand. For resource types including network bandwidth, storage, computation and sensor of all kinds, all are connected with some nodes in the private cloud. There might be a variation in the type and quantity of resources connected with each node, which is a good reflection of this variation for both private cloud and nodes. In the model mentioned above, an edge of the interrelated nodes in a private cloud is known as undirected edge, reflecting the intersection among nodes in the private cloud, while the connection edge between the node and resource is known as directed edge. In other word, in a node-to-resource direction, the resource at the direction end are possessed by the node[28, 29]. A detailed description as well as some definitions of the resource virtualization model are presented as follows.

Definition 1. Resource set.

Let resource set be $R = \{r(i,j)|i \in \text{TypeID}, j \in \text{QuantitySet}\}$, $\text{TypeID} = \{\text{Bandwidth} : 0, \text{CPU} : 1, \text{Memery} : 2, \text{Sensor} : 3, \cdots\}$ which refers to the code set of resource types, $\text{QuantitySet} = \{M, \text{CountNumber}, \cdots\}$ refers to the quantifier set of the resources, and different types of the resources have different quantifiers.

Resource set expresses an unified representation method for all sorts of resources including their types and quantities; the method is for not only the convenience in the search and converging of the resources, but also the preparation of resource virtualization.

Definition 2. Connected edge and resource edge.

Let a private cloud set be $G = \{G_{ij}|i \in \mathbf{N}\}$ and the node set be $N(G_i) = \{n_{ij}|j \in \mathbf{N}\}$ of G_i. Then,

1) A connected edge is the undirected edge, the connected edge set among private cloud is $E_{G-G} = G\&G$, and the connected edge set of node G_i is $E_{i-N} = N(G_i)\&N(G_i)$.

2) A resource edge is the directed edge which is directed from node to resource, the resource edge set of G_i is

$$E_{N-R}(G_i) = N(G_i) \times R = \{\langle n_{ij}, r(k,l)\rangle | n_{ij} \in N(G_i), k \in \text{TypeID}, l \in \text{QuantitySet}\}.$$

In the model, there are two kinds of edges, which are the connected edge connected to a particular private cloud or node, and the resource edge connected to resource. The definition mentioned above makes an excessive distinction among private cloud, node and resource, and gives a clear demonstration of relationship between nodes and resources.

Definition 3. Private cloud.

Any private cloud G_i can be described as an ordered triple record as $G_i = \langle V, E_1, E_2\rangle$:

1) G_i is the undirected connected graph or complete graph $K_{|V|}$.

2) $V \neq \varnothing$ is the node set of G_i which represents all nodes in the private cloud, $V = N(G_i)$ and $V \subseteq \cup N(G_i)$, $\cup N(G_i)$ is the union of all $N(G_i)$.

3) $E_1 \neq \varnothing$ refers to the connected edge of V, which connects nodes within a private cloud, and $E_1 \subseteq N(G_i)\&N(G_i)$.

4) $E_2 \neq \varnothing$ refers to the resource edge of G_i, and $E_2 \subseteq E_{N-R}(G_i) = N(G_i) \times R$.

5) $1 \leqslant |V| \leqslant +\infty$, if $|V| = 1$, then a single node can form a single cloud G_i.

Definition 4. Resource virtualization model.

It can be presented as an ordered triple recorded as $M = \langle G, E, W\rangle$,

1) G is the set consising of private clouds, $G = \{G_0, G_1, \cdots, G_n\}$.

2) M is the undirected connected graph or complete graph $K_{|G|}$.

3) $E \neq \varnothing$, $E \subseteq E_{G-G} = G\&G$ is the set of connected edges, which connects private clouds together.

4) Given that $W : E \rightarrow \mathbf{R}$, where $\mathbf{R}$ is a real number set, for any connected edge $e \in E$, $W(e)$ is the weight of e.

The complexity in the cloud computing environment decides the complexity of description. In order to make a clear account of the cloud computing environment, a clear description of entities such as cloud internal structure, inter-cloud connection, cloud resource, as well as their interrelated logical relationships are required. On the basis of the clarification of the relationship among the entities, it is possible to describe data stream and establish the demanded logical relationships among task allocation, resource virtualization, resource allocation, etc. The RVM meets the requirements mentioned above well, and can not only describe the relationship between private clouds and the nodes within these private clouds in a simple convenient way, but also demonstrate all resources including both the type and quantity clearly. With the help of the RVM, it is easy to realize inter-cloud resource searching, allocating and sharing, and also apply all kinds of cloud computing technologies with ease.

Definition 5. Private cloud neighborhood.

In the model $M = \langle G, E, W\rangle$, $\forall g \in G$, let $N_M(g) = \{h | h \in G \wedge (g, h) \in E \wedge g \neq h\}$ be the neighborhood of g.

Two private clouds are called neighbors to each other as long as they can communicate in the RVM directly. Similarly the neighborhood of nodes within a private cloud can be defined as follows:

Definition 6. Node neighborhood.

In a private cloud $G_i = \langle V, E_1, E_2\rangle$, $V_{ij} \in V$, let $N_G(n_{ij}) = \{n_{ix} | n_{ix} \in V \wedge (n_{ij}, n_{ix}) \in E_1 \wedge n_{ij} \neq n_{ix}\}$ be the neighborhood of n_{ij}.

Definition 7. Private cloud joining and exiting.

1) Private cloud joining: If M is an RVM and g is a private cloud, then $M + g$ refers to that G joins into M. If $M^{'}$ is the new model with the joining of g, then

$$M^{'} = \langle G^{'}, E^{'}, W^{'}\rangle$$
$$G^{'} = G \cup \{g\}$$
$$E^{'} = E \cup \{(g, h) | h \in N_{M^{'}}(g)\}$$
$$W^{'} = W \cup \{W(e) | e \in \{(g, h) | h \in N_{M^{'}}(g)\}\}.$$

2) Private cloud exiting: If M is an RVM model and g a private cloud, $M - g$ refers to that g exits from M. If the new model $M^{'} = M - g = \langle G^{'}, E', W'\rangle$, then $G^{'} = G - \{g\}$, $E_3^{'} = E_3 - \{(g, h) | h \in N_M(g)\}$, $W^{'} = W - \{W(e) | e \in \{(g,) | h \in N_M(g)\}\}$.

The joining and exiting of a private cloud will change the structure of the cloud computing environment. Nevertheless, the environment structure still conforms to the definition of the RVM model and maintains the essence after those changes.

Definition 8. Joining and exiting of resources.

1) The joining of resources: For any resource $r(x, y)$, where $x \in$ TyperID and $y \in$ QuantitySet, $G_i + r(x, y)$ represents that $r(x, y)$ joins into the private cloud G_i. If node v within the private cloud is related to $r(x, y)$, $v \in V(G_i)$, then

$$E_2^{'}(G_i) = E_2(G_i) \cup \{\langle v, r(x, y)\rangle\}.$$

2) The exiting of resources: For any resource $r(x, y)$, where $x \in$ TypeID and $y \in$ QuantitySet, $G_i - r(x, y)$ represents that the resource $r(x, y)$ exits from the private cloud G_i. If v is the node related to $r(x, y)$ in the private cloud, then

$$E_2^{'}(G_i) = E_2(G_i) - \{\langle v, r(x, y)\rangle\}.$$

The joining and exiting of resources do not change the node structure of the private cloud nor the overall cloud computing environment. They also will not affect the stability of the cloud computing system. But they will cause changes in the resource edge set of some nodes and affect the resource capacity of the private cloud. A private cloud can choose to join into or exit from the cloud computing environment. As a matter of fact, the joining and exiting of resources are common cases in the RVM model. Thus, it needs constant updating and the RVM can be called a dynamically expandable model.

4 Resource converging method in RVM

4.1 Virtualized category resource pools

One core of the cloud computing is resource virtualization. The overall system resource is regarded as a huge resource pool, which can only be realized by aggregating the dispersed and capability-limited resources into the resource with huge capability or the resource pool which is available for centralized scheduling. To be specific, a category converging of resources (in terms of their various types) is necessary to provide users with a transparent massive cloud processing capability by aggregating the resources (including bandwidth, computing, storage, sensor, etc) into a lager resource pool. Seen from another perspective, the resource converging is convenient for a dynamic migration and allocation of the resource, which will benefit the user with an access to the transparent but limitless resources. These resources are always somewhere in the certain private cloud.

The RVM model is in favor of this kind of resource converging. In the model, the resource can always be found somewhere in the nodes of a certain private cloud. It can be drawn from Definition 1 that both type and quantity are able to be defined to bring great convenience to aggregate resources according to their categories. Resource converging can be implemented as follows. First of all, set a node search in the private cloud by means of certain algorithm. Once the type and quantity of resource related to these nodes can be obtained with little effort according to Definition 2. Then, set up a virtualization category resource pool for each private cloud to get the obtainable resources into the resource pool. Finally, set up a virtualization resource pool of the overall cloud computing system, which can be directly set up by aggregating all categorized resources in each private cloud resource pool. The consolidation of resource enables the user to have an access to a huge computing capability. Some attentions should be particularly paid to the fact that the RVM is dynamic since private cloud or resource may dynamically join or exit. Therefore, in the process of resource converging, a corresponding mechanism should be built up to cope with such dynamic changes. In the rest of the paper, resource converging algorithm based on the RVM is presented after a presentation of Definition 9.

Definition 9. Virtualization resource pool.

The resource pool of a private cloud G_i is $P_{\text{private}}(G_i) = \{p(x) \mid p(x) \subseteq E_{N-R}(G_i), x \in \text{TypeID}\}$, in which $p(x)$ is the categorized resource set, $p(x) = \{\langle n_{ij}, r(x,l)\rangle | n_{ij} \in N(G_i), l \in \text{QuantitySet}\}$.

According to the definition mentioned above, the resource pool of G_i in terms of resource type is composed of different subsets, with each containing all resource edges of their own resource type. Therefore, by means of resource pool, the distinction of category is possible to be found. In addition, by means of categorized resource set, corresponding types and quantity of resource as well as nodes which store resources will be found with ease. Therefore, it will be of a great convenience to the deployment and application of resource. Even when the resource is in dynamic changes, the definition still makes the updating of resource pool easy. According to the type of the resource joining and exiting, the resources and the nodes in the private cloud can have a modification or an addition or a deletion in corresponding $p(x)$. The definition of resource pool given here is drawn from a strict conformity to the dynamically expandable characteristics of the cloud computing environment.

4.2 Resource converging algorithm

In short, resource converging in cloud computing environment is in demand of gathering up resources to set up a virtualization categorized resource pool. Thus it can reach the purpose of a unified deployment and application of resources. The algorithm of creating the private pool for any private cloud is presented as follows:

Algorithm 1. Set up a virtual resource pool for a private cloud.

Input: Any private cloud G_i

Output: Private resource pool $P_{\text{private}}(G_i)$ of G_i.

Initialization: Resource set $p(x) = \text{null}$, $x \in \text{TypeID}$, the node queue $Q_1 = \text{null}$, and the node access vector flag $(n_{ij}) = 0$.

1) Take a random node n_{i0} to be the starting node, and put n_{i0} into Q_1, and flag $(n_{i0}) = 1$;

2) If $Q_1 = \text{null}$, then go to 7).

3) Suppose $\text{DeQueue}(Q_1) = n_{ij}$, get all the nodes with access vector flag = 0 from the neighborhood $N_G(n_{ij})$ and put them into Q_1. At the same time, modify their access vectors to 1).

4) If the resource edge set $E_{N-R}(n_{ij}) = \text{null}$, then go to 2).

5) Search $E_{N-R}(n_{ij})$, collect the corresponding resource edge into resource set $p(x)$ according to their resource type x, i.e., $p(x) = p(x) \cup \{\langle n_{ij}, r(x,l)\rangle\}$.

6) If searching $E_{N-R}(n_{ij})$ is completed, then go to 2).

7) $P_{\text{private}}(G_i) = \{p(x) | x \in \text{TypeID}\}$.

8) Return $P_{\text{private}}(G_i)$.

Algorithm 1 makes it possible to establish the categorized resource pools for each private cloud (i.e., $P_{\text{private}}(G_i)$), and the private cloud can be further converged into the global pool of the cloud computing system. After a good search for each private cloud $P_{\text{private}}(G_i)\,(G_i \in G)$, the resource converging algorithm directly aggregates each $p(x)\,(x \in \text{TypeID})$ into a larger categorized resource pool which is further combined into the global pool for the purpose of a unified allocation and virtualization. For the search for resources around nodes, the converging of the private cloud is done in a way similar to the establishment of $P_{\text{private}}(G_i)$. The converging of corresponding $P_{\text{private}}(G_i)$ is preceded by the search of each private cloud. The resource converging of global pool (P_{global}) has its process as follows:

Algorithm 2. Resource converging algorithm (RC-RVM)

Input: Private cloud set $G = \{G_0, G_1, \cdots, G_n\}$.

Output: Global categorized resource pool $P_{\text{global}}(G)$.

Initialization: Global categorized resource set $gp(x) = \text{null}$, $x \in \text{TypeID}$, and private cloud queue $Q_2 = \text{null}$, private cloud access vector $\text{flag}(G_i) = 0$.

1) Let any private cloud G_0 be the starting cloud, and put it in Q_2, its access vector $\text{flag}(G_0) = 1$;

2) If $Q_2 = \text{null}$, then go to 6).

3) Suppose $\text{DeQueue}(Q_2) = G_i$, get all the clouds with access vector flag = 0 from the neighborhood $N_M(G_i)$, put them into Q_2, and modify their access vectors to 1.

4) In accordance with respective type x, merge $p(x)$ into $gp(x)$, i.e., $gp(x) = gp(x) \cup p(x)$, $p(x) \in P_{\text{private}}(G_i)$.

5) If resource merging of $P_{\text{private}}(G_i)$ is completed, then go back to 2), else go back to 4).

6) $P_{\text{global}}(G) = \{gp(x) | x \in \text{TypeID}\}$.

7) Return $P_{\text{global}}(G)$.

The random resources of the global pool are represented by resource edges which record the resource type, resource quantity, private cloud which the resource belongs to, and the corresponding nodes. Therefore, the global pool can give a clear description of orderly-organized structure of re-

sources in the overall cloud computing system, which makes it possible to provide users with a transparent and virtualized resource service with ease and convenience.

4.3 Dynamic updating of the virtualization resource pool

The dynamic change exists normally in cloud computing environment, private cloud and global pool obviously need timely updating in accordance with these dynamic changes in order to maintain the reliability and availability of the resource pool, which is of great importance.

The dynamic updating of the global resource can be done by means of periodic polling, namely, by setting up the modifiable periodic time T_{global}. During the period of T_{global}, an inquiry signal will be sent to each $P_{\text{private}}(G_i)$ for the knowledge that whether some updating has happened. If some updating has occurred to $P_{\text{private}}(G_i)$, the message of updating will be sent back to $P_{\text{global}}(G)$ for the sake of modifying corresponding resource. It is the same case as the private cloud pool $P_{\text{private}}(G_i)$. It can adopt a similar way by setting up a modifiable periodic time T_{private}, and sends an inquiry signal to each node for the information of updated resource during the period of time. When the resource is updated, the algorithm sends this particular message to $P_{\text{private}}(G_i)$ for the sake of an overall modification. In order to achieve the goal, another two signals are required to be set up, one is $U_{\text{private}}(G_i)$ which marks the resource pool′s updating of private cloud G_i, and the other is the $U_{\text{node}}(n_{ij})$ which marks the resource updating of node j of private cloud G_i. The algorithm of pool updating is presented as follows.

Algorithm 3. Resource pool updating algorithm

Input: $P_{\text{global}}(G)$, T_{global}.

Output: Updated $P_{\text{global}}(G)$.

Initialization: $U_{\text{private}}(G_i) = 0$, $G_i \in G$, it means no resource updated in $P_{\text{private}}(G_i)$. $U_{\text{node}} = 0$, it means no resource updated in node n_{ij}. T_{private} is initialized.

1) While a new T_{private} does not start, do {if necessary, then go to 7), else wait.}

2) Inquire each node. If $U_{\text{node}}(n_{ij}) = 1$, then {fetch the updated resource information of node n_{ij} and send it back to $P_{\text{private}}(G_i)$, modify $U_{\text{node}}(n_{ij}) = 0$.}

3) According to the fetched information, update $p(x)$ in $P_{\text{private}}(G_i)$, $U_{\text{private}}(G_i) = 1$. If T_{private} terminates, then go to 1).

/*Following steps 4)–6), execute concurrently with 1)–3)*/

4) While a new T_{global} does not start, do {If necessary, then go to 7), else wait.}

5) Inquire each private cloud. If $U_{\text{private}}(G_i) = 1$, then {fetch the updated resource information of cloud G_i, $U_{\text{private}}(G_i) = 0$.}

6) According to the fetched information, update $gp(x)$ in $P_{\text{global}}(G)$. If T_{global} terminates, then go to 4).

7) Modify the values of T_{global} and T_{private} in accordance with demands and go to 1) or 4) or just end the algorithm.

The algorithm can guarantee that the resource pool possesses the most updated resource information, which is of great significance. Whether updating is required is up to the dynamic change in resource, which will accordingly urge the updating of composition of the resource virtualization model. The method of the updating is recorded in details as follows: If the resource $r(x, y)$ joins into G_i, it can be presented as $G_i + r(x, y)$. Let the node relevant to the resource $r(x, y)$ be j, then record the added resource edges as $\langle n_{ij}, r(x, y)\rangle$ and set the value of $U_{\text{node}}(n_{ij})$ to be 1. It indicates that there is resource updated in this node for the inquiry from $P_{\text{private}}(G_i)$. And it is the same case as $G_i - r(x, y)$, where the resource exits from G_i. Some attention should be paid to the fact that the joining or exiting of resource is sometimes merely relevant to the variation in the quantity of resources of this kind, but not other factors. In this case, no adding or deletion of resource edges is required, only the quantity of resource of this kind will be modified. Nevertheless, the value of $U_{\text{node}}(n_{ij})$ should be set to be 1 for the sake of inquiry from the resource pool.

5 Experiments and result analysis

5.1 Experiment design

The experiment in this paper is to achieve three main goals: 1) Design an experimental platform according to the RVM to simulate cloud computing or an approximate cloud computing, and deploy or describe resource with reference to relative definition of the RVM. 2) Realize algorithm on the platform including the converging as well as dynamic changes of resources to verify the reliability of the algorithm. 3) According to the variation in experimental data, conduct a supervision over the working situation of the algorithm and the dynamic and complex situation of resource, node, private cloud data quantity, thus the efficiency of the algorithm can be verified.

The present available experimental platform adopts hyper parallel processing (HPP) systematic framework which has absorbed the advantages of computers like Cluster and multiple parallel processing (MPP). Its application server is Tomcat 7.0, operating system is Red Hat 2.6, and virtual resource management platform is VMWare Workstation 8.04. In addition, the computing nodes contain 92 DawningCB65-F blade servers, providing 736 processor cores of 2.6 GHz and an internal memory of 1.5 TB, and its branch storage system contains 4 data servers, an original server and a set of first-class real-time storages. What is more, the experimental platform adds an extra 50 sets of hosts with their CPU being AMD Athlon(TM) 64 X2 3600 + 2.8 GHz to aid calculating nodes. We also have developed the platform of Fujian University of Technology cloud simulation (FLUT-CS). FJUT-CS is able to logically categorize and present private cloud, nodes in the private cloud and resources in the domain. The platform identically conforms to the RVM depicted as Fig. 2. In addition, to simplify the experiment based on the relevant definition about the resource, we have developed resource simulation software (RSS) to enable an independent operation on each node as well as a simulation

of all kinds of resource. The resource simulator can not only have a random adding or depletion in resource type and quantity, but also have the ability of autonomously producing a large quantity of random resource data in accordance with demands. Resource simulator is in favor of flexible selection for experimental data and algorithm verifying under various conditions.

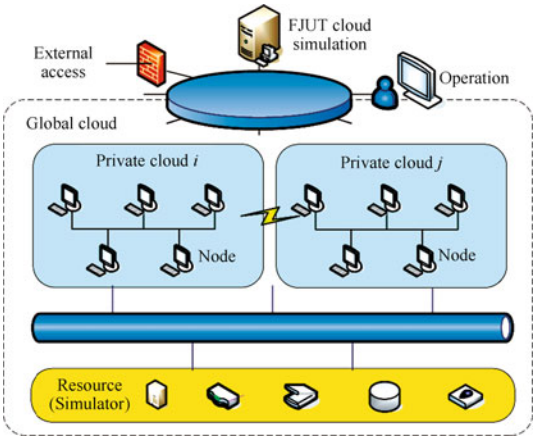


Fig. 2 FJUT-CS frame structure

5.2 Experimental results and analysis

First of all, the experiment uses the data arranged in advance to verify the correctness of the algorithm, including whether the given resource can be found accurately and whether the judgement of resource location, type and quantity are right[30, 31]. Suppose the actual scenarios to be: 4 private clouds which can communicate with each other and locate around different network segments. The 4 private clouds respectively get a possession of nodes with a variation of number from 1 to 4, the nodes of each private cloud can communicate with each other freely, and their detailed resource deployment are shown in Table 1. The representation of the data is identical to the relevant definition mentioned above. For example, $\langle n_{2-2}, r(0, 200\,\mathrm{M})\rangle$ is a resource edge showing that a second node n_{2-2} of the private cloud G_2 has a possession of network bandwidth resource as much as 200 M, and $\langle n_{2-2}, r(3, 6)\rangle$ indicates that it is in possession of 6 sensors. To set the data in advance as shown in Table 1, it aggregates resources into the private resource pool and then into the global pool by means of the algorithm. The result is shown in Fig. 3. It indicates the reliability and practicability of the RVM and the algorithms. Through verifications the result is identical to the given data, which indicates the reliability and practicability of the RVM and the algorithms.

Table 1 Resource deployment of nodes

Private clouds	Nodes	Resource edges of nodes
G_1	n_{1-1}	$\langle n_{1-1}, r(1, 1)\rangle$,$\langle n_{1-1}, r(2, 2)\rangle$,$\langle n_{1-1}, r(3, 1)\rangle$
G_2	n_{2-1}	$\langle n_{2-1}, r(0, 300\,\mathrm{M})\rangle$,$\langle n_{2-1}, r(1, 4)\rangle$,$\langle n_{2-1}, r(3, 2)\rangle$,
	n_{2-2}	$\langle n_{2-2}, r(0, 200\,\mathrm{M})\rangle$,$\langle n_{2-2}, r(1, 9)\rangle$,$\langle n_{2-2}, r(2, 5)\rangle$,$\langle n_{2-2}, r(3, 6)\rangle$
G_3	n_{3-1}	$\langle n_{3-1}, r(0, 500\,\mathrm{M})\rangle$,$\langle n_{3-1}, r(3, 50)\rangle$
	n_{3-2}	$\langle n_{3-2}, r(0, 600\,\mathrm{M})\rangle$,$\langle n_{3-2}, r(2, 30)\rangle$,$\langle n_{3-2}, r(3, 6)\rangle$
	n_{3-3}	$\langle n_{3-3}, r(0, 100\,\mathrm{M})\rangle$
G_4	n_{4-1}	$\langle n_{4-1}, r(0, 400\,\mathrm{M})\rangle$,$\langle n_{4-1}, r(1, 60)\rangle$
	n_{4-2}	$\langle n_{4-2}, r(0, 300\,\mathrm{M})\rangle$,$\langle n_{4-2}, r(3, 60)\rangle$
	n_{4-3}	$\langle n_{4-3}, r(1, 12)\rangle$,$\langle n_{4-3}, r(2, 60)\rangle$,$\langle n_{4-3}, r(3, 15)\rangle$
	n_{4-4}	$\langle n_{4-4}, r(1, 6)\rangle$,$\langle n_{4-4}, r(3, 100)\rangle$

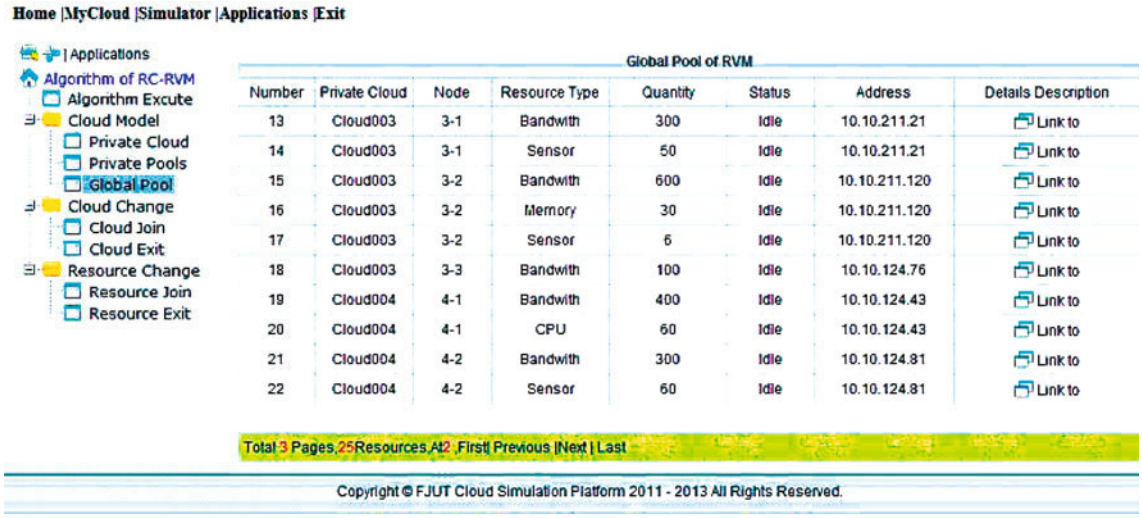


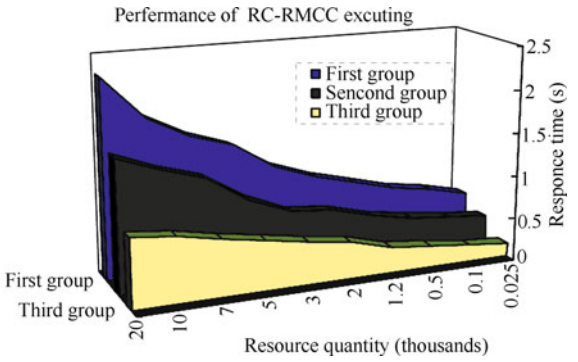
Fig. 3 The global pool after the operation of RC-RVM algorithm

Then under the condition of the same group of data, delete one private cloud in the group and continue the operation of the RC-RVM algorithm after resource converging. The algorithm will detect the missing of the private cloud. Simultaneously, the global pool that we have obtained also contains no resources of this private cloud after a thorough examination. In addition, a respective adding and deletion of resources are conducted to find that the global pool is identical with the actual data after the converging. All these experiments verify the correctness and reliability of the RC-RVM algorithm in the RVM.

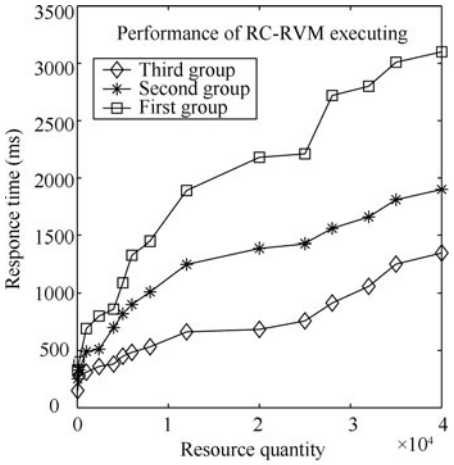
Later, the RC-RVM algorithm will undergo verification using a lot of random data. Experiment conducted in this section mainly aims to further test the efficiency and reliability of the algorithm by introducing a large quantity of random data. As a matter of fact, in the real cloud computing environment, he resources are various in type and massive in quantity. On the other hand, they are in the constant dynamic changes and heterogeneous environment. It is complex with dynamic variation from invalidation or set free or availability. It is really a challenge to aggregate and supervise such complex and complicated resources, which requires the algorithm to have a good reliability and high efficiency. During the experiment, the resource simulation performs in each node and randomly produces massive resources which are divided into different groups. With a gradual increase of data quantity in each group, the state of resource is accordingly set to be idle or invalidation in some random way. Each group respectively aggregates the resource by the performance of the RC-RVM algorithm, and the experimental results are gathered for the sake of an observation of the private cloud pool and the global pool. The observation especially focuses on the efficiency of the RC-RVM algorithm and the difference of each group′s experimental results.

Two tests are designed for the algorithm, resource quantity of the second test is approximately twice as the first test, and each test is done in three groups respectively with each group having a different distribution of resource. The characteristic of the resource distribution for the first group is well distributed, which means that not only the resource types and quantity but also their increasing trend which resource simulator produces in each node of the private cloud have variations. Resource distribution in the second group is relatively concentrated, which means that the resource produced in each private cloud is similar in terms of type, quantity and increasing trend. But when it comes to the internal of the specific private cloud, resource is mainly concentrated in several nodes, which leads to the relatively concentrated distribution of resources in the private cloud. The concentration characteristic of resource distribution is greatly evident for the third group, the resource distribution only concentrates in a few nodes of a limited-number private clouds.

According to the different distributions of aforementioned three kinds of resource, we conduct "algorithm execution" to gather the experimental results and further analyze them in a statistical method as shown in Fig. 4. The results of the two tests reflect the response time to be a roughly similar trend.



(a) First test



(b) Second test

Fig. 4 The RC-RVM algorithm execute comparison of different resource distribution

According to Fig. 4, three groups of each test have a variation in time performance which also varies differently according to the changed increasing of data quantity. The third group gets the best time performance with the second and third groups following. From the perspective of variation trend, the variation in the third group is the slowest with the second and first groups following. An conclusion can be drawn from the analysis that the time in the process of network transmission is indeed a fundamental part for each group, which has great influence on the experimental results. Nevertheless the transmission time exists in each group, it will not affect the comparison of the experimental results of the three groups. The differences of the experimental results of the three groups mainly demonstrate themselves in the distribution of resource. In the experiment of group 3, because of the concentration of statistics, the RC-RVM algorithm mainly conducts the search limited to several nodes in several private clouds. Accordingly, it has the least time complexity. Even there is an increase in the data quantity, the algorithm only increase its computing volume a bit without enlarging the search for private

clouds or nodes Consequently, the increase of time performance is relatively slow. On the contrary, the experiments of both the second and first groups are in search of more private clouds and nodes than the third group, thus their times are bigger and they increase more obviously. Especially in the first group, a thorough search for all nodes in all private clouds is required, therefore its response time is the longest of the three and its variation of response time is most obvious. It should be pointed out that the experimental result for the first group shows the time performance is in accordance with an increase in data quantity. It demonstrates that a sharp increasing occurs when the algorithm is confronted with numerous private clouds, nodes and data. The main reason lies in the fact that the RC-RVM algorithm is conducted to start a thorough execution for the overall cloud computing system, which leads to the concentration of computing quantity. If combined with distributed design, it will enable us to cope with a sharp increasing in response time which is a next problem for us to improve the RC-RVM algorithm.

6 Conclusions

This paper puts forward an RVM resource virtualization model based on the characteristic of cloud computing environment. On one hand, the model can give a full description of its characteristic by representing the private clouds, nodes in private clouds, resources (including its type and quantity) and their relevance to each other. On the other hand, the model can depict the logics among these entities. The RVM can also simulate the dynamic variation of the cloud computing environment and make a dynamic adjustment to the variation in private clouds or resource to maintain the reliability of the description of cloud computing resource. The RC-RVM algorithms which can conduct the resource converging and dynamic updating is designed for the RVM. To verify the RVM and RC-RVM algorithms proposed in this paper, a simulation platform of cloud computing whose resource infrastructure conforms to the characteristic of the RVM model and a resource simulator which is used to produce massive simulated resource for experiments are designed. Through the experiments, the algorithm can aggregate different kinds of resources with relatively good time performance in a reliable way. Nevertheless, some problems do exist as well. For example, the resource distribution affects the algorithm a lot though it has no influence on the correctness of the algorithm execution. The distributed computing designs should be strengthened to reduce the affection by the resource quantity and distribution in the algorithm.

This paper aims to provide an innovative theoretical inspiration for the description of cloud computing resource as well as its converging. At the same time, a resource converging method based on the specific model is put forward. The method and model description will be of great significance to the development and application of the cloud computing technologies. As for the aspects such as how to get a model that enables a thorough detailed description of cloud computing environment, how to improve its practicability, and how to put the technologies mentioned above into applications in real systems are what we should study in the future.

Acknowledgement

Here, we would like to acknowledge and extend our heartfelt gratitude to our colleagues for their generous assistance.

References

[1] I. Foster, Y. Zhao, I. Raicu, S. Lu. Cloud computing and grid computing 360-degree compared. In *Proceedings of the Grid Computing Environments Workshop*, IEEE, Austin, TX, USA, pp. 1–10, 2008.

[2] R. Buyya, S. Y. Chee, S. Venugopal, J. Broberg, I. Brandic. Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation Computer Systems*, vol. 25, no. 6, pp. 599–616, 2009.

[3] H. L. Truong, S. Dustdar. Cloud computing for small research groups in computational science and engineering: Current status and outlook. *Computing*, vol. 91, no. 1, pp. 75–79, 2011.

[4] J. Z. Luo, J. H. Jin, A. B. Song, F. Dong. Cloud computing: Architecture and key technologies. *Journal on Communications*, vol. 32, no. 7, pp. 3–21, 2011.

[5] Y. W. Zhu, L. S. Huang, G. L. Chen, W. Yang. Dynamic trust evaluation model under distributed computing environment. *Chinese Journal of Computers*, vol. 34, no. 1, pp. 55–64, 2011.

[6] B. Charron-Bost, A. Schiper. The Heard-of model: Computing in distributed systems with benign faults. *Distributed Computing*, vol. 22, no. 1, pp. 49–71, 2009.

[7] P. Reeser, R. Hariharan. An analytic model of web servers in distributed computing environments. *Telecommunication Systems*, vol. 21, no. 2–4, pp. 283–299, 2002.

[8] A. Lastovetsky. Adaptive parallel computing on heterogeneous networks with MPC. *Parallel Computing*, vol. 28, no. 10, pp. 1369–1407, 2002.

[9] A. Clematis, A. Corana. Modeling performance of heterogeneous parallel computing systems. *Parallel Computing*, vol. 25, no. 9, pp. 1131–1145, 1999.

[10] C. T. Chang, C. Y. Chang, J. P. Sheu. BlueCube: Constructing a hypercube parallel computing and communication environment over Bluetooth radio systems. *Journal of Parallel and Distributed Computing*, vol. 66, no. 10, pp. 1243–1258, 2006.

[11] A. Chervenak, I. Foster, C. Kesselman, C. Salisbury, S. Tuecke. The data grid: Towards an architecture for the distributed management and analysis of large scientific datasets. *Journal of Network and Computer Applications*, vol. 23, no. 3, pp. 187–200, 2000.

[12] Q. Liang, Y. Yang, K. J. Liang. Guarantee and control of quality of service on grid system: A survey. *Control and Decision*, vol. 22, no. 2, pp. 121–126, 2007. (in Chinese)

[13] K. Amin, G. von Laszewski, M. Hategan, R. Al-Ali, O. Rana, D. Walker. An abstraction model for a Grid execution framework. *Journal of System Architecture*, vol. 52, no. 2, pp. 73–87, 2006.

[14] Q. Liang, Y. Z. Wang. The representation and computation of QoS preference with its applications in grid computing environments. *Annals of Telecommunications*, vol. 65, no. 11–12, pp. 705–712, 2010.

[15] W. Q. Tong, W. K. Miao. Performance predictable servicebsp model for grid computing. *Wuhan University Journal of Natural Sciences*, vol. 12, no. 5, pp. 871–874, 2007.

[16] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, R. Buyya. CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and Experience*, vol. 41, no. 1, pp. 23–50, 2011.

[17] K. Q. Li, L. T. Yang, X. M. Lin. Advanced topics in cloud computing. *Journal of Network and Computer Applications*, vol. 34, no. 4, pp. 1033–1034, 2011.

[18] D. G. Feng, M. Zhang, Y. Zhang, Z. Xu. Study on cloud computing security. *Journal of Software*, vol. 22, no. 1, pp. 71–83, 2011.

[19] K. J. Liang, Q. Liang. A dynamically scalable model for cloud computing based on graph structure. *International Journal of Advancements in Computing Technology*, vol. 4, no. 17, pp. 125–134, 2012.

[20] K. J. Liang, Q. Liang. A dynamic method of model constructing for cloud computing. *International Journal on Advances in Information Sciences and Service Sciences*, vol. 4, no. 17, pp. 174–182, 2012.

[21] D. Ergu, G. Kou, Y. Peng, Y. Shi, Y. Shi. The analytic hierarchy process: Task scheduling and resource allocation in cloud computing environment. *The Journal of Supercomputing*, vol. 64, no. 3, pp. 835–848, 2013.

[22] S. Subashini, V. Kavitha. A survey on security issues in service delivery models of cloud computing. *Journal of Network and Computer Applications*, vol. 34, no. 1, pp. 1–11, 2011.

[23] G. H. Tian, D. Meng, J. F. Zhan. Reliable resource provision policy for cloud computing. *Chinese Journal of Computers*, vol. 33, no. 10, pp. 1859–1872, 2010.

[24] P. Zheng, L. Z. Cui, H. Y. Wang, M, Xu. A data placement strategy for data-intensive applications in cloud. *Chinese Journal of Computers*, vol. 33, no. 8, pp. 1472–1480, 2010.

[25] A. Iosup, S. Ostermann, M. N. Yigitbasi, R. Prodan, T. Fahringer, D. H. J. Epema. Performance analysis of cloud computing services for many-tasks scientific computing. *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 6, pp. 931–945, 2011.

[26] Y. Z. Wang, C. Lin, P. D. Ungsunan, X. M. Huang. Modeling and survivability analysis of service composition using stochastic Petri nets. *The Journal of Supercomputing*, vol. 56, no. 1, pp. 79–105, 2011.

[27] Y. Z. Wang, M. Yu, J. Y. Li, K. Meng, C. Lin, X. Q. Cheng. Stochastic game net and applications in security analysis for enterprise network. *International Journal of Information Security*, vol. 11, no. 1, pp. 41–52, 2012.

[28] J. S. Qu, J. H. Dai. The concept research on task/resource graph modeling method for complex discrete real-time system. *Journal of System Simulation*, vol. 12, no. 6, pp. 600–603, 2000. (in Chinese)

[29] T. Gu, H. K. Pung, D. Q. Zhang. A service-oriented middleware for building context-aware services. *Journal of Network and Computer Applications*, vol. 28, no. 1, pp. 1–18, 2005.

[30] M. Arun, A. Krishnan. Functional verification of signature detection architectures for high speed network applications. *International Journal of Automation and Computing*, vol. 9, no. 4, pp. 395–402, 2012.

[31] X. J. Chen, J. Zhang, J. H. Li, X. Li. Resource reconstruction algorithms for on-demand allocation in virtual computing resource pool. *International Journal of Automation and Computing*, vol. 9, no. 2, pp. 142–154, 2012.

Quan Liang graduated from Southwest Jiaotong University, China in 1996. He received the M. Sc. degree from Central South University of Forestry Science and Technology China in 2004 and the Ph. D. degree from Beijing University of Science and Technology, China in 2008. He is currently an associate professor at the School of Information Science and Engineering, Fujian University of Technology, China. He has published about 60 refereed journal and conference papers, in which proximately 40 papers are indexed by SCI or Ei. He is a reviewer or presiding committee member of *International Journal of Computational Information System, Journal of Grid Computing, Journal of Network and Computer Application, ICECE 2010* and *ICCSIT 2011*.

His research interests include network computing and sensor networks.

E-mail: liangquanlq@gmail.com (Corresponding author)

Yuan-Zhuo Wang graduated from Yanshan University, China in 2001. He received the M. Sc. degree from Yanshan University, China in 2004 and the Ph. D. degree from Beijing University of Science and Technology, China in 2008. He is currently an associate professor at the Institute of Computing Technology, Chinese Academy of Sciences China.

His research interests include network computing and analysis of performance.

E-mail: wyzking@163.com

Yong-Hui Zhang graduated from Central South University, China in 1997. He received the M. Sc. degree from Central South University, China in 2003, and the Ph. D. degree from Central South University China in 2010. He is currently an associate professor at the school of Information Science and Engineering, Fujian University of Technology.

His research interests include wireless sensor networks.

E-mail: 13012478@99.com