Hybrid Cloud Computing Platform for Hazardous Chemicals Releases Monitoring and Forecasting

Xuelin Shi
Beijing University of Chemical Technology, Beijing, China
Email: shixl@mail.buct.edu.cn

Yongjie Sui and Ying Zhao
Beijing University of Chemical Technology, Beijing, China
Email: suiyongj@yahoo.com.cn , zhaoy@mail.buct.edu.cn

***Abstract*—While hazardous chemicals pollutions occurring, monitoring and forecasting are very important for emergency planning and evacuation. To implement the object, huge amounts of wireless sensors may be distributed in large area. The data gathered by the sensors are sent back to data centers for preprocessing and analysis, which need massive computing utility. Therefore a high effective computing paradigm is essential for the hazardous chemicals releases monitoring and forecasting. This paper brought out a hybrid cloud computing platform to solve the problem. It is an Infrastructure as a Service (IaaS) platform, which provides High Performance Computing (HPC) resources, virtualized computing resources, storage resources for hazardous chemicals releases data processing and analysis. Especially we designed a scheduling algorithm and QoS policy to assure efficiency of the platform. At last the platform capability and performance were demonstrated by application scenarios.**

***Index Terms*—Cloud Computing, Scheduling, QoS, Harzadous Chemicals**

I. INTRODUCTION

Releases of hazardous chemicals are main cause of major accidents, which includes accidents in the chemical industry as well as terrorist attacks [1]. While hazardous chemicals pollutions occurring, monitoring and forecasting are very important for emergency planning and evacuation. To implement the object, huge amounts of wireless sensors may be distributed in large area. The data gathered by the sensors are sent back to data centers for preprocessing and analysis, which need massive computing.

With development of Information and Communication Technology, computing will one day be the 5th utility (after water, electricity, gas, telephony) [2]. Computing resources are always distributed dispersedly, which connected with networks. Therefore a high effective computing paradigm is essential for the hazardous chemicals releases monitoring and forecasting.

A number of computing paradigms have been proposed: cluster computing, Grid Computing, and more recently cloud computing. Cloud computing has been an emerging model which aims at allowing customers to utilize computational resources and software hosted by service providers [3].

Buyya defined a definition for cloud as follows: "A Cloud is a type of parallel and distributed system consisting of a collection of inter-connected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resource(s) based on service-level agreements established through negotiation between the service provider and consumers." [2]

According this definition, clouds appear to be a combination of clusters and Grids. Clouds often deal with large amount of resources including processors, memories, storages, visualization equipment, software, and so on.

In this paper we examine how monitoring and forecasting of hazardous chemicals can take advantage of cloud computing to offer timely emergency decision support. The data processing involves the analysis of chemical components, the tracing of chemical reaction, the simulation of computational fluid dynamics (CFD), and so on. It is complex, time consuming, and expensive. To lower costs and improve efficiency, cloud computing is an affordable solution in this area.

Clouds shift the responsibility to install and maintain hardware and basic computational services away form the users (e.g., a laboratory or chemical scientist) to the cloud vendor. Higher levels of the application stack and administration of sharing remain intact, and remain the users' responsibility [4].

Our key contributions are the following: (1) an IaaS platform for cloud computing named Hazardous Chemicals Monitoring Cloud (HCMC), which is a hybrid cloud supporting High Performance Computing uniform computing services and data storage services; (2) a resource scheduling algorithm encompassing both customer-driven service management and computational risk management to sustain Service Level Agreement (SLA)-oriented resource allocation; and (3) a QoS policy to meet different users' demands and improve scheduling efficiency.

The remainder of the paper is organized as follows. Section 2 gives the architecture of HCMC. In section 3,

doi:10.4304/jcp.7.9.2306-2311

we introduce the resource scheduling model and evolutionary algorithm applied in HCMC. Then in section 4, we show how a QoS policy can be used to assure important jobs be processed in high priority. We give an application scenario of HCMC in section 5. At last section 8 give concluding remarks.

II. ARCHITECURE OF HCMC

Data sensing and processing are key problems of hazardous chemicals releases monitoring and forecasting (HCRMF), computing service of which is different from commercial public clouds. The commercial public clouds, such as Amazon Elastic Computing Cloud (EC2), Google App Engin (GAE), often provide several kinds of computing services to meet users' different needs based on virtualization technology. But HCRMF computing platform should not only support the HPC for massive data processing, but also virtualization technology for domain users' computing requirements. Therefore, this paper presents a hybrid cloud platform architecture, Hazardous Chemicals Monitoring Cloud (HCMC), which provides hybrid management for HPC and Virtual Private Cloud.

This section first describes the overall framework of HCRMF, including data sensing, computing platform, user interface and so on. Secondly architecture of HCMC is given.

A. HCRMF Framework

HCRMF system provides functions as: data collection, transmission, analysis, monitoring, and so on. Fig. 1 gives the framework of HCRMF.

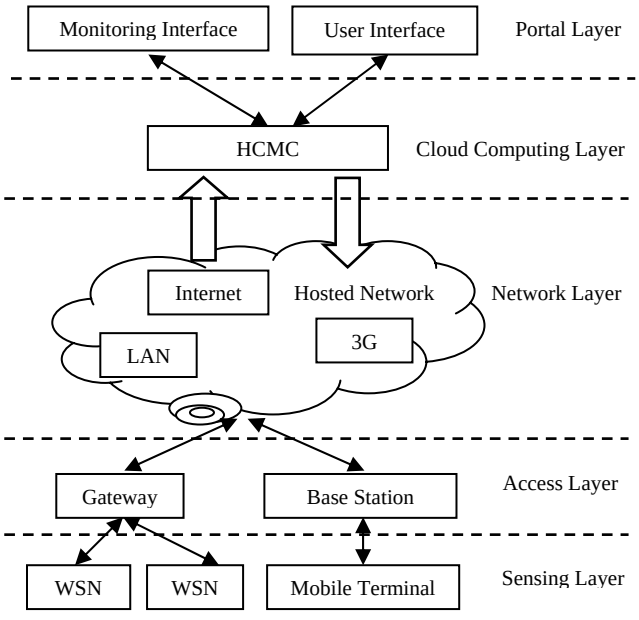

Figure 1. The HCRMF Framework

HCRMF has five layers: sensing layer, access layer, network layer, cloud computing layer and portal layer.

Sensing layer is responsible for hazardous chemicals monitoring and data collection, which consists of Wireless Sensors Network (WSN) and other mobile terminals. WSN is composed of many widely distributed automatic devices, which use sensors to collect the environmental data. Besides of WSN other mobile terminals also can be used, such as police cars equipped with the GPS, cameras, and detection instruments. These mobile terminals can become data-sensing nodes and transmit data to the base stations of the network layer.

Access layer includes base stations and access gateway, which provide functions of sensing layer devices control, data collection and protocol conversion. In the layer, not only the data from sensing layer will be transformed to suit transmission in network layer, but also the instructions from upper layers are distributed to sensing layer.

Network layer is holder of data transmission, which can consist of Internet, LAN, and 3rd generation mobile communications networks. It is connection of the access layer and the cloud computing layer.

Cloud computing layer is the core of the HCRMF framework, which manage HPC servers, virtual machines, and data storage devices. It provides computing utility for data processing and analysis gathered by devices of sensing layer. In addition, it responds computing requests submitted by domain users with portal. This paper presents a hybrid cloud platform HCMC to achieve the above functions.

Portal layer provides monitoring interface and user interface for users to monitor environment and submit computing requests. By real-time monitoring interface, users can retrieve monitoring data and analysis and issue control instructions. As the cloud computing layer provides some software services, such as CFD software, evacuation simulation software and so on, users can submit computing requests by user interface.

B. Hybrid Cloud Arthitechture

Our HCMC is a unified hybrid computing model of managed HPC and other computing resources.

HCMC is not only a pool of virtual machines, but also a set of cluster-level servers. Running HPC workloads on virtual machines was considered impractical for two key reasons: (1) the overhead of virtualization; (2) the virtual machine instances were not backed by a high performance interconnect and storage [5]. Because hazardous chemicals monitoring need massive computing utility, HPC cluster servers with special hardware, such as low-latency interconnects, storage-area networks, multi-core nodes, hundreds of Gigabytes of main memory, are necessary. Therefore HCMC provides two kinds of resources management mechanism: Workload and Resource Management System (WRMS) for HPC cluster servers and Dynamic Infrastructure Management Service (DIMS) for private cloud, i.e. a pool of virtual machines. Fig.1 gives the architecture of HCMC.

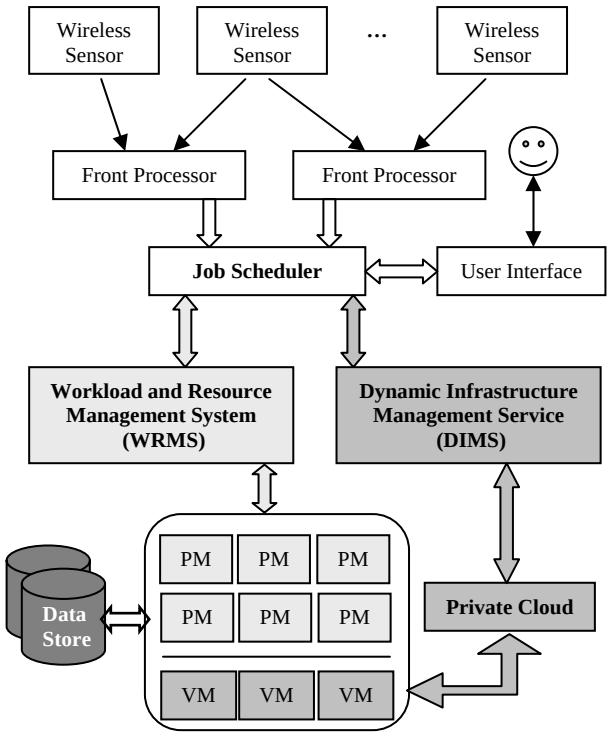

Figure 2. The HCMC Architecture

HCMC are candidates for several roles in hazardous chemicals monitoring, ranging from compute services to data storage. Daily log processing often uses basic machine, while emergency planning and forecasting may rely on clusters of high-performance machines. As shown in Fig.2, HCMC is a unified model of managed HPC and Cloud resources. The working nodes of HCMC can run either traditional physical machines (with the operating system statically bound to the hardware), denoted by PM, or virtual machines (with the operating system dynamically bound to the hardware), denoted by VM.

Jobs enter the HCMC through the Job Scheduler who puts them in a submit queue, decides when to dispatch them to the WRMS or DIMS.

Distributed wireless sensors gather data on hazardous chemicals releases and send them to front processors. The abnormal or key data needing processing will be submitted to Job Scheduler. These jobs are fixed and should be executed on HPC resources according to a set of scheduling policies defined by the system administrators. Therefore they were dispatched to WRMS by Job Scheduler.

WRMS supports three kinds of functionality: (1) resource management, including resource status and capability; (2) job management, including creation, queuing, monitoring, and control; (3) job scheduling, i.e., mapping a job to a set of resources and allocating the resources to that job for a certain time. In HCMC, WRMS is developed with Load Sharing Facility (LSF).

Furthermore users (e.g. a laboratory or chemical scientist) also can submit jobs by user interface. These jobs may be data mining of daily logs, simulation computing, and so on, which are often not emergent and throughput-intensive jobs. Therefore Job Scheduler will dispatch them to DIMS.

DIMS supports two kinds of functionality: (1) physical resource management, including resources status and capability; and (2) service management, including service creation (and the related allocation of underlying physical resources), monitoring, migration, and destruction. In HCMC, DIMS is implemented with OpenNebula. It is an open source, virtual infrastructure manager that deploys virtualized services on both local pool of resources and external IaaS clouds [6].

HCMC also has a data store, which provides distributed, transparent and reliable cloud storage service.

III. RESOURCES SCHEDULING OF HCMC

We proposed a scheduling model for HCMC and designed evolutionary algorithm. In the model priority weight is introduced to implement differential services. Then the algorithm is explained in detail.

A. Scheduling Model

Consider ***C*** as a set of machines (maybe PMs or VMs) in a HCMC environment. Assume that there are m machines in ***C***. Also, suppose ***J*** is the set of jobs and there are n jobs in ***J***.

According to known parameters often used in the existing techniques [7], we defined parameters about jobs and machines are as follows:

t_{ij} total processing time required for the ith job if assigned to the jth machine
d_i deadline of the ith job
r_i ready time of the ith job
p_j the price/unit time for the jth machine
l_i weight of the ith job

A job is the description of a compute task, which consists of its resource requirements (i.e. t_{ij}) and its timing constraints (i.e. d_i and r_i). Furthermore we add l_i, a priority weight parameter, which can be predefined for differential services objective.

So the good cloud scheduler not only minimize the costs of machines (i.e. $\sum t_{ij}\, p_j$), but also take d_i and l_i into account to avoid the loss. The job with big weight should be assigned to machines preferentially in order to assure it can be done before deadline, otherwise loss occurs. To quantize the loss, we use to $l_i \times$ time delay represent it.

Therefore, we got the cloud scheduling model for HCMC:

Problem:

Job scheduling problem

Instance:

J = $\{j_1, j_2, \ldots, j_n\}$, the set of n jobs. ***C*** = $\{c_1, c_2, \ldots, c_m\}$, the set of m machines. **T** = $[t_i]$, the processing time matrix. **P** = $\{ p_1, p_2, \ldots, p_m \}$, the set of price/unit time for m machines. **D** = $\{d_1, d_2, \ldots, d_n\}$, the set of deadline for n jobs. **R** = $\{r_1, r_2, \ldots, r_n\}$, the set of ready time for n jobs. **L** = $\{l_1, l_2, \ldots, l_n\}$, the set of weight for n jobs.

Output:

$$\Sigma(t_{ij} p_j + max(0, l_i (s_i + t_{ij} - d_i))) \quad (1)$$

where s_i is start executing time of the ith job, $max(0, l_i (s_i + t_{ij} - d_i))$ is loss of the job$_i$, if the job$_i$ completes before its deadline time, the loss is 0; otherwise, the loss is $l_i (s_i + t_{ij} - d_i)$. It creates a priority among jobs. $\Sigma(t_{ij} p_j + max(0, l_i (s_i + t_{ij} - d_i)))$ is objective function.

B. Evolutionary Scheduling Algorithm

After constructed the scheduling model, we designed an evolutionary algorithm to obtain the optimal solution. In this section we introduce its design and implementation.

Genetic algorithms (GAs) are search algorithms based on the mechanics of natural selection and natural genetics [8]. Evolutionary algorithms (EAs) are GAs with special data structures or special encodings of solutions or genetic operator based on the problem [8]. Being an EA, the basic components to implement our algorithm are chromosome encoding, fitness function and population mutation and crossover.

To minimize $\Sigma(t_{ij} p_j + max(0, l_i (s_i + t_{ij} - d_i)))$, it is clear that not only to assign the job$i$ to the machine with highest performance for executing it (i.e. t_{ij} is smallest), but also to assure the jobs be executed in time to avoid loss. We can think jobs with higher weight (i.e. l_i) and urgent deadline (i.e. d_i) have a higher rank. The scheduling is equivalent to search the combinations of jobs and machines to select the optimized schedule.

Therefore we designed the combination of job$_i$ (let it be β_i) and machine$_j$ (let it be γ_j) as gene (β_i, γ_j), which form sets called chromosomes.

Under this gene encoding, a chromosome of such a subpopulation in which each job appears once and only once are raw schedules. The objective of our algorithm is to find an optimized schedule by selection the chromosomes of the offspring subpopulations.

To evaluate an offspring, we defined the fitness function as followed:

$$fitness = \Sigma(t_{ij} p_j + max(0, l_i (s_i + t_{ij} - d_i))) \quad (2)$$

Usually a random population is used as initial population in EAs, but if the quality of the initial population is better than the average of the random population the efficiency of the EAs can be increased. In our algorithm, in order to avoid precocity we still select genes randomly as initial population, $\boldsymbol{P}$. Then genes (β_i, γ_j) of $\boldsymbol{P}$ are sorted in non-decreasing order of theirs $t_{ij} p_j + max(0, l_i (s_i + t_{ij} - d_i))$ to generate template population $\boldsymbol{TP}$. In each iteration, the new population $\boldsymbol{newP}$ is constructed by

$$\boldsymbol{newP} = \boldsymbol{TP} \cup \{x\% \text{ elites in } \boldsymbol{P}\} \quad (3)$$

where x is a predefined parameter. In order to keep elites of parent population, x is let to be 50% in our algorithm.

Recombination and mutation operators change chromosomes genes and create chromosomes for the template populations. The simple crossover mutation is performed. The algorithm works iteratively till stopping criterion is satisfied, i.e. evolutionary generations reach the predefined parameter.

IV. QOS POLICY OF HCMC

QoS concerns operational characteristics of a service that determine its utility in an application context. QoS offers a basis for service differentiation and is also an important factor of competition [9]. In the same way QoS becomes an important concern in cloud computing. As users rely on cloud providers to supply more of their computing needs, they will require specific QoS to be maintained by their providers in order to meet their objectives and sustain their operations. Cloud providers will need to consider and meet different QoS parameters of each individual consumer. In a highly competitive cloud environment, QoS is one of the crucial means for satisfying various demands from resource users and providers [10].

Therefore an effective cloud scheduling policy should take QoS into account. A dynamic environment like HCMC, it is necessary to assure important jobs getting computing resources with priority. In order to solve the problem we designed a QoS policy to improve scheduling efficiency. The QoS policy can sustain operations of important jobs by different levels of QoS.

Our QoS management framework mainly consists of two components: QoS concepts and QoS mechanism.

A. QoS Concept

A consistent and widely applied set of Qos concepts is essential for effective QoS-aware service chaining. QoS concepts enable quality of service to be modeled, specified, communicated, negotiated, controlled and managed. ISO 13236 (ISO/IEC, 1998) defines a generic set of QoS concepts that can be applied for modeling QoS in information technology (IT) systems that provide distributed processing services [9].

In our QoS management framework we defined the concepts as followed: user requirement, use pattern and predictive resource reservation.

User requirement is a quantifiable aspect of quality that is desirable in an interaction or that is necessary for user satisfaction. According to our economic scheduling model, user requirements can be presented by processing time of his or her job executing in some node of cloud, budget, and penalty price.

To promote service efficiency, it is very useful to be able to predict when a given computational resource will be idle, becoming available for gird applications. The resource Use Pattern Analysis (UPA) method is often used to predict resource, which based on the assumption that resource availability at each node can be modeled [11]. Based on the concept of user requirement, we defined another QoS concept: use pattern. At present we only consider CPU use pattern.

The last QoS concept is predictive resource reservation. Based on UPA the user's requirement can be predicted, therefore some resource can be reserved for "important jobs". Resource reservation also expresses different levels

of services: resources reserved for "important jobs" but not allocate to arrived jobs. In our QoS management framework resource reserving is a threshold, which defines when and how many resources should be reserved.

B. QoS Mechanism

After defined QoS concepts, we designed QoS mechanism: scheduling with predictive resource reserving, i.e. based on UPA pre-reserving resources for "important jobs".

The effective approach to acquire resource UP is categorizing log of resource using, such as CPU use, available RAM, disk space, swap space, network and disk I/O. Presently most of operating systems have such functions to generate system logs.

These system logs are often semi-structured data, and there are many categorization methods for semi-structured text which can be generally classified into supervised learning, also called classification, unsupervised learning, also called clustering, and semi-supervised learning [12]. In our work, unsupervised learning is used for UPA mining.

Firstly, original log data is preprocessed and represented in the Vector Space Model (VSM) [13]. In this model, each log record is identified by a feature vector in a space in which each dimension corresponds to a distinct item log associated with a numerical value indicating its weight. At present we just considered CPU use and hoped to find users' use pattern in some period. The resulting representation of one node log in one period is, therefore, equivalent to a n-dimensional vector:

$$d = <(t_1,w_1),(t_2,w_2)\ldots(t_n,w_n)> \quad (4)$$

In the vector w_j represents the numerical CPU usage value of the user t_j in one period d. As a result, we get VSM to represent all cloud nodes resource usage as a matrix: $X \in R^{n\times d}$, where n represents the nodes number of the cloud, and d is node log. To each element X_{ij} of this matrix is assigned the number of CPU usage of user t_j on node d_j. After constructed resource usage matrix (simply named as X), we used k-NN to classify very row of $\boldsymbol{X}$, i.e. vector d.

K-NN finds the k nearest neighbors of the test document, and then uses majority voting among the neighbors in order to decide the log category. Similarity between two vectors is used to decide whether neighbors are near or far, and it is measured by the cosine between the vectors:

$$sim(d_i,d_j) = \frac{\sum_{k=1}^{r} w_{ik} * W_{jk}}{\sqrt{\sum_{k=1}^{r} w_{ik}^2} * \sqrt{\sum_{k=1}^{r} w_{jk}^2}} \quad (5)$$

When new resource usage matrix is given to the k-NN algorithm, for each row of $\boldsymbol{X}$ (i.e. vector d), the similarities among vectors are computed, then they were classified into several categories.

With the above mechanism, the rules of resource usage are found from system logs. When scheduling jobs, these rules can be used to predict when the user will submit a job and how long this job will be processed on a node. Therefore resources can be reserved for "important jobs" to improve scheduling efficiency.

V. Application Scenarios

In this section, we describe how scheduling algorithm and QoS policy deployed on HCMC. Then a scenario of an example job execution workflow is given.

HCMC is a hybrid cloud platform, which provides HPC and virtual computing services. Therefore job scheduling and QoS policy should work based on WRMS and DIMS, which is implemented by a layered scheduling system. It consists of a global job scheduler, WRMS schedulers and DIMS schedulers. As previously stated, HCMC provides hybrid computing resources, this system is effective for managing different resources at distributed sites.

Fig.3 gives the scheduling system deployment on HCMC.

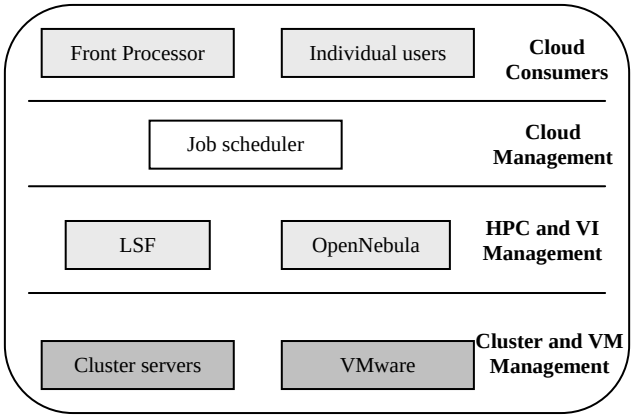

Figure 3. The deployment of HCMC

Key component of the system is the global job scheduler. All machines of HCMC are connected with computer network. Global job scheduler performs top-down scheduling decisions in the cloud platform, while HPC and Virtual Infrastructure (VI) managers wait instructions from the job scheduler to execute jobs.

The layered scheduling management architecture has advantage to effectively control workflow execution on hybrid cloud like HCMC. So our scheduling solution can be performed effectively.

A scenario of an example job execution workflow is following: first, a user submits the computing requests by the portal of HCMC, or a front processor sends data to HCMC for analysis.

Second, the request is sent to the job scheduler. Thirdly the job scheduler set priority weight and other parameters of the job. Then it dispatches the job to HPC manager or VI manager to perform the computation according to the job's requirement. HPC manager (i.e. LSF) or VI manager (OpenNebula) locate one or more machines to

execute the job. Finally the computing results are returned to the user or administrators. Where the requests are submitted, how the computing requests is allocated, and the monitoring and management for computing tasks are all completed by the HCMC transparently.

VI. Conclusion and Future Work

This paper presented a hybrid cloud computing platform HCMC. HCMC is an IaaS platform, which provides HPC and virtualized computing resources, storage resources for hazardous chemicals releases data processing and analysis. Especially we designed a scheduling algorithm and QoS policy to assure efficiency of the platform. At last the platform capability and performance were demonstrated by application scenarios.

Now HCMC is still in theory research stage, but in future it will put into service for some key areas' hazardous chemicals monitoring. Many chemical domain specialists believe that clouds represent the next generation of mass computing services. HCMC is very suitable for processing data from huge numbers of distributed wireless sensors.

Our work is a step toward realizing how cloud computing is used in hazardous chemicals releases monitoring and forecasting to support emergency planning. In future work, a friendly cloud portal for users to access computing and storage resources is one key problem. Furthermore security of the cloud platform is also important.

Acknowledgment

This work was supported in part by the National Grand Fundamental Research 973 Program of China (No. 2011CB706900).

References

[1] Xiaoping Zheng, Zengqiang Chen: Inverse Calculation Approaches for Source Determination in Hazardous Chemical Releases. Journal of Loss Prevention in Process Industries, 2011, doi:10.1016/j.jlp.2011.01.002.

[2] Rajkumar Buyya, Chee Shin Yeo, Srikumar V., James B., Ivona B.: Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. Future Generation Computer Systems. 25, 599--616 (2009).

[3] Hong-Linh Truong, Schahram Dustdar: Composable cost estimation and monitoring for computational applications in cloud computing environments. Procedia Computer Science 1, 2169-2178(2010).

[4] Arnon Rosenthal, Peter Mork, Maya Hao Li, etc.al.: Cloud Computing: A New Business Paradigm for Biomedical Information Sharing. Journal of Biomedical Informatics, 43(2010), 342-353.

[5] Gabriel Mateescu, Wolfgang Gentzsch, Calvin J. Ribbens.: Hybrid Commputing – Where HPC meets grid and Cloud Computing. Future Generation Computer Systems 27 (2011), 440-453.

[6] B. Sotomayor, R. Montero, I. Llorente, I. Foster: Virtual infrastructure manaagement in pirvate and hybird clouds. IEEE Internet Computing, 13(5)(2009), 14-22.

[7] Subodha Kumar, Kaushik Dutta, et.al.: Maximizing Business Value by Optimal Assignment of Jobs to Resources in Grid Computing. European Journal of Operational Research 194, 856-872 (2009).

[8] D.E. Goldberg. Genetic Algorightms in Search. Optimization and Machine Learning, Addison-Wesley, 1988.

[9] Richard Onchaga, Quality of service management framework for dynamic chaining of geographic information services. International Journal of Applied Earth Observation and Geoinformation, 8, 137-148 (2006).

[10] Chunlin Li, Layuan Li: A distributed multiple dimensional QoS constrained resource scheduling optimization policy in computational grid. Journal of Computer and System Science, 72(4), 706-726 (2006).

[11] Marcelo Finger, Germano C. Bezerra, Danilo R.: Conde. Resource use pattern analysis for opportunistic grids. MGC'08, December 1-5, Leuven, Belgium (2008).

[12] Soumen Chakrabarti: Data mining for hypertext: A tutorial survey. SIGKDD Explorations, 1 (2) 1-11 (2000).

[13] G. Salton, C. Yang, A. Wong: A vector space model for automatic indexing. Communications of the ACM, 613-620 (1975).

Xuelin Shi Beijing, China. Birthdate: November, 1977. is Computer Application Technology Ph.D., graduated from Dept. Computer Science Beijing Institute of Technology. And research interests on cloud computing and resource scheduling.

She is a lecturer of School of Information Science and Technology Beijing University of Chemical Technology.

Yongjie Sui Beijing, China. Birthdate: December, 1987. is graduate student of School of Information Science and Technology Beijing University of Chemical Technology.

Ying Zhao Beijing, China. Birthdate: September, 1966. is Control Theory and Control Engineering Ph.D., graduated from School of Information Science and Technology Beijing University of Chemical Technology. And research interests on computer network and computing architecture.

He is professor of School of Information Science and Technology Beijing University of Chemical Technology.