International Journal of Grid and Distributed Computing
Vol. 9, No. 3 (2016), pp.157-172
http://dx.doi.org/10.14257/ijgdc.2016.9.3.18

Building Efficient Resource Management Systems in the Cloud: Opportunities and Challenges

[1]Lin Xu and [2]Jing Li*

[1,2]School of Computer and Science Technology
University of Science and Technology of China, Hefei, China, 230026
[1]linxu@mail.ustc.edu.cn, [2]lj@ustc.edu.cn

Abstract

Cloud computing is designed as computing as a utility. Customers rent computing resources in the cloud to complete their work. To ensure the quality of service (QoS) requirements defined by customers and guarantee the resource utilization in cloud datacenters, effective resource management systems should be considered. However, with more and more individuals and enterprises migrating their work into the cloud, workloads in the cloud become more and more heterogeneous. Meanwhile, resources are much more heterogeneous as cloud providers constantly scale or update the clusters with new generations of machines. Withal, workloads are dynamic with different resource demands during their execution. Besides, machines are re-engaged frequently after removal due to various reasons such as crushing and updating. The heterogeneity and dynamicity in both workloads and resources are huge barriers to using classic resource management systems. This paper will first introduce the status quo of cloud computing environment, and then give an overview of resource demand prediction and allocation policies. Finally, challenges are proposed to help build adaptive resource management systems.

Keywords*: resource management system; cloud computing; workload; resource demand prediction; resource allocation*

1. Introduction

Cloud computing brings a surge of buzz and excitement, promising low cost, high availability, high reliability, elastic scalability and robust performance. Attracted by these charming characteristics, famous enterprises like Google, Amazon and Facebook, migrate their data into the cloud and process their transactions using large distributed computing frameworks in the cloud, such as MapReduce [1]. Additionally, individuals can access cloud computing services, by renting computing resources via Amazon's EC2. Convenience, economic benefits and flexibility make the cloud computing experience huge expansion.

Generally speaking, to complete jobs in the cloud, customers should submit their jobs to a cloud service provider with QoS requirements and manually requested computing resources. After receiving requests for computing resources, the cloud service provider checks the state of the resource utilization in the cloud datacenter at first. If requested resources are available, the cloud service provider will allocate resources to customers according to certain scheduling policies. The whole procedure is manipulated by a resource management system [2, 3, 4, 5]. A well designed resource management system can satisfy customers by accomplishing their jobs with performance goals, in a budget lower than expected. Meanwhile, it can improve the resource utilization and balance the

Jing Li is the corresponding author.

ISSN: 2005-4262 IJGDC

load in the datacenter, which indirectly brings cloud service providers incremental profit. However, it is challenging to build an efficient resource management system in the cloud due to the heterogeneity and dynamicity of the cloud.

In cloud computing environment, it is an ideal to deal with homogenous workloads [6]. Actually, as workload spans multiple customers, it is likely to be inherently diverse in resource demand. Even workloads from a single customer may vary the resource demand due to different computing objectives. Moreover, the consolidated cloud environment which is constructed by a variety of machines is also heterogeneous. Several generations of machines with different specifications can be possibly encountered as new machines with high performance cost ratio are continuously incorporated to scale the cloud computing clusters. The heterogeneity in both workloads and machines may complicate the method of allocating resources in time to meet the specified performance goals and meanwhile achieve high resource utilization. Furthermore, due to this reason, efficient resource management systems cannot be easily established.

In another perspective, the dynamicity of the cloud also complicates the problem building an efficient resource management system. On the one hand, workloads in the cloud are dynamic. Customers rent virtual machines in the cloud to complete their jobs, each possibly containing a mix of sub-tasks. The diversity of these jobs and their arrival patterns lead to a dynamic range of resource demands with significant variation over time. Traditional methods can hardly predict the future resource usage due to the highly dynamic resource usage. On the other hand, resources in the cloud are also dynamic. Machines are re-engaged after removal due to various reasons such as crushing or updating. The dynamicity of both workloads and the cloud environment motivates techniques for dynamic allocation policies, which is another challenge issue to be solved in building efficient resource management systems.

In conclusion, to accomplish jobs with personalized QoS requirements under budget, a resource management system should be maintained, which firstly predicts the resource usage, secondly allocates the resources and thirdly migrate tasks when necessary. However, due to the heterogeneity and dynamicity in both workloads and cloud computing environment, resource management system with traditional resource demand prediction and allocation policies are no longer adaptive. In this paper, we discuss the opportunities and challenges to build a resource management system in the cloud. From the perspective of architecture, a resource management system must contain various components, such as resource provisioning component, job scheduling component, VM management component, monitoring component and so on. Improvement made on any component will help construct an efficient resource management system. In this paper, we choose the resource provisioning component, which is composed of resource demand prediction and allocation, as the start point to help build an efficient resource management system. First, we will summarize the traditional resource demand prediction methods and allocation policies and point out their limits when dealing with heterogeneous workloads in the cloud. Then we describe the architecture of an efficient resource management system with new components such as SLA designer, resource predictor and so on. Finally, we propose new challenges in building efficient resource management systems.

The remainder of this paper is arranged as follows: Section 2 characterizes workloads in the cloud. Overviews on resource demand prediction methods and resource allocation policies are introduced in section 3 and 4, respectively. Section 5 describes the architecture of a resource management system. Section 6 proposes challenges to build an efficient resource management system. Section 7 concludes the discussion and brings out our future work.

2. Workloads in the Cloud

Before investigating key components in an efficient resource management system, we would like to analyze the workloads in the cloud first. To better serve customers and achieve win-win goals, a comprehensive understanding of workloads is absolutely necessary. Workloads from several organizations have been published for investigation, such as Google workload trace [7, 8, 9, 10, 11]. In this section, we will first classify the common workloads in public clouds. Then workload characterizations in the cloud will be analyzed. Finally, we will introduce workload modeling techniques summarized in [12] to help profile workloads in a real cloud.

2.1. Workload Classification

Workloads in a public cloud are diverse. To facilitate the management and understand the characteristic of each workload, we classify the workloads into different categories. The classification on workloads can be made from various perspectives.

Alexey Tumanov *et al*. [13] classified workloads in the cloud based on the location where they had been processed:

1) **NBODY**: Workloads which are tightly coupled computed-bound can run across many servers in a distributed system, but gain advantage in performance when all of its resources are assigned on a single server.
2) **PI**: Workloads without locality constraints have throughput in linear with resources assigned to them.
3) **LOCAL**: Workloads must be processed on a single server with all required resources on the same server.

Charles Reiss *et al*. [8] categorized workloads based on performance metrics (*e.g*. runtime, throughput):

1) **Long running services**: Services (such as web services) employ a certain amount of resources to provide productions for a long time and provide productions for benefits indefinitely [14].
2) **Short running jobs**: Computational jobs employ computing resources over short periods, where primary metric are measured in seconds (*e.g*. FLOPs, jobs/s, MB/s, I/O rates), as opposed to operations (*e.g*. jobs) per month [15, 16]. For example, MapReduce or Dryad [17] runs many short jobs.
3) **High throughput jobs**: Computational jobs use many computing resources over long periods [18].

Benjamin Farley *et al*. [10] classified workloads based on the bottleneck when running applications (an application can be composed of one or several jobs):

1) **CPU bound jobs**: The limiting factor of solving given computational problem is the speed of the central processor.
2) **Memory bound jobs**: Time to complete a given computational problem is decided primarily by the amount of memory required to hold data.
3) **I/O bound jobs**: Time to accomplish a computation is determined principally by the period spent waiting for input/output operations to be completed.

Other researchers also made reasonable classifications, such as scientific computing, elastic long-running Internet services, data analysis of many scales, software development and test and engineering optimizations, which are classified based on the function of applications. In this paper, we are interested in workload classification based on performance measurements. Using this classification method, we can conveniently observe the relationship between workload characterization and resource consumption, which can help make efficient resource demand prediction and allocation policies. Further,

with the help of these well-designed vital techniques, an efficient resource management system can be established.

2.2. Workload Characterization

According to various analyses of Google trace [19], characterization of workloads in the cloud can be summarized as follows:

1) **Heterogeneous**: As we have already mentioned, workloads in the cloud are diverse. The heterogeneity of workloads is inevitable, in terms of job duration, job shapes and distribution. The duration of a job in the cloud ranges from minutes to months. Generally speaking, production workloads have a higher proportion of long-running jobs. The job shape refers to the resource request, namely the CPU, memory, and bandwidth demand. Jobs in different workloads have different shapes. Due to the high heterogeneity of workloads, there is unlikely any common distribution which will accurately model all the workloads in the cloud.
2) **Dynamic**: Downtime happens constantly. However, more than 99% machines keep working most time and the available rate of machines never fall below 98%. Schedulers in the cloud should deal with several to hundreds jobs per second, which is primarily caused by large amount of short-duration jobs and resubmissions of undone jobs. Along with the raise of resubmissions, crash-loops increase, which often occurs among low priority jobs. High priority jobs crash either, but less. Finally, small jobs and low priority jobs are evicted when resource contention occurs.
3) **Predictable**: The Google workload trace contains resource requests as well as resource usage. The resource requests are unconcernedly made by customers. Interestingly, the resource requests are usually much too higher than actually used. For example, 80% higher in memory and 100% higher in CPU. According to the resource usage records, the overall resource utilization is stable. There are numerous short-duration jobs and a great deal of repeated submissions. So it appears that the resource request can be and should be predicted by history resource usage records.
4) **Constraint**: Some jobs submit resource request with constraints such as minimum memory capacity or storage locality *etc*.

Clearly understanding these characteristics of workloads in the cloud, we can design efficient resource management systems specially tailored for workloads in the cloud.

2.3. Workload Modeling Techniques

Unfortunately, comprehensive characterizations on various workloads cannot be achieved easily. To clearly profile a workload in the cloud, we can use certain workload modeling techniques. More specifically, with measured data (often recorded as a trace, or log of workload-related events), we can model the workloads [20] to evaluate performance, plan capacity, detect anomaly and predict what will happen in the future. The main steps to model a workload can be summarized as follows:

1) **Formulation**: Basic components and characterization levels (*e.g*. level of detail such as job step, interactive command, transactions to a database) of a workload, together with parameters for its description, should be selected. Additionally, criterions for evaluating the model representativeness should be determined either.
2) **Collection of parameters**: The parameters of the workload are collected when the workload is executed.
3) **Statistical analyses of measured data**: When parameters have been collected, we can first filter the outliers and scale the parameters to an adaptive degree. Then we sample from the collected data set for detail analysis since analysis based on the whole data set is a waste of time and computational resources. Finally, we can use

clustering [21], time series [22] and probabilistic graphs [23] to describe the behavior of the workload.

4) **Representativeness**: The performance based criterions, in terms of a vector, is widely applied. The vector comprises of performance indices selected according to the objective of the study, such as response time, throughput, resource utilization *etc*. The accuracy of a workload model is determined by the difference between the model vector and real workload vector.

After above steps, we can have a comprehensive observation on the characterization of workloads in the cloud. Then based on the characterization, resource demand prediction and allocation policies specially tailored for each workload can be well designed.

3. Resource Demand Prediction Overview

In the cloud, pooled computing resources including storage, processing, memory, network bandwidth and virtual machines, are served to multiple customers using a multi-tenant model, with various physical and virtual resources dynamically assigned and reassigned according to customers' demand [24]. Existing cloud service providers ask customers to manually request the resource demand. However, general customers, especially non-IT customers, have no idea to make efficient resource demand applications, which will decrease the resource utilization in the cloud and meanwhile brings in unnecessary monetary cost. As a result, automatic accurate resource demand prediction is meaningful for making a great contribution to resource allocation and job scheduling, which will improve the resource utilization in the cloud and save unnecessary monetary cost for customers.

Resource demand prediction is first proposed to estimate the trend of system resource usage in the immediate future. With the estimation of system resource usage, the resource provider can make timely decisions to scale the cluster and re-schedule jobs. To estimate the future resource usage pattern, the following methods are widely investigated:

1) **Linear time series** was first formulated by Box and Jenkins in 1970s [25], which was analyzed to forecast the future observation based on historical observations of the same measurement. When researchers try to predict the system resource consumption, they consider the resource variations as a linear time series and use various models, such as autoregression model and moving average model, to make the estimation [26, 27, 28]. Such approaches perform well in predicting resource usage for non-computational jobs, for example, web services. As for computational jobs, they would estimate the resource usage with less accuracy.
2) **Wavelet analysis**, or wavelet transform, considers forms of time-frequency representation for continuous-time signals and finds the intuitive insight form the information seemingly complex. Using wavelet analysis, researches consider the resource variations as a superposition of multiple waveforms and predict well with periodical variations [29]. However, it performs considerably terrible when encountering applications with too much randomness. As a result, it is often combined with other techniques, such as support vector machine.
3) **Stochastic process** or random process is a collection of random variations or systems over time. Resource variations are taken as a stochastic process which is based on the assumption that the resource information follows normal distribution [30]. However, for most jobs in practical, the resource usage information may not follow the normal distribution, which limits their application.
4) **Machine learning** concerns the construction and study of systems that can learn from data. Specifically, it focuses on prediction, based on known properties learned from the training data. Resource variations are well learned by machine learning models, such as support vector machine [31], M5P [32], Bagging [33], REPTree [34] and so on. Many researches have demonstrated that machine learning methods

outperform the previously described techniques. There are also recent works [35] trying to introduce more sophisticated learning techniques from other fields such as machine vision [36, 37] and robotics [38]. However, considering the tradeoff of accuracy and computing complexity, appropriate models should be used for different jobs.

Along with the development of resource management system, resource demand prediction is gradually applied in estimating the resource demand of jobs before practically execution. The situation is considerably different with that predicting the resource usage in a server. The latter focuses on analyzing historical resource usage records to predict the immediate future resource usage in the cluster, without considering the individual jobs. While predicting the resource demand for a job to be executed, the information about the job itself is of great importance and different jobs intuitively may need different amount of resources. Current representative cloud service providers, such as Google, ignore the importance of resource demand prediction by submitting the resource demand discretionarily, which directly results in the low resource utilization of t he cluster (Figure 1).

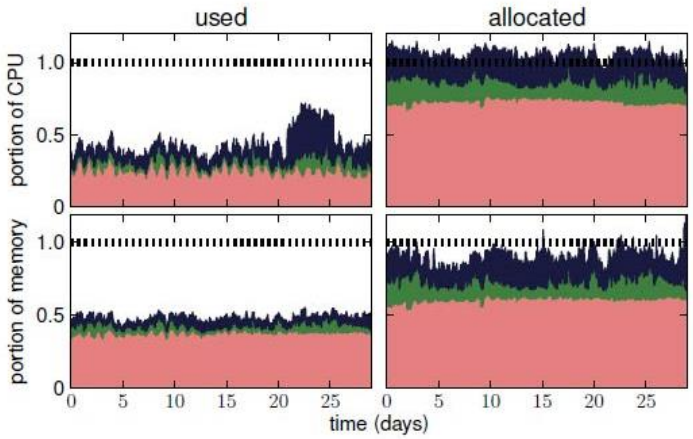

Figure 1. Moving Hourly Average of CPU (Top) and Memory (Bottom) Utilization (Left) and Resource Requests (Right). Stacked Plot by Priority Range, Highest Priorities (Production) on the Bottom (Red), Followed by the Middle Priorities (Green), and Gratis (Blue). The Dashed Line Near the Top of each Plot Indicates the Total Capacity of the Cluster

To predict the resource demand of a computational job, methods introduced above are not entirely appropriate. Time series models cannot predict resource demand for individual computational jobs due to the reason that resource demands of various jobs are not stationary time series. The wavelet analysis cannot handle jobs with abundant randomness. Resource demands required by jobs which do not fit normal distribution cannot be predicted using stochastic process models. Stochastic process algorithms are complex and inefficient. Additionally, under-fitting and over-fitting are often triggered by ill parameters. Machine learning methods are widely used in resource demand prediction, among which linear regression, artificial neural network (ANN) and support vector machine (SVM) are the typical [39].

1) **Linear regression** is one of staple methods in statistics and it finds application in numeric prediction especially where the output (or target class) and the attributes (or features) are numeric. Abhishek Verma predicted the map and reduce slots for a given job with deadline using a linear regression model [40]. Whereas, linear models

are too simple to predict the resource demand which is not linearly related to the given information.

2) **Artificial neural network** is a computational tool originally modeling the interconnection of neurons in the nervous systems of the human brain, which has been applied to problems ranging from recognition to prediction. In the resource demand prediction domain, the characteristics of resource variations are generalized by ANN through proper training. In [41], it is indicated that ANN prediction is more accurate compared with the methods in Network Weather Service (NWS). However, the process of ANN is complex and inefficient. Moreover, the choices of model structures and parameters are lack of standard theory, so that it suffers from over-fitting or under-fitting with ill-chosen parameters.
3) **Support vector machine** is a promising solution to nonlinear regression problems due to its remarkable characteristics such as good generalization performance, the absence of local minima and sparse representation of the solution. SVM is proposed based on the structural risk minimization principle while the traditional regression methods, including neural networks, are based on empirical risk minimization. Prem and Raghavan [31] applied SVM to estimate resource measures and indicated that predictions based on SVM are more accurate.

Additionally, other methods based on source code such as clone techniques [42] are also studied to predict the resource demand for a given job. However, the source code of jobs is usually unavailable in the cloud, which limits the application of source code based resource demand prediction methods.

4. Resource Allocation Overview

To improve the resource management in the cloud, resource allocation is one of the cruxes. Resource allocation is the process of assigning available resources to jobs over the Internet. During this process, various strategies are made to utilize and allocate resources within the limit of cost so as to meet the QoS requirements of jobs. The input parameters to resource allocation strategies and the way of resource allocation vary based on services, infrastructure and the nature of jobs. Consequently, resource allocation strategies can be classified according to execution time, policy, VM, gossip, utility, hardware resource dependency, auction, application and SLA [43].

1) **Execution time**: Resources are reserved based on the estimated execution time of jobs [44]. However, the execution time is hard to estimate and errors are inevitable. If the estimated execution time of a job has a great bias, other jobs will inevitably be affected.
2) **Policy**: Resources are allocated based on the most-fit processor policy 45]. The most-fit policy allocates a job to the cluster, which produces a leftover processor distribution, leading to the maximum number of immediate subsequent job allocations. However, the most-fit policy has high time complexity due to the complex searching process of target cluster.
3) **VM**: Resources are allocated for real time jobs based on the speed and cost of different VMs in IaaS [46]. It allows users to select VMs and reduces the cost for users. In [47], Zhen Kong proposed a resource allocation strategy among selfish VMs in a non-cooperative cloud environment, where VMs care essentially about their own benefits without any consideration for others. They utilized stochastic approximation approach to model and analyze job performance under various virtual resource allocations. However, the method is complex and not practical.
4) **Gossip**: Resources are allocated based on gossip protocol, which means getting information for other local nodes [48, 49, 50]. The gossip protocol used decentralized algorithm to allocate resources and this prototype is not acceptable for heterogeneous cloud environment.

5) **Utility**: Utilities, in terms of response time and profit, are taken into consideration when allocating resources. Zhu *et al*. [51] dynamically allocated the CPU resources to meet QoS objectives by first allocating requests to high priority applications. Minariolli *et al*. [52] proposed the profit based resource allocation for VMs which use live VM migration as the resource allocation mechanism. For multi-tier cloud systems, resource allocation based on response time is proposed by considering CPU, memory and communication resources [53].
6) **Hardware response dependency**: Multiple job optimization schedulers are proposed in [54] to improve the hardware utilization by which resources are allocated based on the categories of jobs (CPU-bound, network I/O-bound, disk I/O bound and memory bound). A CPU usage based resource co-allocation approach was discussed in [55], which mainly focused on CPU and memory resources for co-allocation and did not consider the dynamicity of resource request.
7) **Auction**: A sealed-bid auction mechanism was proposed in [56] to allocate cloud resources. The cloud service provider collected all the users' bids and determined the prices. The resources were finally distributed to the first k_{th} highest bidders under the price of the $(k+1)_{th}$ highest bid. The system simplified the resource allocation problem by transforming it into an ordering problem. However, this mechanism did not ensure profit maximization.
8) **Application**: Virtual infrastructure allocation strategies were designed for workflow based applications where resources were allocated based on the workflow representation of the application [57]. A prototype system was implemented and evaluated for both static and adaptive allocation [58]. This system allocated resources to real time applications and maintained a resource monitor to collect and analyze the real time data. David Irwin *et al*. [59] suggested the integration of high bandwidth radar sensor networks with computing and storage resources in the cloud to design end-to-end data intensive cloud systems. Database replicas allocation strategy was designed in [60].
9) **SLA**: Most resource allocation systems are proposed to fulfill the customers' benefits. Popovici *et al*. [61] took the price and offered load into consideration while designing resource allocation strategies. Lee *et al*. [62] addressed the problem of profit driven service request scheduling in the cloud by considering objectives of both service providers and consumers.

5. Resource Management System

The majority of existing resource management systems focus on the resource allocation (provisioning resources) and job scheduling (scheduling tasks onto proper VM resources), while the quantity and types of resources which should be prepared are always ignored or manually requested by users. However, users who are not domain experts have no idea on the resource demand of their jobs. As a consequence, the resource demands made by users themselves are often freewheeling, which will finally waste system resources or degrade the job performance. Additionally, resources are not encouraged to allocate directly without provisioning. Provisioning resources takes time which will considerably delay the response time in real-time applications. Besides, current resource management systems care little about SLA. Considering those, we proposed a novel resource management system shown in Figure 2.

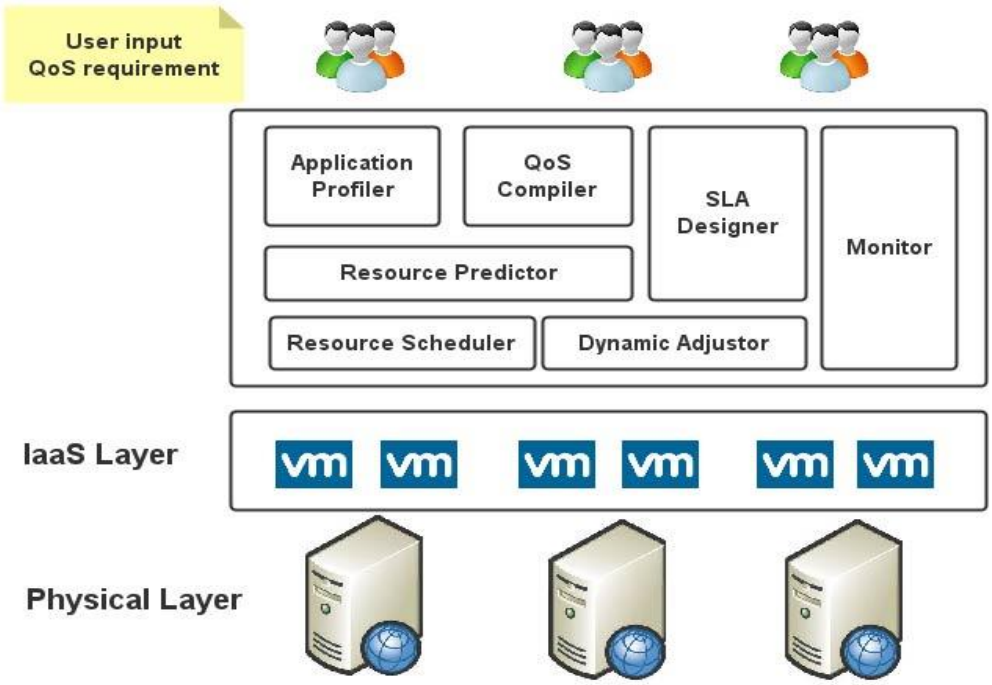

Figure 2. Resource Management System

1) **Application profiler**: For each application, a specific profiler is built to characterize the application. For instance, application configurations and user input parameters which can affect the performance, such as makespan.
2) **QoS compiler**: Users have no ideas on concrete figures to ensure high quality of services. They describe their QoS requirements using natural language, such as high resolution, fluently and so on. Consequently, a QoS compiler which can transfer the natural language into specifications should be designed.
3) **Resource predictor**: Current resource management systems care little on resource demand prediction. Instead, they use machine learning techniques to predict resource demand with the help of application configurations, user input parameters and QoS requirements. Over-provisioning and under-provisioning problems can be effectively decreased by applying resource predictor.
4) **SLA designer**: When resource demand is predicted, we compare it with the available resources captured by the monitor. If there is no confliction, the resource request will be processed and the penalty strategy will be designed. Otherwise, the resource system will bargain for relenting QoS requirements until agreements between system and users are finally made.
5) **Resource scheduler**: When resources are provisioned, a scheduler is needed to schedule jobs, aiming at maximizing the system resource utilization.
6) **Dynamic adjustor**: Practically, the resource predictor cannot make an exact prediction. When jobs are executed, adjustment should be made in case of outrageous resource demand prediction errors.
7) **Monitor**: This module is designed to monitor the resource utilization. The monitoring results can be used to make decisions in SLA designer, resource scheduler and dynamic adjustor modules.

6. Challenges

Designing a resource management system introduced in section 5 will face many challenges.

6.1. Finding the Most Vital Application Parameters

In the application profiler module, we need to understand the most vital parameters of application components and user input file which will directly affect the service performance. Existing solutions can be divided into two categories. The first, represented by Kyriazis [63], asks the application developer to create an XML description of the application components, following a specific template. However, this method does not work well for existing applications lacking such precise descriptions. The second, proposed by Sarkar [42], presents a feedback-based job modeling scheme based on clone detection techniques. In this scheme, resource demand of a newly submitted job is predicted based on historical clones stored in the database. However, this solution requires the knowledge of application source code, which is not available in the cloud.

To the best of our knowledge, there are no efficient solutions to extract key parameters of an application without analyzing the source code. From the application developer's perspective, it is easy to decide the key parameters. However, applications in the cloud are published without detail description from application developers. It remains challenging to mine the key parameters of an application without open source.

6.2. Predicting the Multi-resource Demand

As resource demand prediction is quite an important problem, there are many solutions to be expected [34, 35, 39, 64, 65, 66, 67, 68, 69, 70, 71]. Specifically, workload based resource demand prediction is investigated in [66, 67, 70, 71], which predict the workload variation in the future based on historical resource consumptions. These researches can be used to decide the scale of clusters. Nevertheless, resource demand of individual applications cannot be predicted using such methods. Eun-Kyn Byun investigated resource capacity estimation of workflows using direct acyclic graph (DAG) structure in [68, 69]. The former divided the whole execution time into discrete partition and chose the peak resource demand in each partition as the resource capacity. However, the host requirement was given as prior knowledge, which was impractical. The latter proposed to achieve the maximum utilization through job scheduling policies, in which resource demand of an individual job had been manually determined. Reig *et al*. [34] predicted the CPU and memory usage respectively using machine learning techniques. The idea of their work can be summarized in Figure 3. However, their work is based on the assumption that CPU utilization and makespan have a significant linear correlation. Besides, the resource demand prediction should be made one by one, which means multi-resource demand prediction is not available. George Kousiouris [65] and Johathan Wildstrom [64] predicted the resource demand in another way, which is intuitively shown in Figure 4. Both of the work manually chose some resources first and then predicted the application performance. If the performance metrics can meet the QoS requirements, the manually chosen resources can be accepted for provisioning. Otherwise, resource adjustment would be made to meet QoS requirements of the application. These methods predict the resource demand indirectly, the advantage of which is that multi-resource demand prediction is available. However, the prediction accuracy of such approaches is low because these resource demands are set manually and these approaches focus on finding a feasible solution rather than an optimal solution. As a result, the resource utilization cannot be improved efficiently.

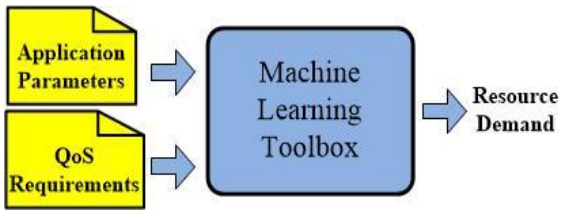

Figure 3. Given Key Parameters which can Characterize an Application and QoS Requirements, One can Predict the Resource Demand using Machine Learning Techniques

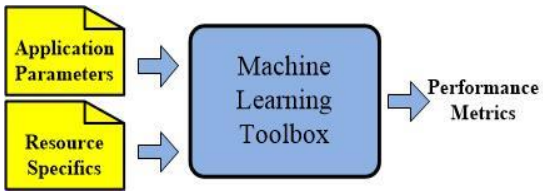

Figure 4. Given Key Parameters which can Characterize an Application and Resource Specifics, One can Predict the Performance Metrics using Machine Learning Techniques

As summarized above, resource demand prediction before execution applications is still a challenging problem, although abundant researches have been done. It needs further investigations on the following unsolved problems:

1) Predict the resource demand for I/O intensive applications. Existing researches focus on prediction resource demand primarily for CPU intensive applications, a few for memory intensive applications. However, they seldom predict the resource demand for I/O intensive applications.
2) Predict multi-resource demands for applications. Existing researches usually predict the CPU and memory resource demands respectively. In fact, there are some correlations between different resources. As a result, predicting multi-resource demands is more meaningful.
3) Improve the prediction accuracy. The most common machine learning techniques used for resource demand prediction is ANN, the accuracy of which is not high. To make a more accurate prediction, other techniques should be investigated.
4) Predict dynamic resource demand. The resource consumption always varies during application execution. If we can predict the dynamic resource demand for applications, especially long running applications, followed by efficient scheduling policies, the system resource utilization will be highly improved.

6.3. Sharing Resources with Other Jobs

Considering a machine holding multiple jobs, the performance of these jobs will be degraded when there are resource contentions. However, not all the resource contentions will lead to SLA violations. Additionally, multiple jobs running on one machine helps improve the system resource utilization. It is challenging to model the performance variation given exhausted resources. Furthermore, it deserves researches to make jobs share the same resources, which can guarantee the SLA of each job and meanwhile maximize the system resource utilization.

6.4. Dynamic Adjustment during Job Execution

Resources provisioned for jobs are not always applicable during job execution. We should monitor the variation of system resource utilization and predict the resource demand periodically from the beginning of job execution. However, frequent predictions bring in extra time cost, which will degrade the performance, especially for real-time applications. Choosing a proper resource demand prediction algorithm and deciding the frequency of predictions are meaningful and challenging.

6.5. Other Challenges

We would like to discuss the resource demand prediction problem in mobile cloud computing environments in the end of this section.

First, most mobile applications are real time. Many researches have been done on cloudlets which aim at reducing the response time. However, it still takes a considerable time to download data from remote datacenter and initialize computing resources. If we can predict users' position and behavior, and provision the resources in advance, response time of computing requests can be further reduced. However, users' position and behavior are hard to learn and the prediction errors may be significant.

Second, mobile devices are energy limited. Energy saving for a mobile device requires more resources, which may increase the response time of other applications due to resource contention. There is a tradeoff between energy saving and resource provisioning. It is challenging to prepare a balanced amount of resources, which can meet the energy saving requirement and avoid bringing in SLA violations of other applications simultaneously.

7. Conclusions

Cloud computing becomes mature gradually over the past decade. Multifarious researches have been done on resource allocation and job scheduling. However, system resource utilization is still low and there is unnecessary monetary cost for users. One of the primarily reasons is that resource demands submitted along with jobs are always requested manually. As a result, an efficient resource management system should be elaborately designed to improve the system resource utilization in the cloud and save monetary cost for user.

In this paper, we discuss the opportunities and challenges of building an efficient resource management system from the perspective of improving resource provisioning policies, which can be composed of resource demand prediction and resource allocation. Before introducing the resource management system, we first introduce the state-of-art of cloud computing. Then we study the workload trace in the cloud and find out the resource utilization is really low. Next, we make an overview on existing resource demand prediction and resource allocation policies. Finally, we introduce a novel resource management system taking into consideration, followed by challenges accomplishing the system.

Acknowledgements

We would like to thank the network information center and supercomputing center of USTC. This research is supported in part by national technical supporting project No. 2012BAH17B03 and the open project of state key laboratory of high performance servers and storage techniques under grant No. 2014HSSA09.

References

[1] J. Dean and S. Ghemawat, "Mapreduce: Simplified Data Processing on Large Clusters". Communications of the ACM, vol. 51, no. 1, **(2008)**, pp. 107-113.
[2] X. Chen, J. Zhang and J. Li. "Resource Management Framework for Collaborative Computing Systems over Multiple Virtual Machines". Service Oriented Computing and Applications. vol. 5, no. (4), **(2011)**, pp. 225-243.
[3] G. Gzajkowski, S. Hahn, G. Skinner, P. Soper and C. Bryce. A Resource Management Interface for Java™ platform. Technical report, **(2003)**.
[4] S. Roy and N. Mukherjee. Multi-agent Framework for Performance-based Resource Management in Computational Grid Environment. Multi-agent Grid Systems, vol. 6, no. 1, **(2010)** pp. 25-53.
[5] L. Wang and M. Zhao, "Application-aware Cross-layer Virtual Machine Resource Management",Proceedings of the 9th International Conference on Autonomic Computing, **(2012)** September 17-21; San Jose, California, USA

[6] M. Schwarzkopf, D. Murray and S. Hand. The Seven Deadly Sins of Cloud Computing Research. Proceedings of the 4th USENIX conference on Hot Topics in Cloud Computing, **(2012)** June 12-13; Boston, MA
[7] A. Mishra, J. Hellerstein, W. Cirne and C. Das. Towards Characterizing Cloud Backend Workloads: Insights from Google Compute Clusters. SIGMETRICS Performance Evaluation Review, vol. 37, no. 4, **(2010)** pp. 34-41.
[8] C. Reiss, A. Tumanov, G. Ganger, R. Katz and M. Kozuch. Heterogeneity and Dynamicity of Clouds at Scale: Google Trace Analysis. Proceedings of 3rd ACM Symposium on Cloud Computing, **(2012)** October , pp. 14-17; San Jose, CA, USA
[9] Z. Liu and S. Cho, Characterizing Machines and Workloads on a Google Custer. Proceedings of 41st International Conference on Parallel Processing Workshops, **(2012)** September 10-13; Pittsburgh, PA, USA
[10] B. Farley, A. Juels, V. Varadarajan, T. Ristenpart, K. Bowers and M. Swift. More for Your Money: Exploiting Performance Heterogeneity in Public Clouds. Proceedings of 3rd ACM Symposium on Cloud Computing, **(2012)** October, pp. 14-17; San Jose, CA, USA
[11] N. Poggi, D. Carrera, R. Gavalda, J. Torres and E. Ayguade. Characterization of Workload and Resource Consumption for an Online Travel and Booking Site. Proceedings of IEEE International Symposium on Workload Characterization, **(2010)** December, pp. 2-4, Atlanta, GA, USA
[12] M. Calzarossa and G. Serazzi. Workload Characterization: A Survey Proceedings of the IEEE, vol. 81, no. 8, **(1993)** , pp. 1136-1150.
[13] A. Tumanov, J. Cipar, G. Ganger and M. Kozuch. Alsched: Algebraic Scheduling of Mixed Workloads in Heterogeneous Clouds. Proceedings of the 3rd ACM Symposium on Cloud Computing, **(2012)**, October, pp. 14-17; San Jose, CA, USA
[14] P. Bodik, A. Fox, M. Franklin, M. Jordan and D. Patterson. Characterizing, Modeling and Generating Workload Spikes for Stateful Services. Proceedings of the 1st AMC Symposium on Cloud Computing, **(2010)** June 10-11; Indianapolis, Indiana, USA
[15] Y. Chen, S. Alspaugh and R. Katz. Design Insights for Mapreduce from Diverse Production Workloads. Technical Report UCB/EECS-2012-17, EECS Department, University of California, Berkeley, **(2012)**.
[16] S. Kavulya, J. Tank, R. Gandhi and P. Narasimhan. An Analysis of Traces from A Production MapReduce Cluster. Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, (2010) May 17-20; Melbourne, Victoria, Australia
[17] M. Isard, M. Budiiu, Y. Yu, A. Birrell and D. Fetterly. Dryad: Distributed Data-parallel Programs from Sequential Building Blocks. Proceedings of the 2nd ACM European Conference on Computer Systems, **(2007)** March 21-23; Amsterdam, the Netherlands
[18] U. Lublin and D. Feitelson. The workload on Parallel Supercomputers: Modeling the Characteristics of Rigid Jobs. Journal of Parallel and Distributed Computing, vol. 63, no. 11, **(2003)**, pp. 1105-1122.
[19] Google Workload Trace. Https://code.google.com/p/googleclusterdata/. **(2011)**.
[20] D. Feitelson. Workload Modeling for Performance Evaluation. Performance Evaluation of Complex Systems: Techniques and Tools, Tutorial Lectures, **(2002)**.
[21] A. Jain, M. Murty and P. Flynn. Data Clustering: A Review. ACM Computer Surveys. vol. 31, no. 3, **(1999)** pp. 264-323.
[22] P. Esling and C. Agon. Time-series Data Mining. ACM Computer Surveys, vol. 45, no. 1, 12:1-12:34, **(2012)**.
[23] D. Ferrari. On the Foundations of Artificial Workload Design. ACM SIGMETRICS. **(1984).**
[24] L. Bill. Cloud Computing: What is Infrastructure as a Service. TechNet Magazine, **(2011)**.
[25] G. Box and G. Jenkins. Time Series Analysis: Forecasting and Control. Holden-Day Incorporated, **(1990)**.
[26] P. Dinda. Design Implementation and Performance of an Extensible Toolkit for Resource Prediction in Distributed Systems. IEEE Transaction on Parallel Distribution Systems, vol. 17, no. 2, **(2006)**, pp. 160-173.
[27] P. Dinda and D. Hallaron. Host Load Prediction using Linear Models. Cluster Computing, vol. 3, no. 4, pp. 265-280, **(2000).**
[28] M. Swany and R. Wolski. Multivariate Resource Performance Forecasting in the Network Weather Services. Proceedings of ACM/IEEE Conference on Supercomputing, **(2002)** November 16-22;Baltimore, Maryland, USA
[29] N. Hiep, Z. Shen and X. Gu. Agile: Elastic Distributed Resource Scaling for Infrastructure as a Service. Proceedings of the 10th International Conference on Autonomic Computing, **(2013)** June 26-28; San Jose, CA, USA
[30] V. Berten and B. Gaujal. Brokering Strategies in Computational Grids using Stochastic Prediction Models. Parallel Computing, vol. 33, no. 4-5 , **(2007)** pp. 238-249
[31] H. Prem and N. Raghavan. A Support Vector Machine based Approach for Forecasting Network Weather Services. Journal of Grid Computing. no. 4, pp. 89-114, **(2006)**
[32] Y. Wang and I. Witten. Induction of Model Trees for Predicting Continuous Classes. Proceedings of the 9th European Conference on Machine Learning, **(1997)** April 23-25; Prague, Czech Republic

[33] I. Witten and E. Frank. Data Mining: Practical Machine Learning Tools and Techniques. Second Edition (Morgan Kaufmann Series in Data Management Systems), Morgan Kaufmann Publishers Incorporated, San Francisco, CA, USA **(2005)**.
[34] G. Reig, J. Alonso and J. Guitart. Prediction of Job Resource Requirements for Deadline Schedulers to Manage High-level SLAs on the Cloud. Proceedings of the 9th IEEE International Symposium on Network Computing and Applications, **(2010)** July 15-17; Cambridge, Massachusetts, USA
[35] L. Xu, J. Cao, Y. Wang, L. Yang and J. Li. Bejo: Behavior based Job Classification for Resource Consumption Prediction in the Cloud. Proceedings of IEEE International Conference on Cloud Computing Technology and Science, **(2014)** December 15-18, Singapore
[36] Y. Wang, J. Feng, Z. Wu, J. Wang and S. Chang. From Low-Cost Depth Sensors to CAD: Cross-Domain 3D Shape Retrieval via Regression Tree Fields. European Conference on Computer Vision, **(2014)** September 6-12; Zurich, Switzerland
[37] Y.Wang, R. Ji and S. Chang. Label Propagation form ImageNet to 3D Point Clouds. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, **(2013)** June 23-28; Portland
[38] Y. Li, Y. Wang, M. Case, S. Chang and P. Allen. Realtime Pose Estimation of Deformable Objects Using a Volumetric Approach. Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, **(2014)** September 14-18, Chicago, IL, USA
[39] A. Bankole and S. Ajila. Predicting Cloud Resource Provisioning using Machine Learning Techniques. Proceedings of 26th IEEE Canadian Conference on Electrical and Computer Engineering, **(2013)** May 5-8; Regina, SK, Canada
[40] A. Verma, L. Cherkasova and R. Campbell. Aria: Automatic Resource Inference and Allocation for Mapreduce Environments. Proceedings of the 8th ACM International Conference on Autonomic Computing, (201) June 14-18; Karlsruhe, Germany.
[41] A. Eswaradass, X. Sun and M. Wu. A Neural Network based Predictive Mechanism for Available Bandwidth. Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium, **(2005),** April 4-8; Denver, CO, USA
[42] M. Sarkar, T. Mondal, S. Roy and N. Mukherjee. Resource Requirement Prediction using Clone Detection Techniques. Future Generation Computing System,vol. 29, no. 4, **(2013)**, pp. 936-952,
[43] V. Vinothina, R. Sridaran and P. Ganapathi, " A Survey on Resource Allocation Strategies in Cloud Computing",International Journal of Advanced Computer Science and Applications, vol. 3, no. 6, **(2012)**, pp. 97-104,
[44] J. Li, M. Qiu, J. Niu, Y. Chen and M. Zong. Adaptive Resource Allocation for Preemptable Jobs in Cloud Systems. Proceedings of the 10th International Conference on Intelligent Systems Design and Applications, **(2010)** November 29 – December 1; Cairo, Egypt
[45] K. Huang and K. Lai, Processor Allocation Policies for Reducing Resource Fragmentation in Multi-cluster Grid and Cloud Environments. Proceedings of International Computer Symposium, **(2010)** December 16-18; Tainan, Taiwan
[46] K. Kumar, J. Feng, Y. Nimmagadda and Y. Lu. Resource Allocation for Real-time Tasks using Cloud Computing. Proceedings of the 20th International Conference on Computer Communications and Networks, **(2011)** July 31- August 4; Maui, Hawaii, USA
[47] Z. Kong, C. Xu and M. Guo. Mechanism Design for Stochastic Virtual Resource Allocation in Non-cooperative Cloud Systems. Proceedings of International Conference on Cloud Computing, **(2011)** July 4-9; Washington DC, USA
[48] D. Niyato, K. Zhu and P. Wang. Cooperative Virtual Machine Management for Multi-organization Cloud Computing Environment. Proceedings of the 5th International ICST Conference on Performance Evaluation Methodologies and Tools, **(2011)** May 16-20; Paris, France
[49] F. Wuhib and R. Stadler. Distributed Monitoring and Resource Management for Large Cloud Environments. Proceedings of IFIP/IEEE International Symposiium on Integrated Network Management, **(2011)** May 23-27; Dublin, Ireland
[50] R. Yanggratoke, F. Wuhib and R. Stadler. Gossip-based Resource Allocation for Green Computing in Large Clouds. Proceedings of the 7th International Conference on Network and Service Management, **(2011)** October 24-28, Paris, France
[51] X. Zhu, D. Young, B. Watson, Z. Wang, J. Rolia, S. Singhal, B. McKee, C. Hyser, D. Gmach, R. Gardner, T. Christian and L. Cherkasova. 1000 Islands: Integrated Capacity and Workload Management for the Next Generation Data Center. Proceedings of the 5th IEEE International Conference on Autonomic Computing, **(2008)** June 2-6; Chicago, Illinois, USA
[52] D. Minarolli and B. Freisleben. Utility-based Resource Allcoation for Virtual Machines in Cloud Computing. Proceedings of IEEE Symposium on Computers and Communications, **(2011)** June 28- July 1; Kerkyra, Corfu, Greece
[53] H. Goudarzi and M. Pedram. Multi-dimensional SLA-based Resource Allocation for Multi-tier Cloud Computing Systems. Proceedings of the 4th IEEE International Conference on Cloud Computing, (2011) July 4-9, Washington DC, USA
[54] W. Hu, C. Tian, X. Liu, H. Qi, L. Zha, H. Liao, Y. Zhang and J. Zhang. Multiple-job Optimization in MapReduce for Heterogeneous Workloads. Proceedings of the 6th International Conference on Semantics Knowledge and Grid, **(2010)** November 1-3; Beijing, China

[55] P. Endo, A. Palhares, N. Pereira, G. Goncalves, D. Sadok, J. Kelner, B. Melander and J. Mangs. Resource Allocation for Distributed Cloud: Concepts and Research Challenges. IEEE Network, vol. 25, no. 4, **(2011)**, pp. 42-46,
[56] W. Lin, G. Lin and H. Wei. Dynamic Auction Mechanism for Cloud Resource Allocation. Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, **(2010)** May 17-20; Melbourne, Australia
[57] T. Huu and J. Montagnat. Virtual Resources Allocation for Workflow-based Applications Distribution on a Cloud Infrastructure. Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, (2010) May 17-20; Melbourne, Australia
[58] W. Iqbal, M. Dailey, I. Ali, P. Janecek and D. Carrera. Adaptive Resource Allocation for Back-end Mashup Applications on a Heterogeneous Private Cloud. Proceedings of International Conference on Electrical Engineering/Electronics Computer Telecommunications and Information Technology. (2010) May; Chiang Mai, Thailand
[59] D. Irwin, P. Shenoy, E. Cecchet and M. Zink. Resource Management in Data-intensive Clouds: Opportunities and Challenges. Proceedings of the 17th IEEE Workshop on Local and Metropolitan Area Networks, (2010) May; Long Branch, New Jersey
[60] P. Xiong, Y. Chi, S. Zhu, H. Moon, C. Pu and H. Hacigumus. Intelligent Management of Virtualized Resources for Databased Systems in Cloud Environment. Proceedings of IEEE 27th International Conference on Data Engineering, (2011) April 11-16; Hannover, Germany
[61] F. Popovici and J. Wilkes. Profitable Services in an Uncertain World. Proceedings of ACM/IEEE Supercomputing, (2005) June 20-22, Cambridge, Massachusetts, USA
[62] Y. Lee, C. Wang, A. Zomaya and B. Zhou. Profit-driven Service Request Scheduling in Clouds. Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, **(2010)** May 17-20; Melbourne, Australia
[63] D. Kyriazis, R. Einhorn, L. Furst, M. Braitmaier, D. Lamp, K. Konstanteli, G. Kousiouris, A. Menychtas, E. Oliveros, N. Loughran and B. Nasser. A Methodology for Engineering Real-time Interactive Multimedia Applications on Service Oriented Infrastructures. In IADIS Applied Computing, **(2010)**.
[64] J. Wildstrom, P. Stone and E. Witchel. Carve: A Cognitive Agent for Resource Value Estimation. Proceedings of the 5th IEEE International Conference on Autonomic Computing, (2008) June 2-6; Chicago, Illinois, USA
[65] G. Kousiouris, A. Menychtas, D. Kyriazis, S. Gogouvitis and T. Varvarigou. Dynamic, Behavioral-based Estimation of Resource Provisioning based on High-level Application Terms in Cloud Platforms. Future Generation Computer Systems, vol. 32, no. 0, **(2014)**, pp. 27-40,
[66] Z. Gong, X. Gu and J. Wilkes. Press: Predictive Elastic Resource Scaling for Cloud Systems. Proceedings of International Conference on Network and Service Management, (2010) October 25-29; Niagara Falls, Canada
[67] K. Sembiring and A. Beyer. Dynamic Resource Allocation for Cloud-based Media Processing. Proceedings of the 23rd ACM Workshop on Network and Operating System Support for Digital Audio and Video, **(2013)** February 27; Oslo, Norway
[68] E. Byun, Y. Kee, J. Kim, E. Deelman and S. Maeng. Bts: Resource Capacity Estimate for Time-Targeted Science Workflows. Journal of Parallel Distributing Computing, vol. 71, no. 6, **(2011)**, pp. 848-862.
[69] E. Byn, Y. Kee, J. Kim and S. Maeng. Cost Optimized Provisioning of Elastic Resources for Application Workflows. Future Generation Computer Systems, vol. 27, no. 8, pp. 1011-1026,
[70] S. Islam, J. Keung, K. Lee and A. Liu. Empirical Prediction Models for Adaptive Resource Provisioning in the Cloud. Future Generation Computer Systems, vol. 28, no. 1, **(2011)**, pp. 155-162,
[71] N. Roy, A. Dubey and A. Gokhale. Efficient Autoscaling in the Cloud using Predictive Models for Workload Forecasting. Proceedings of International Conference on Cloud Computing, **(2011)**, July 4-9; Washington DC, USA.

Authors

Lin Xu, She received the B.E. degree of Computer Science and Technology from the University of Science and Technology of China (USTC) in 2010. Now, she is a Ph.D. student at the School of Computer Science and Technology in USTC. Her research interests include data analytics, cloud computing, high performance computing and data mining.

Jing Li, He received his B.E. degree of Computer Science from the University of Science and Technology of China (USTC) in 1987, and Ph.D. in Computer Science from USTC in 1993. Now he is a Professor at the School of Computer Science and Technology in USTC. His research interests include distributed systems, could computing, big data processing and mobile cloud computing.