A heuristic task deployment approach for load balancing

Gaochao Xu[1], Yunmeng Dong[1], Xiaodong Fu[1], Yan Ding[1], Peng Liu[1], Jia Zhao[2*]

[1] *College of Computer Science and Technology, Jilin University, Changchun 130012, China*

[2] *College of Computer Science and Engineering, Changchun University of Technology, Changchun 130012, China*

Received 1 February 2014, www.tsi.lv

Abstract

The load balancing strategy, which is based on the mission deployment, has become a hot topic of green cloud data centre. For the question that currently the overloaded physical hosts in the cloud data centre causes the load imbalance of the whole cloud data centre, the proposed makes an intensive study which is about the select location question of the deployment tasks on the physical host and then this proposed a new heuristic method which is called LBC. Its main idea consists of two parts: First, based on the function, which denotes the performance fitness of physical hosts, it conducts a constraint limit to all physical hosts in cloud data centre. So a task deployment strategy with global search capability is achieved. Secondly, using clustering methods can further optimize and improve the final clustering results. Thus, the whole way achieves the long-term load balancing of the cloud data centre. The results show that compared with the conventional approach, LBC significantly reduces the number of failure of the deployment tasks, improves the throughput rate of the cloud data centre, optimizes the performance of external services of the data centre, and performs well in terms of load balancing. Besides, it makes the operation of cloud data centres be more green and efficient.

Keywords: load balancing strategy, cloud data centre, task deployment, LBC, clustering

1 Introduction

Cloud computing [1, 2] is the focus of research topic currently which is the most promising and valuable research direction following the utility computing, grid computing and distributed computing. Cloud computing provides users with infrastructure, platform and software services according to user's needs through the Internet. Infrastructure as a Service (IaaS) is the foundation of cloud computing, whose key is to make the data centre cloud computing resources to be a resource pool through virtualization technology. Besides, it allocates according to the task specifications and resource requests, which are submitted by users. In addition, it provides elastic physical or virtual computing, storage and network resources. A large number of physical hosts, which are deployed by the cloud data centre, provide services for users. However, each physical host's resource remaining amount is changing all the time. Therefore, it cannot guarantee to place every task on the physical host, which has the largest remaining amount of resources.

Currently, load balancing is the hot issue in the domain of the cloud data centre study. In order to further optimizing load balancing in the cloud data centre, this proposed presents a heuristic idea [3], load balancing strategy, which aims at finding the physical host whose deployment performance is optimal. And details are as follows: First, giving a constraint value which is based on the resource amount of requested tasks. Then cluster the physical hosts, which are greater than the constraint value in the cloud data centre. Secondly, forming a set of physical hosts by clustering whose similarities are within a certain threshold. The collection of physical hosts got after clustering is the physical hosts collection having the optimal deployment performance, which we want to find. Finally, place the tasks, which need to be processed into the physical hosts which are in the collection to conduct deployment. Clustering the physical hosts in the data centre exactly is the process of finding the physical hosts, which have optimal deployment performance. Therefore, through our deployment task strategy, it cannot only achieve load balancing in the cloud data centre, but also provide efficient external service performance for users.

This paper aims to achieve long-term load balancing in the cloud data centre and provide users with efficient external service performance. And achieving long-term load balancing in the cloud data centre must by means of deploying the task request to the resource pool in the cloud data centre efficiently and rationally. Therefore, it can achieve load balancing in the cloud data centre and improve the efficiency of the cloud data centre. Furthermore, it can show the excellent external service performance of cloud data centre to users. The load balancing strategy proposed cannot only find the physical hosts, which have optimal deployment performance efficiently, but achieve long-term load balancing in the cloud data centre.

[*] ***Corresponding author*** e-mail: zhaiyj049@sina.com

Other parts of this paper are organized as follows: In the second part, we briefly describe the current work that is related to the method, which can achieve the load balancing of cloud data centre. In the third section, we first point out the premise, which proposed in our questions, and then introduce the design and implementation process of our algorithm in details. In the fourth section, we will give the experiments and results, and prove that the algorithm we proposed has high efficiency. The fifth part, we summarize the full paper and future work is put forward.

2 Related works

Load balancing has been a hot research topic of cloud data centre [4] and its goal is to ensure that every computing resource can process tasks efficiently and fast, improve the utilization of resources ultimately. The question is present in a cloud computing environment. When there are some task requests in the cloud data centre, these tasks request will be deployed to the optimal physical host of the cloud data centre, so that the computing performance of cloud computing centres can achieve optimal, while cloud data centres can achieve the load balancing of entire network. Researchers have proposed a series of static, dynamic and mixed scheduling policies. In addition, there are also some studies using live migration technology of virtual machine to meet the cloud data centres' requested tasks which include performance requirements and load limitation. In fact, most problems are just deploying the requested tasks to the cloud data centre's physical hosts.

Existing load balancing strategies are generally divided into two categories: static load balancing and dynamic load balancing. Static load balancing scheduling algorithm [5-8] are commonly used round robin, weighted round robin, least connection method, weighted least connection method and so on. These static algorithms only use some static information, which cannot solve dynamic load changes among servers in cluster effectively and their adaptive ability is poor. Currently, some of the most open-source IaaS platform most use static algorithm to conduct resource scheduling. For example, Eucalyptus [9] platform uses round robin to assign virtual machines to different physical hosts in sequence to achieve load balancing. In Literature [10], Wei Q et al. used the weighted minimum link algorithm, which means that different weights indicate the performance of the physical host. Then, the virtual machine will be allocated to the physical host, which has the smallest ratio of the number and weight. The advantage of static scheduling algorithm is that it is simple to do. But facing the large-scale cloud data centre whose heterogeneous resources are strong and users' consistent demand, load balancing effect is not ideal.

Dynamic load balancing [11-13] is a NP-complete problem which is a classic combinatorial optimization problem. It is mainly used in the field of distributed parallel computing, and its main objective is how to distribute the load more rationally among multiple computers to avoid some phenomenon of calculation node overload and light load. Thus, overall system performance can be improved. Additional communication overhead produced in the process of DLB will reduce the dynamic load balancing system performance. And with the increase of network latency among each node, the influence of the restricting DLB performance of additional communication overhead will further increase. Therefore, how to reduce communication overhead furthest among each node in the process of DLB becomes an important problem, which will influence the performances of DLB. Now aiming at the problem of reducing additional communication overhead in the process of DLB, the solution is mainly using greedy algorithm to process. The LRS algorithm, which is put forward in Literature 9 using the light load preferentially received allocation pattern.

In Literature [14], Lau et al. integrated two strategies which are heavy load priority and light load priority. They put forward an adaptive load distribution algorithm, which helps reduce the load balancing communication overhead effectively. Using the greedy algorithm can solve the problem of load distribution. However, several algorithms above cannot meet greedy choice performance and sub-optimal structural property at the same time. Therefore, load distribution program was often local optima. And the effect of solving the problem of load distribution under certain special circumstances is not ideal. Cloud data centre cannot reach the entire network load balancing. Virtual machine migration placement strategy is the most widely used strategy to achieve cloud computing [15, 16] data centre load balancing currently. VMware load balancing solution is DRS (Distributed Resource Scheduling) [17]. When DRS select the physical host for the virtual machine, it will check the load status of each physical host and choose the placement method to reduce the overall load imbalance. And in the process of running a virtual machine, DRS will continue to monitor the load status of the cluster and use VMware VMotion technology to perform live migration of virtual machines among different physical servers. Thus, it can ensure load balancing and efficient use of physical resources of the entire cluster.

3 LBC algorithm design

In IaaS cloud data centre, when users have requests, the system will deploy the task request to the physical hosts, which are in the resource pool of a cloud data centre. In general, cloud data centres will select physical hosts randomly to deploy. When the requested resources are greater than the physical hosts' remaining resources, physical hosts cannot deploy the task. When the requested resources are in proximity to the physical hosts' remaining resources, it will cause overload of physical hosts. Thus, it will cause load imbalance in the

cloud data centre and result in decreased efficiency and increased energy consumption. Obviously, with regard to the cloud data centre, different deployment task strategies will cause different load allocation in entire system. There is no doubt that the optimal deployment task strategy can make the entire cloud computing system produce the effect of load balancing. Therefore, it is necessary to design and implement an efficient and load-balancing deployment task strategy in the cloud data centre.

3.1 IMPLEMENTATION OF LBC ALGORITHM

The implementation of the LBC algorithm:

Step 1:

Assuming the number of physical hosts in the data centre is n. We need to do a constraint to all physical hosts in the data centre. And in order to meet the physical hosts' performance constraints. We treat the physical hosts' remaining amount of resources L_i as a metric. It is defined as follows:

$$L_i = \alpha L_c + \beta L_{mem}, \tag{1}$$

$$\alpha + \beta = 1. \tag{2}$$

L_i shows the remaining computing resources of physical host node *I*, which mean the usage of CPU and memory usage. L_c is the remaining of CPU. L_{mem} is the remaining of memory. α is the weight of CPU. β is the weight of memory. The value of α and β are obtained through BP neural network study. According to the fitness function (1) and (2) of the physical hosts' performance, it generally obtains the monitoring data of the physical hosts' various performance in the entire data centre through SNMP (simple network management protocol), including CPU and memory data. The remaining resources of n physical hosts in the cloud data centre can be calculated. The constraint value is defined as: The total amount of resources received of task request collection within time Δt, namely:

$$L_{req} = \sum_{i=1}^{n} L_{tk}^{i}. \tag{3}$$

In this equation, L_{req} is the total amount of resources of task request collection, L_{tk}^{i} is the resource of task *i* in the task request collection. There defines an empty set *S* = {}.

According to the equation (3), L_{req} can be calculated. When there is an inequation, $L_i > L_{req}$, a host *i* will be put into set *S*, otherwise, we will continue to find. The set *S*, which is got after comparing n physical hosts with the constrained value is S = { s_1 , s_2 , s_3 …., s_m }, m ≤ n.

Step 2:

We can get the performance values of each physical host based on the fitness function of physical hosts' performance. By restricting the constrained value, we put the physical hosts in the data centre whose performances are relatively good into the set S. Regard the remaining of the physical host's CPU as the physical host's property. Suppose S = { s_1 , s_2 , s_3 , …, s_m } as the set which contains m physical hosts. We arrange CPU remaining of physical hosts in the set S in descending order. Large CPU remaining of physical hosts is arranged in the front. Supposing that s_j is the physical host, which has the largest CPU remaining, we regard s_j as the class-centre. The equations that calculate the similarity are as follows:

$$d(\mathrm{s}_i, \mathrm{s}_j) = \sqrt{\sum_{k=1}^{d} (\mathrm{s}_i^k - \mathrm{s}_j^k)^2}, \tag{4}$$

$$s(\mathrm{s}_i, \mathrm{s}_j) = \frac{1}{d(\mathrm{s}_i, \mathrm{s}_j)}, \tag{5}$$

s_j^1 is a property of the physical host j. It can represent the physical host's CPU remaining. So according to the equations (4) and (5), the similarity of the physical host *i* and the physical host j can be calculated. $s(\mathrm{s}_i, \mathrm{s}_j)$ is the similarity of s_i and s_j.

$$s(\mathrm{s}_i, \mathrm{s}_j) = \frac{1}{\sqrt{(\mathrm{L}_{ci}^1 - \mathrm{L}_{cj}^1)^2}}. \tag{6}$$

Step 3:

Regarding s_j as the class-centre, giving a threshold value $U_{threshold}^{Similarity}$, according to the similarity, we calculate the similarity of s_j and each element of the set S. If the similarity is greater than the threshold value $U_{threshold}^{Similarity}$, we will add this element into the new set S' and not put the class-centre into S'. Then the set S selects class-centre according to the remaining of physical host CPU in descending order and calculate apart the similarity with the elements of S'. Next put the threshold which is greater than $U_{threshold}^{Similarity}$ into the set S'. When the elements of the set S' does not change, the iteration ends. The final clustering result is the set S'. $S' = \{\mathrm{s}_1', \mathrm{s}_2' \ldots \mathrm{s}_q'\}$, q ≤ m ≤ n.

Step 4:

It puts the task request received from the data centre into the collection $S^{'}$ of physical hosts, then physical hosts in the collection $S^{'}$ process the task set in a collection of physical hosts to process the requested task collection. After processing, the results will be returned to users. From the physical hosts in the collection $S^{'}$ starting processing the task until the processing is completed, the period of time is recorded as Δt. The task requirements that the data centre receives within time Δt will be the next task to be processed.

Step 5: Repeat the above process.

3.2 MODEL OF LBC ALGORITHM

Overall, the target of LBC algorithm is also in line with the idea of heuristic algorithms. It mainly because the solution that heuristic idea finds every time is not always the optimal solution. But by constant finding and revising, it can get closer to the optimal solution until infinitely close to the optimal solution. And his process meets the goal of algorithm. From the view of the goal of algorithm, we are committed to achieve the load balancing of data centre. An iteration of the algorithm can achieve load balancing. So it needs repeated iterative algorithm, and find the optimal physical host after each iterative. Therefore, users' requested tasks can be disposed by the optimal physical host in the data centre. In this way, after repeated iterative algorithm, each set of physical hosts we found can get close to the best performance. Data centres can quickly process tasks requested by the user. Thus, the data centre tends to be load-balancing and finally it achieves load balancing. This is a long process, and LBC algorithm ensures that the process can get good results within a reasonable time.

4 Evaluation

This paper uses CloudSim simulator to simulate a dynamic cloud data centre. It supports for dynamic creation of different types of entities at run-time and it can add and delete data centre. In CloudSim platform, it creates a resource pool with 100 physical hosts. These hosts have different computing resources and 50 different tasks request resources. They need different CPU and memory of physical host. LBC model we proposed calls and gets resource information and status of physical hosts in the cloud resource pool regularly. In this section, through load balancing degree, make-span, and external service performance, we compared the LBC deployment strategies we proposed with random deployment strategies and do some experiments. The results shown below:

In this scenario of experiments, we compared the Make-Span of LBC and random methods. Make-Span is the completion time of computing tasks. The results shown in Figure 1, it can be seen from the figure that the increase of two deployment methods with the increasing number of requests tasks and the Make-Span of the collection of requested tasks requests will also increase. As can be seen from Figure 1, comparing with random deployment, the LBC algorithm we proposed has a smaller Make-Span under the same condition. This experiment illustrates that the LBC algorithm not only has good load balancing effect, but has relatively good Make-Span.

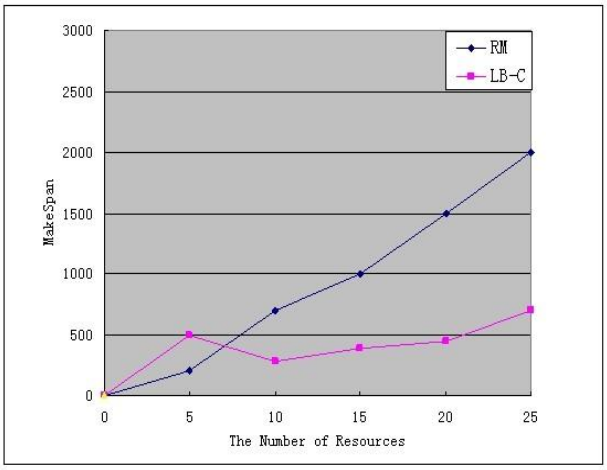


FIGURE 1 Comparison of Makespan

In this set of experiments, we compare the changes of load balancing in the cloud data centre, which influenced by LBC method and stochastic methods over time. As can be seen from Figure 2, with the time increasing during deployments, the load balance degree of stochastic methods and LBC method decrease gradually. The load balance degree of traditional random deployment strategies is always greater than LBC method. It is because the LBC method can quickly find the optimal physical host based on the required CPU resource amount of a requested task. To a certain extent, it ensures the CPU utilization of physical host is much good. From the experimental results, the LBC method we proposed has better load balancing effect. Thus, the resource utilization of the cloud data centre is more effectively improved. And it indirectly saves the power consumption for the cloud data centre.

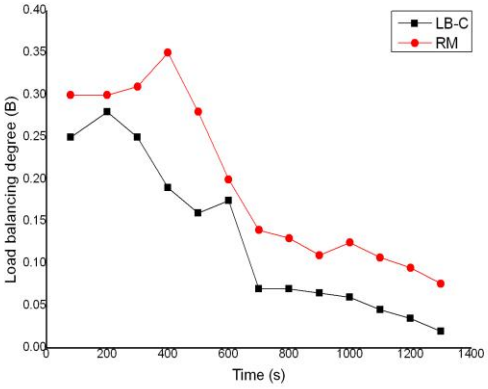


FIGURE 2 Comparison of load balancing degree

The third set of experiments verifies the LBC method from the eternal service performance of the data centre after the deployment of two methods. They selected the throughput as the evaluation criteria of the external service performance of the data centre, because the throughput is usually the overall evaluation of a system and the ability of its assembly units requested ability to process transmission data. Experimental results shown in Figure 3, it can be seen from the figure that using two different deployment methods, external service performances of the data centre are different.

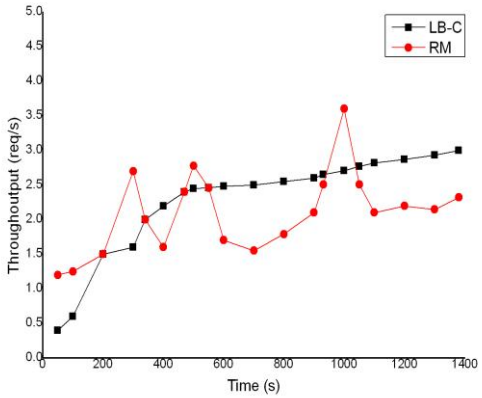


FIGURE 3 Comparison of external service performance

After the random deployment, computing performance of the data centre is much good. With the increase of response time, eternal service performance has a waved trend, and no stability. But using the LBC method to deploy tasks, the initial external service performance of the data centre is not as good as that of the data centre by using the random methods. With the increase of response time, the external service performance of the data centre gradually stabilizes. By comparing the performance of external service performance of the data centre, it can be concluded that using LBC method can be more stable and efficient than random deployment.

5 Conclusion and future work

Based on summarizing the related work, this task proposes a new load balancing strategy, LBC, which is based on task deployment and gives the main idea including process implementation and evaluation. It uses heuristic ideas, which are based on clustering. In these heuristic ideas, first it calculates the total amount of resources of requested tasks according to the fitness function of physical hosts' performance. Then, it compares the amount of tasks with the remaining resources of the physical host in the cloud data centre. And it makes the physical host which is greater than the total amount of resources to be the clustering object. Therefore, LBC has better search capability and adaptive capacity at the beginning. To assess LBC method, there are several experiments done at CloudSim platform. Through four experiments, the result shows that LBC method can deploys the real-time task requests to the resource pool of cloud data centre faster and efficiently. LBC achieves the long-time load balancing and high-efficiency computing capacity of the cloud data centre. It minimizes the number of failures of deployment tasks in the cloud data centre. And to some extent, it improves the throughput of the cloud data centre. In those LBC methods proposed, there are some open questions, which need further research, and some experimental questions, which need a lot of experiments to get a much good solution. The value of CPU weight, α , and memory weight, β , is an empirical question. It needs several experiments to obtain the optimal values so that we can get the equation: $\alpha+\beta=1$.Therefore, LBC method can be more efficient and feasible. In this context, all the parameters are set to the appropriate value. To further improve the performance of LBC, We plan to study the robustness of LB - C in the next step. LBC method should be able to choose one to more physical hosts to deploy tasks in the collection after clustering. Thus, the cloud data centre and users can get the maximum benefit.

References

[1] Erl T, Puttini R, Mahmood Z 2013 *Cloud Computing: Concepts, Technology & Architecture* Pearson Education
[2] Garg S K, Versteeg S, Buyya R 2013 *Future Generation Computer Systems* **29**(4) 1012-23
[3] Yassa S, Chelouah R, Kadima H, Granado B 2013 Multi-Objective Approach for Energy-Aware Workflow Scheduling in Cloud Computing Environments *The Scientific World Journal* **2013** Article ID 350934 13 p
[4] Liu L, Wang H, Liu X, et al 2009 GreenCloud: a new architecture for green data center *Proceedings of the 6th international conference industry session on Autonomic computing and communications industry session* ACM 29-38
[5] Wei Qun, Xu Guangli, Li Yuling 2011 Research on duster and load balance based on linux virtual server *Information Computing and Applications* Berlin: Springer Heidelberg 169-176
[6] Di Yuan, Wang Shuai, Sun Xinya 2013 A dynamic load balancing model based on negative feedback and exponential smoothing estimation http://www.thinkmind.org/index.php?view=article&articleid=icas_2012_2_10_20043 *(24 Feb 2013)*
[7] Chen Wei, Zhang Yufang, Xiong Zhongyang 2010 Research and realization of the load balancing algorithm for heterogeneous cluster with dynamic feedback *Journal of Chongqing University* **33**(2) 2-14
[8] Song S, Lv T, Chen X 2014 A Static Load Balancing algorithm for Future Internet *TELKOMNIKA Indonesian Journal of Electrical Engineering* **12**(6)
[9] Nurmi D, Wolski R, Grzegorczyk C, et al 2009 The eucalyptus open-source cloud-computing system *Cluster Computing and the Grid, 2009 CCGRID'09 9th IEEE/ACM International Symposium on. IEEE* 124-31
[10] Wei Qun, Xu Guangli, Li Yuling 2011 Research on duster and load balance based on linux virtual server *Information Computing and Applications* Berlin: Springer Heidelberg 169-76
[11] Willebeek-LeMair M H, Reeves A P 1993 Strategies for dynamic load balancing on highly parallel computers *IEEE Transactions Parallel and Distributed* Systems **4**(9) 979-93

[12] You T, Li W, Fang Z, et al 2014 Performance Evaluation of Dynamic Load Balancing Algorithms *TELKOMNIKA Indonesian Journal of Electrical Engineering* **12**(4)
[13] Bahi J M, Contassot-Vivier S, Couturier R 2005 Dynamic load balancing and efficient load estimators for asynchronous iterative algorithms *IEEE Transactions on Parallel and Distributed Systems*, **16** (4) 289-99
[14] Lau S M, Lu Q, Leung K S 2006 Adaptive load distribution algorithms for heterogeneous distributed systems with multiple task classes *IEEE Transactions Parallel and Distributed Computing* **66**(2) 163-80
[15] Armbrust M, Fox A, Griffith R, Joseph A D, Katz R, Konwinski A, Lee G, Patterson D, Rabkin A, Stoica I, Zahariaa A 2010 A view of cloud computing *Communications of the ACM* **53**(4) 50-58
[16] Moreno-Vozmediano R, Montero R S, Llorente I M 2013 Key Challenges in Cloud Computing: Enabling the Future Internet of Services *Internet Computing IEEE* **17**(4) 18-25
[17] Muir S, Tavilla R, Verghese B 2012 *Vmware Technica Journal* [*EB/OL*] **1**(1)

Authors

Gaochao Xu, Wuhan, born in 1966

Current position, grades: Changchun, Professor *and* PhD supervisor of College of Computer Science and Technology, Jilin University, China.
University studies: BS, MS and PhD on College of Computer science and Technology of Jilin University in 1988, 1991 and 1995
Scientific interest: Cloud Computing, Mobile Cloud Computing
Publications: SCI 10
Experience: more than 10 national, provincial and ministerial level research projects of China Gaochao Xu was.
Research interests: distributed system, grid computing, cloud computing, Internet of things, information security, software testing and software reliability assessment, etc.

Yunmeng Dong, born in 1989, in Yushu of Jilin province of China

Current position, grades: Changchun, Master
University studies: bachelor degree in computer science at Changchun University of Technology (2012), a postgraduate candidate of the college of computer science and technology of Jilin University
Scientific interest: Virtualization, Cloud Computing, Mobile Cloud Computing
Publications: EI 1
Research interests: distributed system, cloud computing and virtualization technology

Xiaodong Fu, ChangChun

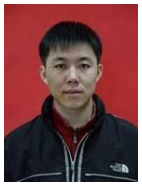
Current position, grades: Senior engineer in the College of Computer Science and Technology， Jilin University of China.
University studies: BSc degree from Jilin University.
Research interests: Distributed System, Grid Computing, Cloud Computing, Internet Things
Publications: 14 research articles

Yan Ding, born in 1988, in Yichun of Heilongjiang province of China

Current position, grades: Changchun, Master, a postgraduate candidate of the college of computer science and technology of Jilin University
University studies: bachelor degree at Jilin University in 2011
Scientific interest: Virtualization, Cloud Computing, Mobile Cloud Computing
Publications: SCI 1
Research interests: distributed system, cloud computing and virtualization technology

Peng Liu, born in 1990, in Jixi of Heilongjiang province of China

Current position, grades: Changchun, Master, is a postgraduate candidate of the college of computer science and technology of Jilin University
University studies: bachelor degree at Daqing Normal University in 2013
Research interest: Virtualization, distributed system Cloud Computing, Mobile Cloud Computing, SDN



Jia Zhao, born in 1982, in Changchun of Jilin province of China

Current position, grades: Changchun, Doctor, PhD candidate of the college of computer science and technology of Jilin University
University studies:
Scientific interest: Virtualization, Cloud Computing, Mobile Cloud Computing
Publications: SCI 4
Research interests Virtualization, Cloud Computing, Mobile Cloud Computing include distributed system, cloud computing, network technology
Experience: participated in several projects