Fog Computing Dynamic Load Balancing Mechanism Based on Graph Repartitioning

SONG Ningning[1*], GONG Chao[2], AN Xingshuo[2], ZHAN Qiang[2]

[1] School of Mechanical Engineering, University of Science and Technology Beijing, Beijing China
[2] School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing China

Abstract: Because of cloud computing's high degree of polymerization calculation mode, it can't give full play to the resources of the edge device such as computing, storage, etc. Fog computing can improve the resource utilization efficiency of the edge device, and solve the problem about service computing of the delay-sensitive applications. This paper researches on the framework of the fog computing, and adopts Cloud Atomization Technology to turn physical nodes in different levels into virtual machine nodes. On this basis, this paper uses the graph partitioning theory to build the fog computing's load balancing algorithm based on dynamic graph partitioning. The simulation results show that the framework of the fog computing after Cloud Atomization can build the system network flexibly, and dynamic load balancing mechanism can effectively configure system resources as well as reducing the consumption of node migration brought by system changes.

Keywords: fog computing; graph partitioning; load balancing

I. INTRODUCTION

Nowadays, the existing cloud computing systems mostly adopt the master-slave structure, and the cloud infrastructure, platform, software and services are all provided by the cloud service providers. Users access to the cloud through the network and obtain the on-demand cloud service. This is a kind of pure and high degree of polymerization service computing model[1]. Cloud computing is just improving the cloud resource utilization. The user as the cloud service user, do not participate in the construction of cloud computing. And a large number of users idle resources do not be used effectively. In addition, cloud service providers (CSP) control over user data absolutely. In the case of the cloud service provider is not fully trusted, the security of user data cannot be guaranteed, and the degree of privacy awareness is low. Cloud service is a high degree of polymerization of service computing. Although cloud service is cheap, simple and convenient, but it consumes a lot of network bandwidth. A large quantity of users ' access will increase the network traffic greatly, and it resulting service interruptions, network latency and other issues. And the centralized calculation of cloud computing will result in uneven distribution of outlets, then the user access response speed of the network will be reduced. Therefore, in the era of big data, we need service computing model with lower degree of polymerization, and we should emphasize on the edge of computing devices and

In this paper, the author construct the system model of fog computing by combining the graph theory and characteristics of fog computing. Furthermore, they propose a fog computing dynamic load balancing mechanism based on graph repartitioning.

scatter computing or storage tasks on physical nodes, rather than centralized.

In the age of big data, data is characterized with extensive, high speed, variety, low value, etc. To meet the business needs of large data, the new basic network features include ultra high speed switching, unified exchange, exchange of virtualization, transparent exchange, etc. However, the existing network infrastructure is far from achieving the request. Big data has a large amount of data and it is very complicated. There are frequent conflicts and cooperation between massive data, and the massive data has a strong redundancy and complementarity and real-time performance. And it is also multi-source heterogeneous data[2]. Therefore, in the case of high real-time requirements, it is a huge challenge for the large amount of data filtering, processing, transmission and application.

Bonomi first proposed the concept of fog computing. Fog computing is a kind of computing between cloud and personal, and it is a Para-virtualization service computing architecture model[3]. Data, data processing and application in the Fog Computing are concentrated in the network edge device. Fog computing expands the network computing model of cloud computing, which is more widely used in various services. Fog computing not only has the advantages of cloud computing, but also has the advantages of client computing. It can make full use of the computing resources and storage resources of each fog node, and it uses cloud services in terms of resource sharing. Fog computing can well solve the service computing problems of delay sensitive application. There are several obvious characteristics of fog computing: low delay and location aware; more extensive geographical distribution; adaptive mobility application; support more edge nodes; P2P architecture, fog nodes not only provide service but also use service. These features make the deployment of mobile business more convenient and meet more extensive node access. Fog computing not only can solve the problem of networking equipment automation, but also it requires less of data transmission. It can improve the local storage and computing power, and eliminate the bottleneck of data storage and data transmission.

Data, (data) processing and application in the Fog Computing are concentrated in the network edge device. Therefore, compared with the cloud computing, it is more frequent for Fog node' changes in state, such as joining, exiting, updating and so on. Due to the relatively low computing and storage capacity of edge equipment, therefore, edge device cannot run multiple virtual machines simultaneously as physical devices in cloud computing. In the fog computing system, multi edge devices are composed of a virtual machine. In order to make full use of the hardware resources of the nodes in the system, and the efficient service as well, on the one hand, it hopes to shorten the idle waiting time by the reasonable allocation of tasks and the balance of the load; On the other hand, it is hoped that reducing the communication overhead as much as possible, and improving the parallel efficiency. Because of the dynamic nature of the system, the dynamic load balance model is needed to realize the load balance of the fog computing system.

As the research of the fog computing is in the initial stage, there are a lot of research on the load balancing and resource scheduling of cloud computing system at present. According to the different control modes, it is mainly divided into centralized policy and distributed strategy[4]. Centralized policy manage the load on each server of the cloud computing system by controlling the central control node of the system, and also for task scheduling, which is easier to maintain, and the system can achieve the load balance quickly, but it is easy to produce the performance bottleneck of the center node[5-9]. Distributed strategy has no central control node, of which each node can search low load servers in the system, and it's the proper one to apply to the large public cloud. This method has good expansibility and do not have the performance bottleneck of the central controller, but the implementation of this strategy is much more complex, and it's communication overhead is large relatively[10-16].

In summary, computing load balancing of the existing large cloud can mainly be the development of load balancing technology in cluster system, however, the existing methods have shortcomings in the aspects of system hierarchy, fault tolerance and load forecasting, which can not be directly applied to the large dynamic and P2P architecture of the fog computing system. On the basis of the research on the architecture of the fog computing system, we builds the graph model of the fog computing in the paper, and use the vertices to represent nodes, in which vertex right means the amount of resources. And the edges represent the data dependencies between tasks, and edge right indicates link bandwidth. On the basis of the above, we innovate the layered modeling of fog and construct the dynamic graph repartition algorithm for the fog computation by use of cloud atomization process. We use the result of the last load balancing as the input of the repartitioning algorithm. To achieve load balance, we gradually change the original block changing the original segment, and we try to reduce the difference between the results of the repartition and the original segmentation results. Under the premise of ensuring the load balance of the fog, we try to minimize the migration overhead after the repartition as well. Therefore, when the resource or the task changes, the system can achieve load balancing with only a small overhead. Experiments have proved that the load of the fog computing can quickly reach the equilibrium conditions through the implementation of the scheme in this paper, and the equilibrium time or the equilibrium cost is better than the static load balancing algorithm.

II. Fog Computing System Framework

Hierarchical framework of the fog computing system is shown in Fig.1. It is divided into four layers from bottom to top—Physical Resource Layer, Resources Layer of Cloud Atomization, Service Management Layer and Platform Management Layer.

(1) Physical Resource Layer consists of the center of the cyberspace, the computing and storage resources of edge nodes and network resources among them.

(2) Resources Layer of Cloud Atomization carries the cloud atomization process on the computing, storage, networking and other resources of the physical recourse layer as the core components of the fog computing system, to form the resource pool of cloud atomization. It atomizes the physical node with high computing and storage resources in the system, forming multi virtual machine nodes. And the node with little resource is atomized into a single virtual machine node. Compared with the virtualization in cloud computing, the range of the particle size of cloud atomization process is wider and the dynamic degree of variation is greater.

(3) Service Management Layer includes cloud atomization management, resource monitoring, load balancing, fault processing and other functions, to achieve monitoring, management and scheduling of the resources of cloud atomization. Platform management layer provides the interface service to users, achieving security, service and management of resources.

(4) Platform Management Layer, Service Management Layer and Resources Layer of Cloud Atomization are logic function layer abstracted from the Physical Resource Layer. They can undertake by a node in the fog computing system with a long online time and stronger computing and storage capacity. The nodes in the system can use the fog computing services simultaneously when providing the resources of atomization.

This paper studies the task-oriented load balancing mechanism of the fog computing. The system carries the cloud atomization process on the physical nodes in the fog computing system according to a certain amount of resources, forming virtual machine nodes by clustering division. Fog computing tasks are assigned to single or multiple virtual machines nodes according to the level of resources required of the task.

III. The Fog Computing System Load Balance Mechanism

Due to the physical nodes of fog computing join and quit system often frequently, task and resource change dynamic. Therefore, in order to achieve the load balancing of the fog computing system, the dynamic changes of the system should be fully considered. First, we define the related elements of the system. Then, we carry the cloud atomization process on the system physical nodes, and abstract physical node graph model into a virtual machine node graph model. Through the graph partitioning of the virtual machine node, we achieve load balancing of task allocation. Based on the situation of the resources change of node and network, when we realize the dynamic changes of the system, we use dynamic load balancing task assignment.

Definition 1, Fog computing system resources $\{cpu, mem, io, net\}$. Resources in the fog computing system include computing resources (cpu), storage resource (mem), throughput resource (io) and network resource (net).

Definition 2, Equivalent resource calculation $res = min(cpu, \alpha \cdot mem, \beta \cdot io, \gamma \cdot net)$, the parameters α, β, γ of them are empirical parameters. Each of them represents a unit of computing resources. It means we require α units of storage resources, β units of throughput resources and γ units of network resources, and they form a complete set to play a role.

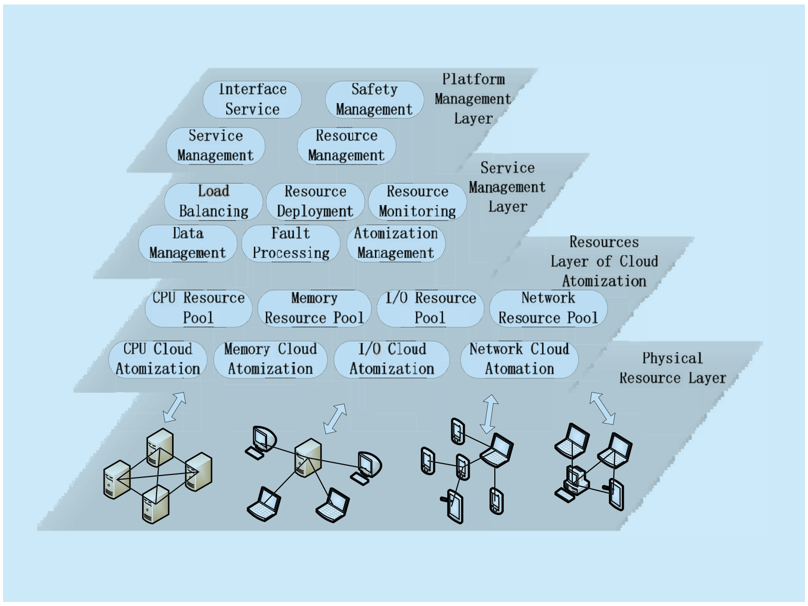

Fig.1 *Hierarchical framework of the fog computing system*

Definition 3, Nodes atomization, we turn the physical nodes which have more computing and storage resources in the system atomization, and then they form multiple virtual machines.

Definition 4, Nodes cloudization, we turn the physical nodes which have less computing and storage resources in the system cloudization, and then they form a virtual machine.

Definition 5, Virtual machine node unit resource $\triangle$, it represents the equivalent amount of resources needed to build a unit virtual machine.

Definition 6, Link weight calculation, we use link bandwidth as link weights, it represents the network resources of the link, and it can reflect the resource distance between nodes.

3.1 Physical nodes in the fog computing cloud atomization

The physical nodes in the fog computing system can be expressed as a non-directed and weight value connected graph $G_P=\{V_P,E_P\}$. V_P represents a collection of physical nodes in fog computing $V_P=\{v_{Pi}, i\in[1,n]\}$, and E_P represents a collection of link between physical nodes $E_P=\{e_{Pj}, j\in[1,m]\}$.

(1) Traverse and calculate the equivalent resource of nodes $res(v_{Pi})$ in V_P. If $res(v_{Pi})\geqslant\triangle$, the v_{Pi} will be atomized into k virtual machine nodes $v_{PiFl}(i\in[1,k])$ and a virtual physical node v_{PiP}. Among them, $k=\lfloor res(v_{Pi})/\triangle\rfloor$. If $res(v_{Pi})<\triangle$, remain unchanged. Virtual physical nodes, the original physical nodes and links compose V_P' together.

(2) The physical nodes and virtual physical nodes in V_P'. Initialization $V_{new}=\{v_\Omega\}$. v_Ω is any one code in V_P', and let $E_{new}=\{\}$.

(3) In the collection E_P, we select the edge $e=<u,v>$ that is the biggest link weight value

$wid(e)$, and node u is an element of the collection V_{new}, and not in the collection V_{new}. When there are multiple links to meet the conditions, we choose the biggest one of $res(v)$, and put v into collection V_{new}, and put e into collection E_{PS}. Repeat operation (3) until all nodes in V_P' are traversed, and form a collection $T=\{V_{new}, E_{new}\}$.

(4) Given a minimum link edge weight value η. According to following multi-objective programming, we remove the edges that satisfy the conditions from T. Formation of forest $F=\{f_w, w\in[1,g]\}$.

$$\begin{cases} wid(e) < \eta \\ \Delta(1+\psi) < \sum res(v_k) < \Delta(1+\zeta), v_k \in f_w \end{cases}$$

(5) Physical nodes and virtual physical nodes of all the trees in the forest $F=\{f_w, w\in[1,g]\}$are aggregated to form a virtual machine node.

(6) The virtual machine nodes those are generated in the step (1) and step (5) and the external links of virtual machine nodes get together, and form the Virtual machine nodes graph model. The virtual machine nodes those are generated in step (1) and the virtual machine nodes that contain virtual physical nodes in the step (5) constitute a complete graph.

Till then, we complete the cloud atomization treatment of fog computing system physical nodes. As is shown in **Fig.2**.

3.2 Virtual machine node dynamic graph partitioning algorithm

After the Cloud atomization treatment, the physical nodes of fog computing system come into being virtual machine nodes. According to the resource distance, task load balancing and other conditions, the virtual machine nodes provide services to the users by Graph partitioning and clustering.

Definition 7, the virtual machine nodes graph partitioning

Suppose that there is a links collection $E_{PVS}\subset E_{PV}$ setting in the abstract graph $G_{PV}=\{V_{PV}, E_{PV}\}$ of the virtual machine nodes, after removing E_{PVS} from the system G_{PV}, we divide the system G_{PV} into h clustering $\prod_{E_{PVS}}\{V_1, V_2, ..., V_h\}$. If it meets the following conditions, we call it a h road segment of the virtual machine node.

(1) $V_i \subset V$ and $V_i \neq \emptyset (1 \leqslant i \leqslant h)$

(2) $V_i \cap V_j = \emptyset (1 \leqslant i, j \leqslant h)$

(3) $\bigcup_{i=1}^{k} V_i = V_{PV}$

In clustering $\prod_{E_{PVS}}\{V_1, V_2, ..., V_h\}$, $V_i(1\leqslant i\leqslant h)$ is called i-th partition blocks, and E_{PVS} become link partition, and all link in the E_{PVS} is called a split link. The weight value of V_i is called the amount of resources $res(V_i) = \sum_{v_r\in V_i} res(v_r)$. The sum of all the resources in G_{PV} is $res(V_{PV}) = \sum_{v_r\in V} res(v_r)$.

Definition 7, Fog computing task sequence $P=\{p_1,p_2,\ldots,p_h\}$.

The amount of resources required to perform the task p_q is $res(p_q)$, and the total amount of resources $res(P)$ required by the task sequence P is $res(P) = \sum_{p_q\in P} res(p_q)$.

Definition 8, Load balance factor $avg(\prod_{E_{PVS}})$ is compatible degree of the distribution of task-needed resources and virtual machine nodes partition clustering resources.

$$avg(\prod_{E_{PVS}}) = \sqrt{\frac{1}{h}\{(\frac{res(v_i)_{\max}}{res(p_q)_{\max}} - \frac{res(V)}{res(P)})^2 + ... + (\frac{res(v_i)_{\min}}{res(p_q)_{\min}} - \frac{res(V)}{res(P)})^2\}}$$

Definition 9, the link partition $cost(\prod_{E_{PVS}})$ is the sum of all the split links weight. That is $cost(\prod_{E_{PVS}}) = \sum wid(e_{PVS_i})$, among them

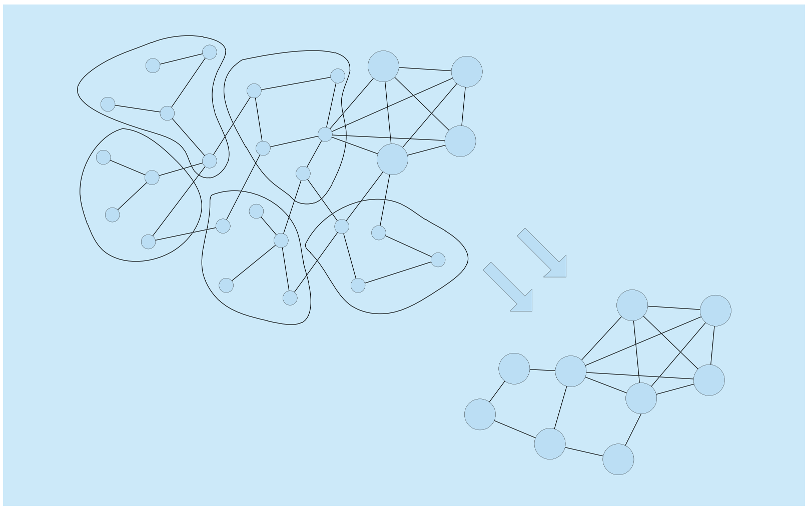
Fig.2 *Cloud atomization treatment of fog computing system physical nodes*

$e_{PVS_i} \in E_{PVS}$.

Construction of virtual machine nodes graph partition algorithm:

(1) Construct the minimum spanning tree T_{PV} of $G_{PV}=\{V_{PV}, E_{PV}\}$.

(2) Given a minimum link edge weight value π, and compatible degree constant ϖ of cluster resource quantity. According to the following multi-objective programming, we remove the edges E_{PVS} that satisfy the conditions from T. Formation of the forest $F_{PV}=\{f_{PV\theta}, \theta\in[1,h]\}$.

$$\begin{cases} wid(e) < \pi \\ avg(\prod_{E_{PVS}}) < \varpi \end{cases}$$

(3) $G_{PV}'=\{V_{PV}, (E_{PV}-E_{PVS})\}$ is the load balancing partition of fog computing virtual nodes about task sequences.

(4) When the system state changes, we repartition the graph based on the historical segmentation information. The changes of the system state including the node join and exit, the increase or decrease of resources and the increase or decrease of link bandwidth. Here we mainly study the situation of node join and node exit. Other types of states can be transformed into this situation.

(5) When the physical nodes join the system, if the new node's resource is greater than the standard value of virtual machine resources $res(v_{Pnew})\geqslant\triangle$, the node will be atomized into k virtual machine nodes $v_{PnewFj}(j\in[1,k])$ and a virtual physical node v_{PiP}, among them $k=\lfloor res(v_{Pi})/\triangle\rfloor$.

(6) $v_{PnewFj}(j\in[1,k])$ is virtual machine collection generated by clustering. The clusters of other virtual machines that are connected with the $v_{PnewFj}(j\in[1,k])$ and the hop that is not more than two assign partial tasks to $v_{PnewFj}(j\in[1,k])$. If the load balance factor of the system is satisfied the condition $avg(\prod_{E_{PVS}}) < \varpi$, the dynamic repartition is over. If it is not satisfied, the task continues to diffuse to the clustering which hop count is not more than two. If satisfied, the dynamic repartition is over. If not satisfied, the whole system will be repartitioned.

(7) If the new node's resource is less than the standard value of virtual machine resources $res(v_{Pnew})<\triangle$, this new node will be cloudization to the smallest connected $\sum res(v_k)$ virtual machine node. If this virtual machine node $\sum res(v_k)_{new}>2\cdot\triangle$, the virtual machine is atomized into two virtual machine nodes. And we will repeat step (5) of the diffusion process. Otherwise, the dynamic repartition ends.

(8) When the physical node exits the system, if the node's resource is greater than the standard value of the virtual machine resource $res(v_{Pnew})\geqslant\triangle$, the task of virtual machine that is atomized by physical nodes diffuses to the connected and no more than 2 hop virtual machine clustering. If the load balance factor of the system is satisfied the condition $avg(\prod_{E_{PVS}}) < \varpi$, the dynamic repartition is over. If it is not satisfied, the task continues to diffuse to the clustering which hop count is not more than two. If satisfied, the dynamic repartition is over. If not satisfied, the whole system will be repartitioned.

(9) If the node's resource is greater than the standard value of the virtual machine resource $res(v_{Pnew})<\triangle$, if the load balance factor of the system is satisfied the condition $avg(\prod_{E_{PVS}}) < \varpi$, the dynamic repartition is over. If it is not satisfied, the task continues to diffuse to the clustering which hop count is not more than two. If satisfied, the dynamic repartition is over. If not satisfied, the whole system will be repartitioned.

IV. Simulation Result Analysis

In order to verify the validity of the load balancing mechanism in the fog computing, in this paper, we have extended the basic framework of the distributed system hadoop to build the simulation platform of fog computing. The physical nodes in the simulation platform are included 2 IBM System x3850 X5 servers (Xeon E7-4820 CPU*2, Eight core 2GHZ CPU, 64G of memory and Multiple disk array), ten computers with i5 processors with the Linux system and 4G memories, and ten intelligent mobile phones with the Android system, 4 core processor and 2GB of memory.

The simulation uses JMeter to generate the application pressure test data. In this paper, we design the load balancing algorithm and the classical hybrid strategy of dynamic load balancing algorithm[17] to simulate the scheduling experiment.

With the change number of join nodes and quit nodes, Fig.3a and Fig.3b show the average nodes number situation of migration when we use the dynamic repartition algorithm and the static segmentation algorithm in this paper. Node join and exit will lead to system load imbalance. If we use a static load static segmentation algorithm, the system needs to recalculate task load of each node. Because the segmentation algorithm may have multiple solutions, the number of nodes need to move is large. In contrast, the dynamic repartition algorithm can effectively uses the existing node load status, and only moving less nodes can make the system to achieve load balancing state. By the figure, compared with the static segmentation algorithm, using dynamic repartition algorithm can effectively reduce the average number of migration caused by load balancing node. Thereby, it can reduce the loss of resources and waiting time.

Fig.4 is a comparison of the operating time between the dynamic repartition algorithm and the classical hybrid strategy(HDLB) along with the increasing number of tasks. With the increase of the number of tasks, the system needs to run the load balancing algorithm to achieve load balancing. System takes a long time to achieve load balancing by using HDLB algorithm because system state information is needed to be recomputed. On the other hand, by using historical information, dynamic repartition algorithm can reduce the time that is used by system achieve load balancing. By the graph, the running time of the algorithm is less than that of the load-balancing algorithm based on the hybrid DLB method. And the more number of tasks, the more superior algorithm is.

V. Conclusions

In this paper, we construct the system model of fog computing by combining the graph theory and characteristics of fog computing. And on this basis, we propose a fog computing dynamic load balancing mechanism based on graph repartitioning. Simulation results prove

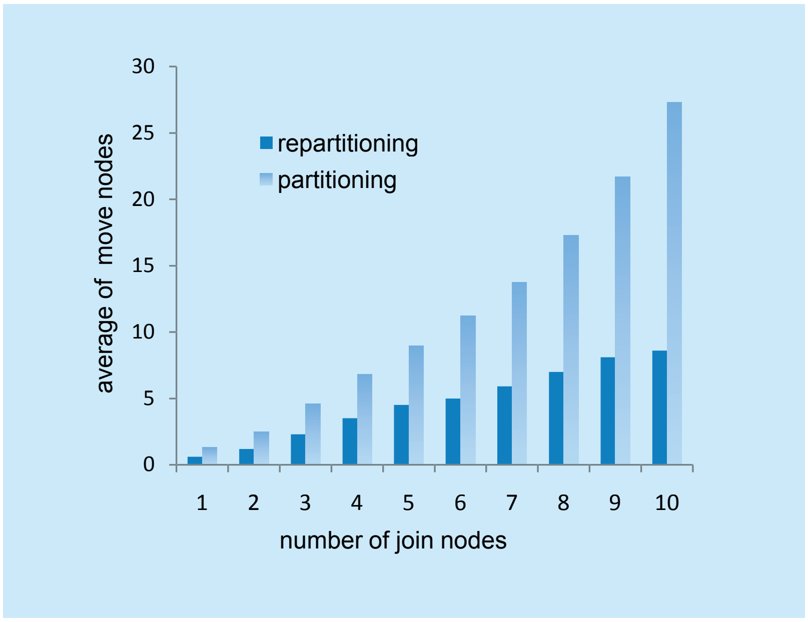

Fig.3a *The change situation of the average quantity of move nodes along with the number of join nodes*

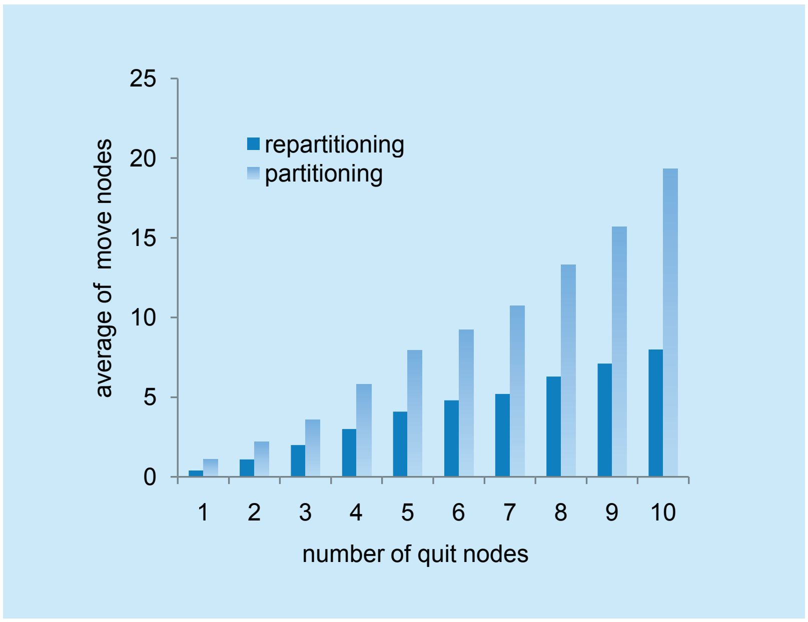

Fig.3b *The change situation of the average quantity of move nodes along with the number of quit nodes*

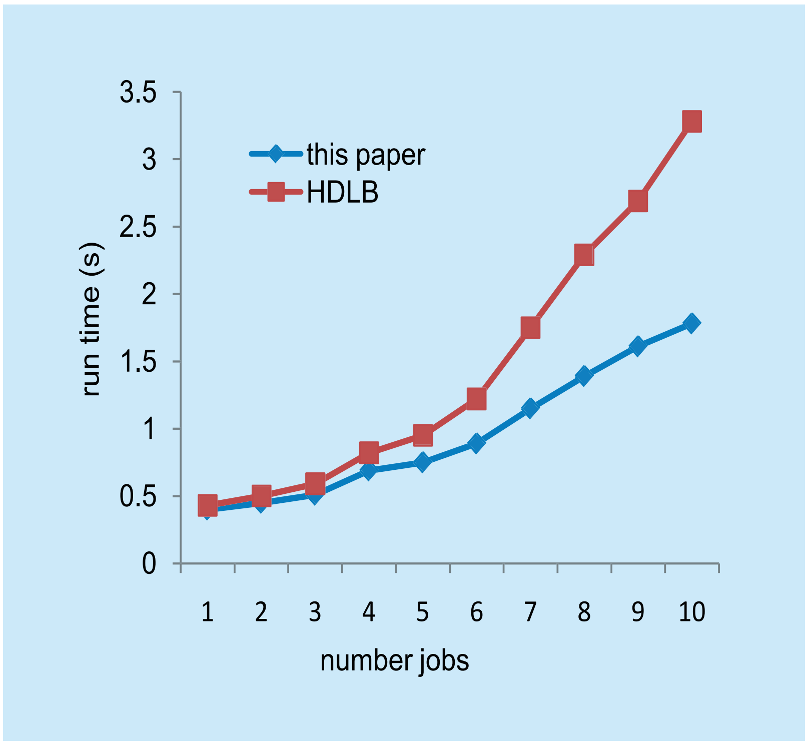

Fig.4 *With the increase of the amount of tasks, the algorithm running time comparison chart*

that this algorithm is effective and can be applied to the dynamic load balancing mechanism in the fog computing system. On this basis, our focus of the next step is to improve and extend the load-balancing algorithm based on the change of the system and the characteristics of the graph repartitioning, and further improve the performance of the dynamic load balancing mechanism.

ACKNOWLEDGEMENTS

This work was supported in part by the National Science and technology support program of P. R. China (No. 2014BAH29F05).

References

[1] M. Armbrust, et al, "A view of cloud computing", *Communications of the ACM*, vol.53, no.4, pp 50-58, 2010.

[2] J. Manyika, et al, "Big data: The next frontier for innovation, competition, and productivity", *The McKinsey Global Institute*, vol.5, no.33, pp 222, 2011.

[3] F. Bonomi, et al, "Fog computing and its role in the internet of things", *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, ACM, pp 13-16, 2012.

[4] G, Vilutis, et al, "Model of load balancing and scheduling in Cloud computing", *Information Technology Interfaces (ITI), Proceedings of the ITI 2012 34th International Conference on*. IEEE, pp 117-122, 2012.

[5] B. Mondal, D. Kousik, and D. Paramartha, "Load balancing in cloud computing using stochastic hill climbing-a soft computing approach", *Procedia Technology*, no.4, pp 783-789, 2012.

[6] P.V. Krishna, "Honey bee behavior inspired load balancing of tasks in cloud computing environments", *Applied Soft Computing*, vol.13, no.5, pp 2292-2303, 2013

[7] K. Mukherjee and G. Sahoo, "Mathematical model of cloud computing framework using fuzzy bee colony optimization technique", *Advances in Computing, Control, & Telecommunication Technologies, ACT'09. International Conference on*. IEEE, pp 664-668, 2009.

[8] G.B. Zheng, "Achieving high performance on extremely large parallel machines: Performance prediction and load balancing", *Ph.D. Thesis. Urbana, Illinois: University of Illinois at Urbana-Champaign*, 2005.

[9] T. Agarwal, "Strategies for topology-aware task mapping and for rebalancing with bounded migrations Urbana", *Illinois: University of Illinois at Urbana-Champaign*, 2005

[10] O.A. Rahmeh, P. Johnson and A. Taleb-Bendiab, "A dynamic biased random sampling scheme for scalable and reliable grid networks", *INFOCOMP Journal of Computer Science*, vol.7,no.4, pp 1-10, 2008.

[11] P.K. Yadav, M.P. Singh and H. Kumar, "Scheduling algorithm: tasks scheduling algorithm for multiple processors with dynamic reassignment", *Journal of Computer Systems, Networks, and Communications*, vol.2008, no.1, pp 1-9, 2008.

[12] T. Rotaru and H-H. Nägeli, "Dynamic load balancing by diffusion in heterogeneous systems", *Journal of Parallel and Distributed Computing*, vol.64, no.4, pp 481-497, 2004.

[13] K. Nishant, et al, "Load balancing of nodes in cloud using ant colony optimization", *Computer Modelling and Simulation (UKSim), 2012 UKSim 14th International Conference on*. IEEE, pp 3-8, 2012.

[14] M.A. Mehta, S. Agrawal and D.C. Jinwala, "Novel algorithms for load balancing using hybrid approach in distributed systems", *Parallel Distributed and Grid Computing (PDGC), 2012 2nd IEEE International Conference on*. IEEE, pp 27-32, 2012.

[15] S.J. Vaughan-Nichols, "New approach to virtualization is a lightweight", *Computer* vol.39, no.11, pp 12-14, 2006.

[16] C. Zou, et al, "Load-based controlling scheme of virtual machine migration", Cloud Computing

and Intelligent Systems (CCIS), 2012 IEEE 2nd International Conference on. IEEE, Vol.1, pp 209-213, 2012.

[17] S.C. Wang, et al. "Towards a Load Balancing in a three-level cloud computing network", *Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on.* IEEE, vol.1, pp 108-113, 2010.

Biographies

SONG Ningning, the corresponding author, email: 286319203@qq. com, received his Ph.D. degrees from University of Science and Technology Beijing, China in 2014, respectively, in Communication and Information System. He is currently a postdoctor in the School of Mechanical Engineering, University of Science and Technology Beijing, Beijing 100083, China, University of Science and Technology Beijing, China. His research interests include cloud computing security, big data and Energy efficiency analysis.

GONG Chao, received Bachelor's degree in Department of Computer and Communication Engineering from University of Science and Technology Beijing, China in 2014. At present, he is pursuing his Ph.D. degree in Department of Communication Engineering, School of Computer and Communication Engineering, University of Science and Technology Beijing. His research interests include indoor positioning,game theory and Delay Tolerant Networks.

AN Xingshuo, received Bachelor's degree in Department of Computer and Communication Engineering from Guangxi University, China in 2011. He received his master degree in Department of Science, University of Science and Technology Beijing,China in2014. At present, he is pursuing his Ph.D. degree in Department of Communication Engineering, School of Computer and Communication Engineering, University of Science and Technology Beijing.His research interests include resource allocation of indoor positioning, game theory, The next generation of mobile communication technology.

ZHAN Qiang, received Bachelor's degree in Department of Computer and Communication Engineering from University of Science and Technology Beijing, China in 2014. He is currently a master in School of Computer and Communication Engineering, University of Science and Technology Beijing, China. His research interestsinclude cloud computingsecurity, security of communication networks and game theory.