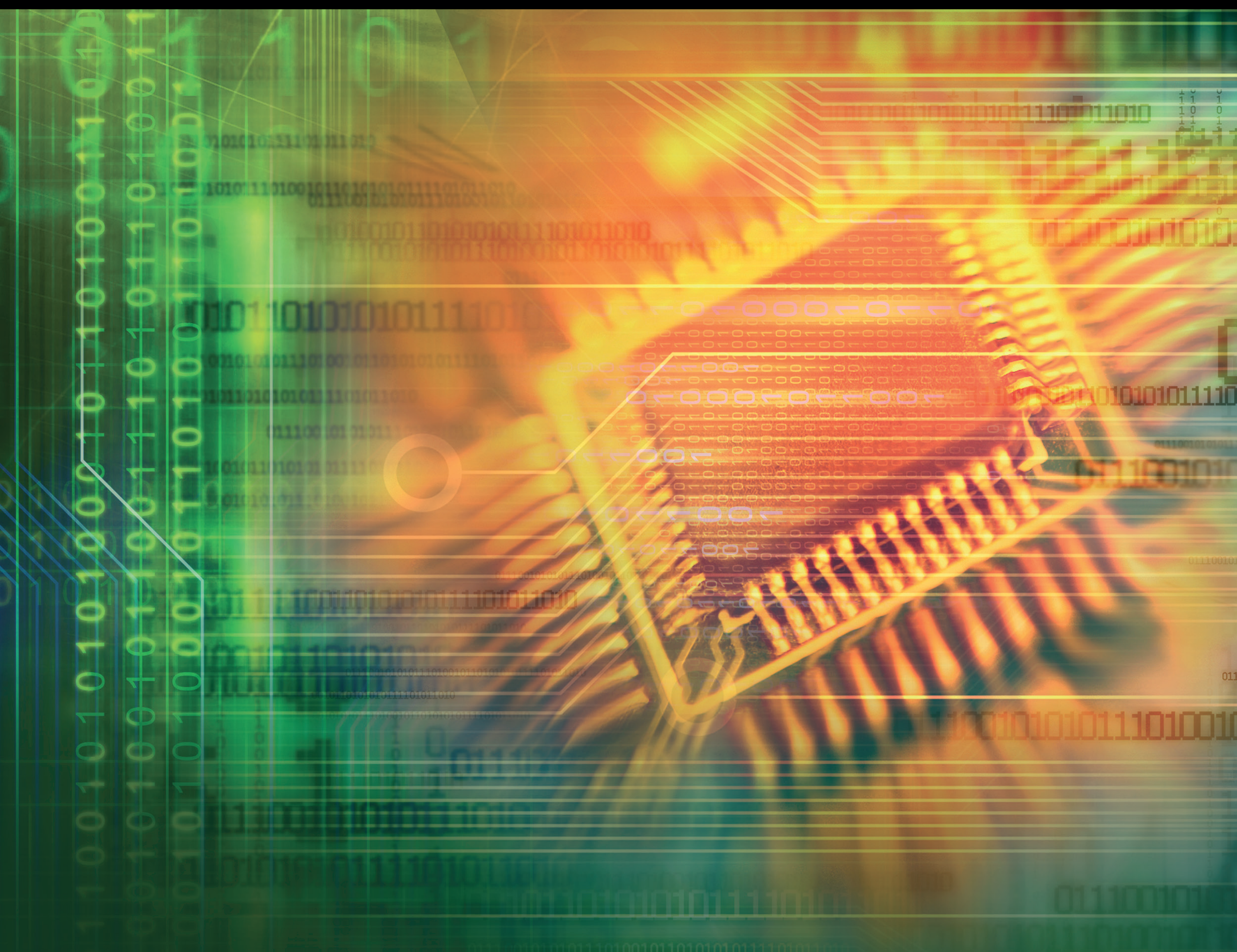
Journal of Electrical and Computer Engineering

Cognition Cloud Model for Next Generation Mobile Internet: Communication, Control, and Application

Guest Editors: Yong Jin, Chaobo Yan, and James Nightingale



Cognition Cloud Model for Next Generation Mobile Internet: Communication, Control, and Application

Journal of Electrical and Computer Engineering

Cognition Cloud Model for Next Generation Mobile Internet: Communication, Control, and Application

Guest Editors: Yong Jin, Chaobo Yan, and James Nightingale

Contents

Cognition Cloud Model for Next Generation Mobile Internet: Communication, Control, and Application
Yong Jin, Chaobo Yan, and James Nightingale
Volume 2016, Article ID 7578298, 2 pages

Cooperative Cloud Service Aware Mobile Internet Coverage Connectivity Guarantee Protocol Based on Sensor Opportunistic Coverage Mechanism
Qin Qin, Yong-qiang He, and Li-ming Nie
Volume 2015, Article ID 128691, 8 pages

Cooperative Optimization QoS Cloud Routing Protocol Based on Bacterial Opportunistic Foraging and Chemotaxis Perception for Mobile Internet
Shujuan Wang, Long He, and Guiru Cheng
Volume 2015, Article ID 641062, 7 pages

User Utility Oriented Queuing Model for Resource Allocation in Cloud Environment
Zhe Zhang and Ying Li
Volume 2015, Article ID 246420, 8 pages

Regression Cloud Models and Their Applications in Energy Consumption of Data Center
Yanshuang Zhou, Na Li, Hong Li, and Yongqiang Zhang
Volume 2015, Article ID 143071, 9 pages

Cloud Platform Based on Mobile Internet Service Opportunistic Drive and Application Aware Data Mining
Ge Zhou
Volume 2015, Article ID 357378, 7 pages

QoE Guarantee Scheme Based on Cooperative Cognitive Cloud and Opportunistic Weight Particle Swarm
Weihang Shi
Volume 2015, Article ID 398709, 9 pages

Hindawi Publishing Corporation
Journal of Electrical and Computer Engineering
Volume 2016, Article ID 7578298, 2 pages
http://dx.doi.org/10.1155/2016/7578298

Editorial

Cognition Cloud Model for Next Generation Mobile Internet: Communication, Control, and Application

Yong Jin,[1] Chaobo Yan,[2] and James Nightingale[3]

[1] *Changshu Institute of Technology, Changshu 215500, China*
[2] *University of Michigan, Ann Arbor, MI, USA*
[3] *University of the West of Scotland, Paisley PA1 2BE, UK*

Correspondence should be addressed to Yong Jin; jinyong@cslg.cn

Received 7 March 2016; Accepted 17 March 2016

It is well known that the next generation of mobile Internet has arrived, bringing with it great technical challenges and immense opportunities to enhance both our personal lives and work environments. A new mobile Internet has been the proliferation of a large and increasingly diverse set of mobile services and applications. To accommodate this rapid development of next generation mobile Internet services and applications, new reliable, trusted, and high performance cognition cloud model-based services for communication, control, and mobile computational services are required.

This special issue consists of six papers. Three of the papers focus on cloud computing: a paper on cooperative cloud service aware mobile Internet coverage connectivity guarantee protocol, a paper on cooperative optimization QoS cloud routing protocol, and a paper on user utility oriented queuing model. Another two papers are focused on data mining: one paper discusses the regression cloud models of data center and the other examines the mobile Internet service opportunistic drive and application aware data mining. The remaining paper studies the cooperative cognitive cloud and opportunistic weight particle swarm.

In the paper "Cooperative Cloud Service Aware Mobile Internet Coverage Connectivity Guarantee Protocol Based on Sensor Opportunistic Coverage Mechanism," Q. Qin et al. proposed the coverage connectivity guarantee protocol for mobile Internet for improving the Internet coverage ratio and providing connectivity guarantee, based on sensor opportunistic coverage mechanism and cooperative cloud service.

In the paper "Cooperative Optimization QoS Cloud Routing Protocol Based on Bacterial Opportunistic Foraging and Chemotaxis Perception for Mobile Internet," S. Wang et al. established the scheme of optimizing the cloud routing efficiency based on opportunistic bacterial foraging bionics for strengthening the mobile Internet mobility management and cloud platform resources utilization. They also put forward a chemotaxis perception of collaborative optimization QoS (Quality of Services) cloud routing mechanism.

In the paper "User Utility Oriented Queuing Model for Resource Allocation in Cloud Environment," Z. Zhang and Y. Li proposed a user utility oriented queuing model for task scheduling. They modeled task scheduling in cloud environment as an M/M/1 queuing system. They classified the utility into time utility and cost utility and built a linear programming model to maximize total utility for both of them.

In the paper "Regression Cloud Models and Their Applications in Energy Consumption of Data Center," Y. Zhou et al. analyzed linear and nonlinear regression energy model based on performance counters and system utilization and proposed a support vector regression energy model. They gave a general linear regression framework and compared three linear regression models.

In the paper "Cloud Platform Based on Mobile Internet Service Opportunistic Drive and Application Aware Data Mining," G. Zhou presented a reliable and efficient data mining cloud platform construction scheme based on the mobile Internet service opportunistic driving and application perception. Their scheme can satisfy the diversity of mobile Internet service and deal with the inefficient data mining problems.

In the paper "QoE Guarantee Scheme Based on Cooperative Cognitive Cloud and Opportunistic Weight Particle Swarm," W. Shi proposed the QoE (Quality of Experience) guarantee mechanism based on construction algorithms of cooperative cognitive cloud platform and optimization algorithm of opportunities weight particle swarm clustering.

It is evident that this special issue does not cover many other important areas of relevance to cloud model for next generation mobile Internet. Nonetheless, we hope that readers will find the information presented to be interesting and useful.

Acknowledgments

We thank all the authors who responded to the call for papers. We also wish to acknowledge all the reviewers for their dedicated efforts in ensuring a high standard for the selected papers.

Yong Jin
Chaobo Yan
James Nightingale

Hindawi Publishing Corporation
Journal of Electrical and Computer Engineering
Volume 2015, Article ID 128691, 8 pages
http://dx.doi.org/10.1155/2015/128691

Research Article

Cooperative Cloud Service Aware Mobile Internet Coverage Connectivity Guarantee Protocol Based on Sensor Opportunistic Coverage Mechanism

Qin Qin,[1] Yong-qiang He,[1] and Li-ming Nie[2]

[1]*College of Computer, Henan Institute of Engineering, Zhengzhou 450007, China*
[2]*School of Software Technology, Dalian University of Technology, Dalian 116621, China*

Correspondence should be addressed to Qin Qin; qq@haue.edu.cn

Received 20 August 2015; Revised 30 November 2015; Accepted 9 December 2015

Academic Editor: James Nightingale

In order to improve the Internet coverage ratio and provide connectivity guarantee, based on sensor opportunistic coverage mechanism and cooperative cloud service, we proposed the coverage connectivity guarantee protocol for mobile Internet. In this scheme, based on the opportunistic covering rules, the network coverage algorithm of high reliability and real-time security was achieved by using the opportunity of sensor nodes and the Internet mobile node. Then, the cloud service business support platform is created based on the Internet application service management capabilities and wireless sensor network communication service capabilities, which is the architecture of the cloud support layer. The cooperative cloud service aware model was proposed. Finally, we proposed the mobile Internet coverage connectivity guarantee protocol. The results of experiments demonstrate that the proposed algorithm has excellent performance, in terms of the security of the Internet and the stability, as well as coverage connectivity ability.

1. Introduction

With the development and integration of wireless network, mobile communication [1], sensor [2], cloud platform, and so forth, the next generation of Internet has been widely used and developed in the field of real-time monitoring, full coverage of mobile network, and poor environment data acquisition and communication. The mapping of Internet nodes and sensor nodes establish the heterogeneous communication, and provide support for the monitoring of information storage and forwarding and real-time processing. However, because of the diversity of environmental monitoring needs and the forwarding of sensor information collection and the network coverage and connectivity [3] of all kinds of bad environment, it is still a hot and key issue of the next generation Internet.

There are some researches of Internet coverage connectivity. A coverage-based hybrid overlay was proposed in article [4], which disseminates messages to all subscribers without uninterested nodes involved in and increases the average number of node connections slowly with an increase in the number of subscribers and nodes. A new notion of intermittent coverage for mobile users was introduced in article [5], which provides worst-case guarantees on the interconnection gap, the distance, or expected delay between two consecutive mobile-AP contacts for a vehicle. Efficient node coverage scheme was proposed for addressing the scheduling issue of nodes of the underground space near surface [6]. The authors of article [7] proposed the supporting plant designer during wireless coverage prediction, virtual network deployment, and postlayout verification sensor opportunistic coverage mechanism. A low-cost way [8] was proposed for public transit operators to enhance quality of experience for passengers who access the Internet. The control flow graph and cyclomatic complexity of the example program [9] were used to find out the number of feasible paths present in the program and compared it with the actual number of paths covered by genetic algorithm.

About the coverage issue of wireless sensor networks (WSNs), Zhao et al. [10] analyzed the changes of opportunistic coverage ratios. The new cooperative opportunistic four-level model for IEEE 802.15.4 wireless personal area network was proposed by Rohokale et al. [11]. A localization scheme named Opportunistic Localization by Topology Control was proposed in [12] for sparse Underwater Sensor Networks. The use of an MAS as an appropriate mechanism was advocated by different stakeholders in [13].

For studying the relationship of cloud platform and coverage issue of Internet, a service aware location update mechanism was proposed in [14], which can detect the presence and location of the mobile device. A novel quality aware computational cloud selection service was proposed and evaluated in [15]. The cost and energy aware service-provisioning scheme [16] was presented for mobile client in mobile cloud, which includes two-stage optimization process. A novel disaster-aware service-provisioning scheme was proposed in [17], which multiplexes service over multiple paths destined to multiple servers/datacenters with many casting. A privacy-aware cross-cloud service composition method was proposed in [18] based on its previous basic version HireSome-I.

However, these research results ignored the resources management issue of cloud platform and the mobile Internet requirements and had a little research of coverage guaranteeing ability of WSNs. Then, we proposed the cooperative cloud service aware mobile Internet coverage connectivity guarantee protocol based on sensor opportunistic coverage mechanism.

The rest of the paper is organized as follows. Section 2 gave the sensor opportunistic coverage mechanism. In Section 3, we proposed the model of cooperative Cloud service aware. The mobile Internet coverage connectivity guarantee protocol was proposed in Section 4. The results of the mathematical analysis and simulation verification are given in Section 5. Finally, we conclude the paper in Section 6.

2. Sensor Opportunistic Coverage Mechanism

In the context of mobile Internet, how to deploy wireless sensor networks to achieve reliable and stable network coverage has become the key technology of wireless sensor networks and the Internet. Separate wireless sensor network coverage rule is specified based on different type of structure. Generally, the same wireless sensor network only includes a single cover rule or a geometric figure. This is not conducive to the combination of large scale sensor networks and mobile Internet. In order to meet the needs of the integration of the Internet and reduce the complexity and robustness of wireless sensor network coverage control, a mobile Internet based on sensor deployment and network coverage algorithm is proposed. The algorithm can ensure the distributed computing and the full connected routing of sensor nodes. The algorithm can be implemented based on the irregular coverage of mobility.

Based on the driving of Internet mobile node, the algorithm combines the opportunity dynamic coverage rule and communication between the mobile node and the sensor node to achieve a high reliability and real-time security.

N denotes the number of mobile nodes which are used to connect the sensor network with the Internet. Wireless sensor networks (WSNs) deployment node number is M. The nearest distance between the mobile node and the sensor node is d_0. The minimum distance between the sensor nodes is d_1. In order to reduce the impact of the Internet node's mobility on sensor network coverage, the deployment distance parameters need to meet the following formula:

$$\frac{S(d_{\max})}{\sum_{i=1}^{M} d_1^i} \leq \frac{\mathrm{MI}(d_{\max})}{\sum_{j=1}^{N} d_0^i}. \quad (1)$$

Here, $S(d_{\max})$ denotes the maximum connection distance of sensor nodes in WSNs. $\mathrm{MI}(d_{\max})$ denotes the maximum connection distance between Internet and WSNs.

At the same time, in order to ensure the deployment of sensor nodes independently, the communication distance between adjacent sensor nodes and the Internet mobile node should be in accordance with formula (2), which is used to ensure that the mobile node and sensor nodes connected with independent characteristics

$$\frac{\mathrm{MI}(d_{\max})}{v_{\max}} \leq t_0 \left[\frac{\mu \cos\alpha}{e^{-S(d_{\max})}}\right]^{N/M}. \quad (2)$$

Here, $v_{\max}$ is the maximum moving speed of Internet node. t_0 is the delay of transmitting signal from Internet node to sensor node. μ denotes the opportunistic connection ratio between Internet node and sensor node. α is the angle of the antenna direction of the Internet node and the sensor node direction.

Sensor node S_i can sense the Internet node MI_i. Internet nodes help sensor nodes to establish network coverage with strong connectivity. The opportunity coverage ratio is shown in the following formula:

$$C_{(\mathrm{S,MI})} = \begin{cases} 0, & t \geq t_{\mathrm{TH}} \\ 1, & t \leq t_{\mathrm{TH}}, \end{cases}$$
$$t_{\mathrm{TH}} = \mu \left(\frac{d_{(\mathrm{S,MI})}}{v_{\mathrm{MI}}}\right)^{\delta}. \quad (3)$$

Here, $C_{(\mathrm{S,MI})}$ denotes the connection ratio between nodes. t_{TH} is the maximum delay of keeping communication. δ is the probability of occurrence and transformation of Internet node movement direction.

The mobile nodes and the sensor nodes would use signal level cooperation communication. The communication between Internet mobile node MI and the m sensor nodes would create the neighbor coverage relationship. They are common sensing coverage and control an area, which are shown in Figure 1. The opportunistic coverage strength OCI

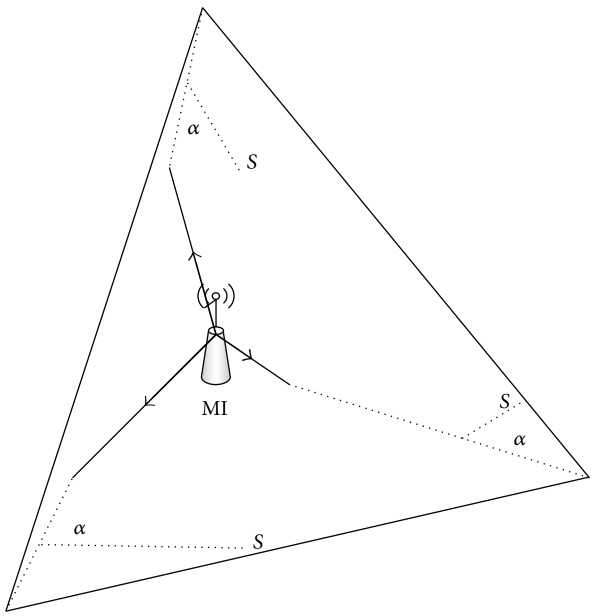


Figure 1: Common sense and coverage control area.

between sensor node and the mobile node can be obtained by the following formula:

$$\mathrm{OCI} = |\gamma - \mu|^2 \sum_{i=1}^{m} C_{i,\mathrm{MI}},$$
$$\gamma = \frac{\sum_{i=1}^{m} t_i}{t_{\mathrm{TH}}}. \tag{4}$$

Here, γ is the probability of connecting time greater than t_{TH} between the Internet nodes and sensor nodes.

Through the combination of the Internet node deployment area and sensor network deployment area of the mobile Internet, the opportunity to cover the intensity MI-OCI can be obtained through the node between the opportunity connection and collaborate sensing, as shown in the following formula:

$$\mathrm{MI} - \mathrm{OCI}(N, M) = \frac{\sqrt{t_{\mathrm{TH}} |v_{\max} - v_{\min}|^2} \sum_{i=1,j=1}^{N,M} d_{i,j}}{\mu |\gamma - \delta|^2 v_{\mathrm{TH}}},$$
$$v_{\mathrm{TH}} = \frac{\sqrt{|S(d_{\max})^2 - \mathrm{MI}(d_{\max})^2|}}{t_{\mathrm{TH}}}. \tag{5}$$

Here, v_{TH} is the Internet node moving speed threshold.

3. Cooperative Cloud Service Aware Model

With the rapid development of the Internet and WSNs, the next generation network convergence and cloud platform business should be improved. So, the cloud platform service business platform is proposed. The service platform has the following characteristics.

(1) To guide the deployment, implementation, and management of WSNs is the center of service support and user requirements.

(2) Data services and forwarding services are capable of supporting multinetwork integration and heterogeneous services and are not independent of the front-end device and forwarding gateway for specific wireless sensor networks.

(3) The size of the cloud platform has a dynamic adjustment ability. According to different network traffic and forwarding data content, the cloud devices have the way of cooperation between the management and control.

(4) The information service quality and wireless sensor network resource utilization can be perceived as an open and reliable access to provide services to the front of the Internet and the sensor.

The architecture model of the cloud platform service business platform is shown in Figure 2. Cloud service business support platform of the Internet application service management capabilities and wireless sensor network communication service capabilities are abstracted as the architecture of the cloud support layer. The support layer is a business entity shared by all cloud services. Cloud service middleware platform provides the deployment and execution environment for all kinds of Internet applications and data services for wireless sensor networks. Cloud physical layer for the Internet and WSNs could provide the core support of various front-end equipment and services and cross layer interface.

The above architecture shows that the perception of cloud services has important significance for the Internet management and WSNs coverage, as well as topology management. Therefore, the application of the Internet could be encapsulated as cloud services. Various types of sensor front-end devices and cloud services could be considered as the perception of objects, through the aggregation of the Internet service and WSNs, which is used to provide a basis for the internet.

The cloud service perception model is shown in Figure 3. Based on the Internet mobility management business security mechanism and the WSNs connectivity guarantee mechanism, the perception model is proposed. The complexity of cloud computing through WSNs communication between different devices could be reduced. Cloud platform through cooperative cloud services for WSNs deployment and coverage could provide a unified positioning business. Cloud platform through the cooperative cloud service could be packaged in the cloud platform middleware layer. Web services of the Internet and WSNs data collection and forwarding services could be provided by the combination of cooperative cloud aggregation services. This service can provide reference information and decision basis for route maintenance and cooperative coverage in the WSNs coverage area. In particular, there are the perception pile between the Internet and WSNs, which is composed of wireless multiple word system bus. In order to improve the parallelism of the system, we designed the control feedback interaction

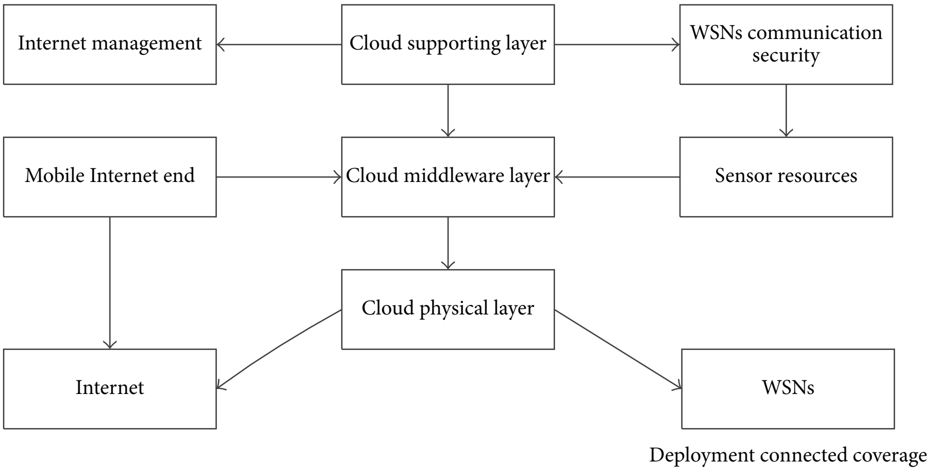


Figure 2: Architecture model of cloud service platform.

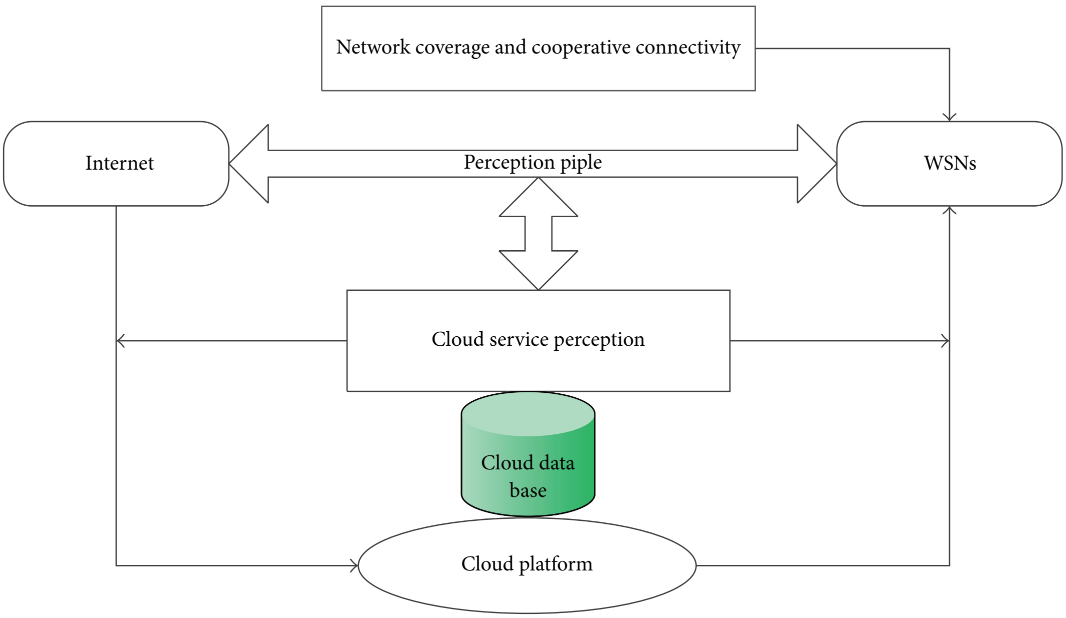


Figure 3: Cloud service perception model.

line, which is from cloud service perception platform to the Internet and WSNs.

4. Mobile Internet Coverage Connectivity Guarantee Protocol

Mobile Internet applications can achieve the application development through the integration of cloud platforms and WSNs. When the application is extended, the problem of mobile management and resource management and reliable information forwarding should be considered. The high coverage ratio could be maintained by deploying a large number of redundant devices or sensor nodes in the cloud platform and WSNs. But about the Internet node service life, WSNs sensor node resource management and cloud computing scheduling, and so forth, the network connectivity could not kept by increasing the size of equipment.

Therefore, it is the key technology to make full use of the opportunity of each sensor node in wireless sensor network, and the effective driving ability of the cloud platform service, which is used to ensure the Internet connectivity. Through the WSNs, the sensor nodes and the Internet nodes are in the mobile state, which can reduce the resource consumption and realize the effective seamless coverage of the network. In order to reduce the impact of the mobile Internet connectivity coverage, the Internet and the cloud platform play a role in the root node of the cloud device. The diffusion of Internet nodes and sensor nodes coverage can be opportunistic, dynamic, and adjusted at the same time to achieve maximum coverage of connectivity and through collaborative optimization control to achieve global connectivity. This can reduce the resource consumption of each node and can reduce the probability of the occurrence of invalid overlap between nodes, as shown in the following formula:

$$C_{\mathrm{RC}} = \sum_{k=1}^{N+M} R_k \frac{S(d_{\max}) + \mathrm{MI}(d_{\max})}{\sum_{i=1}^{M} d_1^i d_0^i},$$
$$P_{\mathrm{IO}} = \sum_{i=1}^{C_N} \rho_i \frac{\sqrt{|\mathrm{SC}_N - \mathrm{SC}_M|}}{\mathrm{SC}_{N+M}}. \tag{6}$$

Here, C_{RC} denotes cover resource consumption. R_k denotes the resource consumption of node. P_{IO} is the probability of invalid overlap. C_N is the cloud number of the platforms. ρ denotes the active ratio of cloud. SC denotes the covering area.

Based on the Internet coverage mechanism and the cooperative security mechanism, the following problems should be considered.

(1) When the mobile node and sensor nodes are used in the model, the sensing distance of the node is related to the computing power of the cloud computing power. According to the node sensing distance, the Internet coverage is irregular geometry area composed by a number of mobile nodes and sensor nodes. Internet coverage area is not affected by node mobility.

(2) The sensor covers the sensing area opportunistically and deploys sensor nodes based on the remaining resources of the Internet node and cloud service needs.

(3) When the sensor network is deployed by the terrain or the impact of a large building, it can be through the cloud platform of collaborative cloud service sensing mechanism, the communication of all nodes to the same plane, as shown in Figure 4.

(4) The hierarchical communication architecture of sensor nodes and the hierarchical protocol of Internet nodes occur through the cloud platform to achieve cross layer interaction. The nodes and the cloud devices in the Internet are covered by the Internet, which can communicate directly with the sensor nodes and the cloud devices, and the heterogeneous communication protocols can be handled transparently.

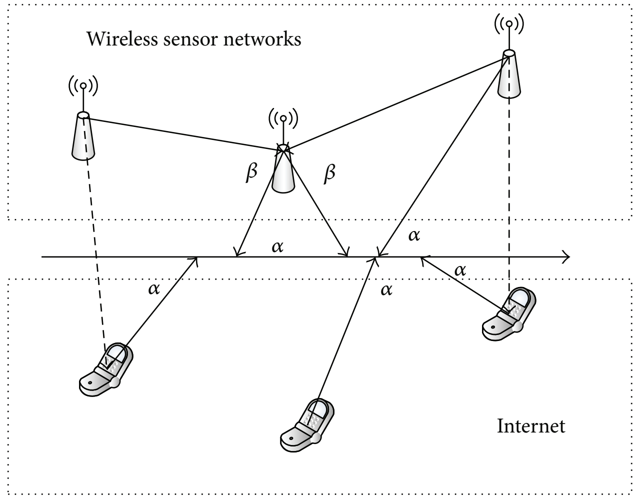


Figure 4: Communications mapping with cooperative cloud aware scheme.

The WSNs opportunity coverage role of Internet mobile coverage connectivity is reflected in the following aspects.

(1) When the sensor nodes have separated from the Internet, these nodes would broadcast a request of the subnetwork separation and reconstruct to the neighbor nodes. After receiving the request, the nodes are calculated by the cloud platform to maintain the network connectivity. If you cannot meet the needs of the Internet seamless coverage, the cloud platform issued a collaborative cloud service perception control in the Internet node and sensor nodes to search for collaborative gaps and opportunity to drive the region for ensuring the full connectivity of the Internet monitoring area and real-time coverage, as shown in the following formula:

$$d_{\max(S,\mathrm{MI})} = \sqrt{\left|d_{\max(S)} - d_{\min(\mathrm{MI})}\right|^2} \sum_{i=1,j=1}^{N,M} d_{i,j},$$
$$\mathrm{SC}_{\mathrm{crevice}} = \frac{\sqrt{\left|\mathrm{SC}^2{}_N - \mathrm{SC}^2{}_M\right|}}{\tan\left(d_{\max(S,\mathrm{MI})}/\theta\right)}. \tag{7}$$

Here, θ is the Internet node and sensor node antenna angle. $\mathrm{SC}_{\mathrm{crevice}}$ is the gap area.

(2) In order to avoid the loss caused by the network nodes and sensor nodes due to overlapping coverage area, the cloud platform and data forwarding provide seamless coverage and full connected routing. We must gradually reduce the communication delay between the nodes in the coverage area, so that the three-party interaction and coverage area of the cloud platform, the Internet, and WSN achieve the best equilibrium state.

(3) When the channel quality is good, it can increase the share of the three parties, the largest consumption of resources to achieve maximum coverage area. When the channel quality is poor, the sensor can increase the opportunity to cover the weight and the cloud service cooperative sensing coefficient and achieve the optimal control in the area of Internet coverage and connectivity.

5. Mathematical Analysis and Simulation Verification

The performance of the proposed algorithm is analyzed in this section, which includes the stability, connectivity, and

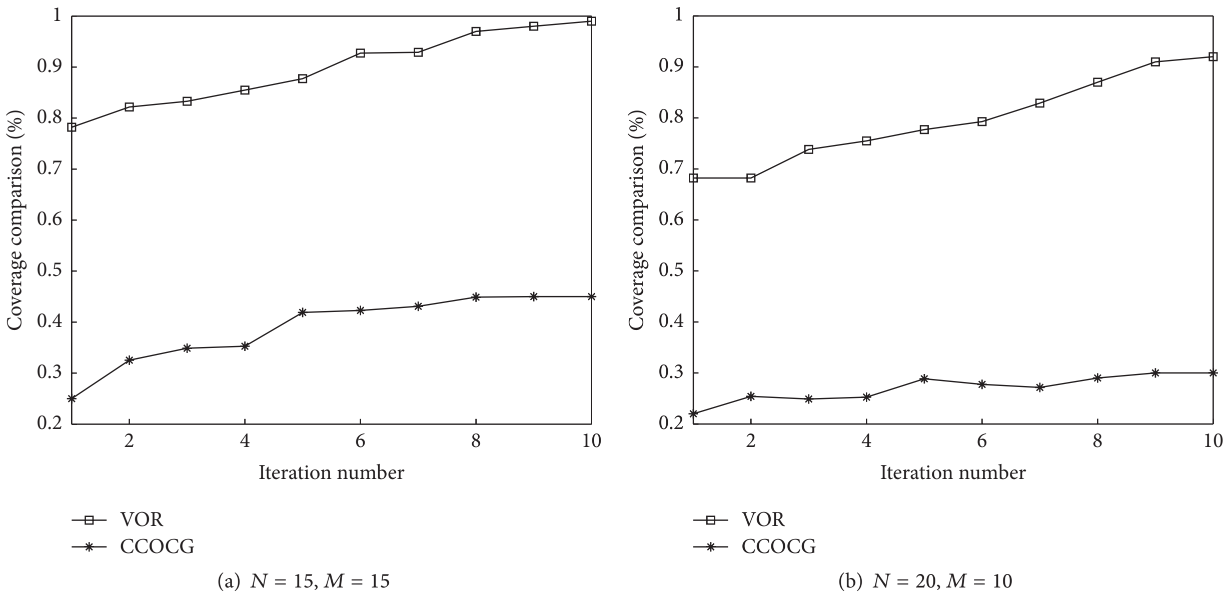


Figure 5: Coverage comparison.

security of the Internet. The coverage stability of the Internet is considered, both the Internet node and the sensor node sensing area coverage and the connectivity security capability. When the security capability is strong, the security time of the proposed algorithm is investigated, including the Internet node iteration time, the cloud computing time, and the forwarding time of the sensor. The time spent on the Internet is determined by the time of the node moving speed, cloud computing delay, and sensor node coverage.

The cost of Internet connectivity includes:

(1) the network nodes in order to maintain the loss of the mobile speed;

(2) the total number of nodes required by the Internet;

(3) the resource consumption of the cloud devices to maintain continuous calculation;

(4) the energy consumption of sensor nodes and the energy consumed in communication.

So, by comprehensive evaluation of coverage connectivity and security costs, we can better analyze the performance of the proposed algorithm (CCOCG) in the Internet node mobility management, cloud service perception, and WSNs opportunity communication and other environmental performance.

In the experiment, the Internet covering model of time domain, frequency domain, and spatial domain is considered. Random mobile model is adopted for the nodes and sensor nodes in the Internet. The cloud devices of cloud platform have the same configuration and deploy evenly. The basic parameters of the integrated network and WSN network are as follows: the maximum communication radius of nodes is 250 m, and the fusion network size is 800 m × 900 m. The average value of the results of the 50 experiments was used as the final experimental results.

In the experiment, we compared and analyzed the performance of the proposed algorithm and VOR algorithm [19]. In the comparison of the experimental results, the change of the distance between two extremes and the sensor is changed, and the effect of the algorithm is analyzed.

When the number of the Internet nodes and the number of sensor nodes are different or equal, the coverage performance of the proposed algorithm and the VOR algorithm have been shown in Figure 5. The experimental results are as follows.

(1) With the increase of the size of nodes, the coverage area of the two algorithms is increased.

(2) When the number of nodes and the number of sensor nodes are different, the coverage performance of the proposed algorithm is better than the VOR algorithm. Because the proposed algorithm uses the opportunity sensor coverage method, the local node coverage information and the global coverage of WSN can be deployed and connected with the Internet to achieve global optimal coverage.

(3) With the increase of the number of iterations, the proposed algorithm based on cloud service area has a stronger coverage restoration and stable full connectivity.

With the different node of the mobile speed of the Internet, which executed 30 iterations and recover update, the coverage of the 2 algorithms of the full connectivity ratio is shown in Figure 6. The proposed algorithm provides a transparent interactive platform through the cloud platform to build a middleware layer between the Internet and WSN. There is mapping relationship between the network nodes and sensor nodes, which weaken the influence of the node moving speed and communication distance on the whole network. And the nodes can adaptively adjust the moving speed with the help of the cooperative cloud service sensing strategy. The network algorithm of full connectivity ratio was significantly higher

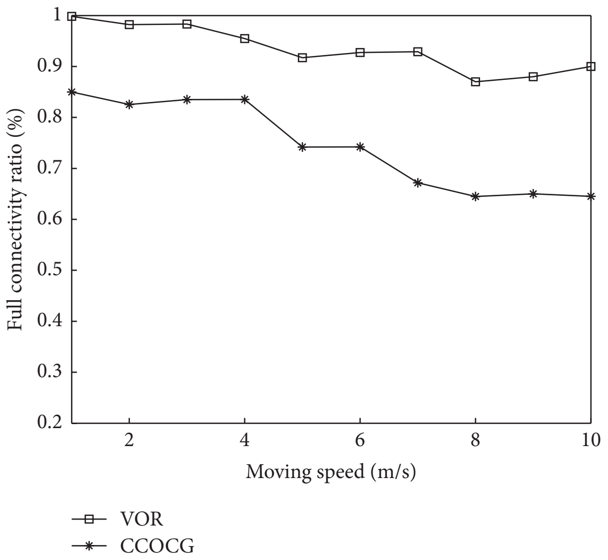


Figure 6: Full connectivity ratio.

than that in the VOR algorithm, and, in highly mobile nodes request, still has high communication rate.

6. Conclusions

How to improve the whole connectivity of the Internet through the integration of cloud platform and WSNs becomes an important technology of Internet application development. First, we studied the sensor opportunistic coverage mechanism based on the scale of Internet nodes, cloud, and sensor nodes. Second, according to the requirements of users and cloud platform computing ability, the cooperative cloud service aware model was given. Third, mobile Internet coverage connectivity guarantee protocol was proposed for resolving the mobility management, resource management and reliable information forwarding, and so forth. Mathematical analysis and simulation verification proved that the proposed scheme is superior to the VOR algorithm, such as coverage rate and full connectivity ratio, as well as connectivity guarantee price.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work is supported in part by Key Scientific Research Project of Henan Province (15A520054) and Science and Technology Project of Henan Province (112102310550).

References

[1] B. Xie and A. Kumar, "A protocol for efficient Bi-directional connectivity between ad hoc networks and internet," *Journal of Internet Technology*, vol. 6, no. 1, pp. 101–108, 2005.

[2] N. Bartolini, G. Bongiovanni, T. F. La Porta, and S. Silvestri, "On the vulnerabilities of the virtual force approach to mobile sensor deployment," *IEEE Transactions on Mobile Computing*, vol. 13, no. 11, pp. 2592–2605, 2014.

[3] A. Sathiaseelan and J. Crowcroft, "Internet on the move: challenges and solutions," *Computer Communication Review*, vol. 43, no. 1, pp. 51–55, 2013.

[4] X. Ma, Y. Wang, and W. Sun, "Feverfew: a scalable coverage-based hybrid overlay for internet-scale pub/sub networks," *Science China Information Sciences*, vol. 57, no. 5, pp. 1–14, 2014.

[5] Z. Zheng, P. Sinha, and S. Kumar, "Sparse WiFi deployment for vehicular internet access with bounded interconnection gap," *IEEE/ACM Transactions on Networking*, vol. 20, no. 3, pp. 956–969, 2012.

[6] X. Huang and J. Chen, "Efficient node coverage scheme of wireless sensor network under multi-constraint conditions," *Journal of Networks*, vol. 8, no. 10, pp. 2277–2284, 2013.

[7] S. Savazzi, V. Rampa, and U. Spagnolini, "Wireless cloud networks for the factory of things: connectivity modeling and layout design," *IEEE Internet of Things Journal*, vol. 1, no. 2, pp. 180–195, 2014.

[8] S. G. Hong, S. Seo, H. Schulzrinne et al., "ICOW: internet access in public transit systems," *IEEE Communications Magazine*, vol. 53, no. 6, pp. 134–141, 2015.

[9] M. Panda and D. P. Mohapatra, "Generating test data for path coverage based testing using genetic algorithms," in *Proceedings of International Conference on Internet Computing and Information Communications*, vol. 216 of *Advances in Intelligent Systems and Computing*, pp. 367–379, Springer, New Delhi, India, 2014.

[10] D. Zhao, H. Ma, L. Liu, and X.-Y. Li, "Opportunistic coverage for urban vehicular sensing," *Computer Communications*, vol. 60, pp. 71–85, 2015.

[11] V. M. Rohokale, S. Inamdar, N. R. Prasad, and R. Prasad, "Energy efficient four level cooperative opportunistic communication for wireless personal area networks (WPAN)," *Wireless Personal Communications*, vol. 69, no. 3, pp. 1087–1096, 2013.

[12] S. Misra, T. Ojha, and A. Mondal, "Game-theoretic topology control for opportunistic localization in sparse underwater sensor networks," *IEEE Transactions on Mobile Computing*, vol. 14, no. 5, pp. 990–1003, 2015.

[13] R. Tynan, C. Muldoon, G. O'Hare, and M. O'Grady, "Coordinated intelligent power management and the heterogeneous sensing coverage problem," *Computer Journal*, vol. 54, no. 3, pp. 490–502, 2011.

[14] Q. Qi, J. Liao, and Y. Cao, "Cloud service-aware location update in mobile cloud computing," *IET Communications*, vol. 8, no. 8, pp. 1417–1424, 2014.

[15] S. Nizamani and A. Kumari, "A quality-aware computational cloud service for computational modellers," *Communications in Computer and Information Science*, vol. 414, pp. 184–194, 2014.

[16] L. Chunlin and L. LaYuan, "Cost and energy aware service provisioning for mobile client in cloud computing environment," *The Journal of Supercomputing*, vol. 71, no. 4, pp. 1196–1223, 2015.

[17] S. S. Savas, F. Dikbiyik, M. F. Habib, M. Tornatore, and B. Mukherjee, "Disaster-aware service provisioning with manycasting in cloud networks," *Photonic Network Communications*, vol. 28, no. 2, pp. 123–134, 2014.

[18] W. Dou, X. Zhang, J. Liu, and J. Chen, "HireSome-II: towards privacy-aware cross-cloud service composition for big data applications," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 2, pp. 455–466, 2015.

[19] B. Wang, J. Qu, X. Wang, G. Wang, and M. Kitsuregawa, "VGQ-Vor: extending virtual grid quadtree with Voronoi diagram for mobile *k* nearest neighbor queries over mobile objects," *Frontiers of Computer Science*, vol. 7, no. 1, pp. 44–54, 2013.

Hindawi Publishing Corporation
Journal of Electrical and Computer Engineering
Volume 2015, Article ID 641062, 7 pages
http://dx.doi.org/10.1155/2015/641062

Research Article

Cooperative Optimization QoS Cloud Routing Protocol Based on Bacterial Opportunistic Foraging and Chemotaxis Perception for Mobile Internet

Shujuan Wang,[1] Long He,[2] and Guiru Cheng[1]

[1] *Changchun University of Technology, Changchun 130012, China*
[2] *Sinopharm A-THINK Pharmaceutical Co. Ltd., Changchun 130012, China*

Correspondence should be addressed to Guiru Cheng; chengguiru@ccut.edu.cn

Received 29 September 2015; Accepted 29 November 2015

Academic Editor: James Nightingale

In order to strengthen the mobile Internet mobility management and cloud platform resources utilization, optimizing the cloud routing efficiency is established, based on opportunistic bacterial foraging bionics, and puts forward a chemotaxis perception of collaborative optimization QoS (Quality of Services) cloud routing mechanism. The cloud routing mechanism is based on bacterial opportunity to feed and bacterial motility and to establish the data transmission and forwarding of the bacterial population behavior characteristics. This mechanism is based on the characteristics of drug resistance of bacteria and the structure of the field, and through many iterations of the individual behavior and population behavior the bacteria can be spread to the food gathering area with a certain probability. Finally, QoS cloud routing path would be selected and optimized based on bacterial bionic optimization and hedge mapping relationship between mobile Internet node and bacterial population evolution iterations. Experimental results show that, compared with the standard dynamic routing schemes, the proposed scheme has shorter transmission delay, lower packet error ratio, QoS cloud routing loading, and QoS cloud route request overhead.

1. Introduction

Research of QoS (Quality of Services) guarantees that routing protocol has been the focus issue in mobile Internet [1]. In particular, some research staff has combined the bionics research results of chemotaxis system [2, 3] and bacterial foraging optimization [4] with the above problem. However, how to improve the QoS routing protocol effect of mobile Internet and Cloud platform has not been in-depth study.

On the one hand, Szymanski [5] presented a Constrained Multicommodity Maximum-Flow-Minimum-Cost routing algorithm, which could compute the maximum-flow routings for all smooth unicast traffic demands within the Capacity Region of a network subject to routing cost constraints. Berger et al. [6] proposed the interdomain traffic routing for dealing with the generalization of the shortest path and path-trading problem. A model of Cascading Failures on the Interdomain Routing System was proposed by Liu et al. [7]. The Virtual Internet Routing Lab platform was shown in article [8], which was used in the training, education, or research of Internet service. A delay-guaranteed energy profile-aware routing (DEAR) algorithm proposed for a green Internet was proposed to resolve the limitations of energy profiles and delay guarantees in article [9].

On the other hand, a variant of the bacterial foraging optimization algorithm was proposed by Tan et al. [10], which included the time-varying chemotaxis step length and comprehensive learning strategy. Naveen et al. [11] formulated the network reconfiguration problem as nonlinear objective optimization problem. A new long term scheduling for optimal allocation of capacitor bank in radial distribution system was presented by Devabalaji et al. [12], which can minimize the system power loss to resolve the equality constraints. The quantitative behavior of one-dimensional classical solutions for a hyperbolic-parabolic system describing repulsive chemotaxis was investigated in article [13]. Based on the knowledge of fuzzy mathematics, microeconomics, and swarm intelligence, the author proposed the flexible QoS unicast routing scheme with QoS satisfaction degree and utility introduced [14]. The novel bacterial chemotaxis optimization method (BCO) to QoS multicast routing scheme was proposed by Yong [15]. Chen et al. [16] proposed the bacterial colony optimization algorithm based on bacterial chemotaxis.

On the basis of the above research results, we studied the bacterial foraging model of mobile Internet and presented the chemotaxis perception cooperative routing protocol. Finally, the QoS cloud routing protocol for mobile Internet was proposed.

The rest of the paper is organized as follows. Section 2 describes the bacterial foraging model. In Section 3, we design the chemotaxis perception cooperative routing protocol. In Section 4, we proposed the QoS cloud routing protocol for mobile Internet. Experiment results are given in Section 5. Finally, we conclude the paper in Section 6.

2. Bacterial Foraging Model for Internet

In the mobile Internet, we set each mobile node as a bacterium. The behavior of mobile nodes searching for the next hop data receiving nodes is defined as bacterial foraging. Mobile nodes movement process is defined as a bacterial motility. When k bacterium is moving, the target was searching in the subnetwork of current bacteria. Set the distance to the d_{bk}. Then, the different directions of the current crawling angle as α were chosen. When k bacterium is in the direction of the movement direction of the best individual in the subnetwork, the attraction force of the bacteria b_{af} is determined by the following formula:

$$d_{bk} = \frac{t_k \sqrt{v_k \sin\alpha}}{\tan\beta},$$
$$b_{\mathrm{af}} = \left[b_p - \sum_{i=1}^{k} \frac{\sin\beta}{d_{bk}} \right]^{-\cos\alpha}. \quad (1)$$

Here, t_k denotes the peristalsis delay of k bacteria. v_k denotes the peristalsis speed of k bacteria. b_p denotes the overall dynamics of bacterial populations. β is the angle of antenna and signal direction.

The mobile Internet is a multidimensional space field of the bacteria. Data transmission route selection problem is decomposed into the problem of the bacteria on the multidimensional field. The food delivery process may be accomplished by a number of bacteria in the same field, and it may be completed by a new structural field S_F, which is derived from the bacterial assemblage from different fields, as defined in

$$V_{S_F} = [v_1, v_2, \ldots, v_n],$$
$$S_F = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} V_{S_F}. \quad (2)$$

Here, V_{S_F} is the bacterial movement velocity in the structure field. v is the mobile speed of node. b is the amount of food carried by the bacteria. The amount of information transmitted is denoted by S_F for the mobile communication of the structure field.

The dynamic changes of the multifield and the structure of the bacteria in the process of the movement and feeding of the bacteria are shown in Figure 1. There is bacteria foraging in multidimensional field, when the same field can not meet the needs of food delivery and other fields through mapping to establish the route, thus generating the structure, and there are various fields to select the best bacteria to join the structure of the field. By this mapping mechanism, it can prevent the local best bacterial ability to limit the population creep performance but also can shorten the time of route update and improve the convergence precision of the searching. The mapping results could be connected with the Internet by combining with the wireless cooperative networks.

Among them, from the dynamic evolution of the multidimensional field to the structural field, the principle of bacterial opportunity to feed is used. In Figure 3, three multidimensional fields and two structural fields are given. When the bacteria foraging from M_{F1} and M_{F2} to M_{F3} could not satisfy the transmission requirements, the opportunity structure field would be constructed based on the optimal direction and speed of the multidimensional field. Therefore, there is the different bacteria structure, the food principle, and the food amount. For example, S_{F1} is composed of good bacteria from M_{F1} and M_{F2}. The food principle is the XOR operation between M_{F1} and M_{F2}. The food amount is the sum of the food scales of M_{F1} and M_{F2}. S_{F2} would be obtained by the fusion of S_{F1} and M_{F1}, M_{F2}, and M_{F3}. The bacteria in the field are dynamic, which could be selected and optimized according to the requirements of food and information transmission requirements in the multidimensional field. The structural parameters of the above structures are shown in the following formula:

$$b_k^t = b_k(\alpha, \beta, b_k),$$
$$S_{F1} = \mathrm{sub}(M_{F1}) \cup \mathrm{sub}(M_{F2}) \cup \mathrm{sub}(M_{F3}) \sum_{I=1}^{K} b_k^t, \quad (3)$$
$$S_{F2} = \delta(M_{F1} \cap M_{F2} \cap M_{F3}) \cup \mathrm{sub}(M_F) \cup S_{F1}.$$

Here, let b_k^t denote transfer peristalsis from the multidimensional field to the structural field of k bacteria. δ is opportunity foraging factor of bacteria, which is determined by the bacteria in real time and the attraction force.

Hence, the bacteria opportunity foraging algorithm in the mobile Internet described is as follows.

Algorithm 1 (BOF (d_{bk}, b_{af}, α, β, δ)).

BOF function Initialization of each Bacteria
α and β to d_{bk}
b_p to b_{af}
Searching for optimal bacteria in M_F

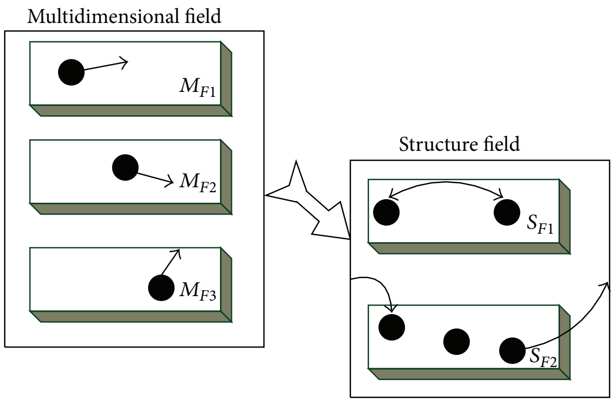


Figure 1: Multidimensional field and structure field of bacterial opportunistic foraging.

For $i = 1$ to k

{

Statistics V_{S_F};

Compute S_F;

$S_{F1} \leftarrow \mathrm{sub}(M_{F1}) \cup \mathrm{sub}(M_{F2}) \cup \mathrm{sub}(M_{F3}) \sum_{I=1}^{K} b_k^t$;

$S_{F2} \leftarrow \delta(M_{F1} \cap M_{F2} \cap M_{F3}) \cup \mathrm{sub}(M_F) \cup S_{F1}\}$

return S_F

3. Chemotaxis Perception Cooperative Routing Protocol

In the process of bacterial foraging, the direction of bacterial motility and the structure of the field are characterized by chemotaxis. All the bacteria in the field can creep along the same direction and can also be spread to the food in the field of structural field after the food is carried and dispersed. The position of a trend of drug resistance is shown in the following formula:

$$b_k^{i+1} = b_k^i \left(\alpha, \beta, b_f\right) + \frac{\sum_{i=1}^{L} S_{F_i}}{\sqrt{\Delta S_F}}. \tag{4}$$

Here, ΔS_F is the update information of the structure field.

On the basis of chemotaxis perception, the behavior of the bacterial foraging population has the characteristics of dormancy, activation, and direction. When bacteria are crawling, the bacteria are perceived to be more resistant and opportunistically release the lure of the bacteria to the neighbor and move along the structural field for the next round. When the bacteria in the active state are all completed by the lure and the movement, the whole structure field becomes an active field, and the relationship between the bacteria and the colony behavior of the two groups is shown in the following formula:

$$\begin{aligned} B_{S_F}\left(b_k, n, m\right) &= \sum_{i=1}^{n} b_k^m B_{S_F}^i \left(b_k^i, i, m\right), \\ B_{S_F} &= \sum_{j=1}^{m} \left[-\gamma \exp\left(\cos\left(\left|\alpha - \beta\right|\right)^2\right)\right]^{m-j}. \end{aligned} \tag{5}$$

Here, B_{S_F} denotes the bacterial population behavior of structure field. n is bacterial number of structure fields. m is the number of neighbors who have become more resistant to the bacteria. With the bacterial motility and structural field of the update, the attraction of the drug is gradually enhanced. Bacterial colony behavior becomes gradually mature. The position and direction of the bacteria are unified, as shown in the following formula:

$$\begin{aligned} B_{S_F}^{i+1}\left(b_k, i+1, m\right) &= \cos\left(\left|\alpha - \beta\right|\right)^2 B_{S_F}^i \left(b_k^i, i, m\right), \\ b_k^{i+1} &= b_k^i \left(\sin\alpha, \arctan\beta, \gamma\right). \end{aligned} \tag{6}$$

After multiple iterations of the search, the driving of the chemotaxis and the total number of bacteria in the food are shown in the following formula:

$$T_{\mathrm{CH}}\left(B_{S_F}^n\right) = \sum_{i=1}^{n} b_p^i \exp\left(\sin^2\alpha - \arctan^2\beta\right). \tag{7}$$

Here, T_{CH} denotes the total foraging amount of bacteria population. In order to make full use of the characteristics of the drug resistance to the bacteria, the mobile node route is established, and the angle parameters of the bacterial motility and the attraction force are calculated.

At the same time, in order to speed up the accumulation of bacteria and route establishment, the bacterial community would be sorted linearly and opportunistically according to the food principle and individual food amount. The bacteria with good performance would be produced through the drug resistance drive. In order to simplify the algorithm, in order to maintain the vitality of the population, the bacteria and the bacteria in the dormant state are consistent with the direction of the movement of the drug.

In the actual Internet communication environment, the transmission data of mobile nodes is affected by external forces or their resources are limited, and the process of routing communication based on bacterial chemotaxis may be destroyed. But the bacterial population is helping to find a more abundant area of food. Therefore, in view of the end-to-end communication, the mechanism of the drug delivery mechanism based on the bacterial cooperative control is beneficial to the bacterial foraging and population stability. In the cooperative routing algorithm, the individual behavior and the behavior of the bacteria are repeated after many iterations, and the bacteria can be a certain probability to the recombinant population or the structure field and will be the advantage of the spread of the food gathering area. The mobile node information fusion scheme can not only protect

Table 1: Internet settings.

Parameter	Value	Parameter	Value
Simulation time	2500 s	Perceived distance of mobile node	200 m
Internet area	100 m ∗ 1200 m	Number of maximum active nodes	30
Concurrent service number	5	Packet size	1200 bytes
Cloud number	8	Packet sending speed	3 packets/s
Disk size of cloud	1 Tbytes	CPU number of cloud	4
Kernel operating frequency	2 GHz	Memory size of cloud	4 Gbytes

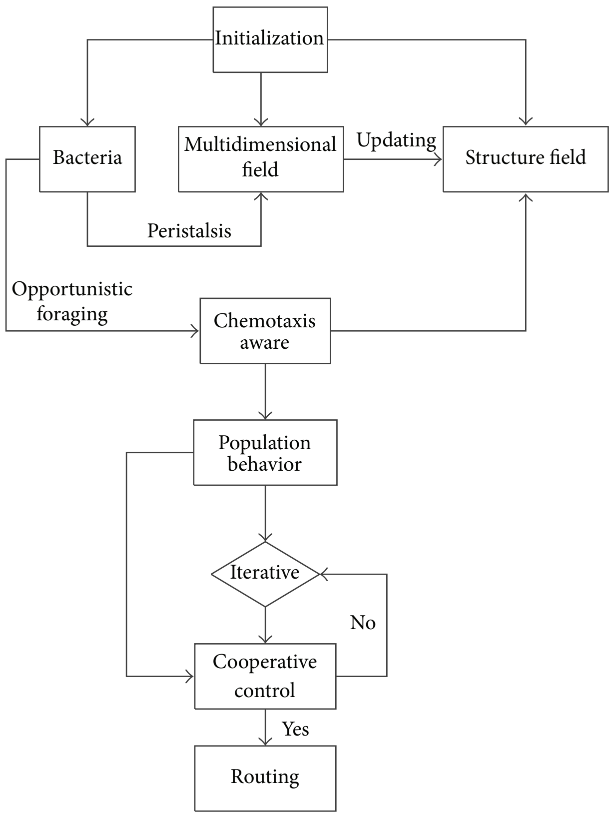


Figure 2: Route establishment based on bacterial opportunity to search for food and drug delivery.

the bacteria from the local optimal value but also enhance the global routing performance.

In summary, the establishment of a route based on bacterial opportunity foraging and the route of drug delivery is described in Figure 2.

4. QoS Cloud Routing Protocol for Mobile Internet

In the mobile Internet to join the cloud platform, the mobile node's bacterial opportunity foraging behavior can express the communication distance between the data sending node and the receiving node. Bacterial chemotaxis of mobile nodes can reflect the quality and stability of the communication link between the Internet and the cloud platform. In the actual communication environment, by detecting the degree of attenuation of the bacterial foraging ability of the mobile nodes, the population behavior of the mobile node is analyzed, such as the direction and speed of the short time of the mobile node, and the probability of the interruption of the link is caused by the structural field update. In the cloud routing mechanism of QoS, the opportunistic optimization of the bacterial population is formed, which will lead to a large loss of multipath gain between the Internet and the cloud platform. Therefore, QoS cloud routing bacterial bionic optimization is based on multipath by establishing mechanism through hedge mapping between iterations in mobile Internet node physical layer of the antenna array and bacterial population evolution. QoS cloud routing path was selected and optimized.

The signal in the end-to-end communication with the bacterial opportunity foraging is shown in the following formula:

$$\mathrm{Sig}_b = \sqrt{h_t^2 h_r^2 \frac{\sin^{\sqrt{G_t G_r}} \alpha}{b_{dk}^n}}. \tag{8}$$

Here, G_t denotes antenna gain of sending node. G_r denotes antenna gain of receiving node. h_t and h_r denote the antenna height of the transmitter and the receiving nodes, respectively. If b_{dk}^n is larger than 1, the bacterial opportunity is not related to the attraction force of the population. In the QoS cloud routing request message in the bacterial population in the process, the activation of the neighbor bacteria opportunity type according to the movement speed and load information of the opportunity to calculate the cloud path of the opportunity weight W is shown in the following formula:

$$w_k = \prod_{k \in S_F} \left(\delta \frac{\rho - b_{dk}}{\rho} \right) \tan \beta. \tag{9}$$

Here, the evolution of the bacteria will occur, which belong to the same field. ρ denotes the peristalsis bacteria foraging radius of node.

As shown in Figure 3, bacteria nodes sent out a signal of the Internet data transmitting. After the opportunity for foraging peristalsis two multidimensional cloud paths are formed in the cloud platform. By formula (9), the opportunity weights of two multidimensional field cloud paths are $w1$ and $w2$. Here, $w1$ is less than $w2$. Although number of active bacteria nodes of dotted cloud routing path is less than one of the solid cloud path, the solid cloud path has more attractive

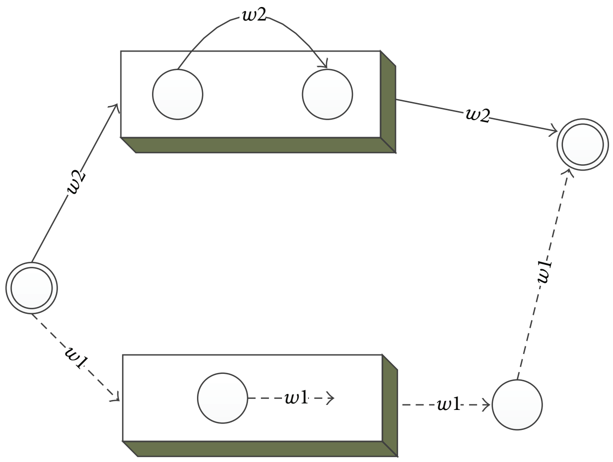


Figure 3: QoS cloud path selection.

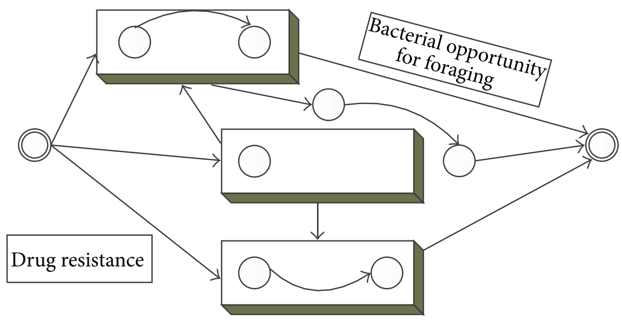


Figure 4: Process of QoS cloud route discovery.

force. In order to avoid breaking the path of the cloud, the current QoS cloud path should be the solid path.

QoS cloud path of cooperative maintenance process is shown in Figure 4. The RREP Format in DSR (Dynamic Source Routing) Options Header is shown in Figure 5. The bacterial neighbor node receives the request message from the data sending end node to the cloud path request message, and the bacteria spread the cloud path request to the bacteria population in the same multidimensional field through the opportunistic feed. There are multiple paths of mobile nodes in the bacterial population through collaborative construction, to maintain the cloud path through the computer.

In order to eliminate bacterial opportunity to feed into the dead cycle, the cooperative maintenance of the QoS cloud path requires the maintenance of the individual in the end to end cloud path of the chemotaxis and motility of the bacterial node sequence, to maintain its monotonic increasing characteristics, as shown in the following formula:

$$\left(-S_{F_{i+1}}, B_{S_F}^{i}\left(b_k^{i}, i, m\right)\right) > \left(S_{F_i}, B_{S_F}^{i-1}\left(b_k^{i-1}, i-1, m\right)\right)$$
$$R_{bd} = \sum_{i=1}^{L} S_{F_i} \frac{\cos\alpha}{\sqrt{\Delta S_F}}. \tag{10}$$

5. Performance Evaluation

In this section, we analyzed and evaluated the performance of the proposed COQCRP (Cooperative Optimization QoS Cloud Routing Protocol based on bacterial opportunistic foraging and chemotaxis perception) routing protocol. Two groups of simulation experiments were designed. The DRP (Dynamic Routing Protocol) with the Internet was compared with the proposed protocol, respectively, based on the mobile speed of the nodes and the size of the Internet in the case of the QoS performance of the Internet. Table 1 records the Internet parameter settings.

In order to evaluate the performance of COQCRP, four performance evaluation criteria of QoS cloud routing are statistically analyzed.

End-to-End QoS Cloud Communication Delay. It is the time interval of data packets transmitted from the Internet node to destination node through cloud.

QoS Cloud Route Load. It is the control packet number of QoS routing cloud cooperative optimizations divided by the number of correctly received data packets.

QoS Cloud Routing Overhead. It is the number of COQCRP cloud routing requests sent or forwarded in the unit time.

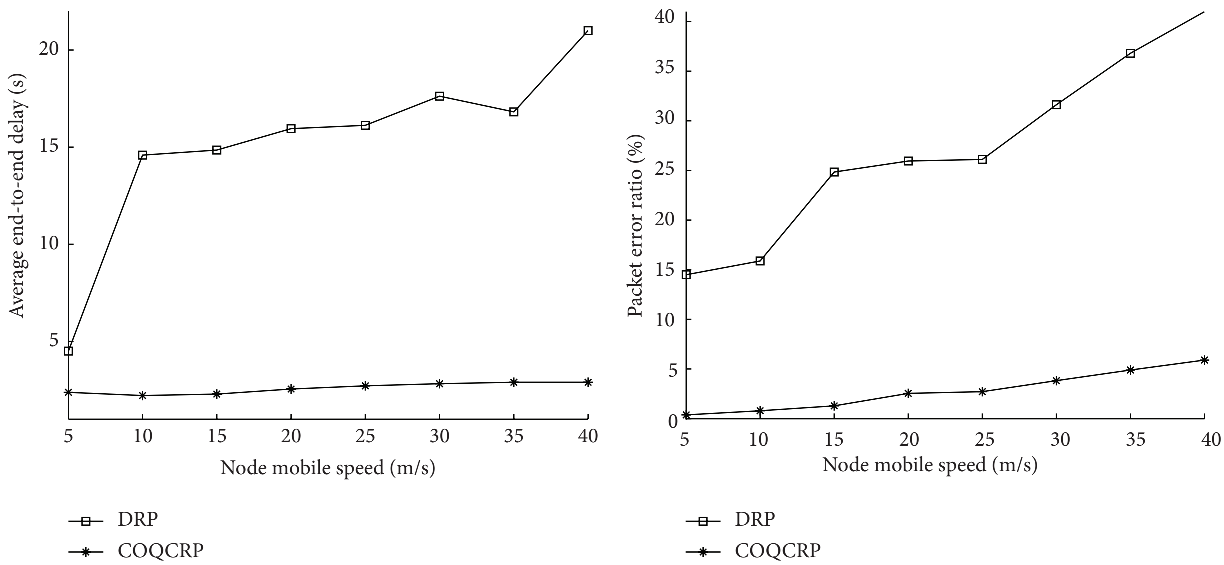
In Experiment 1, the analysis of the Internet based on the mobile speed of the mobile node of the mobile speed of the QoS cloud routing performance, the results are shown in Figure 6. The meaning of coordinates x and y in Figure 6(a) is mobile speed of node and average end-to-end delay. The meaning of coordinates x and y in Figure 6(b) is mobile speed of node and packet error ratio.

The COQCRP significantly improves the end-to-end cloud communication delay, as shown in Figure 6(a). The reason is that, based on QoS cloud, opportunistic bacterial foraging by selection mechanism improves the robustness of the path of the cloud and relationship of cloud routing size for the maintenance delay. COQCRP controlled the message broadcast of cloud path request cooperatively, which improved the Internet link bandwidth utilization and avoided the cloud computing error, as well as the data packet collision probability. With the increase of the node moving speed, the performance of COQCRP and DRP is similar. This is due to the probability of packet collision and the speed of the node of the Internet is proportional to the speed of the node. In Figure 6(b), the simulation results show the effect of the change of node moving speed on the packet loss rate. With the increase of the node moving speed, the path break frequency increases and the packet loss rate increases. COQCRP data transmission path quality is improved. COQCRP can provide an effective guarantee for data packet transmission and packet error ratio is lower than DRP.

In Experiment 2, the analysis of the impact of the scale of the Internet on the QoS cloud routing performance is shown in Figure 7. The meaning of coordinates x and y in Figure 7(a) is cloud scale and routing load. The meaning of coordinates x and y in Figure 7(b) is cloud scale and route request overhead.

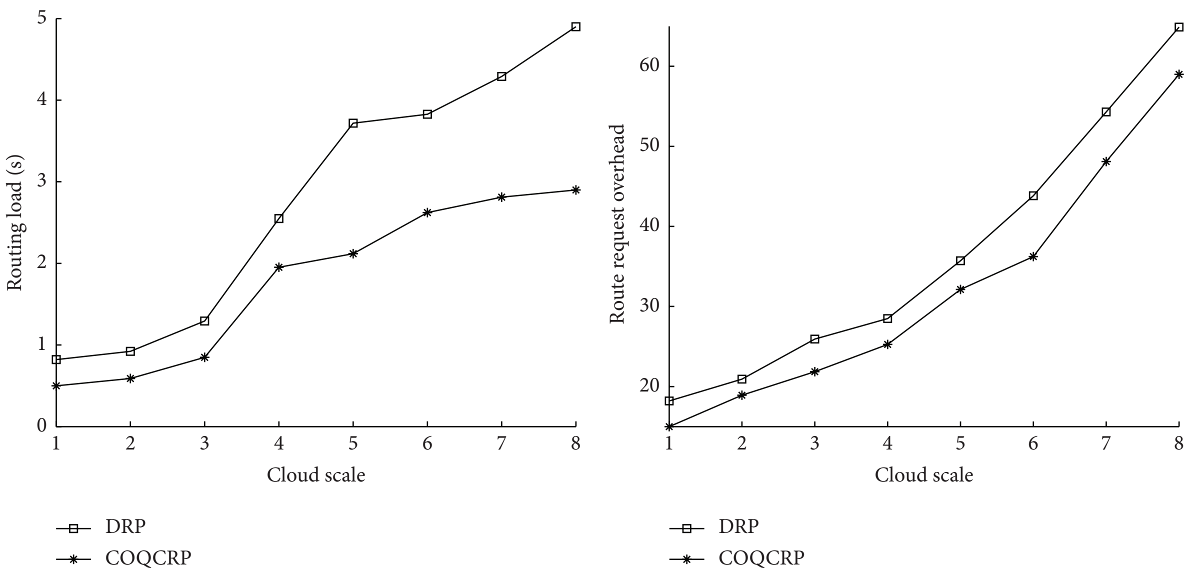
32 bits			
Opportunity foraging weight	Chemotaxis perception	Length of multidimensional field	Length of structure field
Address of sending node			
Cooperative nodes (4–10 bytes)			
Address of receiving node			

Figure 5: The RREP format in DSR options header.



(a) Average end-to-end delay (b) Packet error ratio

Figure 6: Performance analysis with speed.



(a) Routing load (b) Route request overhead

Figure 7: Performance analysis with cloud scale.

Due to the extension of the path lifetime and COQCRP QoS cloud routing load (Figure 7(a)) and cloud routing request overhead (Figure 7(b)) that significantly reduced, COQCRP effectively saved the Internet resources and improved the work efficiency of the cloud platform.

6. Conclusions

According to the mobility management requirements of mobile Internet and cloud platform resource constrained characteristics, a chemotaxis perception cooperative QoS cloud routing optimization mechanism was proposed based on the opportunistic bacterial foraging bionics. First, we assume that each mobile node of the mobile Internet is a bacterium. A data forwarding node group was established through the progress of the bacterial opportunistic feed and the characteristic of bacterial motility. Secondly, based on the analysis of the direction of the movement of the bacteria and the characteristics of the trend of the structural field, the establishment of the mechanism of the drug resistance is established by the distributed load control of all the bacteria in the field. Finally, the cloud platform joined the mobile Internet. We combined the mobile node opportunistic bacterium foraging behavior and bacterial chemotaxis. A cooperative optimization QoS cloud routing mechanism was proposed to guarantee the quality and stability of the communication link between the Internet and cloud platform. The experimental results show that the proposed mechanism has the advantages of real-time, reliability, routing load, and resource utilization of cloud platform in QoS communication.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] M. S. Siddiqui, D. Montero, R. Serral-Gracià, X. Masip-Bruin, and M. Yannuzzi, "A survey on the recent efforts of the internet standardization body for securing inter-domain routing," *Computer Networks*, vol. 80, pp. 1–26, 2015.

[2] Q. S. Zhang and Y. X. Li, "Global existence and asymptotic properties of the solution to a two-species chemotaxis system," *Journal of Mathematical Analysis and Applications*, vol. 418, no. 1, pp. 47–63, 2014.

[3] C.-H. Huang and P. A. Iglesias, "Cell memory and adaptation in chemotaxis," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 111, no. 43, pp. 15287–15288, 2014.

[4] I. A. Mohamed and M. Kowsalya, "Optimal size and siting of multiple distributed generators in distribution system using bacterial foraging optimization," *Swarm and Evolutionary Computation*, vol. 15, pp. 58–65, 2014.

[5] T. H. Szymanski, "Max-flow min-cost routing in a future-internet with improved QoS guarantees," *IEEE Transactions on Communications*, vol. 61, no. 4, pp. 1485–1497, 2013.

[6] A. Berger, H. Roglin, and R. van der Zwaan, "Internet routing between autonomous systems: fast algorithms for path trading," *Discrete Applied Mathematics*, vol. 185, pp. 8–17, 2015.

[7] Y. J. Liu, W. Peng, J. S. Su, and Z. Wang, "Assessing the impact of cascading failures on the interdomain routing system of the internet," *New Generation Computing*, vol. 32, no. 3-4, pp. 237–255, 2014.

[8] J. Obstfeld, S. Knight, E. Kern, Q. S. Wang, T. Bryan, and D. Bourque, "VIRL: the virtual internet routing lab," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 4, pp. 577–578, 2014.

[9] E.-J. Lee, Y.-M. Kim, and H.-S. Park, "DEAR: delay-guaranteed energy profile-aware routing toward the green internet," *IEEE Communications Letters*, vol. 18, no. 11, pp. 1943–1946, 2014.

[10] L. J. Tan, F. Y. Lin, and H. Wang, "Adaptive comprehensive learning bacterial foraging optimization and its application on vehicle routing problem with time windows," *Neurocomputing*, vol. 151, no. 3, pp. 1208–1215, 2015.

[11] S. Naveen, K. S. Kumar, and K. Rajalakshmi, "Distribution system reconfiguration for loss minimization using modified bacterial foraging optimization algorithm," *International Journal of Electrical Power and Energy Systems*, vol. 69, pp. 90–97, 2015.

[12] K. R. Devabalaji, K. Ravi, and D. P. Kothari, "Optimal location and sizing of capacitor placement in radial distribution system using bacterial foraging optimization algorithm," *International Journal of Electrical Power & Energy Systems*, vol. 71, pp. 383–390, 2015.

[13] D. Li, R. Pan, and K. Zhao, "Quantitative decay of a one-dimensional hybrid chemotaxis model with large data," *Nonlinearity*, vol. 28, no. 7, pp. 2181–2210, 2015.

[14] X. Wang, D. Jiang, and M. Huang, *Flexible QoS Unicast Routing Scheme Based on Utility and BCC [EB/OL]*, Sciencepaper Online, Beijing, China, 2010.

[15] H. Yong, "QOS multicast routing algorithm based on bacterial chemotaxis optimisation," *Computer Applications and Software*, vol. 29, no. 11, pp. 269–271, 2012.

[16] L. Chen, L.-Y. Zhang, Y.-J. Guo, J.-Q. Zhao, and Q. Li, "Sequential blind signal separation algorithm based on bacterial colony chemotaxis," *Journal on Communications*, vol. 32, no. 4, pp. 77–85, 2011.

Hindawi Publishing Corporation
Journal of Electrical and Computer Engineering
Volume 2015, Article ID 246420, 8 pages
http://dx.doi.org/10.1155/2015/246420

Research Article

User Utility Oriented Queuing Model for Resource Allocation in Cloud Environment

Zhe Zhang and Ying Li

Institute of Software, Nanyang Normal University, Nanyang, Henan 473061, China

Correspondence should be addressed to Ying Li; lying1024@163.com

Received 20 August 2015; Revised 29 September 2015; Accepted 8 October 2015

Academic Editor: James Nightingale

Resource allocation is one of the most important research topics in servers. In the cloud environment, there are massive hardware resources of different kinds, and many kinds of services are usually run on virtual machines of the cloud server. In addition, cloud environment is commercialized, and economical factor should also be considered. In order to deal with commercialization and virtualization of cloud environment, we proposed a user utility oriented queuing model for task scheduling. Firstly, we modeled task scheduling in cloud environment as an $M/M/1$ queuing system. Secondly, we classified the utility into time utility and cost utility and built a linear programming model to maximize total utility for both of them. Finally, we proposed a utility oriented algorithm to maximize the total utility. Massive experiments validate the effectiveness of our proposed model.

1. Introduction

Providers of cloud services usually provide different computing resources with different performances and different prices, and the requirements of users for performance and cost of resources differ greatly too. So how to allocate available resources for users to maximize the total system utilization is one of the most important objectives for allocating resources and scheduling tasks [1] and is also a research focus in cloud computing.

Traditional resource allocation models mainly focus on the response or running time, saving energy of the whole system, and fairness of task scheduling and do not take user utility into consideration [2]. However, the utility of a user in cloud environment is the usage value of services or resources, and it describes how the user is satisfied with the proposed services or resources while occupying and using them [3, 4]. In order to maximize the total utility of all users in cloud environment, it is necessary to analyze and model user utility first and then optimize it to get a maximum [5]. The modeling of user utility is very complex, as it needs a formal description considering many factors, such as the processing time that tasks have passed by [6], the ratio of finished tasks [7], the costs of finished and unfinished tasks [8], and the parallel speedup [9].

In a cloud server, requests of users, called tasks, come randomly, and a good description of these tasks is the Poisson distribution assumption. At the same time, under the commercialization constraint of the cloud environment, the utility of cloud server becomes much more important. In this paper, we formalized and quantified the problem of task scheduling based on queuing theory, divided the utility into time utility and cost utility, and proposed a linear programming method to maximize the total utility. The contributions of the paper are as follows:

(i) We modeled task scheduling as $M/M/1$ queuing model and analyzed related features in this queuing model.

(ii) We classified utility into time utility and cost utility and built a linear programming method to maximize total utility for each of them.

(iii) We proposed a utility oriented and cost based scheduling algorithm to get the maximum utility.

(iv) We validated the effectiveness of the proposed model with massive experiments.

The rest of the paper is organized as follows. In Section 2, we review related works about resource allocation and task scheduling in cloud computing. In Section 3, we formalize the tasks in cloud environment based on the queuing theory, define a random task model for random tasks, describe our proposed user utility model, and design a utility oriented time-cost scheduling algorithm. Experiments and conclusion are given in Sections 4 and 5, respectively.

2. Related Works

In cluster systems that provide cloud services, there is a common agreement in researchers that the moments, when tasks come into the system, conform to the Poisson distribution, and both the intervals between two coming tasks and the serviced time of tasks are exponentially distributed. In this situation, heuristic task scheduling algorithms, such as genetic algorithm and ant colony algorithm, have better adaptability than traditional scheduling algorithms. However, the deficit of heuristic algorithms is that they have complex problem-solving process, so they can only be applied in small cluster systems. The monstrous infrastructure of cloud systems usually has many types of tasks, a huge amount of tasks, and many kinds of hardware resources, which makes heuristic algorithms unsuitable.

There are a lot of researches about resource allocation or task scheduling in cloud environment, especially for the MapReduce programming schema [10]. Cheng et al. [11] proposed an approximate algorithm to estimate the remaining time (time to end) of tasks in MapReduce environment and the algorithm scheduled tasks with their remaining time. Chen et al. [12] proposed a self-adaptive task scheduling algorithm, and this algorithm computed the running progress (ratio of time from beginning to total running time) of the current task on a node based on its historical data. The advantage of [12] is that it can compute remaining time of tasks dynamically and is more suitable to heterogeneous cloud environment than [11]. In addition, Moise et al. [13] designed a middleware data storage system to improve the performance and ability of fault tolerance.

Traditional task scheduling algorithms mainly focus on efficiency of the whole system. However, some researchers introduce economic models into task scheduling, and the basic idea is optimizing resource allocation by adjusting users' requirements and allocating resources upon price mechanism [14]. Xu et al. [15] proposed a Berger model based task scheduling algorithm. Considering actual commercialization and virtualization of cloud computing, the algorithm is based on the Berger social allocation model and adds additional cost constraints in optimization objective. According to experiments on the CloudSim platform, their algorithm is efficient and fair when running tasks of different users. In addition, with respect to the diversity of resources in cloud environment, more researchers believe that the diversity will increase as time goes on with update of hardware resources. In order to alleviate this phenomenon and ensure quality of services, Yeo and Lee [16] found that while the resources were independently identically distributed, dropping resources that needed three times the number of minimal response time could make the whole system use less total response time and thus less energy.

The study of random scheduling began in 1966, and Rothkopf [17] proposed a greedy optimal algorithm based on the weights of tasks and expected ratios of finished time to total time. If all tasks had the same weights, then this algorithm became the shortest expected processing time algorithm. Möhring et al. [18] proved the optimal approximation for scheduling tasks with random finished time. They began with the relaxation of linear programming, studied the problem of integer linear programming for systems with homogeneous tasks, and got an approximate solution with the lower limit of the linear programming. Based on the above research, Megow et al. [19] proposed a better optimal solution with better approximation. In addition, Scharbrodt et al. [20] studied how to schedule independent tasks randomly. They analyzed the problem of scheduling n tasks on m machines randomly and gave the worst performance of random scheduling under homogeneous environment theoretically, and their result was the best among related works.

All of the above algorithms focus on the response or running time of users' requirements, saving energy of the whole system and fairness of tasks, and do not take user utility into consideration. However, utility of users is very important in cloud service systems. In order to maximize the total utility of all users in cloud environment, we analyze and model user utility first and then optimize it to get a maximal solution.

In addition, Nan et al. [21] studied how to optimize resource allocation for multimedia cloud based on queuing model, and their aim is to minimize the response time and the resource cost. However, in this paper, we deal with commercialization and virtualization of cloud environment, and our aim is maximizing utility. Xiao et al. [22] presented a system that used virtualization technology to allocate data center resources dynamically. Their aim is to minimize the number of servers in use considering the application demands and utility, whereas in this paper we aim to maximize the system's total utility under a certain cloud environment.

3. Proposed Model

3.1. Queuing Model of Tasks. In this paper, we describe randomness of tasks with the $M/M/1$ model of queuing theory, and the model is illustrated in Figure 1. The model consists of one server, several schedulers, and several computing resources. When user tasks are submitted, the server analyzes and schedules them to different schedulers and adds them to local task queue of the corresponding scheduler. Finally, each scheduler schedules its local tasks to available computing resources. In Figure 1, $t(d)$ is the waiting time of a task in the queue and $t(e)$ is the running time.

3.2. Modeling Random Tasks. In the following, we will analyze the waiting time, running time, and queue length of the proposed $M/M/1$ model.

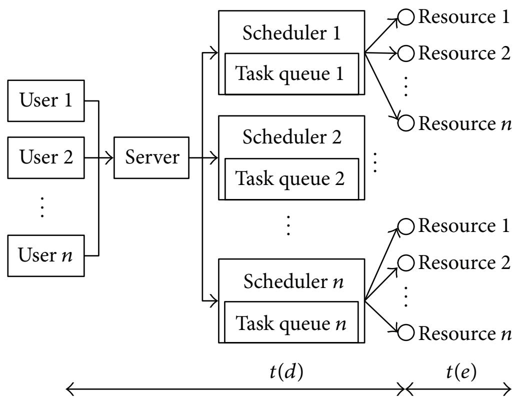


Figure 1: Scheduling model of user tasks in cloud environment.

Definition 1. If the average arrival rate of tasks in a scheduler is λ, the average service rate of tasks in a scheduler is μ, and then the service intensity ρ is

$$\rho = \frac{\lambda}{\mu}. \tag{1}$$

The service intensity describes the busyness of the scheduler. When ρ approaches zero, the waiting time of tasks is short, and the scheduler has much idle time; when ρ approaches one, the scheduler has less idle time, and thus tasks would have long waiting time. Generally speaking, the average arrival rate should be equal to or smaller than the average service rate, otherwise there will be more and more waiting tasks in the scheduler.

Definition 2. If we denote the expected length of tasks in a scheduler as L, the expected length of tasks in queuing as L_q, the expected total time (including both waiting time and running time) of a task as W, and the expected waiting time of a task in queuing as W_q, then we have the following equations according to queuing theory [17]:

$$\begin{aligned} L &= \frac{\lambda}{(\mu - \lambda)} = \rho(1-\rho) \\ L_q &= \frac{\lambda^2}{\mu(\mu - \lambda)} = \rho^2(1-\rho) = L \cdot \rho \\ W &= \frac{1}{(\mu - \lambda)} \\ W_q &= \frac{\lambda}{\mu(\mu - \lambda)} = W \cdot \rho. \end{aligned} \tag{2}$$

In addition, let $P_n = P\{N = n\}$ be the possibility of number of tasks in a scheduler at any moment; then, we have the following equation:

$$P_n = \rho(1-\rho). \tag{3}$$

If $n = 0$, then P_0 is the possibility that all virtual machines are idle.

3.3. Model of User Utility

3.3.1. Time Utility of Tasks. As we can see from Figure 1, the total time that a user takes from submitting a request to getting the result includes both waiting time $t(d)$ and running time $t(e)$. Here, the computing resources are virtual resources managed by virtual machines. Let T be total time; then, we have

$$T = t(d) + t(e). \tag{4}$$

In (4), the running time $t(e)$ is the sum of used time $t(f)$ and remaining time $t(u)$; that is,

$$t(e) = t(f) + t(u). \tag{5}$$

In order to calculate the time requirement of a task, the system needs to calculate the remaining time $t(u)$ and schedules $t(u)$ for different tasks to different virtual machines.

For analyzing the remaining time, we classified tasks into set $P = \{p_i \mid 1 \le i \le m\}$ and nodes into set $V = \{p_j \mid 1 \le j \le n\}$. According to statistical computing, we can get the average executing rate of task p_i on node v_j; that is, $R = \{r_{i,j} \mid 1 \le i \le m,\ 1 \le j \le n\}$, and then the remaining time of p_i on v_j is

$$t(u)_{i,j} = \frac{(w - w_u)}{r_{i,j}}, \tag{6}$$

where w is the number of total tasks and w_u is the number of finished tasks. For computing intensive tasks, w is the total input data and w_u is the already processed input data. Schedulers schedule tasks on virtual machine resources according to their remaining time and assure all tasks are finished on time.

A task can be executed either on one virtual machine or on n virtual machines in parallel, while being divided into m subtasks. We denoted the subtask set as $D = \{d_k \mid 1 \le k \le m\}$. While these subtasks are executed on different virtual machines, especially different physical nodes, the communication cost increases, and we use speedup to measure the parallel performance

$$s = \frac{T_1}{T_p}, \tag{7}$$

where T_1 is the time of a task in one node and T_p is the time of a task in p nodes. In order to make sure $s \le S_0$, all subtasks run in parallel, and total time of the task is

$$T = t(d) + \max\{t(e)_{k,j}\}, \tag{8}$$

where $t(e)_{k,j}$ is the time of subtask d_k on v_j and $\max\{t(e)_{k,j}\}$ is the maximal time of all subtasks.

3.3.2. Cost Utility of Tasks. In this paper, we assume that the cost rate of nodes is proportional to CPU and I/O speed, and tasks of different types consume different energy, different bandwidth, and different resource usage. So different tasks will have different cost rates.

Definition 3. Let $C = (c_{i,j} \mid 1 \leq i \leq m,\ 1 \leq j \leq n)$ be the cost matrix of task p_i on node c_j, and then total cost of a task is the product of node cost and running time; that is,

$$M = C \times T = \sum_{i=1}^{m} \sum_{j=1}^{n} \left(c_{i,j} \times t(e)_{i,k,j} \right), \quad (9)$$

where $c_{i,j}$ is the unit cost of task p_i on node v_j and $t(e)_{i,k,j}$ is the time of subtask d_k on node v_j.

3.3.3. Formalization and Optimization of User Utility

Definition 4. Let the time utility function be U_t and let the cost utility function be U_c; then, the total utility is

$$U = a \times U_t + b \times U_c, \quad (10)$$

where $a + b = 1$, $0 \leq a \leq 1$ and $0 \leq b \leq 1$.

In (10), both time utility and cost utility are between 0 and 1 and a and b are the weights of time utility and cost utility, respectively.

The aim of utility oriented task scheduling is to maximize the total utility, and the constraints are expected time of tasks, expected cost, finished rate, speedup, and so on. In this paper, we classify user tasks into time sensitive and cost sensitive.

For time sensitive user tasks, change of running time for a task will affect the time utility a lot, and its definition is as follows.

Definition 5. The utility model of time sensitive user tasks is defined by the following equations:

$$U = a \times U_t + b \times U_c,$$
$$U_t = \frac{k}{(\ln(t - a) \times b)}, \quad (11)$$
$$U_c = a \times c + b.$$

The constrains are

$$F(D) = 1, \quad (12)$$

$$0 \leq UT < U_t \leq 1, \quad (13)$$

$$0 \leq UC < U_c \leq 1, \quad (14)$$

$$t(d) + t(e) < T_0, \quad (15)$$

$$\frac{L_q}{\lambda} < T_1, \quad (16)$$

$$\max\left\{t(e)_{k,j}\right\} < T_2, \quad (17)$$

$$C \times T < M_0, \quad (18)$$

$$s > S_0, \quad (19)$$

where D is the set of subtasks for all tasks, and the aim is to maximize total utility U.

For cost sensitive user tasks, change of running cost for a task will affect the cost utility a lot, and its definition is as follows.

Definition 6. The utility model of cost sensitive user tasks is defined by the following equations:

$$U = a \times U_t + b \times U_c,$$
$$U_c = \frac{k}{(\ln(c - a) \times b)}, \quad (20)$$
$$U_t = a \times t + b.$$

The constrains are

$$F(D) = 1, \quad (21)$$

$$0 \leq UT < U_t \leq 1, \quad (22)$$

$$0 \leq UC < U_c \leq 1, \quad (23)$$

$$t(d) + t(e) < T_0, \quad (24)$$

$$\frac{L_q}{\lambda} < T_1, \quad (25)$$

$$\max\left\{t(e)_{k,j}\right\} < T_2, \quad (26)$$

$$C \times T < M_0, \quad (27)$$

$$s > S_0. \quad (28)$$

In both Definitions 5 and 6, their aims are maximizing the total utility U, but the differences are the computation of U_c and U_t. Based on the above definitions, we propose a utility oriented and cost based scheduling algorithm. The details of the algorithm are as follows:

(1) Analyze user type for each user and select computing equations for U_c and U_t.

(2) Initialize parameters in constraints for UT, UC, T_0, T_1, T_2, M_0, and S_0.

(3) Compute L_q and W_q for each scheduler according to (1) to (3).

(4) With the results of step (3), tag X schedulers with least waiting time.

(5) Input some data into the X schedulers and set the highest priority for these tasks.

(6) Execute the above tasks, and record the running time and cost (see Pseudocode 1).

(7) Predict running time, cost, and corresponding utility of all tasks with time and cost of results from step 6, and tag the scheduler with the maximal utility.

(8) Schedule tasks in the scheduler with maximal utility, and optimize user utility (see Pseudocode 2).

(9) Wait until all tasks finish, and record the running time, cost, and corresponding utility.

```
if (user task is time sensitive) {
    select nodes with quickest speed, execute the above tasks, such that s < S_0;
} else {
    Select nodes with lowest cost, execute the above tasks, such that s < S_0;
}
```

Pseudocode 1

```
initialize upgrade = 1;
while (task is time sensitive and upgrade = 1) {
    let previous user of current be current user;
    unit time cost of current user = unit time cost × (1 + v%);
    unit time cost of previous user = unit time cost × (1 − w%);
    if (both cost of current user and previous user do not decrease) {
        upgrade = 1;
    else {
        upgrade = 0;
        rescore current user bo be current user;
    }
}
```

Pseudocode 2

Table 1: Hardware configuration parameters.

Number	CPU	Amount	Memory (GB)
1	4-core, 3.07 GHz	10	4
2	4-core, 2.7 GHz	10	4

4. Experiments

4.1. Experimental Setup. We do experiments on two hardware configurations and the configurations are in Table 1. Both of the two hardware configurations run on CentOS5.8 and Hadoop-1.0.1.

There are total 20 computing nodes in our experimental environment, and each computing node starts up a virtual computing node. We start 10 schedulers, and each scheduler manages 2 virtual nodes (computing nodes). The application that we use in the experiments is WordCount.

According to (1) to (3), we computed the service intensity ρ, the expected number of tasks in a scheduler L, the expected length of queuing L_q, the expected finishing time of tasks W, and the expected waiting time of queuing W_q. Figure 2 describes the expected waiting time $T(w)$ on each scheduler. As we can see from the figure, the waiting time from schedulers 1, 3, 5, and 7 satisfied (14) and (23), and thus we can copy and execute some subtasks (data with size 1 KB) on them. If the user task is time sensitive, then we run the task on node with faster speed; and if the user is cost sensitive, then we run the task on node with lower cost.

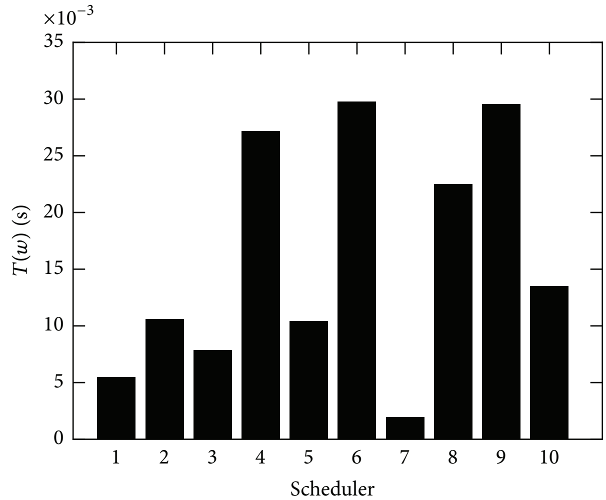


Figure 2: Expected waiting time for each scheduler.

4.2. Experiments for Time Sensitive User Utility Model. In order to select the parameters for time utility and cost utility functions, we normalize them first and get the following equations. Figure 3 describes the curves of the following two equations:

$$U_t = \frac{8}{(\ln(t-20)\times 5)},$$
$$U_c = -\left(\frac{1}{63}\right)\times c + \frac{61}{63}. \tag{29}$$

Based on running time and rate, total time, cost, and utility from schedulers 1, 3, 5, and 7, we set $a = 0.7$ and $b = 0.3$ in (10). Under constrains from (12) to (19), we compute the total utility. In Figure 4, U_t is the predicted time utility, U_c is the predicted cost utility, U' is the predicted total utility, U is

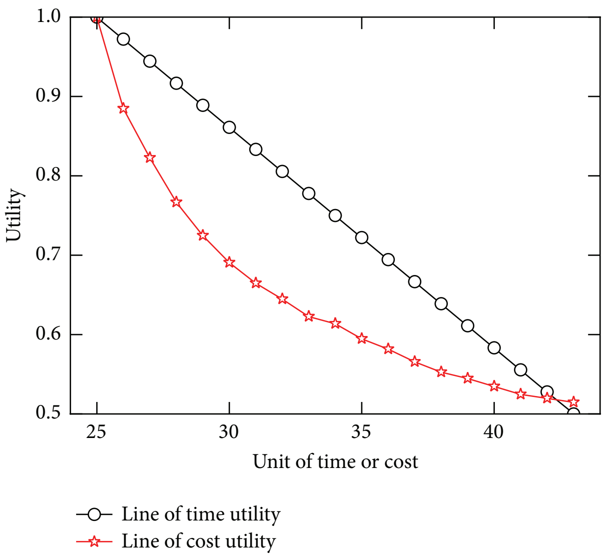


FIGURE 3: Time and cost utility lines for time sensitive user tasks.

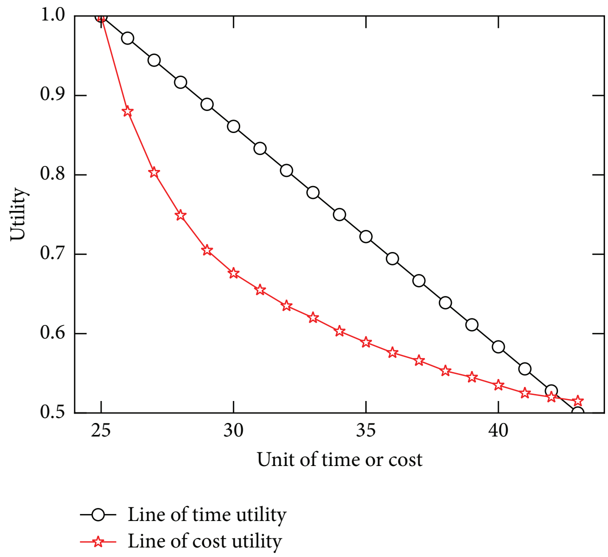


FIGURE 5: Time and cost utility lines for cost sensitive user tasks.

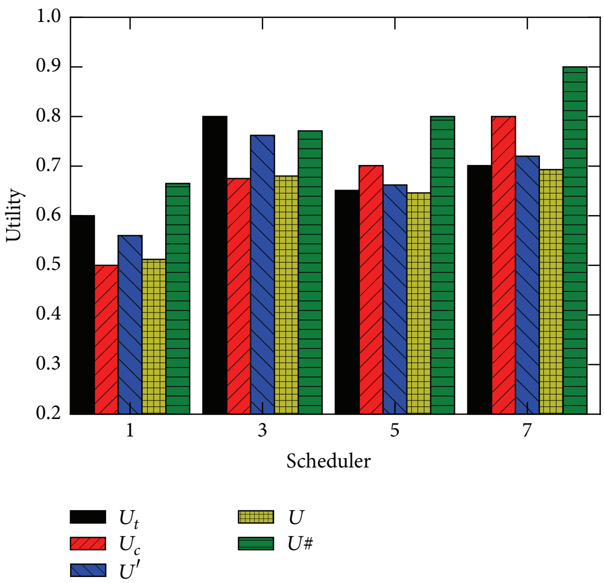


FIGURE 4: Utility distribution of time sensitive user tasks.

the actual total utility, and $U\#$ is the total utility that we get by rescheduling tasks on the above 1, 3, 5, and 7 schedulers.

In Figure 4, for scheduler 1, U_t, U_c, U', and U are all the lowest; for scheduler 3, U_t is the highest, U_c is much lower, and U' is the highest too; for schedulers 5 and 7, although their U_c is higher than scheduler, their U' is lower than scheduler 3. According to the rule of maximizing utility, we should choose scheduler 3 as the scheduler. However, in order to further improve the total utility, we applied the proposed algorithm in Section 3.3.3. By rescheduling the tasks in queuing, we get the actual total utility $U\#$ for each scheduler. In schedulers 5 and 7, $U\#$ is much higher than U' of scheduler 3.

4.3. Experiments for Cost Sensitive User Utility Model. In order to select the parameters of time utility and cost utility functions for cost sensitive user tasks, we also normalize them and get the following two equations. Figure 5 describes the curves of the following two equations:

$$U_t = -\left(\frac{1}{63}\right) \times t + \frac{61}{63},$$
$$U_c = \frac{8}{(\ln(c-20) \times 5)}. \tag{30}$$

From Figure 6 we can see that the predicted total utility U' in scheduler 5 is the highest, and if we schedule tasks on scheduler 5, we would have the highest actual total utility $U\#$. So if user tasks have different time and cost requirements, we can choose different computing nodes to execute them and make the total utility maximal. In addition, after rescheduling the tasks, all tasks have higher actual total utility $U\#$ than predicted utility U' and actual total utility U, which validates the effectiveness of our proposed algorithm.

4.4. Comparison Experiments. In this experiment, we selected 10 simulating tasks and compared our algorithm with both Min-Min and Max-Min algorithms. The Min-Min algorithm schedules minimum task to the quickest computing node every time, and the Max-Min algorithm schedules maximum task to the quickest computing node every time. We implemented two algorithms for both time sensitive and cost sensitive user tasks and denoted them as MaxUtility-Time and MaxUtility-Cost. The experimental result is in Figure 7.

In Figure 7, the total utilities of MaxUtility-Time and MaxUtility-Cost algorithms are higher than the other two algorithms and are also stable; both Min-Min and Max-Min algorithms have lower total utilities, and their values fluctuate very much. Both Min-Min and Max-Min algorithms only consider the running time of tasks and ignore requirements of

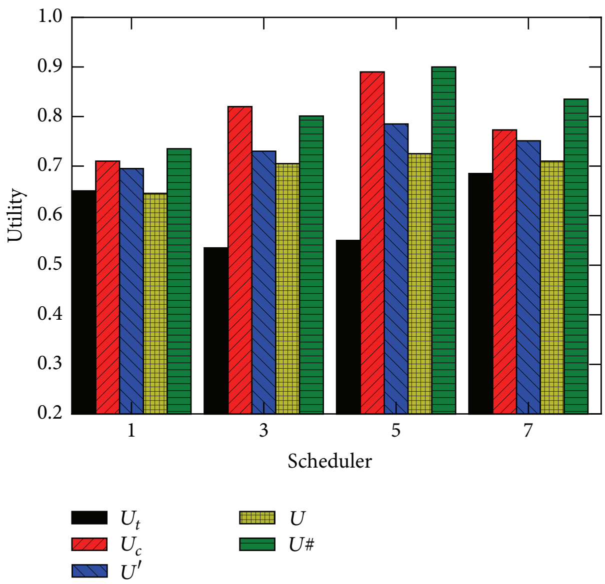


FIGURE 6: Utility distribution of cost sensitive user tasks.

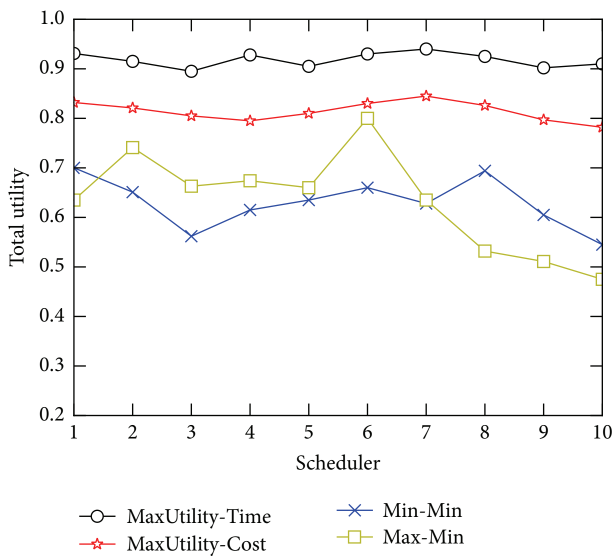


FIGURE 7: Comparison result for different algorithms under simulating tasks.

both time and cost, which makes them get lower total utilities and fluctuate very much. In particular, when running tasks 8, 9, and 10, total utility of the Max-Min algorithm drops quickly. The reason is that it schedules long-running tasks to computing nodes with high performance, which makes the utility very low.

5. Conclusion

In this paper, we introduced utility into the cloud environment, quantified the satisfaction of users to services as utility, and proposed utility oriented queuing model for task scheduling. We classified utility into time and cost utility, rescheduled tasks according to their remaining time, and minimized the total utility by constraints. With the proposed model, we can reschedule remaining tasks dynamically to get the maximum utility. We validated our proposed model by lots of experiments.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] A. Beloglazov and R. Buyya, "Energy efficient resource management in virtualized cloud data centers," in *Proceedings of the 10th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing*, pp. 826–831, IEEE, Melbourne, Australia, May 2010.

[2] C. S. Yeo and R. Buyya, "Service level agreement based allocation of cluster resources: handling penalty to enhance utility," in *Proceedings of the IEEE International Conference on Cluster Computing (CLUSTER '05)*, pp. 1–10, Burlington, Mass, USA, September 2005.

[3] J. N. Silva, L. Veiga, and P. Ferreira, "Heuristic for resources allocation on utility computing infrastructures," in *Proceedings of the 6th International Workshop on Middleware for Grid Computing (MGC '08)*, pp. 93–100, ACM, Leuven, Belgium, December 2008.

[4] G. Song and Y. Li, "Utility-based resource allocation and scheduling in OFDM-based wireless broadband networks," *IEEE Communications Magazine*, vol. 43, no. 12, pp. 127–134, 2005.

[5] T. T. Huu and J. Montagnat, "Virtual resources allocation for workflow-based applications distribution on a cloud infrastructure," in *Proceedings of the 10th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing (CCGrid '10)*, pp. 612–617, IEEE, Melbourne, Australia, May 2010.

[6] Y. Yakov, "Dynamic resource allocation platform and method for time related resources," U.S. Patent Application 10/314,198[P], 2002.

[7] G. Wei, A. V. Vasilakos, Y. Zheng, and N. Xiong, "A game-theoretic method of fair resource allocation for cloud computing services," *The Journal of Supercomputing*, vol. 54, no. 2, pp. 252–269, 2010.

[8] D. López-Pérez, X. Chu, A. V. Vasilakos, and H. Claussen, "Power minimization based resource allocation for interference mitigation in OFDMA femtocell networks," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 2, pp. 333–344, 2014.

[9] X. Wang and J. F. Martínez, "XChange: a market-based approach to scalable dynamic multi-resource allocation in multicore architectures," in *Proceedings of the 21st IEEE International Symposium on High Performance Computer Architecture (HPCA '15)*, pp. 113–125, IEEE, Burlingame, Calif, USA, February 2015.

[10] L. Thomas and R. Syama, "Survey on MapReduce scheduling algorithms," *International Journal of Computer Applications*, vol. 95, no. 23, pp. 9–13, 2014.

[11] D. Cheng, J. Rao, Y. Guo, and X. Zhou, "Improving MapReduce performance in heterogeneous environments with adaptive task tuning," in *Proceedings of the 15th International Middleware*

Conference (Middleware '14), pp. 97–108, ACM, Bordeaux, France, December 2014.

[12] Q. Chen, D. Zhang, M. Guo, Q. Deng, and S. Guo, "SAMR: a self-adaptive mapreduce scheduling algorithm in heterogeneous environment," in *Proceedings of the 10th IEEE International Conference on Computer and Information Technology (CIT '10)*, pp. 2736–2743, IEEE, Bradford, UK, July 2010.

[13] D. Moise, T.-T.-L. Trieu, L. Bougé, and G. Antoniu, "Optimizing intermediate data management in MapReduce computations," in *Proceedings of the 1st International Workshop on Cloud Computing Platforms (CloudCP '11)*, pp. 37–50, ACM, Salzburg, Austria, April 2011.

[14] R. Buyya, D. Abramson, J. Giddy, and H. Stockinger, "Economic models for resource management and scheduling in grid computing," *Concurrency Computation Practice and Experience*, vol. 14, no. 13–15, pp. 1507–1542, 2002.

[15] B. Xu, C. Zhao, E. Hu, and B. Hu, "Job scheduling algorithm based on Berger model in cloud environment," *Advances in Engineering Software*, vol. 42, no. 7, pp. 419–425, 2011.

[16] S. Yeo and H.-H. S. Lee, "Using mathematical modeling in provisioning a heterogeneous cloud computing environment," *Computer*, vol. 44, no. 8, Article ID 5740825, pp. 55–62, 2011.

[17] M. H. Rothkopf, "Scheduling with random service times," *Management Science*, vol. 12, no. 9, pp. 707–713, 1966.

[18] R. H. Möhring, A. S. Schulz, and M. Uetz, "Approximation in stochastic scheduling: the power of LP—based priority policies," *Journal of the ACM*, vol. 46, no. 6, pp. 924–942, 1999.

[19] N. Megow, M. Uetz, and T. Vredeveld, "Models and algorithms for stochastic online scheduling," *Mathematics of Operations Research*, vol. 31, no. 3, pp. 513–525, 2006.

[20] M. Scharbrodt, T. Schickinger, and A. Steger, "A new average case analysis for completion time scheduling," *Journal of the ACM*, vol. 53, no. 1, pp. 121–146, 2006.

[21] X. Nan, Y. He, and L. Guan, "Optimal resource allocation for multimedia cloud based on queuing model," in *Proceedings of the 3rd IEEE International Workshop on Multimedia Signal Processing (MMSP '11)*, pp. 1–6, Hangzhou, China, November 2011.

[22] Z. Xiao, W. Song, and Q. Chen, "Dynamic resource allocation using virtual machines for cloud computing environment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 6, pp. 1107–1117, 2013.

Hindawi Publishing Corporation
Journal of Electrical and Computer Engineering
Volume 2015, Article ID 143071, 9 pages
http://dx.doi.org/10.1155/2015/143071

Research Article

Regression Cloud Models and Their Applications in Energy Consumption of Data Center

Yanshuang Zhou, Na Li, Hong Li, and Yongqiang Zhang

Handan College, Handan, Hebei 056005, China

Correspondence should be addressed to Yanshuang Zhou; shuangshuang223@126.com

Received 13 August 2015; Accepted 27 September 2015

Academic Editor: James Nightingale

As cloud data center consumes more and more energy, both researchers and engineers aim to minimize energy consumption while keeping its services available. A good energy model can reflect the relationships between running tasks and the energy consumed by hardware and can be further used to schedule tasks for saving energy. In this paper, we analyzed linear and nonlinear regression energy model based on performance counters and system utilization and proposed a support vector regression energy model. For performance counters, we gave a general linear regression framework and compared three linear regression models. For system utilization, we compared our support vector regression model with linear regression and three nonlinear regression models. The experiments show that linear regression model is good enough to model performance counters, nonlinear regression is better than linear regression model for modeling system utilization, and support vector regression model is better than polynomial and exponential regression models.

1. Introduction

In a 7×24 running cloud data center, energy model is one of the most important researches in green computing [1]. In a cloud data center, or data center for short, both users and administrators need to know how the energy of servers is consumed according to their behaviors, and thus they can minimize the total consumed energy. Along with support of energy monitoring by hardware, energy consumption of most hardware can be measured in run-time [2]. However, in order to further predict future energy consumption, researchers need to build energy model by their own. During the modeling of energy consumption, researchers need to capture the relationships between hardware resources and their energy consumptions.

Nowadays, as the scale of data center becomes larger and larger, the total energy consumption of data center becomes bigger and bigger and the energy model becomes more complex. Each cloud service, such as SaaS (Software as a Service) [3], PaaS (Platform as a Service) [4], or IaaS (Infrastructure as a Service) [5], has different energy optimization methods, and these methods cannot be migrated to other services directly. However, a unified energy framework [6] includes the following three basic steps. Firstly, it needs to sample or collect factors that affect the energy consumption of the whole system or some hardware components. Commonly used sampling methods are classified into performance counters based and system utilization based. After collecting factors that affect energy consumption, we must combine them together with a suitable model. The more suitable the model, the more accurate the energy prediction we will get in the future. Finally, based on the proposed model, one can focus on factors that affect energy consumption the most, monitor their energy consumptions, and schedule running tasks to minimize the total energy consumption.

In this paper, we analyzed the energy consumption of data center by each computing node and analyzed different regression models and their performances on predicting energy consumption. The contributions of the paper are as follows:

(i) We gave a performance counter based energy model framework and analyzed different linear regression models. According to experiments, we concluded that all kinds of linear models had similar prediction performance, and it was enough to model energy consumption with linear models for the performance counters.

(ii) We analyzed linear and nonlinear regression models based on system utilization and proposed a support vector regression model.

(iii) We did lots of experiments for the system utilization based energy models and had the following conclusion. Nonlinear regression models are better than linear regression model, and our support vector regression model is better than polynomial and exponential regression models.

The rest of the paper is organized as follows. Section 2 describes related works about energy model. In Section 3, we give a performance counter based energy model framework, analyze different regression methods for the system utilization based energy model, and propose a support vector regression energy model. Experiments and conclusion are in Sections 4 and 5, respectively.

2. Related Works

In order to control or reduce the energy consumption of a huge data center, researchers should first understand how energy is consumed, so building energy model for a data center is the basis. Most energy models can be classified into system utilization based and performance counters based [6]. However, there are also some environment oriented models.

2.1. System Utilization Based Model. In system utilization based model, researchers model the energy consumption of the whole system based on components. Fan et al. [7] built a linear regression model for total energy consumption of the whole system using CPU utilization and validated its effectiveness. However, after studying the 177 specimen of SPECpower from years between 2007 and 2010, Hsu and Poole [8] found that energy consumption of servers had changed a lot. Since 2008, both hardware providers and system software providers had proposed their own energy management frameworks, and many of them are not simple linear models. In [8], Hsu and Poole proposed a unitary exponential model for computation intensive applications, and the parameters in the model were from experience, and thus their model were not applicable to other applications. Beloglazov et al. [9] utilized the maximal power of servers, CPU utilization, and the energy ratio of idleness to maximal power to model the energy consumption.

In addition, there are some researches that study the whole system by analyzing each component. These researches detached the whole system into CPU, memory, disks, I/O, and so on and built multiple linear regression models. Economou et al. [10] applied noninvasive method to model energy consumption of the whole system, and the model was built after one calibration. In the experiments, they compared the performance of their model on servers with different energy and workload. Beside hardware components, Lewis et al. [11] took the temperature of environment and bus activities into consideration and predicted the long time energy consumption of servers. Kliazovich et al. [12] modeled the energy of servers by power and frequency of CPUs and a constant. In their model, all components except for CPUs were considered to be constant. In the DENS schedule algorithm, Kliazovich et al. [13] described an energy model of the whole data center, where computing nodes were linear model of CPU utilization, and switch was multivariate function. Lee and Zomaya [14] used the maximal power, the minimal power, and current power of CPUs to model the energy consumption.

2.2. Performance Counter Based Model. In order to further improve the accuracy of energy model, researchers proposed performance counter based energy models along with the development of CPU. By reading data from performance counters together with features of the underlined components, many energy models could be built. For example, according to analyzing memory energy consumption, researchers found that the main energy consumption for memory was read/write throughput [8]. In order to acquire throughput of memory, we can record the missing number of the last level cache, and this can be easily accessed.

In addition, some researchers used data of performance event to build energy model. They studied specified CPUs and fixed the events that needed to be sampled; however their methods could not be applicable to other situations. Isci and Martonosi [15] sampled data for 22 events and built linear energy model based on all events' maximal power. This model is much more complex and needs to sample much data and cannot be deployed into cloud data center.

2.3. Environment Oriented Model. For some specified computing environments, researchers studied environment oriented energy models. In the MapReduce environment, Li et al. [16] analyzed the workload data for each Map, Merge, and Reduce step and built an energy model based on linear regression. They analyzed computing intensive and I/O intensive two classical MapReduce applications. For MPI/OpenMP environment in large scale multicore data center, Wu et al. [17, 18] built an application-centered energy model for CPUs and memory components, and the main target of their research is to differentiate the energy consumption between OpenMP and MPI.

In addition, Farahnakian et al. [19] propose a KNN regression approach for predicting energy consumption in data centers. However, this approach is a heuristic regression model, but we analyze different kinds of regression models, especially on support vector regression. Beloglazov et al. [9] also propose a heuristic energy predicting approach and the underlying model is not a regression model. By experiments, we validate that regression model is good enough for prediction in such situation, so nonregression model is not necessary.

3. Modeling Energy with Regression Methods

3.1. Modeling Performance Counter. Since Pentium series, the Intel processors have introduced performance monitoring system by some specified performance counters. According to these performance counters, we can acquire the performance of processors and thus optimize the performance of the compiler and even the whole system. Generally speaking, the events that can be monitored in processors include instructions, cache, and table lookup buffer. Though these events affect energy consumption of the whole system, they have different weights. By linear equation, we give different events different weights and estimate these weights by historical data. The energy model based on performance counters is as follows:

$$P_c = C_0 + \sum_{i=1}^{n} C_i p_i, \tag{1}$$

where C_0 is a constant, n is the number of monitored events, p_i is the sampled event of performance counter, which represents its energy consumption, and C_i is the weight of event i.

Nowadays, these are many tools that can be used to collect information of performance counters, such as OProfile [20] and Perf [21]. By these tools, applications can acquire energy statistics of hardware components with power management unit. Perf is a system performance optimization tool in Linux kernel and has advantages in adaptation and developing activity. In this paper, we apply Perf to sample performance counters.

3.2. Modeling System Utilization. Except for performance counters, we also apply system utilization to model the energy consumption. In this paper, we mainly focus on system utilization of CPU, memory, storage, and I/O and consider other parts of the system as constant.

3.2.1. CPU Utilization Energy Model. CPU consumes lots of energy in a system, but it can adjust energy consumption conveniently. So adjusting the status of CPUs in a data center is the main energy-saving method.

In [22], the energy model of CPUs is a simple linear regression, which can be described as

$$P = C_0 + C_1 U_{\text{CPU}}, \tag{2}$$

where C_0 is a constant, U_{CPU} is the utilization of CPU, and C_1 is its weight. However, the energy consumption of CPU is not linear with its utilization, so we add another empirical parameter to modify their relation:

$$P = C_0 + C_1 U_{\text{CPU}} + C_2 U_{\text{CPU}}^t, \tag{3}$$

where U_{CPU}^t is a calibrating parameter and C_2 is its weight.

3.2.2. Memory Utilization Energy Model. When memory is idle, we can make it standby, sleep, or even shutdown to save the energy consumption. In each of the above statuses, the energy consumption is lower than working status but the response time increases. According to our observation, the energy consumption of memory is proportional to CPU. That is, when CPU utilization is high, the energy consumption of memory is also high. So, we model the energy consumption of memory as follows:

$$P = C_0 + C_1 U_{\text{CPU}} + C_2 U_{\text{memory}}, \tag{4}$$

where U_{memory} is the utilization of memory and C_2 is its weight.

3.2.3. Storage Utilization Energy Model. Because of the scalability of server, there are usually more than one disk in a server, for example, multiple disks or disk arrays. Different storage systems have different energy consumption. The method for saving energy consumption of storage system is to apply disks with different adjusting functions, such as changing from running status to standby or sleep status, or adjusting the revolutions per minute. In this paper, we consider that the energy consumption of storage is also related to CPU and the equation is

$$P = C_0 + C_1 U_{\text{CPU}} + C_2 U_{\text{disk}}, \tag{5}$$

where U_{disk} is the utilization of storage system and C_2 is its weight.

3.2.4. I/O Utilization Energy Model. We also observed the energy consumption of I/O system. In a server, the I/O energy consumption mainly depends on network interface card and data transition. In this paper, we study the energy model of data center by nodes, and the energy used by data transition between switches is not considered. The energy model of I/O system is as follows:

$$P = C_0 + C_1 U_{\text{CPU}} + C_2 U_{\text{IO}}, \tag{6}$$

where U_{IO} is the utilization of I/O system and C_2 is its weight.

3.3. Regression Methods. Regression method is an effective tool for modeling the relationship between variables, and its purpose is to predict the change of target variable with respect to changes of decision variables. Concretely speaking, we assume energy as the target variable y and utilizations of components as decision vector $\mathbf{x}$. Given some pairs of $\mathbf{x}$ and y, we seek to find the relationship between them; that is, $y = \mu(\mathbf{x})$, and this method is called regression analysis.

3.3.1. Linear Method. In regression analysis, the target variable is called dependant variable, and the decision variables are called independent variables. Multivariate linear regression models the relationship between dependant variable and multiple independent variables. If there are m independent variables, then the regression model is

$$\begin{gathered} y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_m x_m + \varepsilon, \\ E(\varepsilon) = 0, \\ D(\varepsilon) = \sigma^2 < +\infty, \end{gathered} \tag{7}$$

where y is the observed energy consumption, x_i $(1 \le i \le m)$ is the observation of utilization of each component, β_i $(0 \le i \le m)$ is the coefficient of regression model, and ε is the random error that cannot be observed. For random error ε, we assume that its expectation and variance are 0 and σ^2, respectively.

If there are n observation samples $(y_i, x_{i1}, \ldots, x_{im})$, $i = 1, \ldots, n$, then

$$\begin{aligned} y_i &= \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_m x_{im} + \varepsilon_i, \\ E(\varepsilon_i) &= 0, \\ D(\varepsilon_i) &= \sigma^2, \end{aligned} \tag{8}$$

where ε_i $(i = 1, \ldots, n)$ are independent of each other. For convenience, we represent the above equation with matrix and vector, and then we get

$$\begin{aligned} y &= X\beta + \beta_0 + \varepsilon, \\ E(\varepsilon) &= 0, \\ D(\varepsilon) &= \sigma^2 I, \end{aligned} \tag{9}$$

where

$$\begin{aligned} X &= \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{pmatrix} \\ y &= \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix}, \\ \beta &= \begin{pmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \end{pmatrix}, \\ \varepsilon &= \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \varepsilon_4 \end{pmatrix}. \end{aligned} \tag{10}$$

Next, we train the above linear regression model with sampled system utilization. We apply least squares method to estimate the regression coefficient, and the objective is minimizing the following equation:

$$\begin{aligned} &Q(\beta_0, \beta_1, \ldots, \beta_m) \\ &\quad = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_{i1} - \cdots - \beta_m x_{im})^2 \\ &\quad = (y - \beta_0 - X\beta)^T (y - \beta_0 - X\beta). \end{aligned} \tag{11}$$

When we minimize the above equation at $\beta'_0, \beta'_1, \ldots, \beta'_m$ and y', we get the following regression model:

$$y' = \beta'_0 + \beta'_1 x_1 + \cdots + \beta'_m x_m. \tag{12}$$

Generally speaking, it is very hard to get the optimal solution using the least squares method, so we apply the one-by-one regression method. The main idea is introducing one variable at each time, and the condition when introducing a variable is that the bias of the variable is significant. At the same time, after introducing a variable, we check all introduced variables and delete nonsignificant variables. When we have checked all variables, we get the final regression model.

With the one-by-one regression method, we get noncontinuous parameters when normalized. In order to get continuous normalized parameter, we apply lasso regression method after that. The lasso regression method uses absolute value of model parameters to penalize those bigger parameters, and the equation is

$$\begin{aligned} \min_{(\beta_0,\beta)} \Bigg[&\frac{1}{2n} \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_{i1} - \cdots - \beta_m x_{im})^2 \\ &+ \lambda \|\beta\|_1 \Bigg], \end{aligned} \tag{13}$$

where $\|\beta\|_1 = (1/m) \sum_{i=1}^{m} \beta_i$ is the lasso regression and λ is its weight. Compared with traditional model selection method, the lasso method can prohibit overfitting of the selected model. Based on lasso regression in (13), we can make unimportant parameters to 0. When we train the model with system utilization data, we can get a balance between overfitting and underfitting. There is one thing needed to be considered; that is, the selection of λ needs cross-validation.

3.3.2. Nonlinear Methods. Based on linear model, we also introduce nonlinear regression model. By transforming function and adding factors, we get the following nonlinear regression model:

$$f(x) = w \cdot \phi(x) + b. \tag{14}$$

In (14), the transforming function can be of any form. In this paper, we apply the following three functions.

(i) Polynomial + Lasso. Instead of linear model, we use polynomial to model the relationship between system utilization and energy consumption. Here, we set $\phi(x) = \{x_i^a : 1 \le a \le 3,\ 1 \le i \le m\}$, and its coefficient is w. With lasso constraint on $\phi(x)$, the final objective is

$$f(x) = w \cdot \phi(x) + b + \lambda \|\phi(x)\|_1. \tag{15}$$

(ii) Polynomial + Exponential + Lasso. Based on polynomial and lasso regression, we add exponential transform for each coefficient and get the following equation:

$$f(x) = w \cdot \exp(\phi(x)) + b + \lambda \|\phi(x)\|_1, \tag{16}$$

where exponential function $\exp(\cdot)$ is elementwise.

(iii) Support Vector Regression. We also implement a support vector machine based regression model. Given a collection of samples (x_i, y_i), $1 \le i \le n$, where $x_i \in R^m$ are independent

variables, $y_i \in R$ is dependent variable, and n is the number of samples, the regression model is

$$f(x) = w \cdot x + b, \tag{17}$$

and the optimization problem becomes

$$\min_{w,b} \frac{1}{2} \|w\|^2, \quad \left| y_i - w \cdot x_i - b \right| \leq \varepsilon, \quad i = 1, \ldots, n. \tag{18}$$

For solving the above optimization problem, one can transform the original problem into its dual form, and details can be found in [23].

3.3.3. Evaluation Metric. In this paper, we use relative error $\text{Power}_{\text{error}}$ to evaluate the accuracy of the energy model, and its computation is in the following equation:

$$\text{Power}_{\text{error}} = \frac{\text{Power}_{\text{prediction}} - \text{Power}_{\text{real}}}{\text{Power}_{\text{real}}}, \tag{19}$$

where $\text{Power}_{\text{prediction}}$ is the result that we get using our model and $\text{Power}_{\text{real}}$ is the real energy consumption that we measure using electricity meter.

4. Experiments

4.1. Results on Performance Counter Based Model

4.1.1. Experimental Setup. In this experiment, the node of the server contains a Interl Xeon E5620 processor and 48 GB memory; the kernel of the Linux is 2.6.32-220.el6.x86_64. The tool we use to sample performance counters is perf-2.6.32, and the compiler is gcc-4.4.6.

4.1.2. Dataset. In order to sample data efficiently, we use SPEC2006 as our benchmark. The SPEC2006 contains integer and float operations, and we use them separately. For each operation, we use some of data as training data and the remaining as test data. We analyze the effectiveness of different mathematical methods of performance counters to energy consumption and the experimental data is in Table 1.

4.1.3. Sampling and Modeling. We run each Benchmark ten times on the training data and sample the following events: cpu-cycles, L1-dcache-loads, LLC-loads, stalled-cycles-frontend, L1-dcache-load-misses, LLC-load-misses, stalled-cycles-backend, L1-dcache-stores, LLC-stores, instructions, L1-dcache-store-misses, LLC-store-misses, stalled, cycles, per, insn, L1-dcache-prefetches, LLC-prefetches, cache-references, L1-dcache-prefetch-misses, LLC-prefetch-misses, cache-misses, L1-icache-loads, context-switches, branch-instructions, L1-icache-load-misses, cpu-migrations, branch-misses, dTLB-loads, alignment-faults, bus-cycles, dTLB-load-misses, emulation-faults, task-clock, dTLB-stores, branch-loads, page-faults, dTLB-store-misses, branch-load-misses, minor-faults, iTLB-loads, major-faults, and iTLB-load-misses.

Table 1: Experimental data.

Benchmark	Training data	Test data
Integer	bzip2, gcc, sjeng, h264ref, astar, xalancbmk, gobmk	hmmer, libquantum, mcf, omnetpp
Float	bwaves, gamess, zeusmp, gromac, cactusADM, calculix, GemsFDTD, tonto, wrf, sphinx3, dealII, soplex, povray	lbm, leslie3d, milc, namd

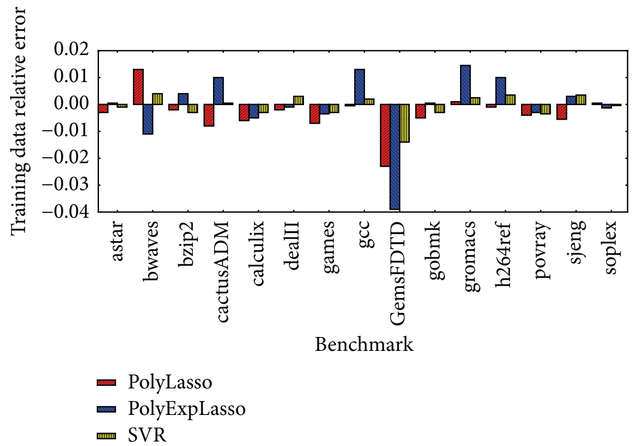


Figure 1: Relative error in training data.

After sampling data from the above events, we apply polynomial lasso regression (PolyLasso), polynomial exponential lasso (PolyExpLasso), and support vector regression (SVR), three regression models to predict the energy consumption.

4.1.4. Results. We divide the data into training data and test data. The relative error in training data reflects fitting of models to training data. In test steps, we abbreviate the real energy consumption as RealVal and compare the predicted power with its real power value.

Figure 1 illustrates the relative error in training data. As can be seen from the figure, all of the three models have small relative errors in training data and they are all between −2% and 2%. At the same time, the SVR model has better fitting advantage. Figure 2 compares the real power usage with the three predicted values on training data, and all of them are very close. Figure 3 describes the relative errors of the three models on test data, and they are all between −4% and 4%. Figure 4 compares the real power usage with the three predicted values on test data, and they are very close too.

From the above experiments we can see that the three regression models have almost the same prediction performance, and SVR does not have advantage in test data. So, in performance counter based energy model, we can apply linear regression.

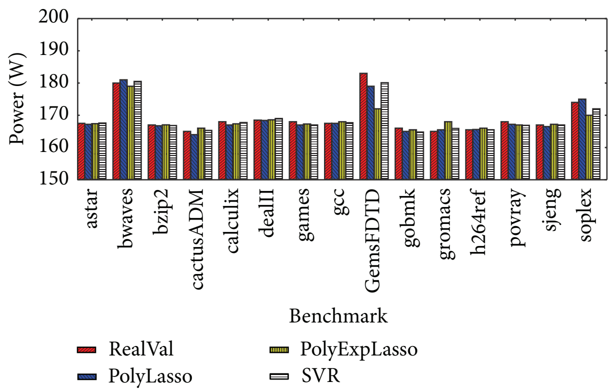


Figure 2: Result comparison on training data.

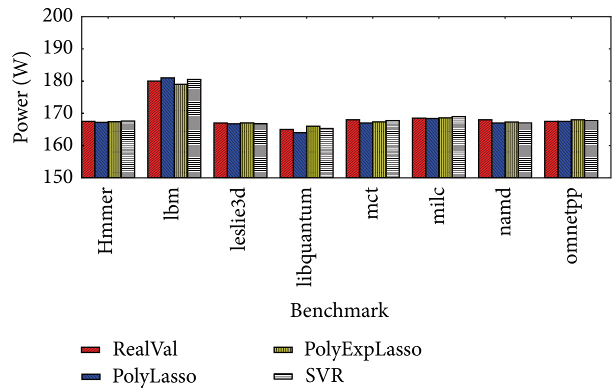


Figure 4: Result comparison on test data.

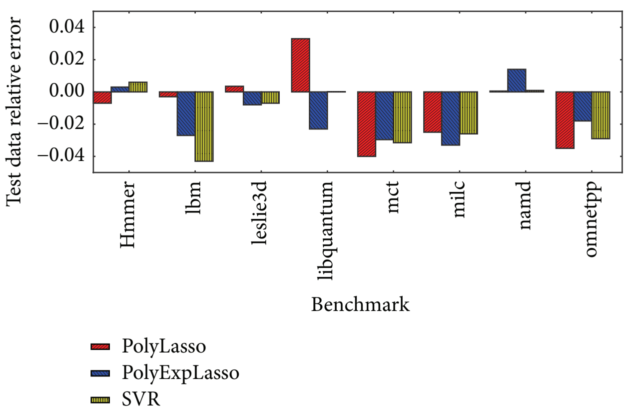


Figure 3: Relative error in test data.

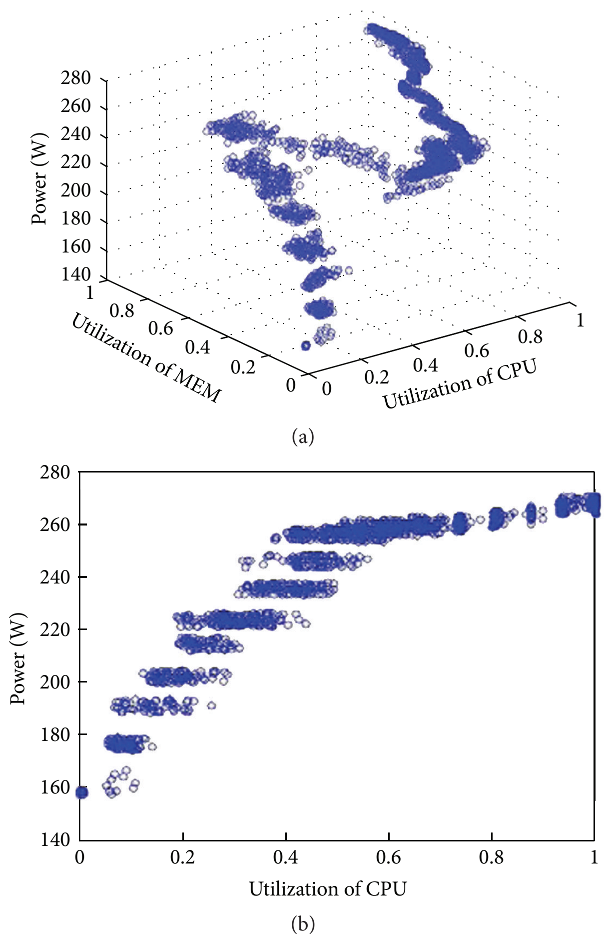


Figure 5: Energy consumption of CPU and memory utilization.

4.2. Results on System Utilization Based Model

4.2.1. Experimental Setup. The experimental environment is the same as before.

4.2.2. Sampling and Modeling. In Linux or Unix system, the utilization of processor includes user status, system status, and idle status, and they represent the statuses by which processor executes user instructions, system kernel instructions, and idle instructions, respectively. The commonly used processor utilization is the total time in which the processor executes nonidle instructions.

In this paper, the experimental operating system is CentOS6.2, and we can acquire the processor utilization with the file “/proc/stat.” In this experiment, the overload of processor is as follows:

cpu 245679 759 89549 345763563 48744 29 2316;

cpu0 196816 395 4962 172780284 45243 0 44;

cpu1 48862 364 84587 172983278 3501 29 2271.

The meaning of each data is as follows:

(i) user: total time of user status since the system startup; it does not contain nice value with negative process;

(ii) nice: the total processor time of nice with negative process;

(iii) system: total time of kernel status;

(iv) idle: total waiting time except for waiting disk;

(v) Iowait: total time for waiting disk I/O;

(vi) irq: total time of hard interrupt;

(vii) softirq: total time of soft interrupt.

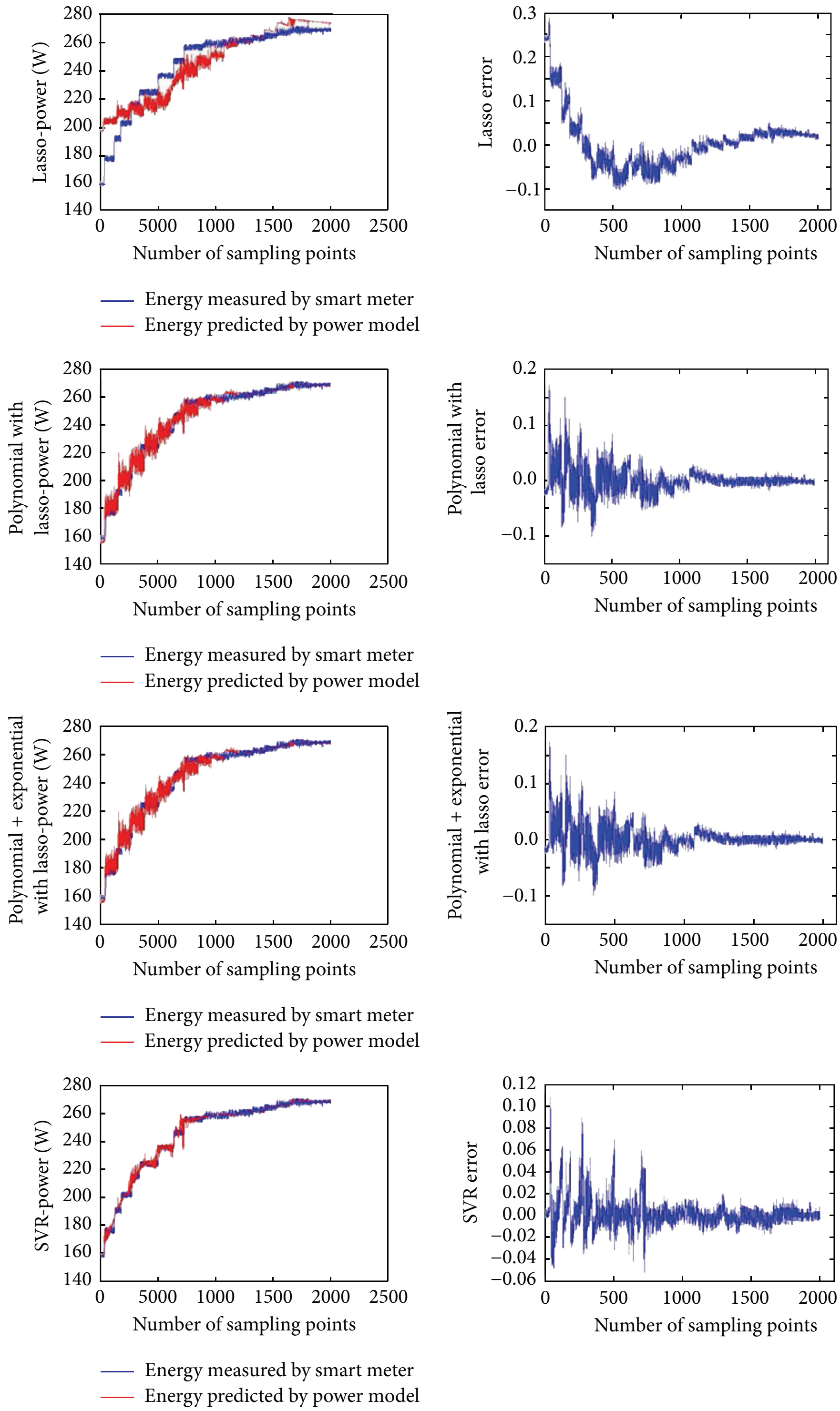


Figure 6: Results of system utilization based models.

Consider the following:

$$\begin{aligned} \text{CPU\_Time} &= \text{user} + \text{system} + \text{nice} + \text{idle} \\ &\quad + \text{iowait} + \text{irq} + \text{softirq}, \\ t_{\text{stat}} &= t_{\text{stat\_now}} - t_{\text{stat\_last}}, \\ \text{Utilization}_{\text{CPU}} &= \frac{t_{\text{user}} + t_{\text{system}}}{t_{\text{user}} + t_{\text{system}} + t_{\text{idle}}}. \end{aligned} \tag{20}$$

With similar idea, we can acquire memory utilization with file "/proc/meminfo." In this experiment, we collect

2000 records, and each of them is a tuple of the form ($\text{Utilization}_{\text{CPU}}$ + $\text{UtilizationM}_{\text{EM}}$ + Power). Details of the records are in Figure 5. Based on the above records, we apply linear lasso regression, polynomial lasso regression, polynomial exponential lasso regression, and support vector regression, four regression models to predict the energy consumption.

4.2.3. Results. For the four regression models, we compared their relative error and the difference between predicted value and real value, and the results are in Figure 6.

From Figure 6 we can see that linear regression has very low predicting accuracy and its error is bigger than others obviously. In addition, polynomial lasso regression and polynomial exponential lasso regression models have very similar predicting results, and most of them are between −10% and 10%. In the four regression models, support vector regression has the best predicting results, and its error is between −6% and 10%.

5. Conclusion

In this paper, we study the energy consumption of data center by computing nodes. For each computing node, we analyzed both performance counter based model and system utilization based model. For performance counter based model, we find that linear method is good enough to model energy consumption; for system utilization based model, we find that nonlinear model is more accurate than linear model and support vector regression is better than polynomial regressions.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work was supported in part by the science and technology planning project of Hebei province (no. 12210334).

References

[1] A. Hooper, "Green computing," *Communication of the ACM*, vol. 51, no. 10, pp. 11–13, 2008.

[2] B. P. Rimal, E. Choi, and I. Lumb, "A taxonomy and survey of cloud computing systems," in *Proceedings of the 5th International Joint Conference on INC, IMS and IDC (NCM '09)*, pp. 44–51, IEEE, Seoul, Republic of Korea, August 2009.

[3] P. Buxmann, T. Hess, and D. W. I. S. Lehmann, "Software as a service," *Wirtschaftsinformatik*, vol. 50, no. 6, pp. 500–503, 2008.

[4] E. Keller and J. Rexford, "The 'platform as a service' model for networking," in *Proceedings of the Internet Network Management Conference on Research on Enterprise Networking (INM/WREN '10)*, p. 4, USENIX Association, San Jose, Calif, USA, April 2010.

[5] S. Bhardwaj, L. Jain, and S. Jain, "Cloud computing: a study of infrastructure as a service (IAAS)," *International Journal of Engineering and Information Technology*, vol. 2, no. 1, pp. 60–63, 2010.

[6] C. Dupont, G. Giuliani, F. Hermenier, T. Schulze, and A. Somov, "An energy aware framework for virtual machine placement in cloud federated data centres," in *Proceedings of the 3rd International Conference on Future Energy Systems: Where Energy, Computing and Communication Meet (e-Energy '12)*, pp. 1–10, IEEE, Madrid, Spain, May 2012.

[7] X. Fan, W.-D. Weber, and L. A. Barroso, "Power provisioning for a warehouse-sized computer," *ACM SIGARCH Computer Architecture News*, vol. 35, no. 2, pp. 13–23, 2007.

[8] C.-H. Hsu and S. W. Poole, "Power signature analysis of the SPECpower_ssj2008 benchmark," in *Proceedings of the IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS '11)*, pp. 227–236, Austin, Tex, USA, April 2011.

[9] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," *Future Generation Computer Systems*, vol. 28, no. 5, pp. 755–768, 2012.

[10] D. Economou, S. Rivoire, C. Kozyrakis, and P. Ranganathan, "Full-System power analysis and modeling for server environments," in *Proceedings of the 14th Interntaional Symposium on Computer Architecture (ISCA '06)*, pp. 70–77, IEEE, Boston, Mass, USA, June 2006.

[11] A. W. Lewis, S. Ghosh, and N.-F. Tzeng, "Run-time energy consumption estimation based on workload in server systems," in *Proceedings of the Conference on Power Aware Computing and Systems (HotPower '08)*, pp. 17–21, December 2008.

[12] D. Kliazovich, P. Bouvry, and S. U. Khan, "GreenCloud: a packet-level simulator of energy-aware cloud computing data centers," *The Journal of Supercomputing*, vol. 62, no. 3, pp. 1263–1283, 2012.

[13] D. Kliazovich, P. Bouvry, and S. U. Khan, "DENS: data center energy-efficient network-aware scheduling," *Cluster Computing*, vol. 16, no. 1, pp. 65–75, 2013.

[14] Y. C. Lee and A. Y. Zomaya, "Energy efficient utilization of resources in cloud computing systems," *The Journal of Supercomputing*, vol. 60, no. 2, pp. 268–280, 2012.

[15] C. Isci and M. Martonosi, "Runtime power monitoring in high-end processors: methodology and empirical data," in *Proceedings of the 36th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO '03)*, pp. 93–104, IEEE Computer Society, San Diego, Calif, USA, December 2003.

[16] W. Li, H. Yang, Z. Luan, and D. Qian, "Energy prediction for MapReduce workloads," in *Proceedings of the 9th IEEE International Conference on Dependable, Autonomic and Secure Computing (DASC '11)*, pp. 443–448, IEEE, Sydney, Australia, December 2011.

[17] C. Lively, X. Wu, V. Taylor et al., "Power-aware predictive models of hybrid (MPI/OpenMP) scientific applications on multicore systems," *Computer Science—Research and Development*, vol. 27, no. 4, pp. 245–253, 2012.

[18] C. Lively, X. Wu, V. Taylor, S. Moore, H.-C. Chang, and K. Cameron, "Energy and performance characteristics of different parallel implementations of scientific applications on multicore systems," *International Journal of High Performance Computing Applications*, vol. 25, no. 3, pp. 342–350, 2011.

[19] F. Farahnakian, T. Pahikkala, P. Liljeberg, and J. Plosila, "Energy aware consolidation algorithm based on K-nearest neighbor regression for cloud data centers," in *Proceedings of the*

6th IEEE/ACM International Conference on Utility and Cloud Computing (UCC '13), pp. 256–259, IEEE, Dresden, Germany, December 2013.

[20] J. Levon, P. Elie, and Oprofile, "A system-wide profiler for Linux systems," 2006, http://oprofile.sourceforge.net.

[21] V. M. Weaver, "Linux perf—event features and overhead," in *Proceedings of the 2nd International Workshop on Performance Analysis of Workload Optimized Systems (FastPath '13)*, p. 80, April 2013.

[22] S. K. Garg, S. Versteeg, and R. Buyya, "A framework for ranking of cloud computing services," *Future Generation Computer Systems*, vol. 29, no. 4, pp. 1012–1023, 2013.

[23] D. Basak, S. Pal, and D. C. Patranabis, "Support vector regression," *Neural Information Processing—Letters and Reviews*, vol. 11, no. 10, pp. 203–224, 2007.

Hindawi Publishing Corporation
Journal of Electrical and Computer Engineering
Volume 2015, Article ID 357378, 7 pages
http://dx.doi.org/10.1155/2015/357378

Research Article

Cloud Platform Based on Mobile Internet Service Opportunistic Drive and Application Aware Data Mining

Ge Zhou

School of Information Engineering, Chongqing Youth Vocational & Technical College, Chongqing 400712, China

Correspondence should be addressed to Ge Zhou; gezhou_cy@163.com

Received 7 July 2015; Accepted 1 September 2015

Academic Editor: James Nightingale

Because the static cloud platform cannot satisfy the diversity of mobile Internet service and inefficient data mining problems, we presented a reliable and efficient data mining cloud platform construction scheme based on the mobile Internet service opportunistic driving and application perception. In this scheme, first of all data selection mechanism was established based on mobile Internet service opportunistic drive. Secondly, through the cloud platform different cloud and channel aware, nonlinear mapping from the service to a data set of proposed perceptual model is applied. Finally, on the basis of the driving characteristics and extraction of perceptual features, the cloud platform would be constructed through the service opportunities of mobile Internet applications, which could provide robust and efficient data mining services. The experimental results show that the proposed mechanism, compared to the cloud platform based on distributed data mining, has obvious advantages in system running time, memory usage, and data clustering required time, as well as average clustering quality.

1. Introduction

The cloud platform has high performance of data management and efficient data mining [1], which has been widely applied in various fields, such as knowledge discovery and Leaf Spot dynamics [2], and ERP applications [3]. So, the cloud platform [4, 5] has played an important role for quality of service guarantee and data mining of mobile Internet.

In data mining, Rakocevic et al. [6] embedded a distributed data mining algorithm into a sensor network, which used local predictors on each sensor node to make a local prediction and offers several original voting schemes. The architecture of the huge amount 3D video data was proposed by [7], which designed a model of Key Encryption Model for protecting the privacy video data. Ronowicz et al. [8] researched the cause-effect relationships between pellet formulation characteristics, as well as the selected quality attribute.

About the service driver, Ghosh et al. [9] developed the cost analysis and optimization framework by using stochastic availability and performance models of an IaaS cloud. The economic model as a Bayesian network was proposed by Ye et al. [10] for selecting and composing cloud services. The importance and challenges in designing event-driven mobile services were discussed [11], which could detect conditions of interest to users and notify them accordingly.

In addition, the fast scalable video coding- (SVC-) based channel-recommendation system for IPTV on a cloud and peer-to-peer (P2P) hybrid platform was studied in [12]. The mechanisms for orchestrating cloud-enabled hardware and software resources were proposed by Petcu et al. [13], which should be supported by a recently developed open-source platform as a service. The novel monitoring architecture was proposed by Alcaraz Calero and Aguado [14], which should be addressed to the cloud provider and the cloud consumers.

However, these research results ignored the relationship between data set of cloud platform and service requirements of mobile Internet and lack the in-depth study of influence of Internet application on cloud platform. Hence, we proposed the cloud platform with mobile Internet service opportunistic drive and application aware data mining.

The rest of the paper is organized as follows. Section 2 describes the mobile Internet services driven data selection mechanism. In Section 3, we design the application perception model. In Section 4, we proposed the efficient

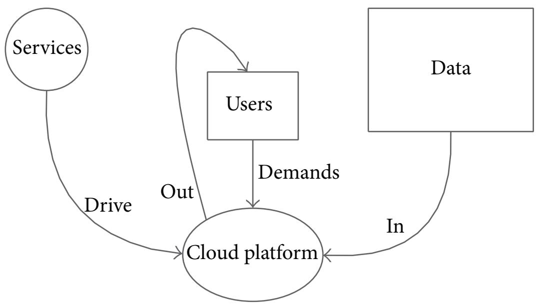


Figure 1: Services architecture in mobile Internet.

data mining cloud platform. Simulation results are given in Section 5. Finally, we conclude the paper in Section 6.

2. Data Selection Mechanism Driven by Mobile Internet Services

In mobile Internet, we assume a service data set function, which was used to optimize data selection for satisfying the service demands and provided the optimal data set for service. This is the role scheme which was used to solve the combination issue of mobile Internet and cloud computing. In general, the mobile Internet service architecture model is shown in Figure 1. Data input was driven by service set and stored in the cloud platform. When the service request was received successfully, the cloud platform would select the data and sent output feedback to the user according to the request.

There are some problems that should be considered in the progress of data selection, which are as follows:

(1) whether the data is corresponding to the service and whether it can meet the service needs,

(2) lack of user information feedback mechanism which is used to judge whether the user is satisfied with the data,

(3) how to coordinate multiple user needs,

(4) how to coordinate the input and output of data without increasing the load of the cloud platform.

The mapping relationship of user set and data set is shown in Figure 2. Here, the relationship was constructed by the services, which include the following cases:

(1) single user to intersection of multiple data sets,

(2) single data set to multiple users,

(3) single user to the union of data sets.

According to the above problems, we give the following service driven definition:

(1) $\mathrm{SE}_{\mathrm{set}}(n)$: service set and its length which is n,

(2) $\mathrm{DA}_{\mathrm{set}}(m)$: data set and its length which is m,

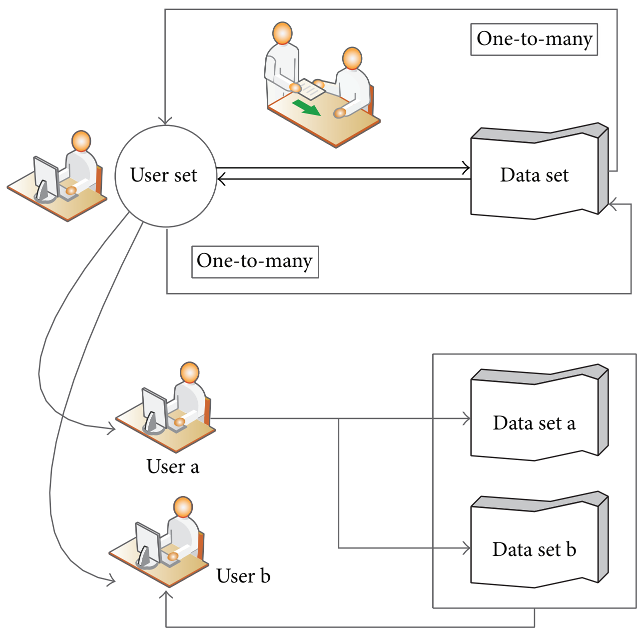


Figure 2: Mapping relationship and correlation between users and data set.

(3) $\mathrm{US}_{\mathrm{set}}(p)$: user set and its length which is p,

(4) $\mathrm{QE}(i)$: service satisfaction of the i data,

(5) $L_{\mathrm{CP}}(j)$: the number of data services in the i unit time,

(6) $T(i)$: the time consumed by the k data service,

(7) $L_{\mathrm{CP}}(a,b)$: the number of service sessions between a and b.

According to the above stations, there is the relationship shown as formula (1) in mobile Internet application:

$$\begin{aligned} N &= L_{\mathrm{CP}}\left(\mathrm{US}_{\mathrm{set}}(1), \mathrm{DA}_{\mathrm{set}}(u) \mid 1 < u \le m\right) \\ &\quad + L_{\mathrm{CP}}\left(\mathrm{US}_{\mathrm{set}}(v) \mid 1 < v \le n, \mathrm{DA}_{\mathrm{set}}(1)\right), \\ T &= \sum_{i=1}^{N} \mathrm{QE}(i)\, T(i), \\ T_h &= \sum_{i=1}^{N} \mathrm{SE}_{\mathrm{set}}(i)\, L_P L_{\mathrm{CP}}(i). \end{aligned} \tag{1}$$

Here, let N denote the number of sessions. Let L_P denote the length of data packet. Let T denote the time of completing the N services. Let T_h denote the throughput.

So, formula (2) gives the data selection model based on service opportunistic driven. Here, service request launched the service opportunistic driven in accordance with the probability. The data sets to meet the service request through the cloud platform. For the request of hybrid multiple data sets of service, the data set could be computed with XOR operation

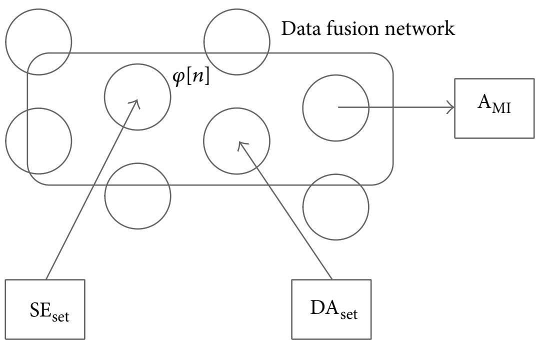


Figure 3: Nonlinear mapping relationship.

to form a single data set, for selecting the data without interference and avoiding the performance degradation:

$$DA_{set}(SE) = \begin{cases} T_h, & \text{dataType} = 0, \\ \sum_{\text{type}=1}^{T_y} DA_{\text{type}}(T_h) \oplus DA_{\text{type}+1}(T_h), & \text{dataType} = 1, \end{cases}$$
$$SE_{set} = [SE_1, SE_2, \dots, SE_n],$$
$$D_{rv} = SE_{set}\begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix}. \tag{2}$$

Here, let D_{rv} denote the service opportunistic request driven factor. Vector $[a_1, \dots, a_n]$ denotes the service opportunistic weight. The value of dataType could be set according to the ability of satisfying the service requests. If dataType is 0, the current data set can be independent of the service request. If dataType is 1, more than 2 data sets could satisfy the current service request. Let T_y denote the number of data sets. $DA_{set}(SE)$ denote the data set which gives feedback to users.

3. Application Perception Model

In the processing progress of mobile Internet applications, cloud platforms can be achieved through nonlinear mapping between different clouds and channel from the service to the data set, which could meet the needs of different data sets from different cloud data fusion. Here, it is important to coordinate transmission and fusion of multicloud applications, as shown in Figure 3. Here, the cycle symbol of Figure 3 denotes the cloud of platform.

Based on the same service request and being located in the same area, data transmission coordination network composed of multicloud needs to be aware of the application and keep the time synchronization. When the application requests are met in different regions, in order to guarantee the space synchronization, the time must keep asynchronous, as shown in formula (3). Based on the above time and space of synchronous request and asynchronous operation, formula (4) illustrated the application of perception matrix and data output matrix and relationship:

$$\begin{gathered} T(i) = T(j), \\ S_{\text{pace}}(DA_{set}(i)) = S_{\text{pace}}(DA_{set}(j)), \\ T(i) \neq T(j), \\ S_{\text{pace}}(DA_{set}(i)) \neq S_{\text{pace}}(DA_{set}(j)). \end{gathered} \tag{3}$$

Here, i and j represent the different application request. Let function $S_{\text{pace}}(DA_{set}(i))$ denote the data set location of the application mapping:

$$\begin{aligned} A_{MI}[n] &= T \otimes (\nabla A_{MI}[n-i] \oplus N(i)), \quad i = 1, 2, \dots, \sqrt{n}, \\ \varphi[n] &= \ln\left(\frac{\nabla A_{MI}[n-i]}{\sqrt{n}}\right), \\ O_{MI} &= e^{A_{MI}[n]} DA_{set}(SE)^{\varphi[n]}. \end{aligned} \tag{4}$$

Here, let $A_{MI}[n]$ denote application perception matrix. Let $\varphi[n]$ denote the multicloud coordination flux. Let O_{MI} denote the output matrix of data set.

Hence, the data output flow of mobile Internet application perception is as follows:

(1) initializing the vector and parameters according to the principles and models,

(2) computing the matrix A_{MI} according to formula (4),

(3) O_{MI} is set to 1, which means service request is active and waiting for data output,

(4) data selection and integration after the activation of cloud devices,

(5) the consistency and time of the region are judged according to formula (3),

(6) if the space is consistent, the time synchronization is obtained by formula (4),

(7) If the space is not the same, the time is asynchronous. The cloud platform is in a stable state. In an asynchronous cycle, the multiple cloud segmentation data are activated. The end of the independent data set is obtained after data integration.

4. Efficient Data Mining Cloud Platform

According to the service opportunities driving characteristics and extraction of perceptual features, how to build cloud platform for mobile Internet applications to provide robust and efficient data mining service has become a key issue of QoS guarantee in mobile Internet. In cloud platform, the customer service satisfaction $QE(DA_{set})$ could be computed by adding each data service satisfaction as shown in formula (5):

$$QE(DA_{set}) = \sum_{i=1}^{n} QE(i). \tag{5}$$

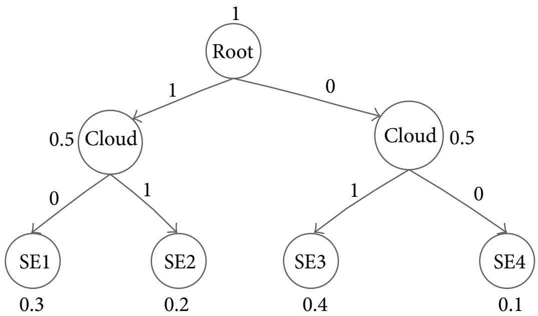


Figure 4: Service driven tree.

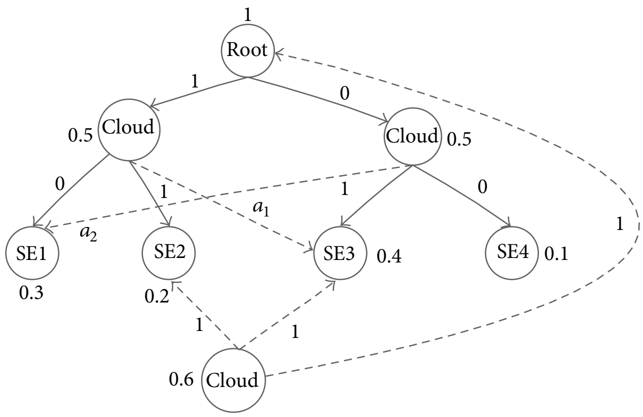


Figure 5: Service opportunistic driven tree.

If customer service demand satisfaction of data sets is larger than the user specified minimum value, the data sets are robust. Otherwise, the data sets should be reconstructed through data mining.

In order to facilitate the service opportunity to play the driving characteristics, we defined the service drive tree. The tree consists of K leaf nodes. Each leaf node represents a service identifier and the opportunity probability. The intermediate node indicates the subtree ability which is denoted by the summary of service opportunities driven weight of leaf nodes. The service driven tree is shown in Figure 4. If the current node and the child nodes are time synchronization, the link will be recorded as 1; otherwise, it is denoted by 0. From the root node to any leaf node, a set of binary bit strings could be obtained. Each leaf node can access the service based on adaptive driving vector, which could reduce data mining complexity. The service opportunities driven tree is shown in Figure 5.

Here, we set 4 leaf nodes and service initiation probability of each leaf node is given. The sum of 4 leaf nodes probability is equal to 1. The service opportunity driven weight of root node is equal to 1. The intermediate nodes are composed of 2 clouds, which are shown as in Table 1.

As shown in Table 1, although the SE4 opportunity weight is low, the service driven path vector is all asynchronous, the data mining complexity is low, and the data clustering service can be obtained with higher real-time. Although the SE3 has the best chance, but the service driven path is heterogeneous,

Service opportunistic drive

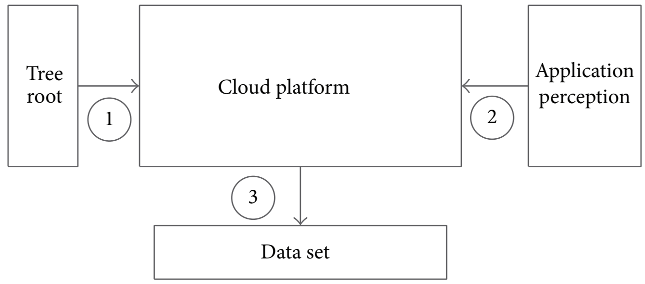


Figure 6: Cloud platform architecture for efficient data mining.

Table 1: Service driven.

Service ID	Weight	Service vector	Data mining complexity
SE1	0.3	10	High
SE2	0.2	11	Low
SE3	0.4	01	High
SE4	0.1	00	Low

Table 2: Service opportunistic driven.

Service ID	Weight	Service vector	Cloud weight	Data mining complexity
SE1	a_2/0.3	10/00	0.5/0.6	High/low
SE2	0.2	11	0.5	Low
SE3	a_1/0.4	01/11	0.5/0.6	High/low
SE4	0.1	00	0.5	Low

the data mining is complex, so the data clustering is poor, and the real-time performance is poor. In addition, the value of SE1, SE2, SE3, and SE4 could be computed based on the service requests and service driven tree as shown in Figure 4.

As shown in Table 2, with opportunistic driven scheme, the data mining complexity of 4 leaf nodes has been reduced. For SE2 and SE3, we can establish the path of multiple data transmission through the opportunity weighting, so as to ensure the real-time and accuracy of data clustering. In addition, the cloud platform could be constructed by choosing the same area of the current leaf nodes and service opportunities driven. Based on the above service opportunity driving features and application of data clustering mining, the architecture of the cloud platform is as shown in Figure 6. Here, ① expresses the analysis of service opportunistic drive characteristics; ② denotes the feature extraction of application perception; ③ denotes the clustering data mining. The construction algorithm is described as follows.

Algorithm 1. Construction of cloud platform is as follows.

Input. It includes $\mathrm{SE}_{\mathrm{set}}(n)$, $\mathrm{DA}_{\mathrm{set}}(m)$, $\mathrm{US}_{\mathrm{set}}(p)$, $\mathrm{QE}(i)$, and data mining complexity threshold $\mathrm{DM}_{\mathrm{th}}$.

Output. It includes data set $\mathrm{DA}_{\mathrm{set}}(\mathrm{C\_P})$ and cloud platform $\mathrm{C\_P}_{\mathrm{DM}}$.

Begin

While (1)

Initial parameters: $L_{\mathrm{CP}}(j)$, $L_{\mathrm{CP}}(a, b)$;

Obtain the value of N

While $i \geqslant N$

If QE(i) = 1 or QE(i) = 3

Count time;

i++;

End while

Computing the value of T and T_h based on QE(i) and $T(i)$;

Obtain the service opportunity request drive factor D_{rv};

Constructing the service opportunity driven tree;

Analyzing the data mining complexity DM;

If DM is larger than $\mathrm{DM_{th}}$

Updating the tree;

Else

If $S_{\mathrm{pace}}(\mathrm{DA_{set}}(i)) = S_{\mathrm{pace}}(\mathrm{DA_{set}}(j))$

Time synchronization

Else

Time asynchronous

End if

End if

Computing multicloud coordination flux $\varphi[n]$

Select the optimal clouds to construct the cloud platform $\mathrm{C\_P_{DM}}$

Activate the multicloud segmentation data and obtain the final independent data set $\mathrm{DA_{set}}(\mathrm{C\_P})$ after the fusion data

End while

End

5. Performance Evaluation

In order to verify the cloud platform performance and data mining performance of the proposed cloud platform based on opportunities in the service of mobile Internet driver and data mining of application perception (CP-SAD), we designed the experiment with 1 GB storage space and 100 data sets randomly distributed in 10 pieces of cloud equipment and evaluated with the transaction data base server. Verification metrics include the system running time and memory occupancy rate, the time required for data clustering, and the average clustering quality, which compared with the cloud computing mechanism with distributed data mining (CP-DDM); the results are as shown in Figure 7.

From Figure 7(a), we found that CP-DDM mechanism system running time increases linearly with increasing data set, while the operating time of the proposed CP-SAD mechanism remained stable, because the use of service opportunities driven model and real-time analysis of the characteristics of the driver can accurately grasp the service request, optimize the system initialization and service driven and application aspects of perception in the delay, not only shorten the operation time but also smooth the delay jitter.

Figure 7(b) proved that the proposed mechanism could optimize the memory capacity, because of the application of perceptual features extraction mechanism, which can not only be aware of the applications required data scale and distribution, but also optimize the composition of the cloud platform, by combining with the opportunity to serve driver tree. The proposed mechanism can increase the utilization ratio of memory, avoiding the cloud platform system performance degradation caused by memory overflow.

From Figure 7(c), we can see that the required time of data mining clustering increases first and then decreases with the increase of cloud scale activation. This is because the large computation result is more time consuming in initialization phase. However, the proposed CP-SAD mechanism can complete cloud equipment initialization and reduce data mining time faster and earlier. The CP-SAD mechanism can complete initialization after the first cloud device activation, and CP-DDM mechanism complete initialization when three-cloud device is activated, which wasted a lot of time for data mining clustering stage. This is because the distributed data mining is composed of a cloud platform data distributed storage and application features and service requests are not built for nonlinear mapping, resulting in data mining delay being large.

Clustering quality average is shown in Figure 7(d) with the increase of the size of the cloud. In the activation of the cloud number from 0 to 4, the average clustering quality of two mechanisms changes in the same law, and the gap is about 10 percentage points. With a further increase in the cloud, the average clustering quality of CP-DDM mechanism began to decline, until 10 clouds are activated to rise to 60%. However, the average clustering quality of CP-SAD mechanism has stalled when there are four clouds and then increases linearly. This is because the CP-SAD mechanism can effectively grasp the opportunity to drive characteristics and extraction of perceptual features and construct, for mobile Internet applications to provide robust and efficient data mining service cloud platform, mobile Internet applications for improving the efficient and reliable quality of service.

The iteration number analysis results of time complexity of CP-DDM and CP-SAD were shown in Figure 7(e). The proposed mechanism could compute the data efficiently and mine the data accurately with less iteration number than one of CP-DDM, which has the very low time complexity.

6. Conclusions

In order to improve the cloud platform of the mobile Internet services with quality assurance and data mining efficiency, we researched the opportunities in the service of mobile Internet driver, as well as the reliable and efficient application aware

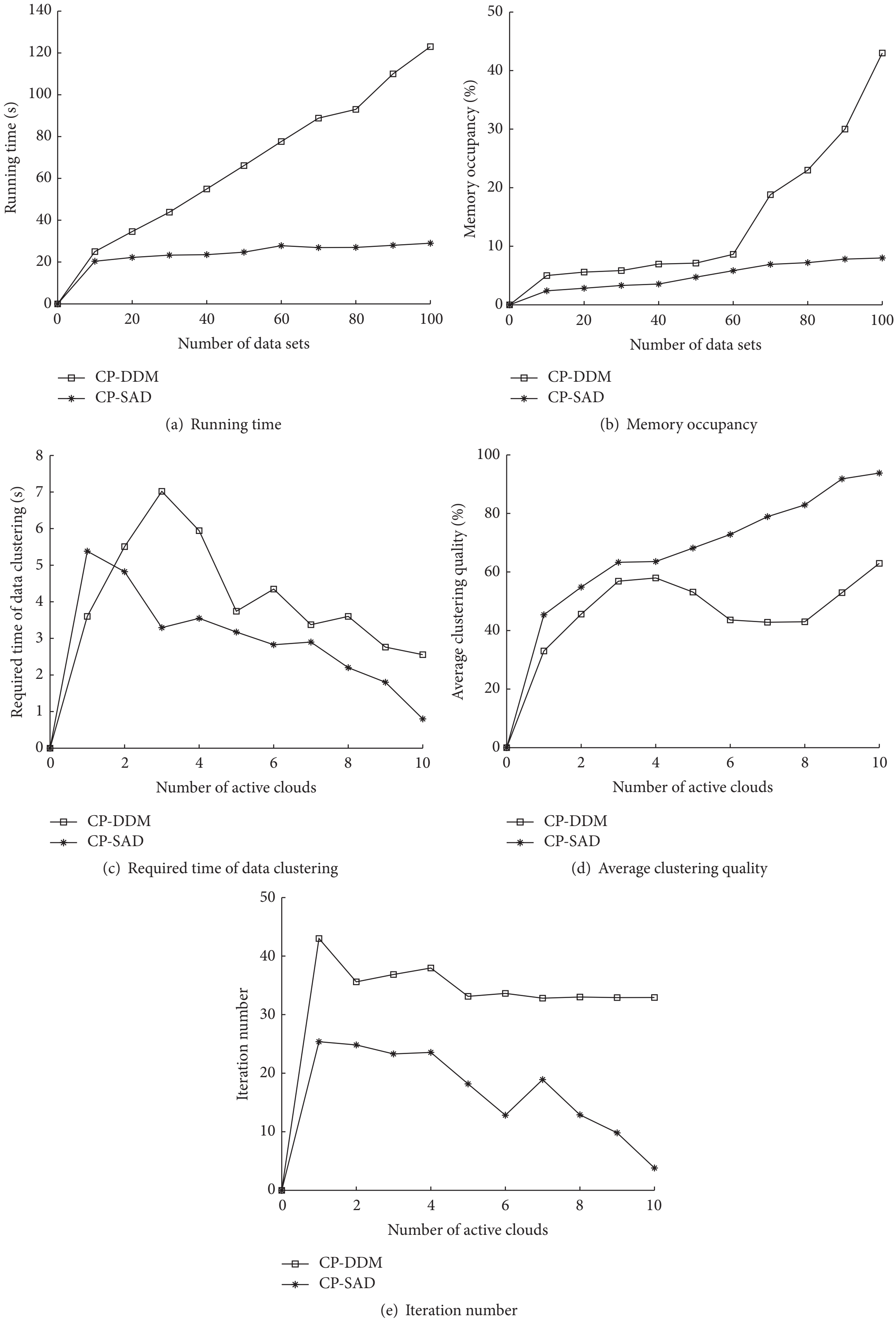


Figure 7: Performance evaluation.

data mining. Firstly, a data selection mechanism is designed based on the analysis of the characteristics of the mobile Internet service opportunity driving. Second, based on the relationship between the Internet service and data set, the nonlinear mapping between cloud and the Internet channel was established, and it established an Internet application perception model. Third, based on the characteristics of service opportunity driven and application perception, a cloud platform is proposed to provide efficient and robust data mining services for mobile Internet applications. The experimental results show that the proposed mechanism in the runtime and memory occupancy rate, data clustering, and the time required for the average clustering quality are better than the cloud platform based on distributed data mining.

Conflict of Interests

The author declares that there is no conflict of interests regarding the publication of this paper.

References

[1] A. Cuzzocrea, "Models and algorithms for high-performance data management and mining on computational grids and clouds," *Journal of Grid Computing*, vol. 12, no. 3, pp. 443–445, 2014.

[2] A. K. Tripathy, J. Adinarayana, K. Vijayalakshmi et al., "Knowledge discovery and Leaf Spot dynamics of groundnut crop through wireless sensor network and data mining techniques," *Computers and Electronics in Agriculture*, vol. 107, pp. 104–114, 2014.

[3] C.-S. Chen, W.-Y. Liang, and H.-Y. Hsu, "A cloud computing platform for ERP applications," *Applied Soft Computing*, vol. 27, pp. 127–136, 2015.

[4] J. Carretero and J. G. Blas, "Introduction to cloud computing: platforms and solutions," *Cluster Computing*, vol. 17, no. 4, pp. 1225–1229, 2014.

[5] H. Qi, M. Shiraz, J.-Y. Liu, A. Gani, Z. Abdul Rahman, and T. A. Altameem, "Data center network architecture in cloud computing: review, taxonomy, and open research issues," *Journal of Zhejiang University SCIENCE C*, vol. 15, no. 9, pp. 776–793, 2014.

[6] G. Rakocevic, Z. Tafa, and V. Milutinovic, "A novel approach to data mining in wireless sensor networks," *Ad-Hoc & Sensor Wireless Networks*, vol. 22, no. 1-2, pp. 21–40, 2014.

[7] T. Xu, W. Xiang, Q. Guo, and L. Mo, "Mining cloud 3D video data for interactive video services," *Mobile Networks and Applications*, vol. 20, no. 3, pp. 320–327, 2015.

[8] J. Ronowicz, M. Thommes, P. Kleinebudde, and J. Krysiński, "A data mining approach to optimize pellets manufacturing process based on a decision tree algorithm," *European Journal of Pharmaceutical Sciences*, vol. 73, pp. 44–48, 2015.

[9] R. Ghosh, F. Longo, R. Xia, V. K. Naik, and K. S. Trivedi, "Stochastic model driven capacity planning for an infrastructure-as-a-service cloud," *IEEE Transactions on Services Computing*, vol. 7, no. 4, pp. 667–680, 2014.

[10] Z. Ye, A. Bouguettaya, and X. Zhou, "Economic model-driven cloud service composition," *ACM Transactions on Internet Technology*, vol. 14, no. 2-3, Article ID 2651420, pp. 255–273, 2014.

[11] A. Boukerche, A. A. F. Loureiro, E. F. Nakamura, H. A. B. F. Oliveira, H. S. Ramos, and L. A. Villas, "Cloud-assisted computing for event-driven mobile services," *Mobile Networks & Applications*, vol. 19, no. 2, pp. 161–170, 2014.

[12] H.-Y. Chang, C.-C. Lai, and Y.-W. Lin, "A fast SVC-based channel-recommendation system for an IPTV on a cloud and P2P hybrid platform," *Computer Journal*, vol. 57, no. 12, pp. 1776–1789, 2013.

[13] D. Petcu, S. Panica, C. Crăciun, M. Neagul, and C. Şandru, "Cloud resource orchestration within an open-source component-based platform as a service," *Concurrency and Computation: Practice and Experience*, vol. 27, no. 9, pp. 2443–2469, 2015.

[14] J. M. Alcaraz Calero and J. G. Aguado, "MonPaaS: an adaptive monitoring platformas a service for cloud computing infrastructures and services," *IEEE Transactions on Services Computing*, vol. 8, no. 1, pp. 65–78, 2015.

Hindawi Publishing Corporation
Journal of Electrical and Computer Engineering
Volume 2015, Article ID 398709, 9 pages
http://dx.doi.org/10.1155/2015/398709

Research Article

QoE Guarantee Scheme Based on Cooperative Cognitive Cloud and Opportunistic Weight Particle Swarm

Weihang Shi

College of Software Technology, Zhengzhou University, Zhengzhou, Henan 450002, China

Correspondence should be addressed to Weihang Shi; shiwh@zzu.edu.cn

Received 15 July 2015; Revised 8 September 2015; Accepted 14 September 2015

Academic Editor: James Nightingale

It is well known that the Internet application of cloud services may be affected by the inefficiency of cloud computing and inaccurate evaluation of quality of experience (QoE) seriously. In our paper, based on construction algorithms of cooperative cognitive cloud platform and optimization algorithm of opportunities weight particle swarm clustering, the QoE guarantee mechanism was proposed. The mechanism, through the sending users of requests and the cognitive neighbor users' cooperation, combined the cooperation of subcloud platforms and constructed the optimal cloud platform with the different service. At the same time, the particle swarm optimization algorithm could be enhanced dynamically according to all kinds of opportunity request weight, which could optimize the cooperative cognitive cloud platform. Finally, the QoE guarantee scheme was proposed with the opportunity weight particle swarm optimization algorithm and collaborative cognitive cloud platform. The experimental results show that the proposed mechanism compared is superior to the QoE guarantee scheme based on cooperative cloud and QoE guarantee scheme based on particle swarm optimization, compared with optimization fitness and high cloud computing service execution efficiency and high throughput performance advantages.

1. Introduction

With the rapid development of cloud computing technology and diversification of user requirements of mobile Internet, how to provide the scalable service and how to optimize the hardware and software platforms have been the hot research issue [1]. Particularly, according to the computing service of the cloud platform [2], we have to obtain the comprehensive understanding of user experience requirements and quality evaluation, which could not only create the maximum interests of services providers but also satisfy the requirements of users. The cloud platform could be adjusted with the real-time dynamic state information adaptively, to further enhance the cloud computing service support capabilities. A series of research results have been obtained, such as Seamless QoE Support [3], Context-Aware QoE [4], and Policy-Based and QoE-Aware Content Delivery [5].

About cooperative cognition technology and cloud computing algorithm, Kaewpuang et al. [6] provided the guarantee of mobile applications by studying and sharing the radio and computing with mobile cloud computing environment. Lei et al. [7] proposed a novel cognitive cooperative vehicular ad hoc network for solving the contradiction between the increasing demand of diverse vehicular wireless applications and the shortage of spectrum resource. Feteiha and Hassanein [8] addressed the area of heterogeneous wireless relaying vehicular clouds and devised an advanced vehicular relaying technique for enhanced connectivity in densely populated urban areas. The spectrum leasing strategy based on cooperative relaying for cognitive radio networks was proposed in [9]. The implementation of the CCRN framework applied to IEEE 802.11 WLANs was proposed by [10].

On the other hand, the fast cloud-based web service composition approach was proposed according to the characteristics of notion of Skyline [11]. The task-based system load balancing method using particle swarm optimization (TBSLB-PSO) was proposed for achieving system load balancing by only transferring extra tasks from an overloaded VM instead of migrating the entire overloaded VM [12].

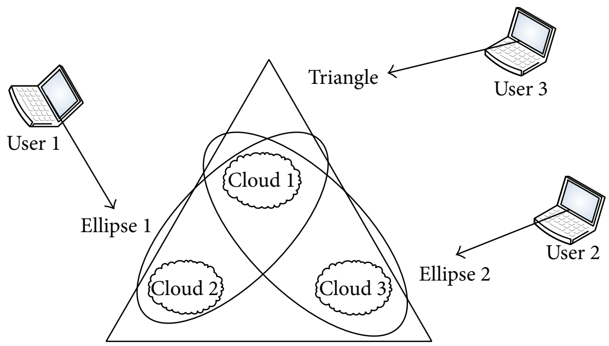


FIGURE 1: Architecture of cloud platform.

This paper puts forward a cloud theory-based particle swarm optimization (CTPSO) algorithm proposed by Ma and Xu [13], which is used to solve a variant of the vehicle routing problem. Multiple Strategies Based Orthogonal Design PSO was presented by Qin et al. [14], which is used with a small probability to construct a new exemplar in each iteration. The hybrid methods for fuzzy clustering were proposed in [15].

In view of the deficiency of the above research results, we studied the cloud platform construction method based on cooperative cognition of data processing units. Then we researched the optimization algorithm with opportunistic weight particle swarm. Finally, we proposed a reliable and efficient QoE guarantee scheme.

The rest of the paper is organized as follows. Section 2 describes the cooperative cognitive cloud. In Section 3, we design the opportunistic weight particle swarm. The QoE guarantee scheme is proposed in Section 4. Experiment results are given in Section 5. Finally, we conclude the paper in Section 6.

2. Cooperative Cognitive Cloud Platform

The cloud platform can store the large scale data. For achieving load balance and dynamic adjustment, the cloud platform would provide different services for different Internet users. The cloud platform can avoid the system performance degradation, which may be caused by user service competition. The cloud platform can also satisfy the service quality reliability simultaneously. The cloud platform architecture is shown in Figure 1.

In Figure 1, there are three clouds in the cloud platform. Every cloud is made up of the computer. According to the needs of different users, the sub platform is different. The service of User 1 is provided by elliptical 1 cloud platform consisting of cloud 1 and cloud 2. The service of User 2 is provided by elliptical 2 cloud platform consisting of cloud 1 and cloud 3. The service of User 3 is provided by triangle cloud platform consisting of cloud 1, cloud 2, and cloud 3.

Cloud platform is defined as the state vector $S_{\text{cp}} = \{C_N, \text{RAM}_{\text{MAX}}, \text{RAM}_{\text{AVE}}, \text{DIS}_{\text{MAX}}, \text{DIS}_{\text{AVE}}, \text{CH}_{\text{MAX}}, \text{CH}_{\text{AVE}}, \text{CPU}_{\text{MAX}}, \text{CPU}_{\text{AVE}}\}$.

Here, C_N denotes the number of computers in subcloud platform, RAM_{MAX} denotes the total member size of subcloud platform, RAM_{AVE} denotes the average member size, DIS_{MAX} denotes the total disk size, DIS_{AVE} denotes the average disk size, CH_{MAX} denotes the maximum channel bandwidth, CH_{AVE} denotes the average channel bandwidth, CPU_{MAX} denotes the maximum CPU operating frequency, and CPU_{AVE} denotes the average CPU operating frequency.

The user cloud computing service request is set to 1. The request includes 4 metrics, which are E_{RAM}, E_{DIS}, E_{CH}, and E_{CPU}. Here, $E_{\text{RAM}} + E_{\text{DIS}} + E_{\text{CH}} + E_{\text{CPU}} = 1$.

The matching vector F between cloud platform and user demand is defined as $\{C_N, E_{\text{RAM}} * \text{RAM}_{\text{MAX}}, (E_{\text{RAM}}/C_N) * \text{RAM}_{\text{AVE}}, E_{\text{DIS}} * \text{DIS}_{\text{MAX}}, (E_{\text{DIS}}/C_N) * \text{DIS}_{\text{AVE}}, E_{\text{CH}} * \text{CH}_{\text{MAX}}, (E_{\text{CH}}/C_N)\text{CH}_{\text{AVE}}, E_{\text{CPU}} * \text{CPU}_{\text{MAX}}, (E_{\text{CPU}}/C_N)\text{CPU}_{\text{AVE}}\}$.

Hence, cloud platform and user request evaluation results are as shown in the following formula:

$$\begin{aligned} a &= E_{\text{RAM}}\text{RAM}_{\text{MAX}} + E_{\text{DIS}}\text{DIS}_{\text{MAX}}, \\ b &= E_{\text{CH}}\text{CH}_{\text{MAX}} + E_{\text{CPU}}\text{CPU}_{\text{MAX}}, \\ E_{(S_{\text{CP}},\text{User})} &= \frac{a+b}{C_N}. \end{aligned} \tag{1}$$

In order to satisfy the request of users, the cloud platform load is shown in the following:

$$\begin{aligned} \text{RAM}_T &= \sum_{t=1}^{T}\sum_{i=1}^{C_N} E^t{}_{\text{RAM}}\text{RAM}^t_{\text{AVE}}, \\ \text{DIS}_T &= \sum_{t=1}^{T}\sum_{i=1}^{C_N} E^t{}_{\text{DIS}}\text{DIS}^t_{\text{AVE}}, \\ \text{CH}_T &= \sum_{t=1}^{T}\sum_{i=1}^{C_N} E^t{}_{\text{CH}}\text{RAM}^t_{\text{CH}}, \\ \text{CPU}_T &= \sum_{t=1}^{T}\sum_{i=1}^{C_N} E^t{}_{\text{CPU}}\text{RAM}^t_{\text{CPU}}. \end{aligned} \tag{2}$$

Whether the cloud platform is able to satisfy the user request or not could be judged by the following formula:

$$\begin{aligned} &Y \;\left(\mathrm{RAM}_T + \mathrm{DIS}_T + \mathrm{CH}_T + \mathrm{CPU}_T\right) \le E_{(S_{\mathrm{CP}},\mathrm{User})}, \\ &N \;\left(\mathrm{RAM}_T + \mathrm{DIS}_T + \mathrm{CH}_T + \mathrm{CPU}_T\right) > E_{(S_{\mathrm{CP}},\mathrm{User})}. \end{aligned} \tag{3}$$

Here, Y denotes that the could platform can satisfy the requirements and N denotes that it cannot do this.

After the subcloud platform satisfies the requirements, the user should broadcast the signal to clouds. The neighbor users could listen to the channel and receive the signal. The collection of service requests for multiple neighbor users is shown in the following formula:

$$U_R^{N_U} = \bigcup_{i=1}^{N_U} \sum_{i=1}^{4} E_i {U^i}_R. \tag{4}$$

Here, $U_R^{N_U}$ denotes the collection of user cognitive service requests, N_U denote the user number of joining the cognitive networks, E denote the 4 aspects of comprehensive evaluation of user's cloud computing service request, and U_R denote the evaluation value of a neighbor's service request.

The channel bandwidth is shown in formula (5) between the sending request user and the cognitive users:

$$\begin{aligned} S_T &= \sum_{i=1}^{C_N} {\mathrm{CH}^i}_T, \\ S_{\mathrm{AVE}} &= \frac{a+b}{C_N}, \\ a &= E_{\mathrm{RAM}}\mathrm{RAM}_{\mathrm{AVE}} + E_{\mathrm{DIS}}\mathrm{DIS}_{\mathrm{AVE}}, \\ b &= E_{\mathrm{CH}}\mathrm{CH}_{\mathrm{AVE}} + E_{\mathrm{CPU}}\mathrm{CPU}_{\mathrm{AVE}}, \\ B_W &= \sum_{i=1}^{N_U}\sum_{j=1}^{C_N} \ln\left({S^i}_T {S^j}_{\mathrm{AVE}}\right). \end{aligned} \tag{5}$$

Here, S_T is used to analyze the resource of subcloud platform and S_{AVE} is used to analyze the average data processing ability of subcloud platform. The channel bandwidth could be obtained by the combination of neighbor users and subcloud platform resources. So, the cloud platform could be constructed based on the cooperative cognition of sending request user, cognitive users, and computers of subcloud platform. The architecture is shown in Figure 2.

To sum up, the optimization of cloud resource management platform could be provided by the cooperative control of cloud platform data processing units and cooperative transmission of the sending request user and the cognition neighbor users, which is shown in the following formula:

$$\begin{aligned} &\text{minimize:} \quad \sum_{i=1}^{\mathrm{SCP}_N} {S^i}_{\mathrm{CP}} \\ &\text{subject to:} \quad \begin{cases} \displaystyle\sum_{i=1}^{N_U}\sum_{j=1}^{C_N} \ln\left({E^i}_{\mathrm{RAM}}{\mathrm{RAM}^j}_{\mathrm{MAX}}\right) \ge E\left[\sum_{j=1}^{C_N}{\mathrm{RAM}^j}_{\mathrm{AVE}}\right] \\ \displaystyle\sum_{i=1}^{N_U}\sum_{j=1}^{C_N} \ln\left({E^i}_{\mathrm{DIS}}{\mathrm{DIS}^j}_{\mathrm{MAX}}\right) \ge E\left[\sum_{j=1}^{C_N}{\mathrm{DIS}^j}_{\mathrm{AVE}}\right] \\ \displaystyle\sum_{i=1}^{N_U}\sum_{j=1}^{C_N} \ln\left({E^i}_{\mathrm{CH}}{\mathrm{CH}^j}_{\mathrm{MAX}}\right) \ge E\left[\sum_{j=1}^{C_N}{\mathrm{CH}^j}_{\mathrm{AVE}}\right] \\ \displaystyle\sum_{i=1}^{N_U}\sum_{j=1}^{C_N} \ln\left({E^i}_{\mathrm{CPU}}{\mathrm{CPU}^j}_{\mathrm{MAX}}\right) \ge E\left[\sum_{j=1}^{C_N}{\mathrm{CPU}^j}_{\mathrm{AVE}}\right]. \end{cases} \end{aligned} \tag{6}$$

Here, the function E is used to compute the mean of all data processing unit of the relevant resources in cloud platform.

3. Opportunistic Weight Particle Swarm

Through considering the various types of user requests and the opportunistic weight, we used and improved the particle swarm optimization algorithm to realize the optimization objectives of cooperative cognitive cloud platform and guarantee quality of the user experience.

Assume that the cognitive cooperative cloud platform is a particle swarm and composed of m particles, which is denoted as $\mathrm{CC} = \{\mathrm{CC}_1, \mathrm{CC}_2, \dots, \mathrm{CC}_m\}$.

The j particle expresses the data service progress of the data packet, which is denoted by vector $KP = \{KP_1, KP_2, \dots, KP_K\}$. Here, K denotes the user experience quality of the data packet, which includes the real-time performance, reliability, size, number of hops, and distance.

The sending power of the j particle Pj is $\{P_{j1}, P_{j2}, \dots, P_{jK}\}$. The extremal optimization cloud platform

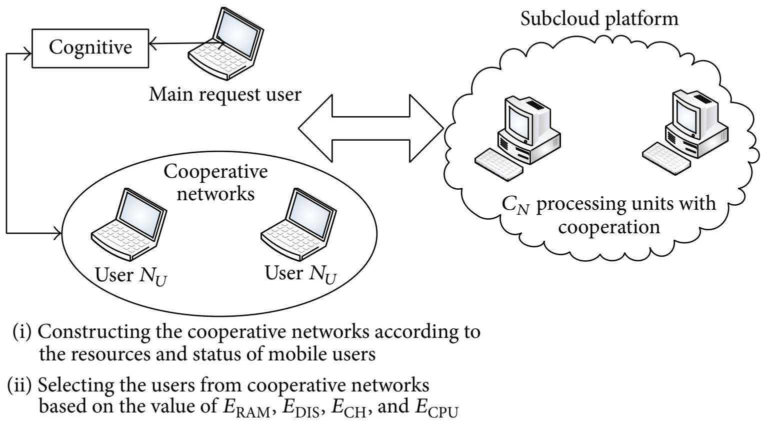


Figure 2: Cooperative cognitive cloud platform.

LO is $\{\mathrm{LO}_1, \mathrm{LO}_2, \ldots, \mathrm{LO}_K\}$. Extremal optimization cloud platform FO is $\{\mathrm{FO}_1, \mathrm{FO}_2, \ldots, \mathrm{FO}_K\}$.

Transmission power opportunity renewal model is shown in the following formula:

$$P_{j+1} = P_j + \sum_{i=1}^{K} \lambda_i KP_i + \sqrt{\left|{P_j}^2 - {P_{j-1}}^2\right|}. \quad (7)$$

Here, the optimization of j particle is realized based on j and $j-1$ particle. The sending power could be updated with the opportunistic dynamical scheme.

Resource opportunity renewal model is shown in the following formula:

$$\begin{aligned}
\mathrm{RAM}_{j+1} &= \sum_{i=1}^{C_N} E^j{}_{i,\mathrm{RAM}} \sum_{l=1}^{K} KP_l \mathrm{RAM}^j_{i,\mathrm{AVE}}, \\
\mathrm{DIS}_{j+1} &= \sum_{i=1}^{C_N} E^j{}_{i,\mathrm{DIS}} \sum_{l=1}^{K} KP_l \mathrm{DIS}^j_{i,\mathrm{AVE}}, \\
\mathrm{CH}_{j+1} &= \sum_{i=1}^{C_N} E^j{}_{i,\mathrm{CHS}} \sum_{l=1}^{K} KP_l \mathrm{CH}^j_{i,\mathrm{AVE}}, \\
\mathrm{CPU}_{j+1} &= \sum_{i=1}^{C_N} E^j{}_{i,\mathrm{CPU}} \sum_{l=1}^{K} KP_l \mathrm{CPU}^j_{i,\mathrm{AVE}}.
\end{aligned} \quad (8)$$

Request weight updating model is shown in the following formula:

$$E^{j+1}{}_{(S_{\mathrm{CP}},\mathrm{User})} = \frac{\sum_{i=1}^{K} KP_i \left(E^j{}_{(S_{\mathrm{CP}},\mathrm{User})} + E^{j-1}{}_{(S_{\mathrm{CP}},\mathrm{User})}\right)}{\sqrt{N_U C_N \left|E^j{}_{(S_{\mathrm{CP}},\mathrm{User})} - E^{j-1}{}_{(S_{\mathrm{CP}},\mathrm{User})}\right|}}. \quad (9)$$

In order to guarantee and sustain dynamic characteristic and diversity of cloud platform in the particle swarm dynamic evolution process, based on subcloud platform optimization extreme and cloud platform extremal optimization, the real-time evaluation function is established and shown in the following formula:

$$\begin{aligned}
\mathrm{RH}\big|_{j \to j+l} &= \frac{\sqrt{j+l}}{l} \sum_{i=1}^{K} KP_i \mathrm{RH}_{j+i}, \\
\mathrm{RH}_j &= \log_2 \left(\sum_{i=1}^{K} KP_i\right)^j \sum_{i=1}^{K} \frac{\mathrm{LO}^j{}_i}{\mathrm{FO}_i}.
\end{aligned} \quad (10)$$

Flow of the cooperative cognitive cloud platform with opportunity weight particle swarm optimization algorithm is shown in Figure 3; the optimal fitness is shown in Figure 4. From the results of Figure 4, based on the best adaptation degree and the average fitness, this opportunity weight particle group optimization algorithm can satisfy the demand for data optimization cooperative cognitive cloud platform well. With the growing particle size, the best fitness increases gradually and the data clustering effect remains good, which not only enhance the local search ability but also achieve good global optimization effect.

4. QoE Guarantee Scheme

In view of the cooperative cognitive cloud platform, with the opportunity weight particle swarm optimization algorithm, the QoE guarantee mechanism is established from the perspective of the user. The QoE guarantee problem can be transformed into a multiobjective optimization problem, as shown in the following formula:

$$\begin{aligned}
&\mathrm{Min} \quad \mathrm{FO}(kp) \\
&\mathrm{s.t.} \quad kp \in KP.
\end{aligned} \quad (11)$$

Here, based on the global optimization, the user experience quality guarantee scheme is established based on multiobjective optimization.

Based on the user experience quality optimization model, we defined the user data service expectations of the target function UE and the expected execution efficiency of EE.

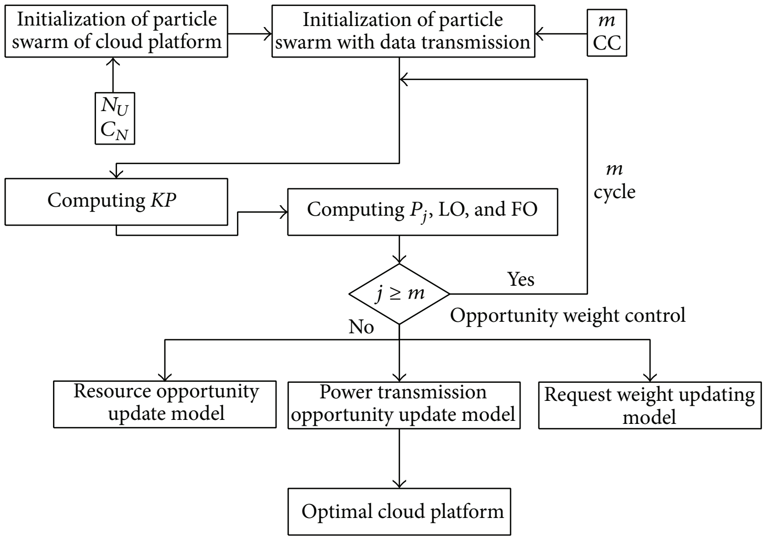


Figure 3: Flow of the weight particle swarm optimization algorithm.

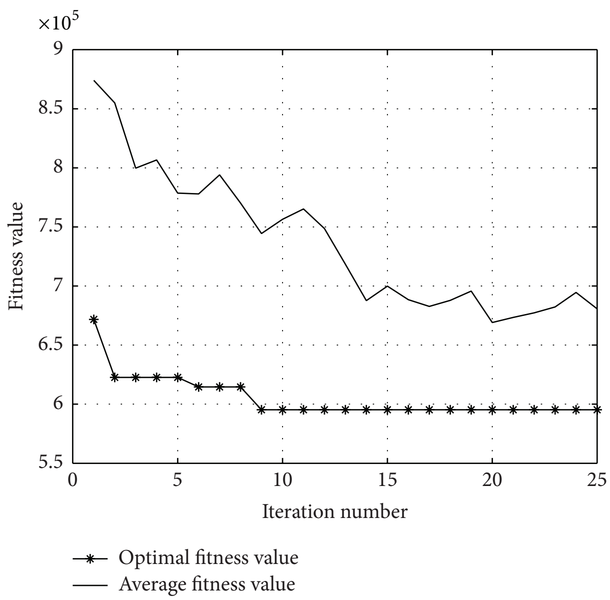


Figure 4: Particle swarm optimization.

After sending the service request to cognitive cooperative cloud platform by users, the expectation objective function UE of user data service is used to realize the optimization goal based on the cloud platform during initialization, particle swarm optimization, updating a series of operations, and so forth. This function is shown in formula (12).

The desired execution efficiency of EE: this means the execution time of the data service tasks and the ratio of the required cloud platform resources, which are shown in formula (13):

$$\mathrm{UE} = \ln\left(\frac{\max\left\{\mathrm{RAM}^{j}{}_{\mathrm{MAX}}, \mathrm{DIS}^{j}{}_{\mathrm{MAX}}, \mathrm{CH}^{j}{}_{\mathrm{MAX}}, \mathrm{CPU}^{j}{}_{\mathrm{MAX}}\right\} - S_{\mathrm{AVE}}}{1 + \max\left\{\mathrm{RAM}^{j}{}_{\mathrm{AVE}}, \mathrm{DIS}^{j}{}_{\mathrm{AVE}}, \mathrm{CH}^{j}{}_{\mathrm{AVE}}, \mathrm{CPU}^{j}{}_{\mathrm{AVE}}\right\}}\right), \tag{12}$$

$$\mathrm{EE} = \ln\left(\frac{\mathrm{UE} - \sum_{i=1}^{K} KP_i\left(E^{j}{}_{(S_{\mathrm{CP}},\mathrm{User})} + E^{j-1}{}_{(S_{\mathrm{CP}},\mathrm{User})}\right)}{1 + \left|E^{j}{}_{(S_{\mathrm{CP}},\mathrm{User})} - E^{j-1}{}_{(S_{\mathrm{CP}},\mathrm{User})}\right|}\right). \tag{13}$$

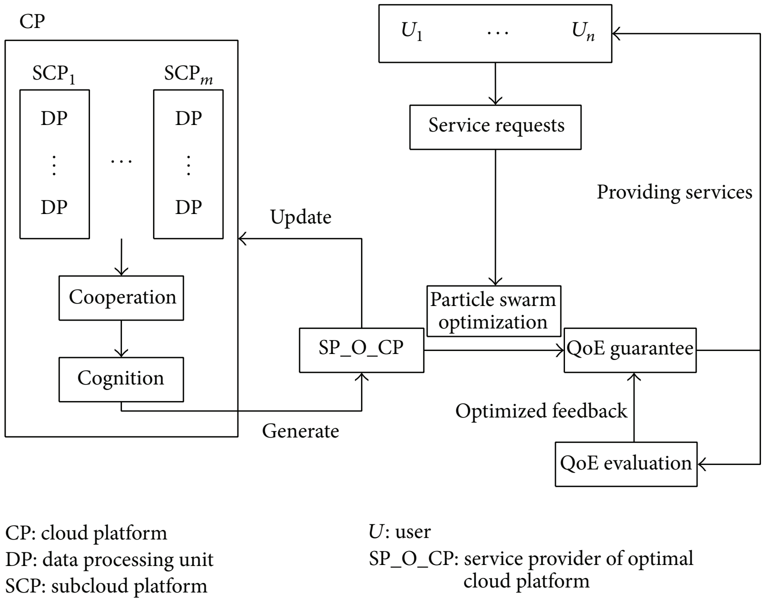


Figure 5: QoE guarantee mechanism architecture.

Above all, based on formulas (11), (12), and (13), the objectives of QoE guarantee mechanism are shown in formula (14), which could minimize the parameters of UE, P, and FO and maximize EE:

$$\begin{aligned}
\text{Min UE} &= \ln \sum_{i=1}^{K} KP_i \left(\frac{\max\left\{\text{RAM}^j{}_{\text{MAX}}, \text{DIS}^j{}_{\text{MAX}}, \text{CH}^j{}_{\text{MAX}}, \text{CPU}^j{}_{\text{MAX}}\right\} - S_{\text{AVE}}}{1 + \max\left\{\text{RAM}^j{}_{\text{AVE}}, \text{DIS}^j{}_{\text{AVE}}, \text{CH}^j{}_{\text{AVE}}, \text{CPU}^j{}_{\text{AVE}}\right\}} \right), \\
\text{Max EE} &= \ln \left(\frac{\text{UE} - \left(E^j{}_{(S_{\text{CP}},\text{User})} + E^{j-1}{}_{(S_{\text{CP}},\text{User})}\right)}{1 + \left|E^j{}_{(S_{\text{CP}},\text{User})} - E^{j-1}{}_{(S_{\text{CP}},\text{User})}\right|} \right)^{1/\sum_{i=1}^{K} KP_i}, \\
\text{Min } P &= \frac{1}{P_N} \sum_{j=1}^{P_N} \lambda_j \sum_{i=1}^{K} KP_i + \sqrt{\left|P_j{}^2 - P_{j-1}{}^2\right|} P_j, \\
\text{Min FO} &= \frac{\alpha \text{UE} + \beta \text{EE} + \gamma P}{\sqrt{\alpha^2 \beta^2 \gamma^2}}.
\end{aligned} \tag{14}$$

The architecture is shown in Figure 5. The workflow of the proposed scheme is described as follows.

Step 1. Construct the cooperative cognitive cloud platform according to the cloud platform state and user requests.

Step 2. Divide the subcloud platform by selecting cooperative users and updating the cloud state.

Step 3. Execute the opportunistic weight particle swarm algorithm.

Step 4. Establish the QoE guarantee mechanism from the user's point of view.

5. Performance Evaluation

In order to validate the performance of the QoE guarantee mechanism based on cooperative cognitive clouds and opportunity weight particle swarm (Q-CCC-OWPW), we designed and developed 6 clouds in the rectangular area of 4000 meters ∗ 4500 meters. Each cloud includes several computers. The setting of the experiment is shown in Table 1. The topology of experiment is illustrated as in Figure 6.

The QoE guarantee scheme based on opportunity weight particle swarm (Q-OWPW) and QoE guarantee scheme based on cooperative cognitive clouds (Q-CCC) and Q-CCC-OWPW are compared and analyzed from the aspects of

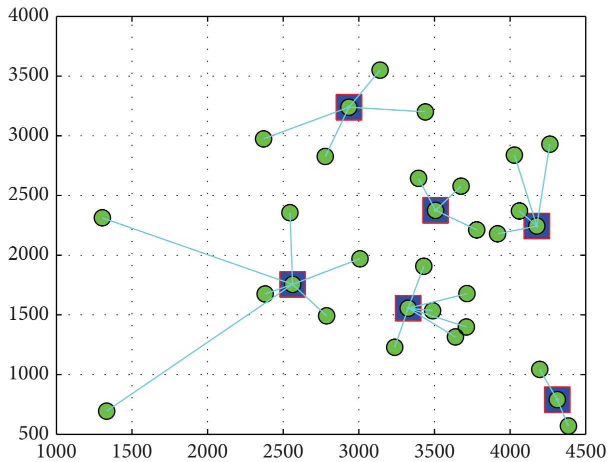


Figure 6: Experiment platform.

Table 1: Cloud settings.

Cloud	Number of computers	Location	Configuration	Disk usage
1	4	(2500, 2400)	Same	30%
2	5	(3400, 2600)	Same	10%
3	5	(4100, 2400)	Same	20%
4	3	(2400, 1700)	Same	25%
5	6	(3500, 1600)	Same	15%
6	3	(4300, 800)	Same	35%

optimal adaptive degree, throughput rate, and user's demand error and execution efficiency. The results are shown in Figure 7.

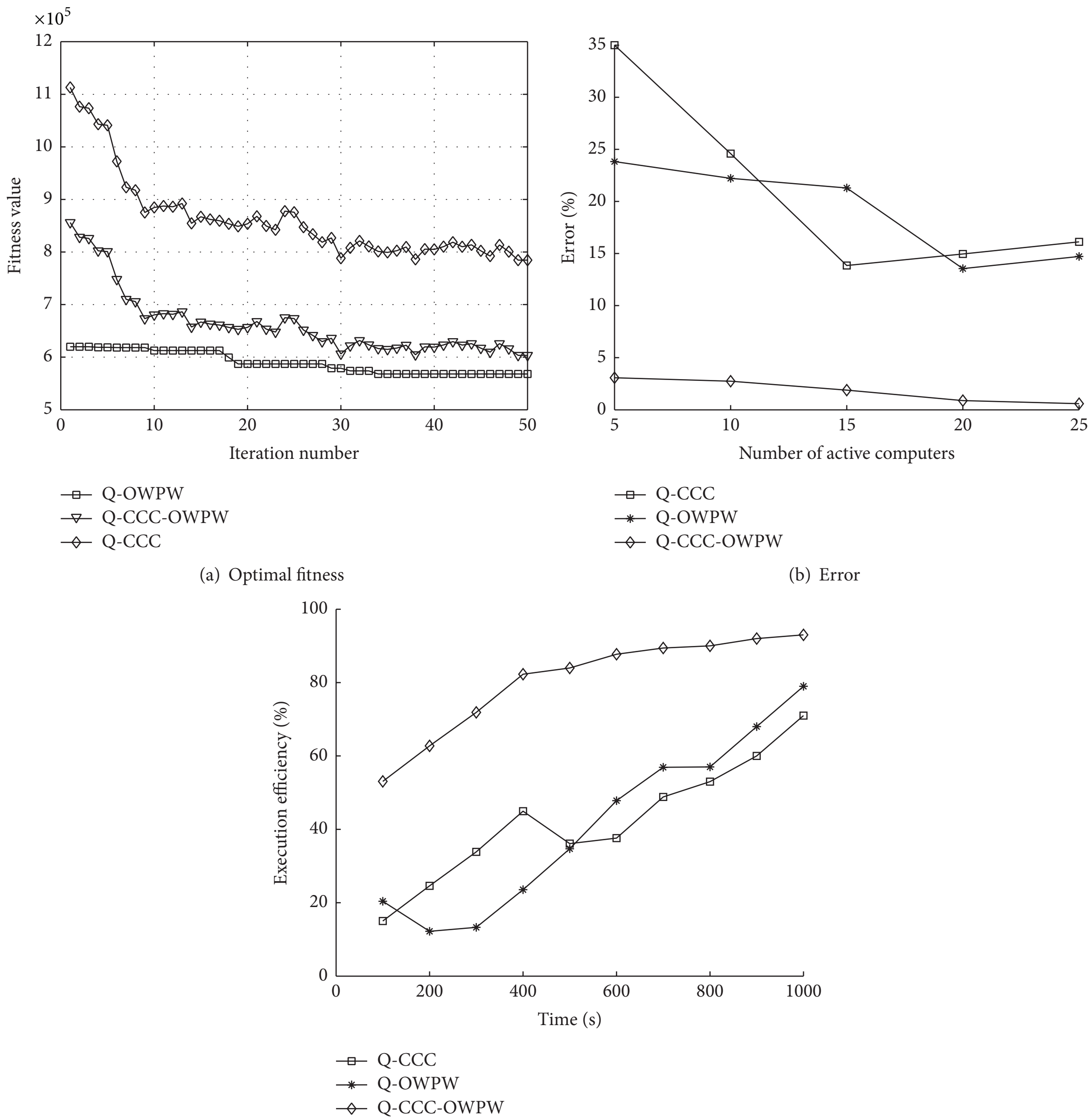
With the increase of the number of iterations, the optimal fitness curves of three QoE guarantee mechanisms are shown in Figure 7(a). Among them, Q-CCC-OWPW has the best fitness and gradually tends to be stable. This has benefited from the cooperation of the data processing computer and the opportunity particle swarm optimization, not only the perception of the cloud platform resources and dynamic adjustment according to the channel quality. However, Q-CCC can only sense the channel quality and Q-OWPW can only perceive the cloud platform resources, so the performance is poor.

Figure 7(b) gives the error of the throughput of the three mechanisms of and the throughput of the user's request. It was found that the error of Q-CCC and Q-OWPW is greater than that of the proposed Q-CCC-OWPW. When the active computer number is between 11 and 19, the performance of Q-OWPW is better than the one of Q-CCC. This is because the Q-OWPW cannot obtain the information of cloud resources.

User service request execution efficiency is shown in Figure 7(c). The execution efficiency of the three mechanisms gradually increases as the experimental time increases. This is because the cloud platform initialization, the cloud platform and user channel competition, and resource allocation stage require a certain period of time; after the completion of the above operations, the execution efficiency of the system increased significantly. However, the efficiency of Q-OWPW prior to 500 s is lower than that of Q-CCC, which is due to the long time required for the optimization of particle swarm optimization. After more than 500 s, the efficiency of Q-CCC is higher than that of Q-OWPW. This is because the particle swarm optimization algorithm obtains the gain of data transmission. For Q-CCC-OWPW, the construction of the opportunity weight particle swarm optimization, to achieve global optimization, has higher efficiency.

6. Conclusions

In order to improve the execution efficiency of cloud and guarantee the quality of experience, we put forward the efficient and reliable QoE guarantee mechanism based on the cooperative cognitive cloud platform construction algorithms and opportunities weight particle swarm clustering optimization algorithm. First, according to the sending requests user and the cognitive neighbor users, the computers of subcloud platform through cooperative cognitive scheme cloud platform would be constructed and updated. Then, in order to realize the optimization objectives of cooperative cognitive cloud platform and improve quality of user experience, particle swarm optimization algorithm was improved for clustering by considering the user end of various types of requests the weight changes in the dynamic of opportunities. Finally, based on the cooperative cognitive cloud platform and the opportunity weight particle swarm optimization algorithm, the QoE guarantee mechanism was proposed from the user's point of view. The experimental results show that the proposed mechanism has obvious advantages in optimizing the degree of adaptation, the efficiency of the implementation of cloud computing services, and the throughput rate.



(a) Optimal fitness

(b) Error

(c) Execution efficiency

Figure 7: QoE guarantee performance.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] M. Jarschel, D. Schlosser, S. Scheuring, and T. Hoßfeld, "Gaming in the clouds: QoE and the users' perspective," *Mathematical and Computer Modelling*, vol. 57, no. 11-12, pp. 2883–2894, 2013.

[2] J. He, Y. Wen, J. Huang, and D. Wu, "On the cost-QoE tradeoff for cloud-based video streaming under Amazon EC2's pricing models," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 4, pp. 669–680, 2014.

[3] G. Kim, S. Lee, and S. G. Lee, "Seamless QoE support for mobile cloud services using IEEE802.21 MIH and the GENI future internet framework," *International Journal of Software Engineering and Knowledge Engineering*, vol. 24, no. 7, pp. 1039–1063, 2014.

[4] K. Mitra, A. Zaslavsky, and C. Ahlund, "Context-aware QoE modelling, measurement, and prediction in mobile computing systems," *IEEE Transactions on Mobile Computing*, vol. 14, no. 5, pp. 920–936, 2015.

[5] M. Olli, Y. F. Zarrar, M. Petteri, and J. Lessmann, "Policy-based and QoE-aware content delivery using Q-learning method,"

Wireless Personal Communications, vol. 83, no. 1, pp. 315–342, 2015.

[6] R. Kaewpuang, D. Niyato, P. Wang, and E. Hossain, "A framework for cooperative resource management in mobile cloud computing," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 12, pp. 2685–2700, 2013.

[7] Z. Lei, L. Tao, L. Wei, Z. Siting, and L. Jianfeng, "Cooperative spectrum allocation with QoS support in cognitive cooperative vehicular ad hoc networks," *China Communications*, vol. 11, no. 10, pp. 49–59, 2014.

[8] M. F. Feteiha and H. S. Hassanein, "Enabling cooperative relaying VANET clouds over LTE-A networks," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 4, pp. 1468–1479, 2015.

[9] K. Ma, J. Yang, G. Hu, and X. Guan, "Cooperative Relay-Aware Spectrum leasing based on Nash bargaining solution in cognitive radio networks," *International Journal of Communication Systems*, vol. 28, no. 7, pp. 1250–1264, 2015.

[10] M. B. Pandian, M. L. Sichitiu, and H. Dai, "Optimal resource allocation in random access cooperative cognitive radio networks," *IEEE Transactions on Mobile Computing*, vol. 14, no. 6, pp. 1245–1258, 2015.

[11] S. Wang, Q. Sun, H. Zou, and F. Yang, "Particle swarm optimization with skyline operator for fast cloud-based web service composition," *Mobile Networks & Applications*, vol. 18, no. 1, pp. 116–121, 2013.

[12] F. Ramezani, J. Lu, and F. K. Hussain, "Task-based system load balancing in cloud computing using particle swarm optimization," *International Journal of Parallel Programming*, vol. 42, no. 5, pp. 739–754, 2014.

[13] Y. Ma and J. Xu, "A cloud theory-based particle swarm optimization for multiple decision maker vehicle routing problems with fuzzy random time windows," *Engineering Optimization*, vol. 47, no. 6, pp. 825–842, 2015.

[14] Q. Qin, S. Cheng, Q. Zhang, Y. Wei, and Y. Shi, "Multiple strategies based orthogonal design particle swarm optimizer for numerical optimization," *Computers & Operations Research*, vol. 60, pp. 91–110, 2015.

[15] T. M. Silva Filho, B. A. Pimentel, R. M. C. R. Souza, and A. L. I. Oliveira, "Hybrid methods for fuzzy clustering based on fuzzy c-means and improved particle swarm optimization," *Expert Systems with Applications*, vol. 42, no. 17-18, pp. 6315–6328, 2015.