Two-stage cloud service optimisation model for cloud service middleware platform

Hsu-Yang Kung\textsuperscript{1}, Ting-Huan Kuo\textsuperscript{2}, Chi-Hua Chen\textsuperscript{3}, Yu-Lun Hsu\textsuperscript{4}

\textsuperscript{1}Department of Management Information Systems, National Pingtung University of Science and Technology, Pingtung, Taiwan
\textsuperscript{2}Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung, Taiwan
\textsuperscript{3}Department of Information Management, National Kaohsiung University of Science and Technology, Kaohsiung, Taiwan
E-mail: chihua0826@gmail.com

Published in The Journal of Engineering; Received on 10th November 2017; Revised on 19th December 2017; Accepted on 6th January 2018

Abstract: In this study, a cloud service optimisation model is presented for resource adaptation control, in which the cost to the user, various levels of virtual machine efficacy, and rental price are considered. A cloud service middleware that supports flexible web services by using heterogeneous platforms is proposed. Developers can conveniently register web services and deploy them to the middleware. The middleware provides an energy-minimisation load balance (EM-LB) mechanism, which considers the temperature of the machine, its power consumption, and the computing resources of virtual machines. The EM-LB mechanism maintains a safe temperature threshold and minimises power usage, thereby reducing hotspots and conserving energy. Furthermore, a two-stage cloud service optimisation model is proposed to analyse users' requirements and existed computing resources for finding the Pareto optimality, and the aspirational level is found in accordance with the Pareto optimality for the lower cost and lower execution time. The simulation results show the efficiency of the proposed cloud service optimisation model and EM-LB mechanism. Finally, the proposed EM-LB mechanism is implemented to prove the feasibility of the model.

1 Introduction

In recent years, the cloud computing industry has flourished. Worldwide enterprises are actively promoting and developing cloud services that reduce business costs and increase profits\textsuperscript{[1]}. Cloud computing services are divided into infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS). The PaaS market is the smallest segment of the cloud computing industry, but it creates a large environment of partners and developers, producing products that are easy to use, customise, and develop. More than 150 companies currently offer PaaS products. Service-oriented architecture (SOA) is the most popular type of paradigm, and it comprises some web-service standardised components that provide a flexible and uniform interface, integrating software from multiple sources. Web service is a basic tool that applies to open data formats (e.g. XML, XML Schema) and defines the standard service interfaces of web communications protocols. However, traditional middleware architecture cannot supply dynamically adaptable computing resources based on the demands of the cloud service without wasting resources and reducing the quality of service. The rapid growth of the cloud computing industry has increased network traffic, information, and the need for data storage capacity, and the energy consumption of data centres has increasingly become a concern.

A cloud service middleware (CSM) platform is proposed that integrates an energy-minimisation load balance (EM-LB) mechanism to disperse user service requests, maintain system stability, and reduce energy consumption. According to the variation in user requests, the cloud service optimisation model optimises the computing resources. Furthermore, previous studies focused on finding the Pareto optimality according to the limited resources\textsuperscript{[2–9]}, but the aspirational level has not been investigated. This study proposes a two-stage cloud service optimisation model to analyse users' requirements and existed computing resources for finding the Pareto optimality, and the aspirational level is found in accordance with the Pareto optimality for the lower cost and lower execution time.

This paper is organised as follows: In Section 2, this study provides background knowledge of SOA and load balancing mechanisms. In Section 3, this study proposes the architecture of CSM. This study gives the experiment results and discussions in Section 4. Finally, this study summarised the contributions of this paper in Section 5.

2 Literatures review

This section discusses the previous studies of SOA and load balancing mechanisms.

2.1 Cloud computing and service-oriented architecture

SOA is a flexible and software platform architecture. In service-oriented computing (SOC), basic services are used to construct rapid, low-cost, secure, and reliable applications. It reduces the need to develop new system components each time a new business process arises, making enterprises more competitive. Web service definition language provides service calculations to execute complicated programmes in distributed environments, whereas simple object access protocol (SOAP) and representational state transfer (REST) link with external communication interfaces. In a cloud computing environment, working with SOC is challenging. Therefore, a cloud web service that can be used to simultaneously operate and communicate between clients and servers without a platform is proposed\textsuperscript{[10]}. Although the performance of REST may be higher than the performance of SOAP\textsuperscript{[11]}, several online systems based on SOAP which existed to obtain services in Taiwan\textsuperscript{[12, 13]} could not be replaced with REST for cost issues. Therefore, the frameworks for composing SOAP, non-SOAP and non-web services were proposed based on enterprise service bus (ESB) or business process execution language\textsuperscript{[14–16]}. This study gives a case study of SOAP-based services for the evaluation of the proposed two-stage cloud service optimisation model, and the REST-based services or non-web services can be adopted as the other case studies in the future.

Some studies developed the SOAs based on cloud computing and virtualisation techniques. For instance, the publish-subscribe mechanism was adopted into a middleware framework in cloud computing environments for the dynamic orchestration\textsuperscript{[17]}. Furthermore, the issues of fault tolerant and execution performance were discussed and solved\textsuperscript{[18, 19]}. However, these studies focused to find the Pareto optimality according to the limited resources, but
the aspirational level has not been investigated. Therefore, this study proposed a two-stage cloud service optimisation model to find the aspirational level to provide web service solutions with higher performance and lower cost.

2.2 Load balancing and scheduling techniques for cloud computing

Load balancing is one mechanism that distributes the dynamic workload evenly across the nodes in a computer network, and it can be adapted and scaled to various systems. In load balancing, multiple computers or computer resources are connected to increase the capacity of the system and reduce response time, facilitating a high ratio of user satisfaction to resource utilisation. Moreover, load balancing enables scalability, avoids bottlenecks and over-provisioning, reduces response time, and facilitates the implementation of fail overs [20]. To achieve green, or environmentally friendly, cloud computing, the load balance must reduce the overall energy consumption and carbon emissions. Regarding energy consumption, a balanced load and workflow prevents overheating the nodes of a cloud, reducing the amount of energy consumed. A system that consumes a large amount of energy emits a large amount of carbon. When the energy consumption is reduced using load balancing, carbon emission is reduced and green computing can be achieved.

Task assignment and workflow scheduling are the important issues for the design of load balancers in cloud environments [21, 22]. Therefore, some studies proposed task assignment and scheduling techniques to analyse the optimisation of waiting time, cost, or task profits. For instance, a cost-effective load balancing technique was proposed to analyse the demands of tasks and fixed limited resources (e.g. the number of processors and the communication bandwidth ability) for finding the Pareto-optimality of solutions with lower cost (i.e. single-objective) [2]. For the optimisation of waiting time, a particle swarm optimisation was proposed to perform the Pareto optimal task scheduling based on function fitness with given fixed resources [3, 4]. Moreover, a Pareto distribution model and the Bayes theorem were proposed to maximise the total of profits based on the constraint of given resources [5, 6]. Although these studies could find the Pareto-optimality of solutions, they only considered the optimisation of single-objective (e.g. waiting time, cost, or profits). Therefore, some studies discussed the multi-objective task assignment and scheduling techniques for the complex practical environments. For example, the Pareto optimality of cloud bursting was proposed to analyse the optimisation of the makespan and the total cost based on fixed limited resources [7]. Furthermore, the particle swarm optimisation and the entropy optimisation were proposed to find the Pareto optimality solutions based on fixed resources for minimising the processing time and the total cost [8, 9]. However, these studies assumed unchangeable resource pools to find the Pareto optimality solutions for the scheduling and load balancing in cloud environments. The aspirational-level solutions have not been investigated. Therefore, this study proposed two-stage cloud service optimisation model to analyse the multi-objective optimisation based on changeable resource pools and dynamic recontracting with virtual machine providers for reaching aspirational level.

3 Cloud service middleware platform

The proposed CSM platform supports web service and operates independently of other services or interfaces. Fig. 1 shows the architecture of CSM platform, which is divided into the client application, service load balance middleware, web service, and virtual machine layers.

3.1 Client application layer

Standard web service user applications are developed by programmers based on the demands of users. Programmers compile codes into executable files that are supplied to clients. If the service needs change or adjustments, programmers create new codes and updated executable files. This development approach is difficult to maintain. The CSM model provides users with rich internet applications, allowing them to use a web browser to customise their cloud service.

3.2 Service load balancing middleware layer

In the service load balancing middleware layer, the CSM provides an EM-LB mechanism for redirect servers. The redirect server possesses a temperature, power consumption, and virtual machine clusters for computing resources and loading. By using service-level agreement (SLA) and this information to disperse user service requests, the CSM assigns users an available location. Figs. 2 and 3 depict the EM-LB mechanism and the physical and virtual machine selection phases.
In the physical machine selection phase, the CPU thermal model is used, which is described in (1) and includes the dynamic section \((-x_0/x_2) \times e^{-x_2 \tau}\), static section \((x_1/x_2) \times P\), and ambient temperature \((E_0)\) [23]. According to the calculated oscillatory power consumption, the CPU temperature \(E\), with time \(\tau\), is used to determine the safe temperature range for a physical machine. The values of constants \(x_0, x_1,\) and \(x_2\) are measured and determined by historical records. Furthermore, the values of other constants (e.g. \(E_0\) and \(P_{\text{max}}\)) are determined in accordance with [23, 24]

\[
E(t) = \frac{x_0}{x_2} \times e^{-x_2 \tau} + \frac{x_1}{x_2} \times P + E_0
\]

The power model was used to determine the current power utilisation of a physical machine [24]. The model is described in (2), where \(k\) is the power ratio of idle status, \(y\) is the usage percentage of the CPU, and \(P_{\text{max}}\) is the maximal power of a physical machine

\[
P(y) = k \times P_{\text{max}} + (1 - k) \times P_{\text{max}} \times y
\]

In the virtual machine selection phase, a virtual machine is selected from the physical machines. The virtual machine is configured based on various resources. Formula (3) describes the resource use \(U_{\text{vm}}\) of a virtual machine

\[
U_{\text{vm}} = \frac{\text{MIPS}_{\text{usage,vm}}}{\text{MIPS}_{\text{total,vm}}}
\]

The service time \(T_{\text{service}}\) of a web service is described in (4) and includes the million instructions per second (MIPS) usage of the service \(\text{MIPS}_{\text{service}}\) and the virtual machine surplus of MIPS. In this phase, an SLA for service time is established, and the \(T_{\text{service}}\) must be less than the SLA

\[
T_{\text{service}} = \frac{\text{MIPS}_{\text{service}}}{\text{MIPS}_{\text{total,vm}} - \text{MIPS}_{\text{usage,vm}}}
\]

3.3 Web service layer

The web service layer is the computing unit of the CSM model. In this layer, service providers register and deploy web services to virtual machine layer by using the service load balancing middleware layer. Service providers can quickly delete and update their web services because the middleware allows for flexibility. To accommodate high web service loads, the multiple service locations were used and the redirect server assigns an appropriate service location (e.g. a uniform resource identifier) based on the demands of the user.

3.4 Virtual machine layer

In the virtual machine layer, the virtualisation software, e.g. Virtualbox, VMware, and Xen, is used to divide the physical machine resources into several virtual machines. The virtual machine is a complete system platform of the operating system and application environment. The virtualisation technology supports repeatable and scalable virtual machines, and various virtual machines can operate simultaneously without affecting each other. The cloud service optimisation model for resource adaptation control combines various virtual machines for cost and efficacy according to the dynamic demand \(q\). This model includes two stages: (i) finding the Pareto optimality according to the limited resources and (ii) finding the aspirational level according to the cost in Stage (1).

Stage (1): Finding the Pareto optimality according to the limited resources

The total budget \(B\) is used calculate the execution time \(T_{\text{execute}}\) based on the constraints in (5).

(i) \(u_l\) is the number of levels of the virtual machine
(ii) \(n_l\) is purchased the virtual machines
(iii) \(t_l\) is the execution time of the virtual machine
(iv) \(r_l\) is the allocation of demand

\[
T_{\text{execute}} = \min \left( \sum_{l=1}^{L} \left( t_l \times \frac{r_l}{u_l} \right) \right)
\]

s.t. \(\sum_{l=1}^{L} r_l = q;\)

\(u_l \leq n_l\) \quad \forall 1 \leq l \leq L

\(B \geq C\)

According to these parameters, the batch processing number \(\left\lceil \frac{n_l}{u_l} \right\rceil\) is calculated, and the longest execution time \(t_l \times \frac{r_l}{u_l}\) is calculated for various levels to determine the minimal \(T_{\text{execute}}\) value. For instance, Fig. 4 shows that the minimal value results from the longest execution time. In this case, Type 1 is the cheapest type of virtual machine (i.e. \(c_1 < c_2 < \cdots < c_L\)), but the execution time of Type 1 virtual machine is the longest for each demand (i.e. \(t_1 > t_2 > \cdots > t_L\)). There are eight demands (i.e. the value of \(q\) is 8), and these demands are distributed to four types of virtual machines (i.e. the value of \(L\) is 4). One virtual machine is purchased for each type (i.e. \(n_1 < n_2 < \cdots < n_4\)). The resource adaptation strategies of Stage (1) include the following
distributions: (1) one demand is distributed to Type 1 virtual machine (i.e. the value of \( r_1 \) is 1), and the total execution time of Type 1 virtual machine is \( t_1^1 \); (2) one demand is distributed to Type 2 virtual machine (i.e. the value of \( r_2 \) is 1), and the total execution time of Type 2 virtual machine is \( t_2^1 \); (3) three demands are distributed to Type 3 virtual machine (i.e. the value of \( r_3 \) is 3), and the total execution time of Type 3 virtual machine is \( t_3^3 \); (4) three demands are distributed to Type 4 virtual machine (i.e. the value of \( r_4 \) is 3), and the total execution time of Type 4 virtual machine is \( t_4^3 \) (i.e. \( t_4^3 = t_4 + t_4 + t_4 \)). The longest execution time is \( t_1^1 \), so the value of \( T_{\text{execute}} \) is equal to \( t_1^1 \). Furthermore, the total cost \( C' \) in Stage (1) is shown as follows:

\[
C' = \sum_{l=1}^{n_l} c_l \times u_l \tag{6}
\]

Stage (2): Finding the aspirational level according to the cost in Stage (1).

In this stage, the longest \( T_{\text{execute}} \) time and total cost \( C' \) were calculated in (5) and (6), where the minimal cost \( C'' \) in Stage (2) is based on constraints. For finding the aspiration level, the number of various virtual machine levels \( u_l \) was calculated using (7). For instance, Fig. 5 shows the adaptation controls that are used when the costs are expensive or there is an excess amount of virtual machines. In this case, the cost of Type 4 virtual machine is very expensive. In Stage (2), a contract is negotiated with the cloud computing provider to purchase more Type 1 virtual machines (i.e. the value of \( n_1 \) is increased up to 4) and release the Type 4 virtual machine. The resource adaptation strategies of Stage (2) include the following distributions: (1) four demands are distributed to the four Type 1 virtual machines (i.e. the value of \( r_1 \) is 4), and the total execution time of Type 1 virtual machine is \( t_1^1 \); (2) one demand is distributed to Type 2 virtual machine (i.e. the value of \( r_2 \) is 1), and the total execution time of Type 2 virtual machine is \( t_2^1 \); (3) three demands are distributed to Type 3 virtual machine (i.e. the value of \( r_3 \) is 3), and the total execution time of Type 3 virtual machine is \( t_3^3 \) (i.e. \( t_3^3 = t_3 + t_3 + t_3 \)). Therefore, the longest execution time which is \( t_1^1 \) is same with the execution time of the Pareto optimality in Stage (1). Furthermore, the minimal cost \( C'' \) in Stage 2 is lower than the cost \( (C') \) in Stage (1) if the cost of Type 4 virtual
machine is four times expensive than the cost of Type 1 virtual machine (i.e. \(4 \times c_1 < c_4\))

\[
C'' = \min \left( \sum_{i=1}^{L} c_i' \times u_i' \right)
\]

subject to \(\max_{i=1}^{L} \left( t_i \times \frac{r_i}{u_i'} \right) \times \frac{r_i}{u_i'} = T_{execute}\)

\[
\sum_{i=1}^{L} r_i = q;
\]

\[
C' \geq C''
\]

(7)

In (8), \(u_i'\) and \(n_i\) are used to obtain the adapted resource value \(a_i\), where \(a_i > 0\) is required to add this level virtual machines and \(a_i < 0\) is required to decrease the overage of virtual machines

\[
a_i = u_i' - n_i
\]

(8)

4 Implementation and experiments

In this section, the EM-LB and cloud service optimisation model are discussed. In addition, an EM-LB is developed and implemented for the CSM platform.

4.1 Energy-minimisation load balance

The CloudSim toolkit is an open source tool that simulates the operation of physical and virtual machines in a cloud computing centre [25, 26]. The datacentre models, host, VM, and EM-LB were modified to be used in CloudSim. For the simulation, relevant CPU thermal model studies were referenced, and the ambient temperature was defined as 45°C [27, 28]. Table 1 shows the environmental parameters.

Tables 2 and 3 show the parameters for physical and virtual machines. Two quad-core CPUs were used: a 2933 MIPS machine and a 3067 MIPS machine. Both machines possessed 8 GB of RAM, and a network bandwidth of 1 Gbit/s. Each VM had a distinct amount of CPU and memory, and all machines operated on a 100 Mbit/s network bandwidth.

In this simulation, the daily energy consumed by the host, availability of datacentres, and the variation of average temperature were estimated. Formula (9) was used to calculate the total daily energy consumption

\[
\text{Energy}_{\text{Total}} = \sum_{i=1}^{n} \text{Host}_{i, \text{energy}}
\]

(9)

The datacentre host availability was calculated using (10), which verified that the EM-LB mechanism maintained a high level of host datacentre availability

\[
\text{Host}_{\text{Availability}} = 1 - \left( \frac{\text{Host}_{\text{violation time}}}{\text{Host}_{\text{total time}}} \right)
\]

(10)

The relationship between the temperature and energy of a host is positive. Formula (11) calculates the variation of average temperature to facilitate the verification of energy consumption and the analysis of host availability

\[
\text{Temperature}_{\text{Average}} = \frac{\sum_{i=1}^{n} \text{Host}_{i, \text{temperature}}}{n}
\]

(11)

The CloudSim migration mechanism was used to analyse the EM-LB mechanism. Pursuant to the presence of a migration mechanism, the location of the virtual machines is not fixed, which will migrate between each physical machine because of physical machine use and energy consumption.

Figs. 6–8 show the migration mechanism used to evaluate the proposed EM-LB mechanism. Fig. 6 shows that the energy...
consumptions of random, round-robin and EM-LB mechanisms. The results indicate that the proposed EM-LB mechanism performs the optimised load balances and reduces unnecessary power consumption for each machine, so the total energies of the proposed EM-LB mechanism are lower than other methods. Furthermore, the proposed M-LB mechanism combined with a migration mechanism can provide higher availability based on the consideration of physical machine status. However, the status of physical machine is not considered by other methods, so the availabilities of these methods are lower. For the analyses of temperature, Fig. 8 shows that increased energy consumption causes temperatures to rise.

4.2 Cloud service optimisation model

The cloud service optimisation model was calculated using LINGO software. Hourly rental prices and various levels of virtual machine efficiency were referenced to differentiate the service time from Amazon EC2. Table 4 shows the parameters for the cloud service optimisation model. Assuming the hourly total cost is $15 US dollars per hour, the maximal demand is 140 units and the minimal demand is 60 units. The results show (1) the purchase of virtual machines is considered in the minimal cost; (2) the total cost is considered in the minimal cost; and (3) the total cost is considered in the highest efficiency.

Fig. 9 shows that the purchase of virtual machines is considered in the analysis of minimal cost. If the purchase of virtual machines is considered in resource adaptation, the maximal virtual machine demand must be obtained. The results show that the longest execution time is 1.5 s with a demand of 140 units of virtual machines. When the demand is reduced to 120, 100, 80, or 60 units, the longest execution time is 1.5 s. The second phase of resource adaption can effectively reduce the overall cost.

Fig. 10 shows that total cost is considered in the analysis of minimal cost. If the resources are adapted according to the total cost, the combination of various virtual machine levels is based on demand. In the first phase of resource adaptation, the execution times vary as demand increases or decreases. With a demand of 140 units, the longest execution time is 1.5 s, and the costs are less than the cost on demand 120 units. Since the high-level virtual machines can adapt to the low-level virtual machine, the overall efficiency will be improved. On the contrary, their cost will increase.

Fig. 11 shows the most efficient model according to the total cost analysis. The purchase of virtual machines was not considered in this analysis. In the first phase of resource adaptation, the longest execution time for demands of 60, 80, 100, 120, and 140 units, was 0.75s, 1s, 1.25, 1.25, and 1.5 s, respectively. The overall total time efficiency for demands of 60, 80, 100, 120, and 140 units, was 30, 56, 85, 90.75, and 139.25 seconds, respectively. The second phase of resource adaptation was used to improve the overall efficiency, raising the total time for demands of 60, 80,100, 120, and 140 units, to 25.75, 44.5, 65.75, 89.5, and 120.75 s, respectively.

Table 4 Cloud service optimisation model parameters

| Type     | Rental price, hourly | Service times, s |
|----------|----------------------|------------------|
| small    | $0.076               | 0.25             |
| middle   | $0.153               | 0.5              |
| large    | $0.305               | 1                |
| extra large | $0.61            | 2                |

This is an open access article published by the IET under the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0/)

J. Eng., 2018, Vol. 2018, Iss. 3, pp. 155–161
doi: 10.1049/joe.2017.0824
5 Conclusions and future work

Cloud computing organisations strive to achieve green computing and optimising energy consumption in datacentres is the current focus of this trend. The cloud service optimisation model is proposed for resource adaptation and control, guaranteeing that the cloud service is in agreement with the service level. The model combines various levels of virtual machine resources based on demand and reduces costs and the waste of resources. Furthermore, the energy-minimisation load balancing mechanism is proposed to efficiently allocate the requests of users, reduce energy consumption, and improve machine availability. Furthermore, this study proposes a two-stage cloud service optimisation model to find the aspirational level to provide web service solutions. In Stage 1, the Pareto optimality is found according to the limited resources. In Stage 2, the aspirational level is found according to the cost in Stage 1. Therefore, the solutions of the proposed two-stage cloud service optimisation model can be provided with higher performance and lower cost.

In the experimental results, migration proved a useful mechanism in the cloud environment. However, the virtual machine migration process increases energy consumption. In the future, a solution mechanism should be designed that balances energy consumption and host availability, and the real-time demands should be analysed to adapt accordingly. Furthermore, the detection of deteriorating physical machine [29] and the optimised service composition in the cloud [30–32] can be investigated.

6 Acknowledgment

The research was supported by the Ministry of Science and Technology of ROC under the grant no. MOST 105-2221-E-020-024.

7 References

[1] Wang L., von Laszewski G., Young A., et al.: ‘Cloud computing: a perspective study’, New Gener. Comput., 2010, 28, (2), pp. 137–146
[2] Sheeja Y.S., Jayakrishmi S.: ‘Cost effective load balancing based on honey bee behaviour in cloud environment’. Proc. 2014 First IEEE Int. Conf. Computational Systems and Communications, Tiruvandrum, India, December 2014
[3] Beegom A.S.A., Rajasree M.S.: ‘A particle swarm optimization based pareto optimal task scheduling in cloud computing’, Lect. Notes Comput. Sci., 2014, 8795, (1), pp. 79–86
[4] Alla H.B., Alla S.B., Ezzati A.: ‘A novel architecture for task scheduling based on dynamic queues and particle swarm optimization in cloud computing’. Proc. 2016 2nd IEEE Int. Conf. Cloud Computing Technologies and Applications, Marrakech, Morocco, May 2016
[5] Chiang Y.J., Ouyang Y.C., Cremers A.B., et al.: ‘A load-based scheduling to improve performance in cloud systems’. Proc. IEEE Int. Conf. Robotic Computing, Taichung, Taiwan, April 2017
[6] Zhao J., Yang K., Wei X., et al.: ‘A heuristic clustering-based task deployment approach for load balancing using Bayes theorem in cloud environment’, IEEE Trans. Parallel Distrib. Syst., 2016, 27, (2), pp. 305–316
[7] Farahabady M.R.H., Lee Y.C., Zomaya A.Y.: ‘Pareto-optimal cloud bursting’, IEEE Trans. Parallel Distrib. Syst., 2014, 25, (10), pp. 2670–2682
[8] AlKayal E.S., Jennings N.R., Abulkhair M.F.: ‘Efficient task scheduling multi-objective particle swarm optimization in cloud computing’. Proc. 2016 IEEE 41st Conf. Local Computer Networks Workshops, Dubai, United Arab Emirates, November 2016
[9] Zuo L., Shu L., Dong S., et al.: ‘A multi-objective hybrid cloud resource scheduling method based on deadline and cost constraints’, IEEE Access, 2016, 5, pp. 22067–22080
[10] Al-Shammary D., Khalil I., George L.E.: ‘Clustering SOAP Web services on internet computing using fast fractals’. Proc. 2011 IEEE 10th Int. Symp. Network Computing and Applications, Cambridge, MA, USA, 2011
[11] Kumari S., Rath S.K.: ‘Performance comparison of SOAP and REST-based web services for enterprise application integration’. Proc. 2015 Int. Conf. Advances in Computing, Communications and Informatics, Kochi, India, August 2015
[12] Dai Y.M., Liu K.S.: ‘TACSA: a web-service based system for coastal surveillance and situational awareness’. Proc. 2015 Int. Carnahan Conf. Security Technology, Taipei, Taiwan, September 2015
[13] Weng Y.C., Hsieh S.I.: ‘Design and implementation of a web-based medical drawing management system’, J. Intel. Inf. Syst., 2017, 49, (3), pp. 391–405
[14] Teng J.T., Tsai Y.C., Wang W.C., et al.: ‘A best practice of enterprise information integration platform in telecom industry’. Proc. 2013 15th Asia-Pacific Network Operations and Management Symp., Hiroshima, Japan, September 2013
[15] Kuo T.H., Chen C.H., Kung H.Y., et al.: ‘Applications of the web service middleware framework based on the BPEL’. Proc. 2016 IEEE 5th Global Conf. Consumer Electronics, Kyoto, Japan, October 2016
[16] Lee J., Lee S.J., Wang P.F.: ‘A framework for composing SOAP, non-SOAP and non-web services’, IEEE Trans. Serv. Comput., 2015, 8, (2), pp. 240–250
[17] Valls M.G., Lopez I.R., Villar L.F.: ‘iLAND: an enhanced middleware for real-time reconfiguration of service oriented distributed real-time systems’, IEEE Trans Ind. Inf., 2013, 9, (1), pp. 228–236
[18] Zhao W., Mellar-Smith P.M., Moser L.E.: ‘Fault tolerance middleware for cloud computing’. Proc. 2010 IEEE 3rd Int. Conf. Cloud Computing, Annapolis, MD, USA, 2010
[19] Garcia-Valles M., Cucinotta F., Lu C.: ‘Challenges in real-time virtualization and predictable cloud computing’, J. Syst. Archit., 2014, 60, (9), pp. 726–740
[20] Kansal N.J., Chan I.: ‘Cloud load balancing techniques: a step towards green computing’, Int. J. Comput. Sci. Issues, 2012, 9, (1), pp. 235–246
[21] Guzek M., Bouvy P., Talbi E.G.: ‘A survey of evolutionary computation for resource management of processing in cloud computing’, IEEE Comput. Intell. Mag., 2015, 10, (2), pp. 53–67
[22] Liu L., Zhang M., Liu Y., et al.: ‘A survey on workflow management and scheduling in cloud computing’. Proc. 2014 14th IEEE/ACM Int. Symp. Cluster, Cloud and Grid Computing, Chicago, IL, USA, May 2014
[23] Weissel A., Beloglazov A.: ‘Dynamic thermal management for distributed systems’. Proc. First Workshop on Temperature-Aware Cloud Computing Systems, Munich, Germany, 2004
[24] Beloglazov A., Abawajy J., Buyya R.: ‘Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing’, Future Gener. Comput. Syst., 2012, 28, (5), pp. 755–768
[25] Beloglazov A., Buyya R.: ‘Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers’. Proc. 2013 1st Int. Conf. Concurr. Comput. Pract. Exper., 2012, 24, (13), pp. 1397–1420
[26] Calheiros R.N., Ranjan R., Beloglazov A., et al.: ‘CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms’, Softw. Pract. Exper., 2011, 41, (1), pp. 23–50
[27] Lee K.J., Skadron K.: ‘Using performance counters for runtime temperature sensing in high-performance processors’. Proc. 2005 IEEE 19th Int. Parallel and Distributed Processing Symp., Denver, Colorado, USA, 2005
[28] Wang S., Bettati R.: ‘Delay analysis in temperature-constrained hard real-time systems with general task arrivals’. Proc. 2006 IEEE 27th Int. Real-Time Systems Symp., Rio de Janeiro, Brazil, 2006
[29] Liu J., Wang S., Zhou A., et al.: ‘Using proactive fault-tolerance approach to enhance cloud service reliability’, IEEE Trans. Cloud Comput., in press, 2016, doi: 10.1109/TCC.2016.2567392
[30] Wang S., Liu Z., Sun Q., et al.: ‘Towards an accurate evaluation of quality of cloud service in service-oriented cloud computing’, J. Syst. Archit., 2014, 25, (2), pp. 283–291
[31] Wang S., Zhou A., Hsu C.H., et al.: ‘The provision of data-intensive services through energy-and QoS-aware virtual machine placement in national cloud data centers’, IEEE Trans. Emerg. Topics Comput., 2016, 4, (2), pp. 290–300
[32] Wang S., Zhou A., Yang F., et al.: ‘Towards network-aware service composition in the cloud’, IEEE Trans. Cloud Comput., in press, 2016, doi: 10.1109/TCC.2016.2603504