International Journal of Database Theory and Application
Vol.7, No.2 (2014), pp.1-12
http://dx.doi.org/10.14257/ijdta.2014.7.2.01

Optimizing Web Service Composition for Data-intensive Applications

Dongjin Yu, Chang Li and Yuyu Yin

School of Computer, Hangzhou Dianzi University, Hangzhou, China
yudj@hdu.edu.cn, cherry_728@163.com, yinyuyu@hdu.edu.cn

Abstract

The accessing, transferring and processing of data need to occur parallel in order to tackle the problem brought on by the increasing volume of data on the Internet. In this paper, we put forward the parallel schema for data-intensive Web services and the Benefit Ratio of Composite Services (BROCS) to balance the throughput and cost. Furthermore, we present the method to determine the degree of parallelism (DOP) based on the BROCS model to optimize the quality of the composite services. The experiment demonstrates how DOP affects the benefit ratio of the composite service. Meanwhile, the results based on the 250,000 Microblog items published by Sina show that our proposed parallel schema for Web services could effectively improve the efficiency of the composite Web service.

Keywords: *data-intensive applications, Web services composition, parallel, response time, throughput, cost*

1. Introduction

The access to large amounts of data has opened up exciting new opportunities in commerce, science and computing applications. Therefore, the challenge now is how to effectively and efficiently process the large-scale data. As service-oriented computing and Web services have achieved success in encapsulation and data integration, the academic world and industry have started to adopt Web services and SOA (Service Oriented Architecture) to manage data on the Internet, which promotes the development, operation and management of data-intensive applications. However, traditional functional-centric Web services cannot meet the requirements of data processing, which merely provide the functionality required by the user. On the other hand, data-intensive Web services focus on providing and updating data with large amounts of data operation and exchange. Although there have been lots of successful researches in the field of data-intensive Web services, they do not deal with the parallel technology in Web services composition, let alone consider how the degree of parallelism (*DOP*) affects the efficiency of the execution of composite services.

In practice, multiple elementary services need to be composed to fulfil the complicated business. Therefore, the introduction of parallel programming schema into composite services to deal with large-scale data is of great significance. On the other hand, the composite service should guarantee users an acceptable level of quality for the whole process, such as response time, price, availability and reliability. In this paper, we focus on the response time and cost from the perspective of users and combine the throughput to find how to enable users to get the composite service with the shortest response time and the largest throughput at the lowest cost, namely, by optimizing the composite services from the perspective of users. As for Web service composition, there are several basic composition structures, including the sequential structures, the parallel structure and the conditional structure. The application of parallel structures to elementary services will enable the parallel processing of data so as to shorten

ISSN: 2005-4270 IJDTA

the response time and improve the efficiency of composite services. Determining the degree of parallelism (*DOP*) to assure an acceptable response time, cost and the largest amount of data processed is the key point of this paper. Our contributions are as follows. (1) We parallel the Web services in the processing of large-scale data. (2) We establish the Benefit Ratio of Composite Services (*BROCS*) model to balance the throughput and cost. (3) We deduce the calculation of the *DOP*, which affects the *BROCS*, and propose an algorithm used to obtain the *DOP*.

Our paper is organized as follows. Section 2 presents related work. Section 3 describes how to parallel Web services. After that, in Section 4 we detail the *BROCS* model and combine the *BROCS* with the composite service based on parallelism to make sense of the solution of the *DOP* decision. Section 5 presents the results of the experiment. Finally, Section 6 concludes the paper and outlines the future work.

2. Related Work

The data-intensive application Web service has been a hot area of research for recent years. Many scholars have been researching Web services composition for data-intensive applications, such as data transferring, data integration, optimization and management.

Some researches focus on the approaches to data transmission for data-intensive applications. Habich et al introduce a seamless extension, BPELDT, to BPEL for the handling of massive data sets, which explicitly represents the dataflow in the process of data-intensive service composition and integrates special data propagation tools to speed up the execution efficiency of composite services [1]. In data-centric environments, Spiros Koulouzis et al propose a Web service framework, which solves the large data sets in SOAP messages during service invocation [2]. Data-Grey-Box (DGB) Web services separate parameters and data when describing the interface. They also propose the approach to the integration of DGB services with special data propagation tools. To tackle the problem of accessing, moving, and processing large dataset, Amer Yahia *et al.*, present ProxyWS, which uses myriad protocols to transport large data sets and serves as an interface for developing new Web services [3]. However, it may be inefficient to exchange XML data across the Internet because the data-intensive Web service usually involves massive data sets. Quirino Zagarese et al expand the WSDL of data-intensive Web services and propose an efficient data exchange method based on negotiation [4].

There exist many researches on combining Web services with MapReduce to implement parallelism. For instance, the approach in [5] is used to select appropriate services from among a large number of candidate services. Lu *et al.*, propose the concept of dataflow constructs based on MapReduce and some other data processing technology in [6] and [7].

With regard to the optimization and management of service composition, Zhang *et al.*, introduce how to utilize two cost estimation models to realize mediator-based data-intensive service composition [8]. Besides, Glatard *et al.*, propose a workflow management system based on the Web service-MOTEUR, which reduces the execution time through multi-task parallelism and service classification [9]. The approach is especially suitable for data-intensive applications based on the grid. Meanwhile, Dirk Habich *et al.*, describe the concept of “data cloud Web service”, by which the original work that requires explicit data transmission between different Web services now just needs to change the data reference in the data cloud [10]. Tim Dornemann *et al.*, analyse the dynamic data dependency between workflow steps and weighs possible data transmission between nodes and the runtime load of various nodes in the cloud computing environment [11]. They introduce a dataflow driven cloud computing resource scheduling algorithm based on a BPEL workflow. As for the problem experienced by many organizations when accumulating data to construct large-scale archives, the data-intensive services are used to publish large-scale archives [12]. The fundamental functional properties include enhancing searching, preprocessing and asynchronous transferring. As the size of data increases, the execution time becomes much longer. Murakami *et al.*, consider storing service invocation results to avoid rerunning the workflow [13]. Although a lot of researches have been done, none mentioned the optimization based on the parallelism of Web services.

In this paper, we adopt the parallel schema for web services. As for QoS, there is no lack of research. The main direction of the QoS of service composition is to select appropriate services from among candidate services within the limits of cost and response time and some other non-functional properties. He *et al.*, introduce a QoS driven approach which provides assistance in selecting services for SaaS developers [14]. Dhore et al introduce four kinds of forms of the objective function under the premise of users' demand [15]. Wu et al provide a composition architecture based on the research into QoS for an improved algorithm in the cloud based on distance [16].

Unlike the existing work, our optimization aims at maximizing the throughput within the limit of funds. We combine the data-intensive service composition with the parallel schema and optimize the composite service by deciding on an appropriate *DOP*. We also propose the concept of Benefit Ratio of Composite Services and the *BROCS* model in terms of throughput and cost. In addition, we provide an algorithm used to compute *DOP*.

3. Web Services in Parallel

Generally speaking, the data-intensive applications have the following characteristics. (1) In order to boost performance, a large number of independent data processing jobs shall be distributed on different nodes in a loosely coupled cluster. (2) The high volume of I/O throughput involves huge amounts of data. (3) Most data-intensive applications are dataflow driven. Since data-intensive applications are intended to process data, it is advised that some data processing technology could support the Web service composition for data-intensive applications.

The composite service is composed of elementary services through the basic composite structures, which include *Conditional* structure, *Sequential* structure, *Parallel* structure and *Loop* structure, as indicated in Figure 1, where the circle represents the elementary Web service and the arrow represents the transition of data between services. In this paper, we focus on the services applied with *Parallel* structures, which means multiple services are executed on multiple nodes at the same time. The combination of the parallel computation of Web services is designed for the efficient and systematic processing of data, including data parallelism and aggregation.

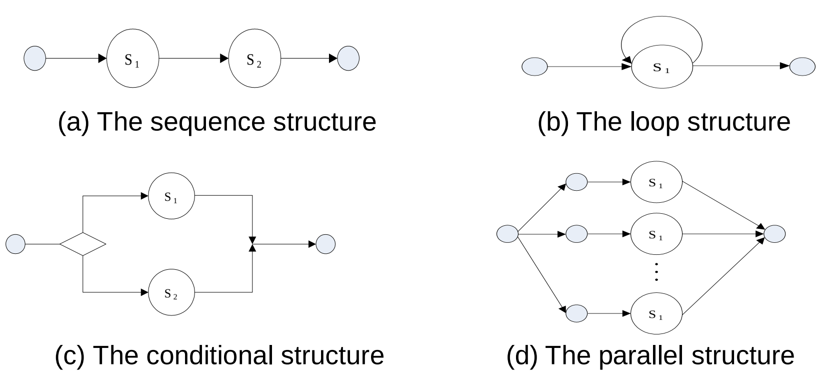


Figure 1. Four Kinds of Composite Structures

4.BROCS Model

4.1. Proposing the BROCS Model

For composite services dealing with large-scale data, paralleling the elementary services would reduce computation cost and improve their ability to process data. In practice, the user usually has some specific requirements for the composite service, such as the acceptable response time, the lowest cost and the largest benefit. The user's benefit mentioned here refers to the amount of data processed by the composite service. The key factor lies in those services which exist in parallel. Although the response time is reduced by the parallel execution of Web services, it does not indicate that it would be better if *DOP* becomes higher, simply because the user has to pay the extra cost for the parallel service replicas. In this paper, we consider how to obtain the most efficient composite service in terms of a given fund from the user's perspective. We focus on three QoS properties: the cost, the response time and the throughput.

The response time refers to the time from the request for the service to the end of its execution. The response time of the composite service is the sum of response time of each task. The throughput of the composite service is the amount of data handled per unit of time. In order to maximize the amount of data processed at the minimum cost, we propose the concept of the *Benefit Ratio of Composite Services*, or *BROCS*. More specifically, the higher the value of *BROCS* is, the lower the fee the user will pay and the greater the amount of data that will be processed. For the sake of clarity, we list the notations and their definitions in Table 1.

Table 1. Related Symbols and Definition

Notation	Definition
C	The highest fee users can pay
P	The maximum throughput required by the user
T	The maximum acceptable response time
Q^C	The total cost of the composite service
Q^P	The total throughput of the composite service
Q^T	The total response time of the composite service
C_i	The price of Web service S_i
T_i	The response time of Web service S_i
P_i	The throughput of Web service S_i
m	The number of tasks in the composite service
n	The number of parallel services
A	A constant greater than 0
B	A constant greater than 0

The objective function is indicated as (1).

$$\begin{cases} \max \dfrac{Q^P}{Q^C} \\ \qquad\qquad T \geq Q^T \\ \text{subject to}\; C \geq Q^C \\ \qquad\qquad p \leq Q^P \end{cases} . \tag{1}$$

where Q^C is equal to the sum of price of each compositional service, and Q^T is equal to the sum of the response time of each compositional services.

$$Q^C = \sum_{i=1}^{m} C_i \ . \tag{2}$$

$$Q^T = \sum_{i=1}^{m} T_i \ . \tag{3}$$

The throughput of the composite service is as follows,

$$Q^P = \frac{Q}{\sum_{i=1}^{m} \frac{Q}{P_i}} \tag{4}$$

4.2. Applying the BROCS Mode

In this section, we discuss how to optimize the composite service based on Web services parallelism. Figure 2 shows an example of a composite service based on four basic structures, where the circle represents the elementary services, the box encircles the nodes in parallel, and the curving line represents the loop execution of the service. As shown in Figure 2, the service denoted by S_4 is deployed with replicas on multiple nodes and is executed in parallel. Although the parallel execution improves the efficiency of the composite service, the user has to pay for the service replicas. In other words, a higher *DOP* is not necessarily better. Of course, if the *DOP* is too low, its ability to deal with large-scale data will be limited. Therefore, it is essential to decide an appropriate *DOP*.

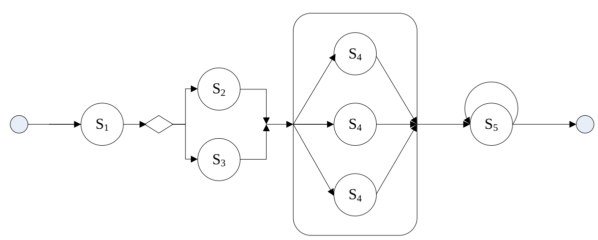


Figure 2. A Composite Service Applied with Multi-structures

4.3. Applying the BROCS Model in General Cases

Figure 2 shows an example process with four basic structures applied. In practice, the composite service may not necessarily be composed of all four structures. In the following, we present the situation where only parallel structures are in place and demonstrate how to apply the *BROCS* model to the process with parallel structures in order to optimize the composite service in terms of its cost and throughput. The approach we propose here will assist the user in achieving their goal for benefit.

According to (1),

$$Q^C = \sum_{i=1}^{j-1} C_i + n * C_j + \sum_{i=j+1}^{m} C_i \ . \tag{5}$$

Since the service S_i is executed in parallel,

$$Q^T = \sum_{i=1}^{j-1} T_i + T_j^{'} + \sum_{i=j+1}^{m} T_i \ . \tag{6}$$

where $T_j^{'}$ is the response time of the services in parallel. In addition, we compute $T_j^{'}$ with the expectation in the following equation:

$$T_j^{'} = \frac{1}{n} \sum_{k=1}^{n} T_{j_k} \ . \tag{7}$$

According to [17], to some extent, before the server reaches the peak load, there is a certain relation between the throughput and the response time. More specifically, response time increases exponentially as the load increases before the peak load. When the system throughput reaches its maximum, the response time increases rapidly and then decreases and finally appears to be steady. According to the server's performance within a normal range, namely before the peak load, the relation between response time T and the throughput P is as follows:

$$T = Ae^{BP} + \xi(\xi \text{ is the response time error}) \ . \tag{8}$$

$$P = \frac{\ln T - \ln A}{B} \ . \tag{9}$$

$$\begin{pmatrix} \ln A \\ B \end{pmatrix} = \begin{pmatrix} 1 & P_1 \\ 1 & P_2 \\ & \vdots \\ 1 & P_n \end{pmatrix}^{-1} \begin{pmatrix} \ln T_1 \\ \ln T_2 \\ \vdots \\ \ln T_n \end{pmatrix} \tag{10}$$

The more balanced the load of each node in parallel is, the more efficient the composite service is. The total throughput of nodes in parallel with DOP n can be calculated as follows.

$$\sum_{k=1}^{n} P_k = n * p_k^{'} \ . \tag{11}$$

where $p_k^{'}$ refers to the throughput of each node in parallel.

According to (1), we can express BROCS as (12) indicates.

$$\text{BROCS} = \frac{\frac{1}{\sum_{i=1}^{m} \frac{1}{P_i}}}{\sum_{i=1}^{j-1} C_i + n * C_j + \sum_{i=j+1}^{m} C_i}. \tag{12}$$

Combined with (10), we can obtain the final BROCS as (13) indicates.

$$\text{BROCS} = \frac{R}{W + \frac{1}{n}\left(\sum_{i=1}^{j-1} C_i + \sum_{i=j+1}^{m} C_i\right)\left(\prod_{i=1}^{j-1} P_i \prod_{i=j+1}^{m} P_i\right) + nC_j * S * U}. \tag{13}$$

Here, R, W are constants with respect to throughput, whereas S, U are respectively expressed as follows,

$$S = \frac{\ln T_j - \ln A}{B} \ ,$$

$$U = P_3P_4 ... P_m + P_1P_4 ... P_m + P_1P_3P_4 ... P_m +, ..., + P_1P_3 ... P_{m-1} \ .$$

As we all know, $a + b \geq 2\sqrt{ab}$, we will obtain the maximum BROCS if

$$\frac{1}{n}\left(\sum_{i=1}^{j-1} C_i + \sum_{i=j+1}^{m} C_i\right)\left(\prod_{i=1}^{j-1} P_i \prod_{i=j+1}^{m} P_i\right) = nC_j * S * U \ ,$$

and

$$n \geq \sqrt{\frac{B\prod_{i=1}^{j-1} P_i \prod_{i=j+1}^{m} P_i \left(\sum_{i=1}^{j-1} C_i + \sum_{i=j+1}^{m} C_i\right)}{C_j * U\left(\ln(T - \sum_{i=1}^{j-1} - \sum_{i=j+1}^{m} T_i) - \ln A\right)}}\ .$$

On the basis of the compositional services selected, the user specifies the total cost C and the acceptable response time T. As for the composite service, the major uncertainty lies in the *DOP*. Therefore, we will consider the impact of *DOP* on the benefit ratio of the composite service.

According to (1), we can conclude that:

$$n \leq \frac{C - \sum_{i=1}^{j-1} C_i - \sum_{i=j+1}^{m} C_i}{C_j}. \tag{14}$$

Consequently, we obtain the possible range of n:

$$\begin{cases} n \geq \sqrt{\dfrac{B\prod_{i=1}^{j-1} P_i \prod_{i=j+1}^{m} P_i \left(\sum_{i=1}^{j-1} C_i + \sum_{i=j+1}^{m} C_i\right)}{C_j * U\left(\ln(T - \sum_{i=1}^{j-1} - \sum_{i=j+1}^{m} T_i) - \ln A\right)}} \\ n \leq \dfrac{C - \sum_{i=1}^{j-1} C_i - \sum_{i=j+1}^{m} C_i}{C_j} \end{cases}. \tag{15}$$

Otherwise, if the floor of n is greater than the ceiling, it means the fee is not enough for the execution of the composite service or we might need to re-select the elementary services.

Table 2 presents the algorithm used to determine an appropriate *DOP*.

Table 2. The Algorithm used to Determine *DOP*

Input:	
C	//The highest fee users can pay,
P_i	//The throughput of the Web service S_i
T	//The maximum response time required by the user
T_i	//The response time of the Web service S_i
C_i	//The price of the Web service S_i
Q	//The size of data
A, B	//Constants greater than 0
Output:	
n	// the degree of parallelism

calculate the floor (n')and ceiling (n'') of n as following:

$$n' = \sqrt{\frac{B\prod_{i=1}^{j-1} P_i \prod_{i=j+1}^{m} P_i \left(\sum_{i=1}^{j-1} C_i + \sum_{i=j+1}^{m} C_i\right)}{C_j * U\left(\ln(T - \sum_{i=1}^{j-1} - \sum_{i=j+1}^{m} T_i) - \ln A\right)}}$$

$$n'' = \frac{C - \sum_{i=1}^{j-1} C_i - \sum_{i=j+1}^{m} C_i}{C_j}$$

if
    $\lfloor n' \rfloor + 1 \le n''$
then
    $n \in \left[\lfloor n' \rfloor + 1, \lfloor n'' \rfloor\right]$
else
    exit
end if

5. Experiment and Results

In our experiment, we utilize the Microblog data as our data sample. We develop the Microblog searching service composed of three elementary services as shown in Figure 3. The *Data-preprocessing* service parses the Microblog data. The *Microblogs-searching* service searches through the Microblogs based on a given user Id. The *Results-collecting* service collects all the microblogs output from the *Microblogs-searching* service. In our experiment, we took 250,000 microblog items published by Sina (www.sina.com.cn) as the testing data and measured the runtime of the whole process by the varying number of data nodes. The results are illustrated in Table 3, where we present the execution time for (1) preprocessing microblogs, (2) searching the microblogs with a given user Id, (3) collecting search results.

Figure 3. The Composite Service of Processing Microblogs

Table 3. The Runtime of each Task of Microblogs Processing

Number of nodes	Runtime in seconds		
	Data-preprocessing	Microblogs-searching	Results-collecting
1	8.12	1001.86	1.02
2	9.07	623.67	1.27
3	9.31	429.54	1.15
4	9.25	362.52	1.23
5	10.02	219.82	1.16
6	12.11	194.75	1.14
7	10.31	179.84	1.21
8	10.56	164.75	1.19

As shown in Table 3, every time we add one node, the whole response time is reduced. Since the data size is fixed, there is almost no change to the time-consumption in the phases of the data preprocessing and the results collecting. Figure 4 displays the values of the P/C with different numbers of nodes.

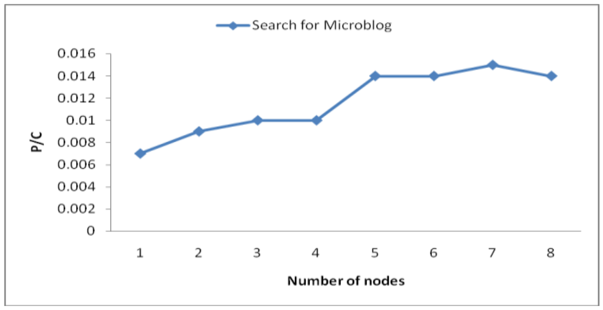


Figure 4. The Benefit Ratio of the Composite Services of Different Number of Nodes

Assuming the user limits the fund to be ten, the price of *Data-preprocessing* Web service and *Results-collecting* Web service to be 0.8, and the price of *Microblogs-searching* Web service to be 1.2. The user specifies the acceptable response time to be 400s. According to Table 3, the average response time of *Data-preprocessing* Web service, *i.e.*, T_1, is 9.84s and the *Results-collecting* Web service, *i.e.*, T_2, is 1.17s. Then we can obtain P_1=2.54, P_2=21.32 (See Table 4).

Table 4. The Response Time and Throughput of Services Data-preprocessing and Results-collecting

Service	Response time	Throughput
Data-preprocessing	9.84	2.54
Results-collecting	1.17	21.32

Now we calculate the arguments A and B according to (10). Based on the above algorithm, we derive the minimum value of n to be six, and the maximum value of n to be seven. Therefore, we set the DOP to be seven when the BROCS is maximum. Compared with the P/C in Table 5, we can conclude that the results computed with our algorithm are approximate to the physical truth.

Table 5. The Value of DOP and BROCS

Number of nodes	The benefit ratio of the composite service (P/C)
1	0.007
2	0.009
3	0.010
4	0.010
5	0.014
6	0.014
7	0.015
8	0.014

In our experiment, we set the value of the properties, such as the price of the Web service. The way in which the decision of *DOP* is carried out must be subject to the reality, which will be more exact.

6. Conclusion

In this paper, we put forward the parallel schema for data-intensive Web services. We consider the QoS optimization of composite services based on services in parallel. We also propose the concept of the *BROCS*, short for the Benefit Ratio of the Composite Service. In addition, we establish the model to balance the throughput and cost. Applying this model to a composite service based on the parallel mode and then deciding the *DOP* to optimize the whole process allow users to obtain the optimal benefit. Finally, we present an algorithm used to obtain the appropriate *DOP*.

As it is time-consuming to transfer data between services, our future work will include the integration of specific transition tools. In addition, we have considered just a few QoS properties in this paper. In the future, we will focus on more properties, such as reliability and availability.

Acknowledgments

The work is supported by Natural Science Foundation of China (No. 60903053), Natural Science Foundation of Zhejiang (No.LY12F02003), the Key Science and Technology Project of Zhejiang (No. 2012C11026-3) and the open project foundation of Zhejiang Provincial Key Laboratory of Network Technology and Information Security. The authors would also like to thank anonymous reviewers who made valuable suggestions to improve the quality of the paper.

References

[1] D. Habich, S. Richly, S. Preissler, M. Grasselt, W. Lehner and A. Maier, "BPEL-DT-Data-aware Extension of BPEL to Support Data-Intensive Service Applications", WEWST, vol. 313, **(2007)**.

[2] S. Koulouzis, R. Cushing, K. A. Karasavvas, A. Belloum and M. Bubak, "Enabling Web Services to Consume and Produce Large Datasets", IEEE Internet Computing, vol. 16, **(2012)**, pp. 52-60.

[3] S. Amer Yahia and Y. Kotidis, "A Web Services Architecture for Efficient XML Data Exchange", Proceedings of the 20th International Conference on Data Engineering, **(2004)**, pp. 523-534.

[4] Q. Zagarese, G. Canfora, E. Zimeo and F. Baude, "Enabling Advanced Loading Strategies for Data Intensive Web Services", Proceedings of International Conference on Web Services, **(2012)**, pp. 480-487.

[5] L. Pan, L. Chen and J. Wu, "Skyline web service selection with MapReduce", Proceedings of the 2011 International Conference on Computer Science and Service System, **(2011)**, pp. 739-743.

[6] X. Fei, S. Lu and C. Lin, "A MapReduce-enabled scientific workflow composition framework", Proceedings of International Conference on Web Services, **(2009)**, pp. 663-670.

[7] X. Fei and S. Lu, "A dataflow-based scientific workflow composition framework", IEEE Transactions on Services Computing, vol. 1, no. 5, **(2012)**.

[8] Y. Zhang, X. Zhou and Y. Gao, "Optimizing the Data Intensive Mediator-Based Web Services Composition", Proceedings of the 8th Asia-Pacific Web Conference, Harbin, China, **(2006)** January 16-18.

[9] T. Glatard, J. Montagnat, X. Pennec, D. Emsellem and D. Lingrand, "MOTEUR: a data-intensive service-based workflow manager, http://www.researchgate.net, **(2006)**.

[10] D. Habich, W. Lehner and S. Richly, "Uwe Assmann. Using Cloud Technologies to Optimize Data-Intensive Service Applications", Proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing, **(2010)**, pp.19-26.

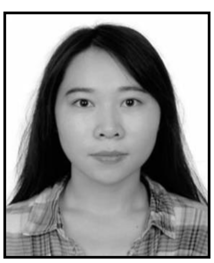
[11] T. Dornemann, E. Juhnke, T. Noll, D. Seiler and B. Freisleben, "Data Flow Driven Scheduling of BPEL Workflows Using Cloud Resources", Proceedings of the IEEE 3rd International Conference on Cloud Computing, **(2010)**, pp. 196-203.

[12] M. Tanaka, Y. Murakami and K. Zettsu, "Data-Intensive Services for Large-Scale Archive Access", Proceedings of the 2012 IEEE Ninth International Conference on Services Computing (SCC), **(2012)**, pp.

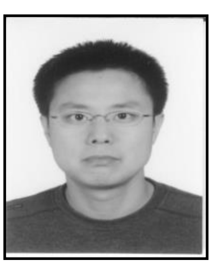
617-624.
[13] Y. Murakami, M. Tanaka, A. Bramantoro and K. Zettsu, "Data-Centered Service Composition for Information Analysis", Proceedings of the 2012 IEEE Ninth International Conference on Services Computing (SCC), **(2012)**, pp. 602-608.
[14] Q. He, J. Han, Y. Yang, J. Grundy and H. Jin, "QoS-Driven Service Selection for Multi-tenant SaaS", Proceedings of the 2012 IEEE 5th International Conference on Cloud Computing (CLOUD), **(2012)**, pp. 566-573.
[15] S. R. Dhore, H. Gangwar, P. Mishra, R. Sharma and R. Singh, "Systematic approach for composing Web Service using XML", Proceedings of the 2012 Third International Conference on Computing Communication & Networking Technologies (ICCCNT), **(2012)**, pp. 1-5.
[16] C. S. Wu and I. Khoury, "QoS-aware dynamic research component composition for collaborative research projects in the clouds", Proceedings of the 2012 IEEE 4th International Conference on Cloud Computing Technology and Science (CloudCom), **(2012)**, pp. 883-888.
[17] X. Xu and T. Xu, "Performance evaluation model of Web servers based on response time", Proceedings of the 2011 13th IEEE Joint International Computer Science and Information Technology Conference (JICSIT), **(2011)**, vol. 3.

Authors

Dongjin Yu is currently a professor at Hangzhou Dianzi University, China. He received his BS and MS in Computer Applications from Zhejiang University in China, and PhD in Management from Zhejiang Gongshang University in China. His current research efforts include service computing, program comprehension and cloud computing. He is especially interested in the novel approaches to constructing large enterprise information systems effectively and efficiently by emerging advanced information technologies. The concern of his research closely relates with real applications of e-government and e-business. He is the director of Institute of Cloud Computing and Big Data, and the vice director of Institute of Intelligent and Software Technology of Hangzhou Dianzi University. He is also a member of ACM and IEEE, and a senior member of China Computer Federation (CCF).



Chang Li was born in July 1988, and received her bachelor's degree in Software Engineering from Hangzhou Dianzi University in 2011. Her primary research area focuses on service computing and software engineering.



Yuyu Yin received the Doctor's degree in computer science from Zhejiang University, Hangzhou, China, in 2010. He is currently an assistant professor in Hangzhou Dianzi University. His research interests include service computing, cloud computing and middleware techniques.