Cloud Model for Service Selection

Shangguang Wang*, Zibin Zheng†, Qibo Sun*, Hua Zou* and Fangchun Yang*
* State Key Laboratory of Networking and Switching Technology; †Department of Computer Science and Engineering
* Beijing University of Posts and Telecommunications; †The Chinese University of Hong Kong
* Haidian, Beijing, China; †Shatin, N.T., Hong Kong
sguang.wang@gmail.com; zbzheng@cse.cuhk.edu.hk; {qbsun; zouhua; fcyang}@bupt.edu.cn

***Abstract*—Cloud computing is Internet-based computing where computing resources are offered over the Internet as scalable, on-demand services. Web services are widely employed for building distributed cloud applications. Performance of web services may fluctuate due to the dynamic Internet environment, which makes the Quality-of-Service (QoS) inherently uncertain. With the increase of Web services in the Internet, selecting the optimal service from a set of functionally equivalent candidates becomes an important research problem. In this paper, we propose an efficient and effective QoS-aware service selection approach. Our approach first employs cloud model to compute the QoS uncertainty for pruning redundant services while extracting reliable services. Then, mixed integer programming is used to select optimal services. The experimental results show that our approach can provide reliable and efficient optimal service selection for users.**

I. INTRODUCTION

Cloud computing is Internet-based computing where computing resources (e.g., machines, network, storage, operating systems, software applications, etc.) are offered over the Internet as scalable and on-demand services [1-3]. Web services are widely employed for building service-oriented applications (SOA) and distributed cloud applications [2, 4]. According to the SOA paradigm, composite applications are specified as abstract processes composed of a set of abstract services (called *service class* or *task*). At service run time, a concrete web service (called *service candidate*) is selected and invoked for each service class. This procedure ensures loose coupling and flexibility of the design [5]. Quality-of-Service (QoS) parameters (e.g., response time, price, throughput, etc.) play an important role in determining the success or failure of the composited system. Service Level Agreement (SLA) is often used as a contractual basis between service consumers and service providers on the expected QoS level [6-7]. QoS-aware service selection aims at efficiently finding the best combination of web service candidates that satisfies a set of end-to-end QoS constraints in order to fulfill a given SLA.

The Internet environment is highly dynamic. QoS values of Web services can change dynamically due to the update of server hardware/software or workload change of servers. Moreover, some of the selected services may become unavailable suddenly at run-time while new service candidates may be launched [8-9]. Therefore, quick responses and auto-adaptation to the dynamic environments are very important for the service-oriented systems and cloud applications. Performance of the service selection algorithm can have a great influence on the overall performance of the composed system. A number of service selection approaches have been proposed recently [6, 10-13]. However, these previous approaches have two major limitations.

Firstly, previous selection approaches do not consider the uncertainty of QoS seriously. QoS values of the composite applications are usually computed by aggregated QoS values of the remote Web services, which may come from different organizations, implemented by different programming languages, and run on different platforms [9]. In the unpredictable Internet environment, any changes in location, network condition, time and many other factors may impact the quality of these Internet web services. In addition, the QoS values may not precisely reflect the actual performance of a web service. For example, service providers of an e-commerce application may not always deliver their "promised" quality level because of "intentional" deceptions [14], aiming at attracting a large number of users in a short time and obtain lots of illegal profits. It is worth noting that a web service with consistently good QoS performance is typically more desirable than a web service with a large variance on its QoS performance. Therefore, consistency should be considered as an important criterion for service selection. Unfortunately, most existing service selection approaches [6, 10-13] have not considered consistency seriously in their QoS models.

Secondly, with the increase of web services in the Internet, computation time of the web service selection approaches becomes larger. Real-time optimal web service selection becomes more and more difficult. Existing approaches focused too much on the optimization of selection algorithms themselves to reduce computation time and neglected the basic impact factor (i.e., the exponential increase in the number of web services). From work [15], there has been a more than 130% growth in the number of published web services in the period from Oct. 2006 to Oct. 2007. The statistics published by the web services search engine *Seekda!* also indicated that the number of web services increased exponentially over the last three years. In addition, the pay-per-use business model promoted by cloud computing paradigm will enable service providers to offer their services to their users in different service levels [16]. Thus, service users will be soon faced with a huge number of variations of the same services offered at different QoS levels. The need for

The work presented in this study is supported by the National Basic Research and Development Program (973 program) of China (No. 2009CB320406); the Foundation for Innovative Research Groups of the National Natural Science Foundation of China (No. 60821001); the National High Technology Research and Development Program (863program) of China (No. 2008AA01A317); "HeGaoJi"-Important National Science & Technology Specific Projects (2009ZX01039-001-002-01); and the National Natural Science Foundation of China (No. 60872042, 61070205 and 61070206).

efficient Web service selection approaches is becoming more and more urgent.

To address these challenges, we present a cloud model based web service selection approach. The main contributions of this paper can be summarized as follows.

(1) We address the problem of web service selection and demonstrate the influence of uncertainty of QoS on the service selection process.

(2) We propose a novel concept, called QoS uncertainty computing, to model the inherently uncertain of Web service QoS. We adopt cloud model to compute the uncertainty of QoS. According to the three numerical characteristics of cloud model, the web services with a large variance on their QoS can be pruned. To the best of our knowledge, this is the first work that computes the uncertainty of QoS for web service selection.

(3) Based on cloud model, we propose a fast and reliable QoS-aware service selection approach. We evaluate our approach experimentally on 10,258 real-world Web services with detailed QoS values as well as on randomly generated QoS values.

The remainder of this paper is organized as follows. Section II introduces the background of service composition. Section III describes the service selection approach. Section IV shows the experiments and Section V concludes the paper.

II. BACKGROUND

A. Related Concepts

In this section we introduce some related concepts of service composition. An abstract composite service can be defined as an abstract representation of a composition request $\mathbb{SC}=\{S_1,\cdots,S_n\}$, where $\mathbb{SC}$ refers to the required service classes. A concrete composite service can be defined as an instantiation of an abstract composite service. A concrete composite service can be obtained by binding each abstract service class in $\mathbb{SC}$ to a concrete service s_j, where $s_j \in S_j$ and $S_j=\{s_{j1},\cdots,s_{jl}\}$ contains $l(l>1)$ functionally equivalent services with different QoS values.

QoS attributes can be divided into two categories: positive and negative QoS attributes. Positive attributes mean that the higher the attribute value is, the better the quality is (e.g., reliability, availability). Negative attributes means inversely with the positive attributes. Since positive attribute values can be easily converted into negative attribute values (i.e., positive attribute values is multiplied by -1), we only consider negative attributes in this paper for the sake of simplicity.

The QoS values of a composite service are aggregated by the selected service candidates. Table I lists the QoS aggregation functions of the sequential composition model. Other models (e.g., parallel, conditional and loops) can be transformed to the sequential model using techniques described in [17].

In the service composition, the service candidates have different values of the QoS attributes. Utility functions are usually employed to map the vector of QoS values *Qs* into a single real value, to enable sorting and ranking of service candidates. The QoS utility function in the paper is similar to [12]. For example, the minimum and maximum aggregated values of the *k*-th QoS attribute of $\mathbb{S}$ are computed as follows:

$$U(s)=\sum_{k=1}^{r}\frac{Q_{j,k}^{\max}-q_k(s)}{Q_{j,k}^{\max}-Q_{j,k}^{\min}}.w_k \quad (1)$$

$$U(\mathbb{S})=\sum_{k=1}^{r}\frac{Q_{k}^{\max}-q_k(\mathbb{S})}{Q_{k}^{\max}-Q_{k}^{\min}}.w_k \quad (2)$$

with

$$Q_k^{\max}=\sum_{j=1}^{n}Q_{j,k}^{\max}, Q_{j,k}^{\max}=\max_{\forall s_{ji}\in S_j} q_k(s_{ji})$$

$$Q_k^{\min}=\sum_{j=1}^{n}Q_{j,k}^{\min}, Q_{j,k}^{\min}=\min_{\forall s_{ji}\in S_j} q_k(s_{ji})$$

where $w_k \in R^+(\sum_{k=1}^{r} w_k=1)$ represents users' preferences, $Q_{j,k}^{\min}$ is the minimum value of the *k*-th attribute in all service candidates of the service class S_j and similarly $Q_{j,k}^{\max}$ is the maximum value, $Q_k^{\min}$ is the minimum value of the *k*-th attribute of $\mathbb{S}$ and similarly $Q_k^{\max}$ is the maximum value.

TABLE I. QOS AGGREGATION FUNCTIONS

QoS Attributes	Sequential
Price, Response time	$q(\mathbb{S})=\sum_{j=1}^{n}q(s_j)$
Throughput	$q(\mathbb{S})=\min_{j=1}^{n}q(s_j)$
Reputation	$q(\mathbb{S})=\frac{1}{n}\sum_{j=1}^{n}q(s_j)$
Availability, Reliability	$q(\mathbb{S})=\prod_{j=1}^{n}q(s_j)$

The service selection with global QoS constraints is an optimization process. The optimal selection for a given service composition $\mathbb{S}$ must meet the following two conditions:

1) For a given vector of global QoS constraints $\mathbb{CS}=\{C_1,\cdots,C_m\}$ $(0\le m\le r)$, $q(\mathbb{S})\le C$ $(\forall C_k\in\mathbb{CS})$, where $q(\mathbb{S})$ is the aggregated QoS value of the composition service.

2) The maximum overall utility value $U(\mathbb{S})$ in the composition service.

However, finding the optimal composition requires enumerating all possible combinations of service candidates, which can be very expensive in terms of computation time. Furthermore, it is difficult to assure the reliability of the services selected.

B. Related work

To overcome the problem of Web service selection, researchers proposed many approaches [2, 6, 10-13]. In [2], a Web services framework was proposed to offer higher level abstraction of clouds in the form of a new technology. This study made possible the provision of service publication, discovery and selection based on dynamic attributes which expressed the current state and characteristics of cloud services and resources. Work [10] proposed a novel service selection optimization approach, which contained three main idea: (1) loops peeling is adopted in the optimization; (2) if a feasible solution for the service composition problem does not

exist, negotiating QoS parameters is performed; and (3) a new class of global constraints is introduced. When facing a great number of service candidates with uncertainty QoS, due to the fail to reduce the search space (eliminate redundant service candidates), this approach has a long computation time. In [12], heuristic algorithms were used to find a near-to-optimal solution. The algorithms are more efficiently than exact solutions and are suitable for making runtime decisions. However, these algorithms do not consider the uncertainty of QoS. The selected services may deviate from the actual execution results in the dynamic service environment. Other related studies include dynamic service selection [11], heuristic service selection [13], Skyline service selection [6], etc.

III. SERVICE SELECTION VIA CLOUD MODEL (SSCM)

As shown in Fig. 1, the proposed SSCM approach contains two phases. The first phase is QoS uncertainty computing (Section III-A), in which we adopt cloud model to transform the quantitative QoS to qualitative QoS for the QoS uncertainty computation. The second phase is service selection (Section III-B), in which we adopt a mixed integer programming to find the most suitable service from each service class. Eventually, service composition engine [10] invokes the selected services and returns the invocation results to service users.

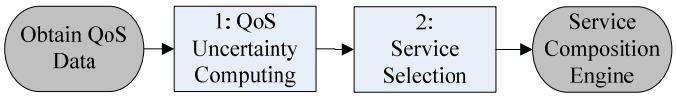

Figure 1. Procedures of SSCM approach

A. QoS Uncertainty Computing

To reduce the fluency of QoS uncertainty on the reliability of service selection, SSCM adopts cloud model to compute the uncertainty by transforming quantitative QoS values (transaction logs) to qualitative QoS concept (uncertainty level). According to the uncertainty level, a web service with consistently good QoS can be distinguished from those services with a large QoS variance.

1) Cloud model

Cloud model [18] is a model of uncertainty transition between a linguistic term of a qualitative concept and its numerical representation. It can be employed for the uncertainty transition between qualitative concept and quantitative description. A cloud model can be defined as:

Definition 1. Let U be the set as the universe of discourse, and C a qualitative concept associated with U. The membership degree of quantitative numerical representation x in U to the concept C, $\mu(x) \in [0,1]$, is a random number with a stable tendency, that is as in (3):

$$\mu : U \to [0,1], \forall x \in U, x \to \mu(x) \quad (3)$$

The distribution of x in the universe of discourse U is called cloud $C(X)$, and x is called a cloud drop.

The literature [18], gave many kinds of cloud models, such as normal cloud, γ cloud. Because a lot of uncertainty concepts behave normal clouds in social and natural phenomena, in our study, we mainly apply normal cloud model. The overall characteristics of cloud model may be reflected by its three numerical characteristics: Expected value (Ex), Entropy (En) and Hyper-Entropy (He). Fig. 2 shows the three numerical characteristics of cloud model, where the number of cloud drops is 1000. In the discourse universe, Ex is the position corresponding to the center of the cloud gravity, whose elements are fully compatible with the linguistic concept; En is a measure of the concept coverage, i.e., a measure of the fuzziness, which indicates how many elements could be accepted to the qualitative linguistic concept; and He is a measure of the dispersion on these cloud drops, which can also be considered as the entropy of En. Then, the vector $NC = \{Ex, En, He\}$ is called the eigenvector of cloud model.

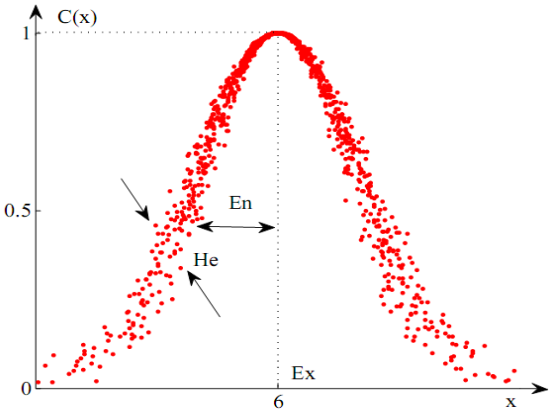

Figure 2. Three numerical characteristics of cloud model

By using the three numerical characteristics Ex, En, and He, the cloud generator [18] can produce many cloud drops. The transforms between a qualitative concept expressed by the three numerical characteristics of a cloud and its quantitative representation expressed by a set of numerical cloud drops are performed by cloud generators: forward cloud generator and backward cloud generator. In this study, we apply these three numerical characteristics of backward cloud generator (Algorithm 1) to denote the uncertainty of QoS by transforming QoS quantitative values to qualitative concept. More information about the cloud model is in [18].

2) Application of the cloud model

We first take two services S and T that offer the similar hotel web service as an example to illustrate the different implications for reliable web services between certain and uncertain QoS. In this example, the performance of S and T is recorded by a series of transaction logs, which helps capture the actual QoS delivered by each provider in practical application. Because in dynamic environment, these service providers operate which causes the uncertainty of their performance, this can be reflected by the fluctuation among different transactions. Although the actual number of transactions should be much larger, for the ease of illustration, we only consider five transactions with services S and T, respectively. These transactions are represented as (s_1,…,s_5) and (t_1,…,t_5) as shown in Table II. Table II gives response-time values of these transactions. The aggregated QoS value

($\underline{s}$ and $\underline{t}$), which are obtained by averaging all transactions, are given in the last row of Table II.

TABLE II. A SET OF SERVICE TRANSACTIONS

Web Service: S		**Web Service: T**	
ID	*Response time (ms)*	*ID*	*Response time (ms)*
s_1	29	t_1	16
s_2	26	t_2	40
s_3	30	t_3	31
s_4	26	t_4	34
s_5	26	t_5	12
$\underline{s}$	27.4	$\underline{t}$	26.6

From Table II, the aggregate QoS values of S is larger than that of T, i.e., $\underline{s}>\underline{t}$. In traditional service selection approach, web services T is usually selected as a service component in service composition because of 26.6<27.4. However, after analyzing each transaction of these two services in great depth, we find the following two important facts that may be ignored by traditional approaches: (1) although the average response time of service T is slightly less than that of S, the three transactions (s_2, s_4, s_5) of service S is less than (t_2, t_4, t_5) of service, i.e., $s_2<t_2$, $s_4<t_4$ and $s_5<t_5$. This means that the response time of service S is less than that of service T in most transactions; (2) the response time of service T is more volatile than that of service S, i.e., service T is with a large variance on its QoS, while service S is with consistently good QoS.

According to the two facts stated above, if service T is selected as a service component, the actual execution result of service T may deviates from $\underline{t}$, leading to poor composition service quality or service selection failure. Thus, it may be obvious that service S is more stable than service T, and S as a service component may be more suitable than service T. So how to compute the uncertainty of web service, that is, how to distinguish one web service with a consistently good QoS from another web service with a large variance on its QoS, is an important issue. In this paper, we adopt the backward cloud generator algorithm of cloud model to distinguish theses services.

Algorithm 1. Backward cloud generator.

Input: n transactions of a web service, i.e., n cloud drops $\{x_1, x_2, \cdots, x_n\}$.

Output: the three numerical characteristics Ex, En, and He of the n cloud drops.

Steps:

1) According to x_i, computing the sample mean $\overline{X} = \frac{1}{n}\sum_{i=1}^{n} x_i$, and the sample variance $S^2 = \frac{1}{n-1}\sum_{i=1}^{n}\left(x_i - \overline{X}\right)^2$;

2) The Expected value of the web service on its QoS can be calculated by $Ex = \overline{X}$;

3) The Entropy of the web service on its QoS also can be calculated by $En = \frac{\sqrt{\pi/2}}{N}\sum_{i=1}^{N}\left|x_i - Ex\right|$;

4) Finally, the Hyper-Entropy can be obtained by $He = \sqrt{S^2 - En^2}$.

We apply the algorithm above to Table II. These response-time values can be seen as quantitative QoS values expressed by five cloud drops, i.e., (s_1,…,s_5) or (t_1,…,t_5). The qualitative QoS concept (uncertainty level) of each service can be expressed by its eigenvector. Then these eigenvectors of services S and T can be calculated as follows: NC_S={27.40, 2.11, 3.10}, and NC_T={26.60, 12.63, 144.25}. Since 2.11<12.63 and 3.10<144.25, the uncertainty level of service S is smaller than that of service T. This means that the QoS of service S is consistently good but service T is with a large variance on its QoS. Thus, the service S should be selected as a service component rather than service T, which is different from the traditional approaches.

In order to apply the cloud mode to Web services, we set the parameters λ and h as the thresholds of En and He according to different web service environments. The web services with a large QoS variance and the web services with a consistently good QoS performance can be distinguished using the conditions $En \le \lambda$ and $He \le h$, respectively. The latter condition will be seen as service candidates prior to the former for reliable service selection. This will guarantee that selected services can be reliably executed. Furthermore, redundant service candidates will be pruned for each service class because of $En > \lambda$ or $He > h$. By this way, cloud model can help reduce the search space of service selection and shorten the computation time in service composition. Because the QoS uncertainty computing is independent of any individual service request, it does not need to be conducted online at request time. Hence, we make use of the cloud model for obtaining reliable service candidates offline in order to speed up the service selection process.

B. Service Selection

After the QoS uncertainty computing, service candidates with consistently good QoS performance can be discovered in each service class. Then a service selection algorithm needs to be designed to find the most suitable service of each class under global QoS constraints. By only focusing on the services which have consistently good performance, we speed up the selection process and are able to select reliable services. Mixed Integer Programming (MIP) is used to solve the optimization problem of service selection based on the obtained services. The MIP has recently been used to solve the service composition problem by several researchers [12-13]. In our study, binary decision variables are used in the problem to represent the service candidates. A service candidate s_{ji} is selected in the optimal composition if its corresponding variable x_{ji} is set to 1 in the solution of the model and discarded otherwise. By re-writing (2) to include the decision variables, the problem of solving the model can be formulated as a maximization problem of the overall utility value given by

$$\sum_{k=1}^{r} \frac{Q_k^{\max} - \sum_{j=1}^{n}\sum_{i=1}^{l} x_{ji} \cdot q_k(s_{ji})}{Q_k^{\max} - Q_k^{\min}} . w_k \qquad (4)$$

subject to the global QoS constraints and satisfying the allocation constraints on the decision as

$$\begin{cases} \sum_{j=1}^{n}\sum_{i=1}^{l} q_k(s_{ji})\cdot x_{ji} \ge C_k, 1 \le k \le m \\ \sum_{j=1}^{l} x_{ji} = 1, 1 \le j \le n \end{cases} \quad (5)$$

By solving (4) and (5) using any MIP solver methods, a list of service candidates are obtained and returned to service composition engine or service broker providing new value-added services for service users.

IV. EXPERIMENTS

In this section, we compare with the approach proposed in [10] by conducting two experiments in Section IV-B and Section IV-C, respectively. The first experiment indicates that our approach has much higher success ratio than other approaches, and the second experiment shows that the computation time consumed by our approach is much shorter than other approaches.

A. Experiment Setup

In this study, we conduct our experiments using two types of datasets. The first is a real-world web service QoS dataset named WS-DREAM[1] from [19]. WS-DREAM dataset contains about 1.5 millions web service invocation records of 150 service users in 24 countries. Values of three QoS attributes (i.e., *Response Time*, *Response Data Size* and *Failure Probability*) are collected by these 150 service users on 10,258 Web services. The second dataset is a randomly generated dataset (named RG) that also contains values of three QoS attributes of 10,000 web services.

In the experiments, we randomly partitioned each dataset into 10 service classes. The threshold values of Entropy and Hyper-Entropy are set to ($\lambda = 3.8$, $h = 5.9$). The number of QoS attributes is set to 3, and the number of QoS constraints is set to 2. The number of service candidates per service class varies from 10 to 100.

All the experiments are conducted on the same computer with Pentium 2.0GHz processor, 2.0GB of RAM, Windows XP SP3, Lp-Solve 5.5, and Matlab 7.6. All approaches are run for 20 times and all results are reported on average.

B. Success Ratio

In service composition system, an important goal of service selection approach is to select reliable services for service users. However, due to the uncertainty of QoS, the selected service often deviates from the user expectations, which may lead to service composition failure in practical application. Thus, the aim of this section is to compare the success ratio of our proposed SSCM approach with other well-known approach in service selection process, i.e., the MAIS approach proposed in [10].

Definition 2. Success Ratio (SR) is how often the ratio of users' QoS constrains (C_i) to monitored aggregated QoS values ($\bar{U}_i(\mathbb{S})$) is greater than or equal to a threshold value (*th*) for n composition services, i.e., $SR = \frac{srn}{n} \times 100\%$, and $srn = \sum_{i=1}^{100} \begin{cases} 1, & \bigcap_{i=1}^{k} \frac{C_i}{\bar{U}_i(\mathbb{S})} \ge th \\ 0, & otherwise \end{cases}$.

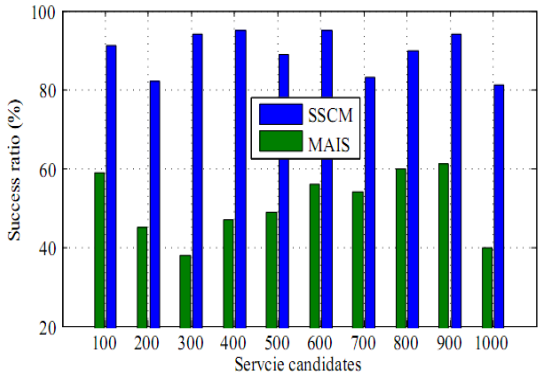

Figure 3. Success ratio comparison with WS-DREAM dataset.

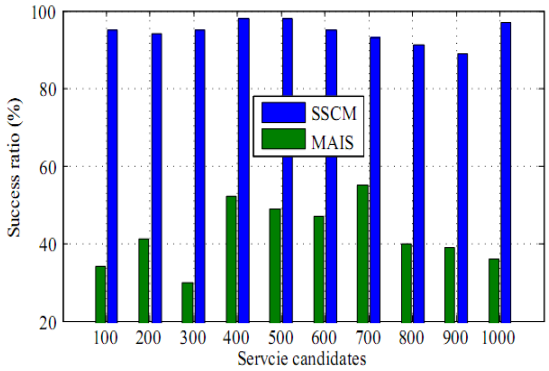

Figure 4. Success ratio comparison with RG dataset.

According to Definition 2, for a service selection approach, the higher its success ratio is, the better its performance. Fig.3 and Fig.4 show the comparison with MAIS on success ratio with two different QoS datasets. In the experiment, the parameters are set as *th* = 0.86, *n* = 200.

From Fig.3 and Fig.4, with different number of service candidates, the success ratio of SSCM is much higher than that of MAIS on both datasets. The overall success ratio of SSCM is 91.95% on average, while that of MAIS is only 47.05%. These experimental results indicate that our approach effectively reduces the influence of QoS uncertainty on the quality of composition service and greatly improves the reliability of service selection, because cloud model is adopted to compute the uncertainty of web service QoS in our SSCM approach. By using *En* and *He to* monitor QoS transactions, SSCM effectively defense from web services with a large variances on its QoS, and reduce the difference between selected services and actual execution results. Hence, the reliability of service selection can be greatly improved.

C. Computation Time

Existing service selection approaches usually ignore reducing the redundant candidates, which will lead to longer

[1]http://www.wsdream.net

computation time. In this section, we conduct experiments to compare SSCM approach with MIAS approach on the computation time.

TABLE III. EXPERIMENTAL RESULTS ON COMPUTATION TIME

The number of Service Candidates	RG Dataset		WS-DREAM Dataset	
	MIAS(s)	SSCM(s)	MIAS(s)	SSCM(s)
100	0.443	0.300	0.344	0.316
200	0.500	0.441	0.828	0.341
300	0.765	0.672	1.110	0.643
400	1.140	1.020	1.578	1.101
500	1.641	1.187	2.140	1.578
600	2.328	1.453	2.828	1.812
700	3.844	1.906	3.640	2.312
800	3.562	2.203	4.422	2.625
900	4.469	2.735	5.953	3.109
1000	5.359	3.125	7.469	3.531

From TABLE III, we can see that computation time (second unit) of SSCM is very short. All results are shorter than 4 seconds. Compared with the MIAS approach, the computation time of SSCM is shorter than that of MIAS on both datasets. The experimental results indicate that SSCM significantly reduces the time cost of service selection, since the searching space is reduced in our approach.

Thus, according to the experimental results on success ratio and computation time, SSCM can efficiently and reliably perform service selection for service-oriented systems and composed cloud applications.

V. CONCLUSIONS

Cloud computing usually provides shared resources to users over the Internet via web applications/web services. To guarantee the reliability and real-time requirement of web service selection, this paper proposes an efficient QoS-aware service selection approach based on cloud model. Our approach applies cloud model to compute the uncertainty of QoS and uses mixed integer programming to identify the most suitable web services. We evaluate our approach experimentally using both real-world and randomly generated web service QoS datasets. The experimental results show that our approach can perform service selection for users efficiently and effectively.

In our future work, we will continue to investigate more efficient Web service selection approaches (e.g., How to set the threshold of En and He accordingly? what relation is between them). We are currently constructing a realistic test-bed and developing a practical middleware for service selection in Planet-Lab. Our goal is to help real-world service users find appropriate services according to their QoS requirements in the near future.

REFERENCES

[1] G. Wei, A. Vasilakos, Y. Zheng, and N. Xiong, "A game-theoretic method of fair resource allocation for cloud computing services," The Journal of Supercomputing, vol. 54, pp. 252-269, 2010.

[2] A. Goscinski and M. Brock, "Toward dynamic and attribute based publication, discovery and selection for cloud computing," Future Generation Computer Systems, vol. 26, pp. 947-970, 2010.

[3] D. Habich, W. Lehner, S. Richly, and U. Assmann, "Using Cloud Technologies to Optimize Data-Intensive Service Applications," Proceddings of th 3th IEEE International Conference on Cloud Computing (CLOUD 2010), 2010, pp. 19-26.

[4] L. Min, Z. Liang-Jie, and L. Fengyun, "An Insuanrance Model for Guranteeing Service Assurance, Integrity and QoS in Cloud Computing," Proceedings of the 8th IEEE International Conference Web Services (ICWS 2010), 2010, pp. 584-591.

[5] G. Canfora, M. Di Penta, R. Esposito, and M. L. Villani, "A framework for QoS-aware binding and re-binding of composite web services," Journal of Systems and Software, vol. 81, pp. 1754-1769, 2008.

[6] M. Alrifai, D. Skoutas, and T. Risse, "Selecting skyline services for QoS-based web service composition," Proceedings of the 19th international conference on World Wide Web (WWW 2010), 2010, pp. 11-20.

[7] V. Cardellini, E. Casalicchio, V. Grassi, and F. Lo Presti, "Flow-Based Service Selection for Web Service Composition Supporting Multiple QoS Classes," Proceedings fo the 5th IEEE International Conference on Web Services (ICWS 2007), 2007, pp. 743-750.

[8] Z. Zibin, Z. Yilei, and M. R. Lyu, "CloudRank: A QoS-Driven Component Ranking Framework for Cloud Computing," Proceedings of the 29th IEEE Symposium on Reliable Distributed Systems (SRDS 2010), 2010, pp. 184-193.

[9] Z. Zibin, T. C. Zhou, M. R. Lyu, and I. King, "FTCloud: A Component Ranking Framework for Fault-Tolerant Cloud Applications," Proceedings of the 21th IEEE International Symposium on Software Reliability Engineering (ISSRE 2010), 2010, pp. 398-407.

[10] D. Ardagna and B. Pernici, "Adaptive service composition in flexible processes," IEEE Transactions on Software Engineering, vol. 33, pp. 369-384, 2007.

[11] S.-Y. Hwang, E.-P. Lim, C.-H. Lee, and C.-H. Chen, "Dynamic Web service selection for reliable Web service composition," IEEE Transactions on Services Computing, vol. 1, pp. 104-116, 2008.

[12] T. Yu, Y. Zhang, and K.-J. Lin, "Efficient algorithms for Web services selection with end-to-end QoS constraints," ACM Transactions on the Web, vol. 1, pp. 1-26, 2007.

[13] M. Alrifai and T. Risse, "Combining global optimization with local selection for efficient QoS-aware service composition," Proceedings of the 18th international conference on World Wide Web (WWW 2009), 2009, pp. 881-890.

[14] Y. Qi and A. Bouguettaya, "Computing Service Skyline from Uncertain QoWS,", IEEE Transactions on Services Computing, vol. 3, pp. 16-29, 2010.

[15] E. Al-Masri and Q. H. Mahmoud, "Investigating web services on the world wide web," Proceedings fo the 17th International Conference on World Wide Web (WWW 2008), 2008, pp. 795-804.

[16] K. S. Candan, W.-S. Li, T. Phan, and M. Zhou, "Frontiers in information and software as services," Proceedigns fo the 25th IEEE International Conference on Data Engineering (ICDE 2009), 2009, pp. 1761-1768.

[17] J.-H. Jang, D.-H. Shin, and K.-H. Lee, "Fast quality driven selection of composite Web services," Proceedings of the 4th European Conference on Web Services (ECOWS 2006), 2006, pp. 87-96.

[18] D. Li, D. Cheung, X. Shi, and V. Ng, "Uncertainty reasoning based on cloud models in controllers," Computers & Mathematics with Applications, vol. 35, pp. 99-123, 1998.

[19] Z. Zheng and M. R. Lyu, "Collaborative reliability prediction of service-oriented systems," Proceedign of the 32th ACM/IEEE International Conference on Software Engineering (ICSE 2010), 2010, pp. 35-44.