Method of Semantic Web Service Discovery and Automatic Composition

Yuemin Wang
Suqian College, 3rd Department, Suqian, Jiangsu, China
Email: wym07@163.com

Jun Steed Huang
Ottawa-Carleton Institute of ECE, Ottawa, Canada
Email: steedhuang@ujs.edu.cn

***Abstract*—This paper investigates the semantic information-providing web service composition problem, and proposes a heuristic method of web service automatic composition based on a planning graph model. According to the characteristics of information-providing semantic web service discovery, the service composition is partitioned into two modules: the construction of the planning graph and the search for composition scheme. Thus the satisfaction of the service composition can be judged in polynomial bounded branching time. The rich relationship information from varying percentages of partial to full matched service network is fully considered in the construction process of the planning graph, and the search module of composition scheme further uses a corresponding heuristic service selection strategy. Finally, the experiments verify the efficiency and the strong adaptability of the automated method.**

***Index Terms*—information-providing; semantic web; heuristic intelligence; service composition**

I. INTRODUCTION

With the continuous development of Service Oriented Architecture (SOA) and the maturity of all kinds of web services standards; the problem of interoperability between heterogeneous systems is being solved effectively [1]. However, the problem of semantic web service composition remains a very complex task. The complexity mainly comes from the growing number of web services and the instability of the service itself plus the inconsistency of service description language [2], until recent progressing of D language. Therefore, the study on the problem of web service composition not only causes widespread concern in the business process management and workflow organization, but also attracts much attention of many experts from Artificial Intelligence (AI) field [3], [6]. Based on above background, the paper considers service binding time, its theory and technology basis for service composition, and focuses on the study of the problem for information-providing web service automatic composition. Inspired by the planning graph model of artificial intelligence [4], [7]; the paper proposes a Heuristic Web Service Composition Approach (HWSCA). The HWSCA makes full use of the service relationship information of service network in the composition process and has high efficiency and strong adaptability, due to its reduced state space property.

The project is supported by Suqian College fundings: Jiangsu Provincial Higher Education Reform Research Fund Grant No. 2013JSJG520.

doi:10.4304/jetwi.6.3.298-304

II. INFORMATION-PROVIDING WEB SERVICE COMPOSITION

According to the classification of the service function [5], [8]; an information-providing web service w usually consists of a set of input parameter $w^{in}=\{I_1,I_2,\ldots\ldots\}$ and a set of output parameters $w^{out}=\{O_1,O_2,\ldots\ldots\}$. If we invoke w with the input parameters w^{in}, it will return a set of parameters w^{out} as its response.

Definition 1. User' s Request: A user's request r can be expressed by a 2-tuple (r_{in} , r_{out}), where r_{in} represents a set of initial input parameters and r_{out} represents a set of target output parameters.

Definition 2. Atomic Service Discovery Problem: Given a user's request r= (r_{in} , r_{out}) and domain ontology T, atomic service discovery problem can be described as follows: Find a web service w, where w meets the following two conditions:

(1) $r_{in} \succ_T w^{in}$ (2) $w^{our} \cup r_{in} \succ_T r_{out}$

Atomic service discovery problem can be resolved as follows. Query each abstract service in the service network. If an abstract service that satisfies definition 2 is found, we only need to return the concrete service that meets the abstract service to the user. Otherwise, if we can't find an abstract service that meets the user's requests, we need to first find a group abstract service set which can potentially meet the user's requests and then select multiple abstract services from the abstract service collection and combine them to accomplish the user's goals. Such problem is called information-providing web service composition.

For information-providing web service composition problem, because it is very similar to the traditional AI planning problem, it can also be formalized by a 5-tuple (P, W, r_{in} , r_{out}, T) for easier programming:

1). P represents the set of all parameters.
2). W represents a set of web services.
3). $r_{in} \subseteq P$ r_{in} represents the initial input parameters.
4). $r_{out} \subseteq P$ represents the target output parameters.
5). T stands for domain ontology.

At the same time, this model can also be expressed by state space model, for easier simulation.

Here the description of the problem of Information-Providing Web Service Composition (IPWSC) formalizes as follows.

Definition 3. IPWSC Problem: Given a user's requirement (r_{in} , r_{out}) and domain ontology T, IPWSC problem can be described as follows. From the existing collection W of abstract services we find the service collection sequence <W_1,W_2,……,W_n> which contains the fewest services with the minimum time steps and which needs to meet the following three conditions:

1). $(r_{in} \succ_T w_{1i}^{in})$, $1 \le i \le |W_1|$;
2). $(r_{in} \cup W_1^{out} \cup W_2^{out} \cup ... \cup W_{i-1}^{out}) \succ_T w_{ij}^{in}, 1 < i \le n, 1 \le j \le |W_i|$;
3). $(r_{in} \cup W_1^{out} \cup W_2^{out} \cup ... \cup W_n^{out}) \succ_T r_{out}$;

The services in each collection of the sequence can be concurrently executed, so if we serialize the services in each collection we will get the service sequence of serial execution. Obviously, the process of resolving the IPWSC problem is to find the fewest service from the available services and make that service serial execution satisfying the user's request.

IPWSC problem is NP complete [9], [10]; therefore, when the quantity of the services is quite large, it will be very difficult to find the optimal solution [11], [12]. This paper mainly reduces the complexity of the problem from the two aspects. First, in order to reduce the number of the services, we put forward service discovery method based on a limited number of service network. Second, in order to improve the efficiency of service composition algorithm, we further propose web service automatic composition method based on a heuristic graph planning model.

III. Composition-Oriented Service Discovery

Actual user's requests in the field of service discovery often can not be satisfied by a single service [11]. At this point it is necessary to extract the services from the service library which are related to the user' request and have a contextual relationship. On this basis, we can get a service composition process that satisfies the user's request by using a service composition process. Based on this observation, the Composition Oriented Service Discovery（COSD）problem is given, and the concept of service inverted table is introduced here.

Definition 4. Composition-Oriented Service Discovery Problem:

Composition-oriented service discovery problem can be described as follows. Given a user's request r, we find a collection of services in the service network and combine part or all of these services, then we get the service collection sequence which meets the request r.

Definition 5. Service Inverted Table: Given the output parameters' concept collection of all the services, service inverted table consists of two parts of key and key values. Where each key corresponds to one concept of the concept collection, and key value contains all the services' list that can produce the concept.

According to the definition of service inverted table, for a concept, a service which can output this concept or a concept which is equivalent to it, is also allowed to output this concept. Therefore, in order to get all the services which can output this concept, we first need to get all sub concepts and equivalent concepts of the concept based on concept hierarchy about domain ontology, and then to get the services' list which can output the concepts from the service inverted table. When the collection which needs the user's request is got, according to the service inverted table, and according to the partially matched and fully matched relations between the services in the service network, we can arrive at the services that meet the input of each service in the collection. These services are added to the service collection. Note that the total amount (in terms of the percentage of overlapping in service and service network) of partial match can be adjusted by the web master or end user, to trade of the final semantic search engine performances, e.g. speed versus accuracy, in real life applications.

We repeat the above process until no service can be added. In this way, we get all these services that are directly or indirectly (to certain extends) related to the user's requests. If there is a composition service which satisfies the user' requests, then the component service in the composition service must be in the returned results. That is, the process can solve composition-oriented service discovery problem. However, the returned result may still contain a large number of unrelated services. In order to further narrow the scale of the service, on the basis of the process, this paper devises an improved algorithm of Composition Oriented Service Discovery. According to the analysis of the service network, the algorithm will look up the largest repetition number of service process by the service relationship, set it as the network average shortest path, and delete those services which have no prior service and the input of which cannot be satisfied by the user.

COSD algorithm is described as follows:

First we judge whether the algorithm is called for the first time, and if so, depending on the service inverted lists, set the variable TS as service collection RS which can satisfy the user's expecting output concept. Otherwise, the variable TS is directly set as the service collection S which has been got. Then, for each service in the TS, first of all the algorithm determines whether the user's input can completely satisfy the service's input; if not, the algorithm judges whether the service has a prior service; if not, the service will be deleted, otherwise, according to the relationship information between services the algorithm adds the services which match the service partially or completely to the TS for given percentage. Repeat this process until it reaches a predefined maximum repeating number N (which is set according to the desired percentage of matching) or until no services can be got. Finally, the algorithm returns the acquired service set S.

IV. SERVICES' AUTOMATIC COMPOSITION ALGORITHM

Since the information-providing web service automatic composition problem is very similar to the AI planning routine, it can usually be solved by using the existing method of AI planning solution. The methods can be roughly divided into Total Order Planning (TOP), Partial Order Planning (POP), Graph Planning (GP), Hierarchical Task Network Planning (HTNP), etc. Graph planning method proposes an unique search space. The connection between the action and proposition of the space is expressed by a compact way, such as to improve the efficiency of planning algorithm greatly. On the basis of the service network, this paper puts together a Graph Based Heuristic Web Service Composition (GBHWSC) algorithm. The pseudo code of the algorithm is shown here.

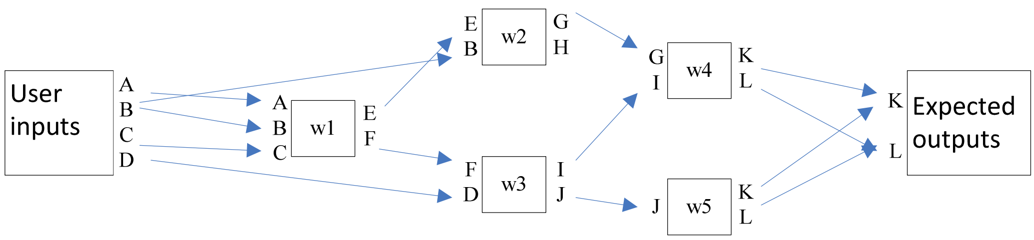

Figure 1 Example of information-providing web service composition

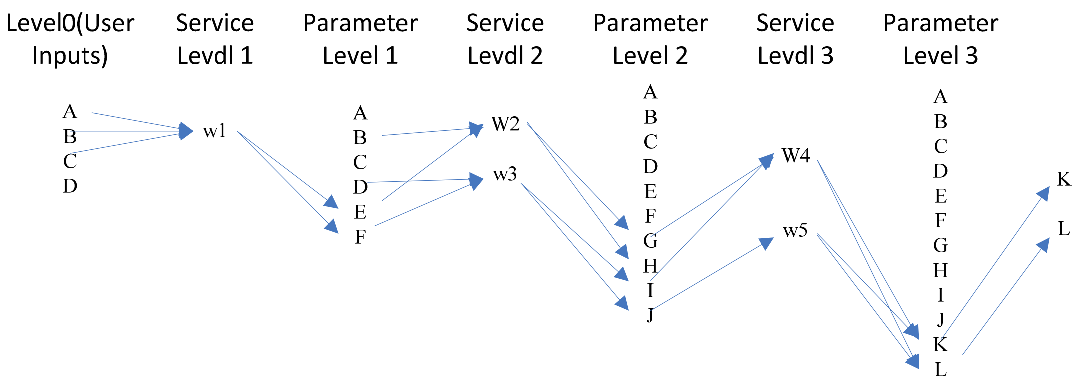

Figure 2 Example of the service planning graph model

Input: abstract service network As=(V_{as}, PF_{as}), a set of abstract services W, user's goal g=<IN_g, OUT_g>, Domain ontology G

Output: a sequence of sets of services Sqlist

(1). planninggraph = pgbuilder(g, W, As, G);

(2). If (planninggraph != Null) {

(3). Sqlist = regression(Planninggraph, g);

(4). Return Sqlist; }

(5). Else

(6). Return failure;

The algorithm firstly uses a forward search process (i.e., the function of the function pgbuider) to build a service planning graph. Then the algorithm uses heuristic way to search for the possible best solution (That is, the function of the function regression).

A. Service Planning Graph Algorithm

Service planning graph is a kind of directed acyclic hierarchical graph that contains two types of nodes and two kinds of edges. The nodes are divided into two categories: parameter nodes and service nodes. Where the parameter nodes appear only in the parameter level and the service nodes appear only in the service level. The parameter level and the service level appear alternately. The first level of service planning graph is a parameter one whose nodes are composed of the input parameters of the user's request. The second level is a service one, and for each service node on it, its input parameters can all be satisfied by the parameters in the first level. The third level is a parameter one, the parameters are composed of the parameters of the first level and the output parameters of all services in the second level. And so on, the last level of the service planning graph can semantically contain the output parameters which the user desires.

The steps of service planning graph algorithm are as follows:

1). The inputs provided by the user are set as the parameters of the first level of the service planning graph.

2). On the basis of the parameter and domain ontology in the previous parameter level, we can get the services which can be performed under this parameter set, and add these services to the current service level.

3). According to the output parameters of all services in the current level together with the parameters of the first parameter level we construct the next parameter level.

4). Repeat 2) and 3) until there is no service which can be added to the current service level or the parameters in the obtained parameter level can semantically contain the output parameters which the user's request desires for.

5). If there is no service which can be added to the current service level, the algorithm returns failure, otherwise, it returns the service planning graph.

Here we give an example to show the algorithm process of service planning graph. Figure 1 shows an example of information-providing web service composition, where w1, …,w5 are five web services. They are indicated by a rectangle respectively and each input or output parameter of the web service is annotated in both side of the rectangle. The user's request is (A, B, C, D), then outputs (K, L). According to the service planning graph algorithm, we can get solution of the service composition problem. It is shown in the figure 2 above.

can be directly taken out, after the judgment, two services can be all performed in the Parameter Level 2, so the two services can be added to the current level (Service Level 3).

7) Merging the output parameters (K, L) of the two services w4 and w5 in the previous level with the parameters of the Parameter Level 2, we can get the parameters (A, B, C, D, E, F, G, H, I, J, K, L) of the next parameter level (Parameter Level 3).

8) For(K, L) ⊆ (A, B, C, D, E, F, G, H, I, J, K, L), service planning graph algorithm stops and returns the result.

B. Service Planning Solution Search Algorithm

In order to obtain the best solution of the problem, this paper proposes a heuristic planning solution search algorithm (regression). The heuristic strategy is as follows.

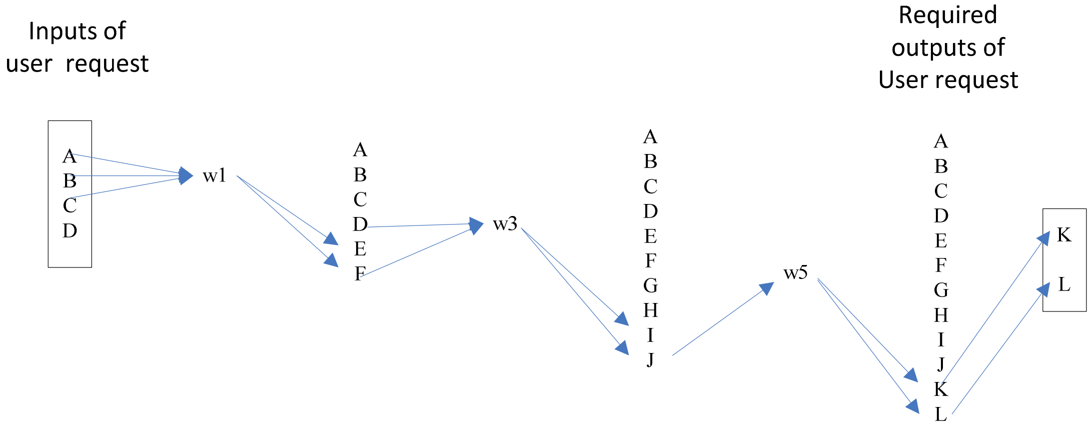

Figure 3 Example for service composition planning solution

In view of the above example, the algorithm pgbuilder constructs the service planning graph as follows:

1) We add the input parameters (A, B, C, D) which are provided by the user's request to the first parameter level (Parameter Level 0).

2) According to the parameters on the Level 0, only w1 is executable in the current case, therefore, we add w1 to the first service level (Service Level 1).

3) We add the output parameters (E, F) of w1 together with the parameters (A, B, C, D) on the first level to the second parameter level (Parameter Level 1), thus the available parameters on Parameter Level 1 contains the six parameters (A, B, C, D, E, F).

4) According to the matching relationship between the services in figure 1, we extract the service w2 and w3, after the judgment, the two services all can be performed on Parameter Level 1, so they can be added to the current service level (Service Level 2).

5) Merging the output parameters (G, H) and (I, J) of the two services W2 and W3 in the previous level with the parameters on Parameter Level 1, we can get the parameters (A, B, C, D, E, F, G, H, I, J) of the next parameter level (Parameter Level 2).

6) Again according to the matching relationship between the services in figure 1, the services w4 and w5

1) From the beginning of the last level of the service planning graph, based on the parameters on the current parameter level, the service from the previous service level which has more outputs and can meet the parameters has a priority to be output. When the number of the outputs is equal, we choose the service that has less input. That is, let w_1 and w_2 be two services, if $|W_1^{out}| > |W_2^{out}|$, we prefer w_1 or if $|W_1^{out}|=|W_2^{out}|$ and $|W_1^{in}|<|W_2^{in}|$, w_1 is preferred.

2) The service selected before has a good chance to be chosen. The algorithm is described as follows.

First the parameters of the last parameter level which can satisfy the user's expecting output are set as the current target. Then, for each sub target in current goal (i.e., target parameter), according to the heuristic strategy, the algorithm finds the appropriate service from the previous service level. If finding it, the algorithm will add the input of the service to the current target as a sub goal, and if not, the algorithm will preserve the sub goal. When search process reaches the first parameter level, the algorithm stops and returns the planning solution.

The algorithm takes the user's requests as input and outputs the service set sequence which meets the user's requests by reversely searching the service planning graph.

As for the service planning graph shown in figure 2, the algorithm invokes the heuristic service planning solution search algorithm to deal with this, and can obtain the planning solution for the service composition problem, which is shown in figure 3. The steps are as follows:

1) Add (K, L) to the current target.

2) W4 and w5 can all satisfy the sub goal K, and both the number of output are equal. But the input number of w5 (which is 1) is less than the input number of w4 (which is 2), therefore the algorithm chooses w5. As for sub target L, because w5 are satisfiable, the algorithm chooses w5 too.

3) Remove K and L from the current target, and add the input J of w5 to the current target.

4) Since there is only the w3 that is satisfiable to J, so the algorithm chooses the w3, and removes J from the current goal. At the same time the algorithm adds the input (D, F) of w3 to the current target.

5) In view of the subgoal F, because only w1 can meet its need, the algorithm chooses w1, and then deletes the subgoal F at the same time. For subgoal D, because there is no any service which can meet its need, the algorithm keeps it in the target.

6) Because a subgoal D can be satisfied directly by the user's input, the algorithm deletes it directly. At this point, the current target is empty, so the algorithm ends.

We can see from the above process, if we do not use the heuristic strategy in step 2), we may choose the w4 or choose w4 and w5 simultaneously. If we only select w4, then in order to meet the two inputs G and I of w4, we need to choose w2 and w3 at the same time, therefore the final service composition will contain five services at the most which are more than the service composition which has three services gained by using heuristic strategy. In this case, the solution that the algorithm regression finds is the optimal planning result.

V. Simulation Experiment And Result Analysis

To verify the superiority of the method GBHWSC, this paper implements another Simple Service Composition Algorithm (SSCA). The algorithm is also based on planning graph. The algorithm does not use composition oriented service discovery service in the network information and the heuristic strategy. The algorithm is simply called SSCA. For the two aspects about the composition efficiency and success rate, the two algorithms are compared as follows.

A. *Comparison of Composition Efficiency*

The experiment specifies three goals G1, G2 and G3, which are shown in table 1.

For each goal, in three service networks of different sizes (150,351 and 579), we carry out twenty times experiments with the two algorithms respectively. The nine results are all shown in figure 4 here. We can see from the graphs that for the first goal G1, the algorithm GBHWSC has not apparent advantage, but for the execution time towards the other two goals, the algorithm GBHWSC clearly shows more advantages. Here the main reason is that the solution of target G1 contains only a single service, so the actual execution process of the two algorithms can essentially be seen as the matching process of atomic services, thus the time difference between them is not significant. But for the other two goals, the optimal solution of each goal contains four services, and because GBHWSC algorithm first use COSD to filter the services and make full use of the relationship information between the services in the process of combination, the efficiency is improved greatly. From the processing time of the goals G2 and G3 we can see that with the increase of scale of the services, the algorithm SSCA will increase the time to 15075.75 ms quickly, while the algorithm GBHWSC increases to 4948.5 ms at maximum. It is clear that the algorithm GBHWSC has good scalability with the increasing of the service scale.

TABLE I.
THREE SPECIFIED TARGET

Goal	Inputs	Outputs
G1	EntrezGene_ID	GenBankRecord
G2	EMBLaccession	PhylogeneticRecord
G3	BiologicalSequence, BioinformaticsDataResource, BioinformaticsAlgorithm	PhylogeneticRecord

B. *Comparison of the Composition Successful Rate*

The successful rate of the algorithm is the ratio that the algorithm can find the solution among all the attempts. In order to compare effectively the successful rate of the two algorithms, the experiment firstly generates four testing service sets which have the scales of 500, 1500, 2500, 5000 respectively. Then the experiment tests each testing set with 100 users' requests generated randomly. The experimental results are shown in figure 5 here. As can be seen, when the scale of the services reaches 500, both two success rates are almost the same; but with the increasing scale of the services, the success rate of the algorithm SSCA falls behind dramatically, and when the service number reaches 5000, the algorithm is not competent to the task of service composition any more. In contrast, the decline of the success rate of the algorithm GBHWSC is relatively smooth. All these show that the algorithm GBHWSC has better adaptability.

VI. Conclusion

First, on the basis of the service inverted list approach, this paper gives a composition oriented service discovery method based on a size controllable service network. A percentage of the semantic partial to full matching concept is used to control this size. The main purpose of the controllable service network based discovery is to reduce the scope of the services, thereby to reduce the complexity of service composition, and speed up the searching process, without loss of semantic meaning.

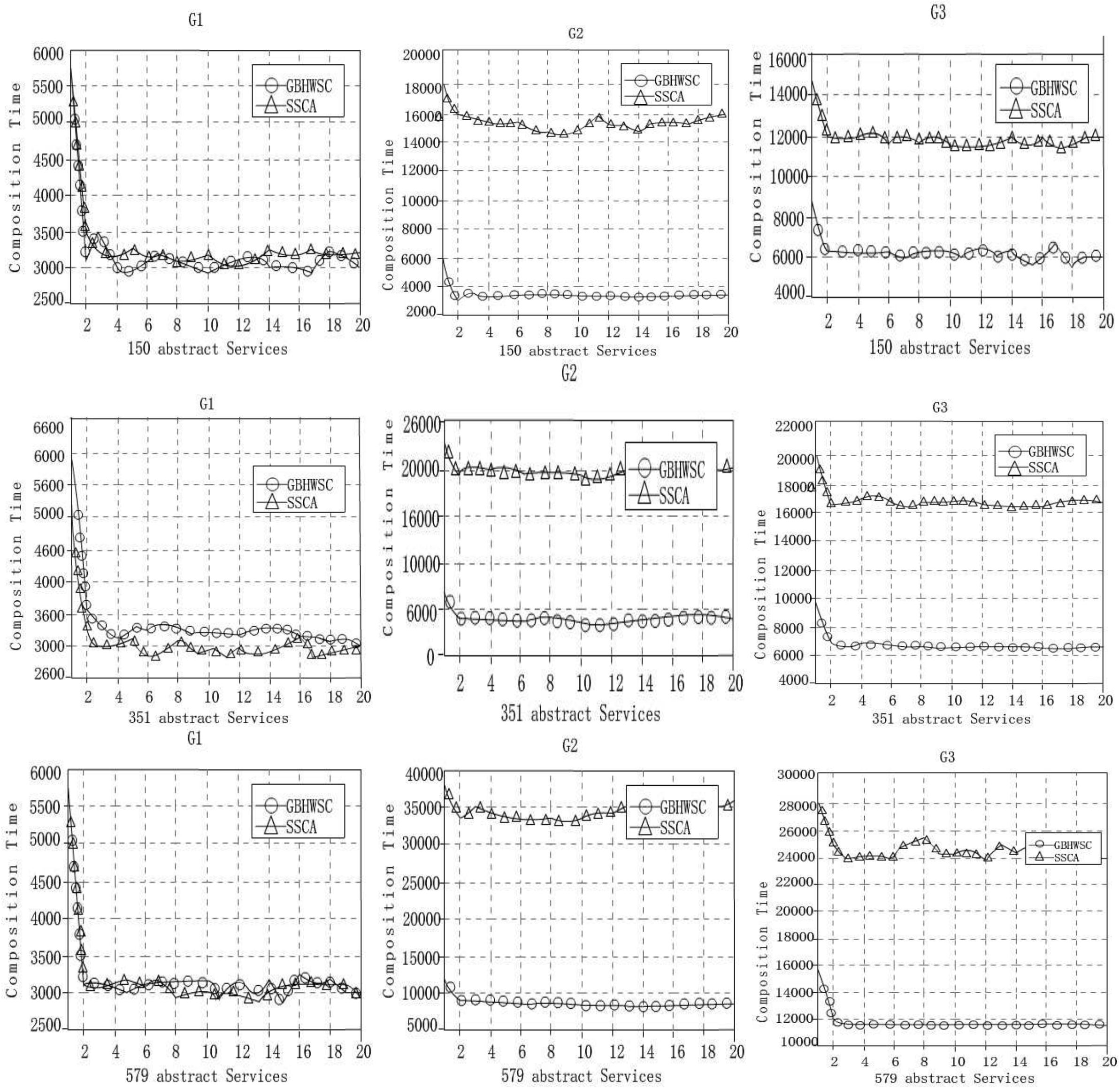

Figure 4 Comparison of the efficiency of service composition algorithms

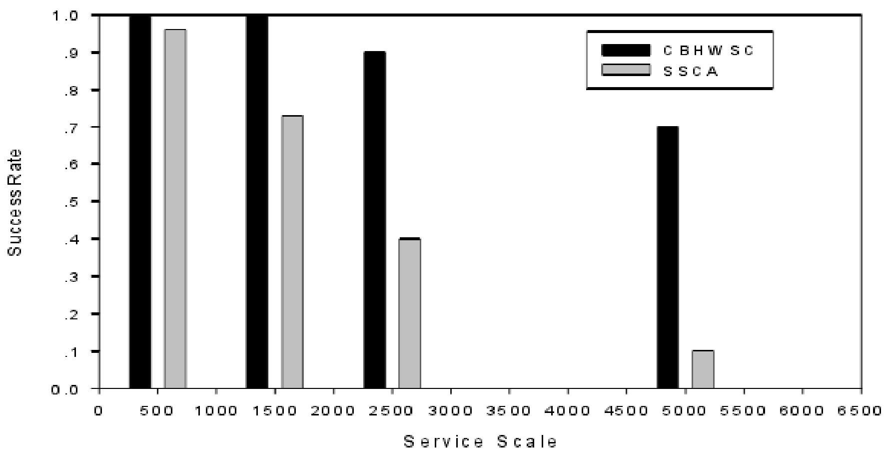

Figure 5 Comparison of the successful rate of service compositions

Second, the paper presents a graph based heuristic web service composition method. The method is divided into two stages:

(1) The construction phase of the service plan, through which we can judge the satisfiability of the problem of service composition in polynomial branching off time;

(2) The search phase of the service composition solution, by using a heuristic search strategy, we can reach suboptimal solution of the problem in polynomial time as well. Finally, the experiments verify the efficiency and strong adaptability of the service heuristic search algorithm.

For future work, we are building a testbed for our industry partner in online store business, using html5 and Java for mobile client side application and with html5 and D languages for cloud side server integration. The purpose is to demonstrate the advantage of applying this adjustable match algorithm to make the personalized search engine applications.

ACKNOWLEDGMENT

The authors wish to thank Prof. Yuhui Shi, Dr. Qihong Yu, Miss Qing Li and Mrs. Helen Liu for improving the paper. This work was supported in part by the grants from Provincial Education Department and the Research Contract from M2M technology; Higher Education Reform Research Fund Grant No. 2013JSJG520.

REFERENCES

[1] Anagnostopoulos, C. and S. Hadjiefthymiades, Semantic Web Service Discovery: Methods, Algorithms and Tools, in Semantic Web Services: Theory, Tools and Applications, J. Cardoso, Editor, 2007, IGI Global, pp. 240-280.

[2] Rao, J.H., et al., “A mixed initiative approach to semantic web service discovery and composition: SAP's guided procedures framework,” IEEE International Conference on Web Services, 2006: Chicago, IL. pp. 401-408.

[3] Han Yong-Guo, Sun Shi-Xin, “Study of Algorithms for Dynamic Service Composition and Optimal Service Composed,” Computer Science, 2005, 32(12):104-105, 139.

[4] Xianrong Zheng, Yuhong Yan, “An Efficient Syntactic Web Service Composition Algorithm Based on the Planning Graph Model,” IEEE International Conference on Web Services, 2008: pp.691-699.

[5] Danilo Ardagna, Barbara Pernici, “Adaptive Service Composition in Flexible Processes,” IEEE Transactions on Software Engineering, 2007, 33(6):369-384.

[6] Guo Yu-Bin , Du Yu-Yue, Xi Jian-Qing, “A CP-Net Model and Operation Properties for Web Service Composition,” Chinese Journal of Computers, 2006, 29 (007):pp1067-1075.

[7] Qian Zhu-Zhong, Lu Sang-Lu, Xie Li, “Automatic Composition of Petri Net Based Web Services,” Chinese Journal of Computers, 2006, 29(007):pp1057-1066.

[8] Lei Lihui, Duan Zhenhua, “Specification and Verification of Composite Web Services Based on Extended Projection Temporal Logic,” Journal of Xi’an Jiaotong University, 2007,41(10): pp 1155-1159.

[9] Hu Jia, Feng Zhi-yong, “Semantic Web service composition framework based on abstract template,” Journal of Computer Applications, 2009, 11(29), pp:3149-3153.

[10] Zhang Pei-yun, Huang Bo, Sun Ya-min, “Web Services Hybrid Composition Based on Semantics and Template,” Journal of Nanjing University of Science and Technology (Natural Science), 2008.32(04):pp.425-428.

[11] Fu Yan-ning, Liu Lei , Zhang Chang-hai, “Composition of process template-driven Web services,” Journal of Jilin University (Engineering and Technology Edition), 2008.38(S2):pp.169-172.

[12] A. De Leon Battista, N. Villanueva-Rosales, M. Palenychka, M. Dumontier, “SMART: A Web-Based, Ontology-Driven, Semantic Web Query Answering Application,” Proceedings of the Semantic Web Challenge 2007, Vol. 295.

Yuemin Wang was born in 1967, China. She received her Master degree in Computer Engineering from Suzhou University, Jiangsu, China, in 2004. Since then, she has been the key founding member of the Third Department, sponsored by Jiangsu University, majors of Computer Science and Electrical Engineering, in Suqian College, where she is currently Vice Chairman and Associate Professor of the Department. Her research interests include intelligent information processing, sensor network applications etc.

Jun Steed Huang was born in Shanghai, China. He received his doctoral degree in 1992 from a Joint Ph.D program between Southeast University China and Concordia University Canada. He worked at a number of places including Bell Canada, Lockheed Martin USA, Carleton and Ottawa Universities. He is also a Distinguished Professor of School of Computer Science and Communication Engineering at Suqian Collegue sponsored by Jiangsu Universities in China. His current research interests include biological sensor network, visible light automotive network and artificial quantum intelligence web service.