Maximum Portfolio: A Query Condition Optimization Method

Zhi Yang
State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, China
Email:yangzhi9721@yahoo.com.cn

Budan Wu, Junliang Chen, Pingli Gu
State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, China
Email: {wubudan, chjl }@bupt.edu.cn, gplqy98@163.com

***Abstract*—Web Service is becoming the next generation of web-based application. With enhancement of quality of services and increasing quantity of services, service discovery becomes important. In the process of discovery, although the user is not familiar with service interfaces, he wants to obtain the best services with some limited simple keywords. Because the keywords are often incomplete or wrong or fuzzy to match with the services in service library, it is difficult to get the satisfying services. This paper provides an approach named as Maximum portfolio to deal with the problem. The aim of maximum portfolio is that the suitable composition is found out from ambiguous input keyword set. The improved service matching algorithm is proposed and applied to the Service Generating Platform. The result of experiment shows the algorithm can improve efficiency of service discovery.**

***Index Terms*—Service Discovery Process and Methodology, Maximum Portfolio, Service Matching Algorithm, Ontology**

I. INTRODUCTION

Web Service is one of implementation modes of distributed systems and becoming the next generation of web-based application. Available Web services are described and advertised, and then they can be discovered and composed by other applications to create new value-added systems [1]. Web services facilitate dynamic formation of Web-service-based software applications according to user's demand [2]. So they are becoming popular and widely accepted due to their accessibility and compatibility [3]. In SOA (Service-Oriented Architecture) , Web service discovery is extremely important. In the process of discovery, the query keyword array is the start of service matching algorithm. Although the user is not familiar with service interfaces, he wants to obtain the best services with some limited simple keywords. Because the keywords are often incomplete or wrong or fuzzy to match the services in service library, it is difficult to get the satisfying services. So how to find out suitable services with the keywords is very important and urgent.

Recently, in the service matching algorithms the keyword array is treated as accurate [6, 7, 8 and 9]. In the algorithm in UDDI [6], it is hypothesized that the keywords are available and accurate. In the algorithms in literature [9], the keywords are divided into IOPE, but it does not treat the accuracy of keywords. Our earlier work [7] which provides a new service matching algorithm did not consider whether the keywords are accurate. The algorithm's premise is that keywords are correct.

Existing Web service matching approaches pay attention to design and realize algorithms while the query keywords input by the user are ignored. Because that the interface descriptions of Web services are often terse and cryptic [5] and that the user does not understand the detailed interfaces of Web service, it is very difficult for the above approaches to solve the above problem. How to deal with the problem is the concerned content of this paper. A solution scheme is proposed. We make use of the idea and approach of maximum portfolio to find out the user's request. Using this method, it is not difficult to solve the above issue. Then we propose an improved service matching algorithm. And in detail, the performance of the matching algorithm is analyzed.

This paper is organized as follows. Section 2 defines the maximum portfolio. An improved service matching algorithm is proposed and analyzed in section 3. Section 4 verifies this algorithm by an experiment. The last section draws to a conclusion.

II. DEFINITION OF MAXIMUM PORTFOLIO

User's input information is inaccurate. So in order to obtain the user potential requirement, the theory of maximum portfolio is provided. The definition of maximum portfolio and portfolio granularity are shown as follows.

Definition 1 Maximum Portfolio: There exists a set $S = \{I_1, I_2, ..., I_n\}(n = 1, 2, ...)$ and there is a

doi:10.4304/jsw.6.12.2486-2494

constraint condition C. $M = \{I_{j_1}, I_{j_2}, ..., I_{j_m}\}(1 \le j \le n, m \le n)$ is a subset of S， and M satisfies C condition. That is, the composition of elements of set S can satisfy C. $R = \{M_1, M_2, ..., M_n\}$ is the set of all the compositions, in which $M_i(i = 1, 2, ..., n)$ can satisfy C condition. Then $MAX\{|M_1|, |M_2|, ..., |M_n|\}$ indicates maximum portfolio.

Definition 2 Portfolio Granularity: In above definition, the element number of composition $M = \{I_{j_1}, I_{j_2}, ..., I_{j_m}\}(1 \le j \le n, m \le n)$ is described by PG. Its value is

$$PG = |M| = |\{I_{j_1}, I_{j_2}, ..., I_{j_m}\}| = m \quad (1)$$

Note that

a. Maximum portfolio describes the relations among elements of set S. In a set $S = \{I_1, I_2, ..., I_n\}(n = 1, 2, ...)$, according to some condition some elements are together and form a new set. The set including the most elements that satisfies the condition is maximum portfolio.

b. Portfolio granularity is an index to describe portfolio degree. By means of this measure index, it is easy to compare among more than one composition.

Generally, in the set which includes n elements, there are three types of portfolio granularity (PG) as follows:

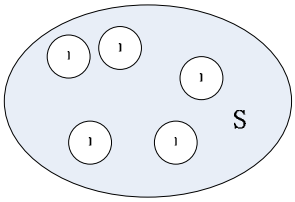


Figure1: PG=1

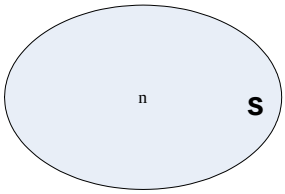
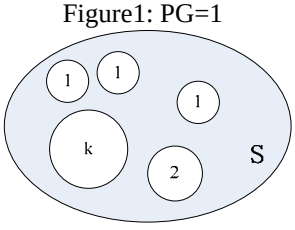


Figure2: PG=n



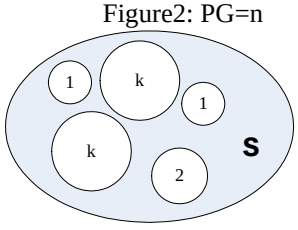


Figure3: PG=k

(k) indicates the set whose portfolio granularity is k($k = 1, 2, ...n$).

(S) indicates the whole set S.

Note: from the above figures, we can see that the biggest maximum portfolio may be all the element of the set (Figure 2) . Namely, set S is a composition and the composition is biggest. All the elements are in one area. The smallest maximum portfolio is one element (Figure1) . That is, there is no relation in any two elements of the set. The common maximum portfolio is shown as figure 3. There is one maximum portfolio in the left figure and more than one maximum portfolios in the right figure.

When maximum portfolio is applied to projects, one critical and challenging problem is how to obtain the maximum portfolio.

Two approaches are recommended as follows:

(1) The first step is that we examine one by one if each keyword of input parameter is effective. If with some keyword we can not get the available query result，then the keyword is deleted from the input parameter set.

The second step is that two keywords are chosen from the input parameter set every time. Then we justify whether the two keywords are effective. If they are available, we store them in the memory space M. Until all the two-keyword compositions are justified, we check the memory space M. If the M is not empty, the elements of M are candidate maximum portfolio. If the M is empty, we go on with the third step.

The third step is that three keywords are chosen every time. Then we justify with the three-keyword composition whether an available query result set is obtained from the ontology library. If the result set is not empty, the elements of memory space M are deleted and the three-keyword composition is put into M. Otherwise, we go on justify other three-keyword composition. Until all the compositions are judged, we check the memory space M. If the composition in the M is two keywords, the compositions in the M are maximum portfolios. Otherwise, we increase number of composition and repeat the third step, until the number of composition is equal to the number of the elements of input parameter set.

(2) The second approach is process of opposite compared with the first approach. It is that the number of elements of input parameter set is n and memory space M is created. The fist step we judge whether the whole input parameter set is available. If it is effective, the whole set is maximum portfolio. Otherwise, we go on with second step.

The second step is that n-1 elements are chosen every time. Then we judge if with the n-1 elements whether an available query result set can be obtained. If it is available, it is stored in the memory space M. Otherwise, another n-1composition is chosen and judged. Until all the compositions are judged, we check the memory space M. If M is not empty, the compositions in M are maximum portfolios. If M is empty, we carry on the next step.

The third step is that n-2 elements are chosen from the input parameter set every time. Then we judge if it is effective. If it is available, it is stored in the memory space M. Otherwise, another composition is chosen and judged. Until all the compositions are judged, we check the memory space M. If M is not empty, the compositions in the M are maximum portfolios. If M is empty, we decrease the number of composition and repeat the third step, until the number of composition is equal to 1. The figure of the process is as follow:

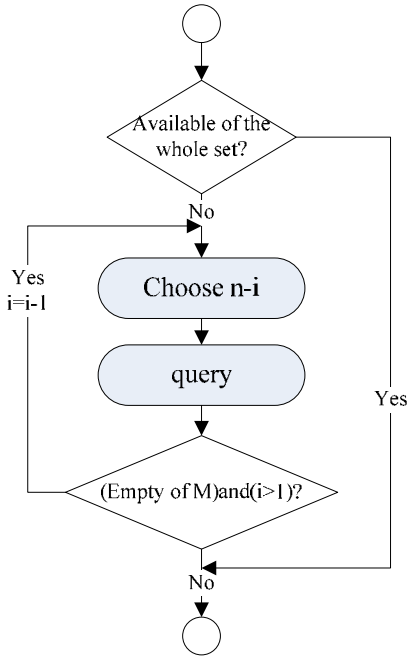


Figure4: The Process of Maximum Portfolio

From the two approaches, we can see that how to obtain the maximum portfolio of a set is a complicated and critical problem. Because that in the second approach, the first satisfying portfolio is maximum portfolio, in this paper the second approach is adopted.

By means of the thought of maximum portfolio we design a Filter to achieve maximum portfolio from the set of user's input keyword array. The process is showed as follow.

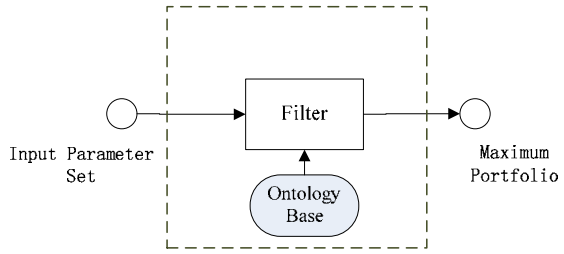


Figure5: Input Parameter Filter

Note that the input is user's input keyword set which is user's request. The output is maximum portfolio. Because it may be more than one, sometimes maximum portfolio is a set. The figure indicates using the ontology library the Filter extracts maximum portfolio set from input parameter set.

III. Improved Service Matching Algorithm

Based on our earlier work, an improved service matching algorithm is provided combined with the maximum portfolio. Firstly we pretreated user's keyword array in ontology library to obtain the maximum portfolio. Then by means of matching algorithm accurate result set is got.

The theoretical knowledge about service matching algorithm is introduced as follows.

A. Query Rewriting [10]

In the set of database relation, $T=\{T_1,T_2,...,T_n\}$ and its view set $V=\{V_1,V_2,...,V_n\}$, query Q are about set T of database relation. If there is a query Q_1 which searches at least a view in the view set V. Moreover, query result of Q_1 is consistent with query result of Q. So we claim that Q_1 is the query rewriting of Q.

B. Ontology

Ontology is a very important semantic technology. It is a description of the objective concepts and relationships [11].Ontology was originally a philosophical concept [12]. In 1998, Studer et al. further studied ontology on the base of study of predecessors and provided that ontology is a clear formal specification of shared conceptual model [13]. As a tool for knowledge representation, ontology structures the relations between knowledge points and provides a description or explanation of the domain knowledge to access the knowledge in a field.

Ontology can be described by OWL (Web Ontology Language) . OWL is the standard of ontology description language recommended by W3C. In order to solve semantic interoperability problem, explicitly it expresses the meaning of gloss and terms and their interrelationships [14]. OWL consists of three parts:

Individual: is the object which we are concerned about in some area;

Property: is a binary relation between individuals. That is, individuals create relations through properties.

Class: is a set of individuals.

Using the formal description method, OWL describes relationship between class and the members of class. At present, database is a very important storage approach of ontology and ontology is organized in the database according to some strategy and accessed by means of manipulation and management capabilities of existing database. As the relational database technology matures, most existing works of ontology data management take database management system of relation or object-relation as back storage. Currently popular relational database stores ontology in the database without losing the semantic.

By use of database storing ontology, there is an obvious advantage. That is, factually the operation of database table is the operation of ontology. Accessing ontology is equivalent to accessing database. So ontology library and database become one.

Generally, ontology can be described with some formats. For example, the following figure is about ontology hierarchical structure.

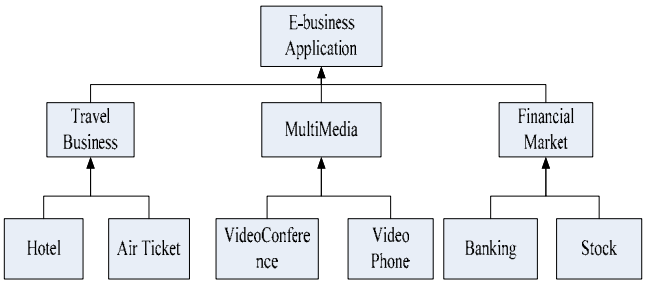


Figure6: domain ontology structure

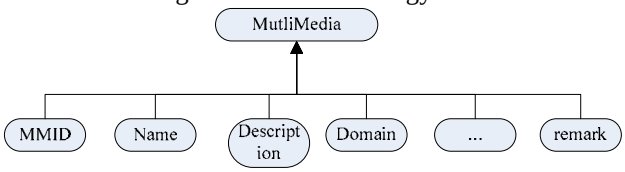


Figure7: the individuals of MutliMedia ontology

Note that the E-businessApplication can be classified into TravelBusiness, MultiMedia and Financial Market. MultiMedia can be classified into VideoConference and Video phone. MutliMedia consists of MMID, Name, Description, and so on. From the figure, we can intuitively see the links between individuals and hierarchical relation. By means of the theory of the relationship between part and whole of ontology, this paper provides an improved service matching algorithm.

C. Service Matching Algorithm

In this algorithm, firstly, the preprocessing is done to the user's input keyword array and we can obtain the maximum portfolio. Then query rewriting by use of ontology is done and in this case we can change query from keywords query to ontology query. The main process is divided two stages: grammar query stage and semantic query stage. After getting user's maximum portfolio requirements, we can analyze logical relationship and find out potential ontology information. Then we search ontology detail information in the database. In the end, we take ontology as query conditions to search data in the database.

First stage: (Grammar query stage)

Obtains the maximum portfolio and eliminates independent items. We adopt the second approach of generating maximum portfolio of set and obtain the maximum portfolio. Then the preprocessing of database set $T = \{T_1, T_2, ..., T_n\}$ and view set $V = \{V_1, V_2, ..., V_n\}$ can get rid of independent records with query and reserve the query-related records. The view set $V' = \{V_1', V_2', ..., V_n'\}$ is got for the second stage.

Second stage: (semantic query stage)

(1) Rewriting query. The user query conditions are taken as a part of the ontology. By using part of ontology in the ontology library the other parts of ontology can be obtained. That is, through the part we can get the whole. The user query is divided into $q(X) :- X_1, X_2, ..., X_n$. Among them, $q(X)$ is logical head which indicates the whole query, $X_1, X_2, ..., X_n$ is the body of conditions (usually, the conditions are not complete) , which is used to search the other parts of ontology in the ontology library. We get the individuals of ontology to compose the whole query condition $C = \{C_1, C_2, ..., C_n\}$.

(2) Matching services. The $C = \{C_1, C_2, ..., C_n\}$ is taken as query condition and through it we can rewrite the user query. That is, we change the query from keywords query to ontology query. Finally an available result set is obtained.

The process of the grammar and semantic query stages is as follows:

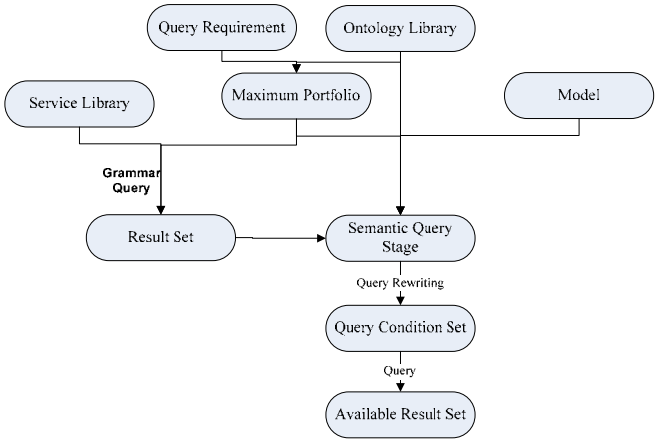


Figure8: grammar semantic query process

In this process, the ideas of ontology and database model are used in the semantic query stage. Its process is that the whole ontology is obtained through the part by means of query. Then we rewrite query and get query condition set. In the end available result set is got.

For service query, firstly we define and describe the service.

[15] Provides uniform definition of services as follows:

{
Primary Information and Provider Information
Functional Description
Quality Description
Other Attributes Description
}

Referring to above four parts, we set down detailed service structure as follows:

{
ServiceName // service name
FunctionDescription// function explanation
Domain// service application domain
ApplyScope //service applicative scope
Input //service input parameters
Output //service output parameters
Precondition // the precondition of service
Postcondition // the post condition of service
Restrictcondition // the restricted condition of service

Namespace // the namespace of WSDL file
Location // the location of WSDL file
}

WSDL [18] (Web Service Description Language Web) is an XML-based language and used to syntactically describe a Web service at the interface and binding levels. At the interface level, abstract interfaces of a Web service are described as interfaces which comprise a set of operations. An operation is in turn defined by its inputs, outputs and fault messages. XML Schema language is used to describe the content of those messages. At the binding level, the service's abstract interface is bound to a particular transport protocol defining specific implementation information such as encoding format and address information [1]. WSDL documents are semi-structured data that describe the functional and non-functional semantics of services [2].

Service description structure with WSDL is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
   xmlns:tns="http://www.zzl.org/Sum"
   xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   targetNamespace="http://www.zzl.org/Sum">
<wsdl:types>
<xsd:schema
targetNamespace="http://www.zzl.org/Sum">
<xsd:element name=" MatchRule">
<xsd:complexType>
<xsd:sequence>
<xsd:element name=" ServiceName " type="xsd: string "
/>
<xsd:element name=" FunctionDescription "
type="xsd:string" />
<xsd:element name=" Domain " type="xsd:string" />
<xsd:element name=" ApplyScope " type="xsd:string" />
<xsd:element name=" Input" type="xsd:string" />
<xsd:element name=" Output" type="xsd:string" />
<xsd:element name=" Precondition" type="xsd:string" />
<xsd:element name=" Postcondition" type="xsd:string"
/>
<xsd:element name=" Restrictcondition"
type="xsd:string" />
<xsd:element name=" Namespace" type="xsd:string" />
<xsd:element name=" Location" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:schema>
</wsdl:types>
</wsdl:definitions>
```

An example is shown to explain the algorithm process.

For example: there exists service ontology Service_Ontology(ServiceID, ServiceName, FunctionDescription, Domain, Input, Output)， in which ServiceID is primary key.

Service relation view Service_view(ServiceID, ServiceName, Domain, ApplyScope, Precondition, Postcondition, Restrictcondition, Namespace, Location)

It links with Service_Ontology by ServiceID.

User query input: Q: FunctionDescription ： startConference, Domain: Multimedia, Input: s_id

User wants to get services of realizing starting conference in multimedia field from Service_view.

The steps of this algorithm are as follows:

Grammar query stage:

Firstly, from the three conditions the maximum portfolio need be obtained. Judging all the input parameters in the ontology library, none is obtained. That is, the user's input parameters are not suitable and we need to generate the maximum portfolio by the second approach of maximum portfolio. The number of elements of composition is two and the result is shown as follow:

Table1: maximum portfolio and granularity

NO	Composition	Availability	Granularity
1	C_1, C_2	Yes	2
2	C_1, C_3	No	1
3	C_2, C_3	No	1

From the above table, the NO1, which is C_1 and C_2 , is available. Therefore, it is not necessary to judge the case in which the number of elements of composition is one. The maximum portfolio is C_1 and C_2 . We take the pretreatment of Service_view. The process is as follow.

Query condition is:

C_1：FunctionDescription like startConference，C_2 ：Domain =Multimedia

Using logic *or* relation between conditions, namely, C_1 *or* C_2 , query result set gets rid of independent records. Grammar query available result set S_1 from Service_view is obtained.

Semantic query stage:

By means of condition C_1 and C_2 , using logic *and* relation between conditions, the query is done in Service_Ontology. Through parts of ontology we can get other parts data. If ontology records are obtained, query condition becomes C set.

So we change query from the keywords query to ontology query. Because that operation to ontology changes into operation to database and ServiceID is primary key, we can use the value of ServiceID in S_1 to get the available result set.

Algorithm analysis:

(1) As the service matching algorithm in UDDI is based on key words matching, matching operation in UDDI matching algorithm is similar with operation in grammar

query stage in the proposed approach in this paper. Namely, this algorithm includes service matching algorithm in UDDI. From our earlier work, performance of the algorithm is nicer than the one of algorithm in UDDI.

In semantic query stage, by means of ontology information, the related data of ontology are obtained. So we can get available result set more accurately.

(2) Because many factors need to be considered in the definition of Web service, such as function description, availability and so on. The interface description of Web service is sophisticated. In the description information the importance of each item is different. Just considering it, we classify description items of Web service and assign different weight. The definition of matching degree is as follow:

$$d(s_i, s_j) = \frac{\sum P_k \times V_k}{|s_i|} \times 100\% \quad . \tag{2}$$

Where, s_i is the Web service which will be matched. s_j is the Web service which will match with s_i. V_k is the weight of the k part of s_i and s_j. The principle of distributing weight is that the weight of basic information is high and the other is low. P_k is the number of items which are consistent in the s_i and s_j. The value is P_k shown as follow:

$$P_k = \sum_{n=1}^{m} E_n \quad . \tag{3}$$

$$E_n = \begin{cases} 1 & s_{i_n} = s_{j_n} \\ 0 & s_{i_n} \neq s_{j_n} \end{cases} . \tag{4}$$

From the matching process of this algorithm, we can see that matching object transforms from key-word matching query to ontology matching query. Because the user is not familiar with the Web service, the matching information is inaccurate. If key-word is used to query to obtain the available result set, matching degree is very low. However if ontology is used to query, because ontology consist of detailed information which are correlated with Web service, the matching degree is very high, even is 100% which indicates matching result set is available and satisfying.

(3) In the process of obtaining the maximum portfolio, the cycle number is n and in every cycle the operation number is C_n^i which is obtained from permutation and combination theory. So the time complexity is $O(n^2)$. In the service matching process, the time complexity is $O(n)$ .So the time complexity of the improved service matching algorithm is

$$O(n^2 + n) = O(n^2) \quad . \tag{5}$$

(4) In the algorithm by query rewriting theory, query changes from keywords to ontology. At the same time, query granularity becomes big. Before, we search through keywords. The keywords are indivisible and minimum granularity. This paper provides an approach by means of ontology and the query granularity largens from keywords to ontology. The ontology operation is more convenient than keywords.

IV. Experiment and Analysis

This algorithm is implemented in the Service Generating Platform (SGP) . SGP is on the base of the architecture of the MVC (Model-View-Control) and refer to the theory of reusing the whole process [4]. In SGP, the language of BPEL (Business Process Execution Language) is used. BPEL has been established as an OASIS (Organization for the Advancement of Structured Information Standards) [17] standard for modeling executable and abstract business processes by orchestrating Web services [16]. Modeling business processes in general and modeling BPEL processes in specific could be enormous time-consuming and error-prone. Reuse has been proved as a valuable approach to avoid reinventing the wheel, take the burden of repeated work off users and improve the quality and efficiency of process modeling.

The platform transforms the abstract process to BPEL-coded process. In the platform we can extract logical processes according to concrete businesses and compose the high-level business process. Therefore, it takes the burden of concrete knowledge of BPEL language off business developers. They only consider the business familiar to them. The aim of the platform decreases developer's burden and increases efficiency of developing SOA applications.

The MutliMedia system is an application of the platform. The data of MutliMedia is stored in the database. The Web services which are stored in the database are minimum granularity. On the base of Web services we can composite models which are divided into two types: Collaborating Model (CM) and Collaborating Model Example (CME) . The different is CM and CME is that the CM is not executable in real-world because only logic structure is preservated and the CME is available in real-world because executable Web services are bound. Their relation graph is as follow:

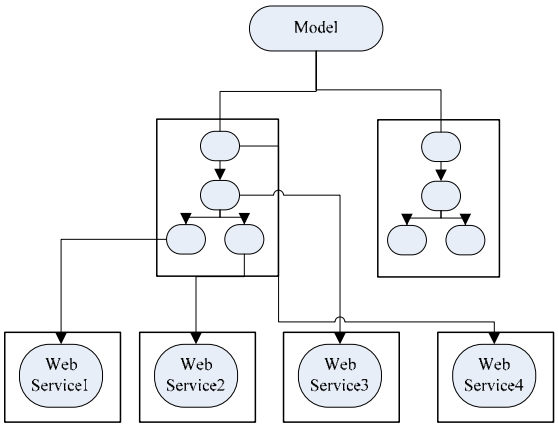


Figure9: CM and CME in Model

In which, the left is CME and the right is CM.

In SGP, we generate a new business process which may be different granularity by CM, CME and Web service. In this case, we can improve development efficiency and save time for developers.

The process of generating models is shown as figure 10. In a segment program with complete structure, four Web services are correlated with the Logic Object. Getting rid of the four Web services, the logic object is remainder. In Logic Object, the logic relation and cited position of Web services are preserved with some variables. This is, except the fours Web services, the remainder is preserved. The Logic Object is called as CM.

The matching process is the inverse process of the generating process. Its intent is that from the database some suitable candidates can be found out to cover for the Web services which are correlated with the Logic Object originally. The process is that from the user's input parameter set we choose the maximum portfolio and using it in the ontology library obtain the ontology which is coincident with user's requirement and get the available result set through querying database.

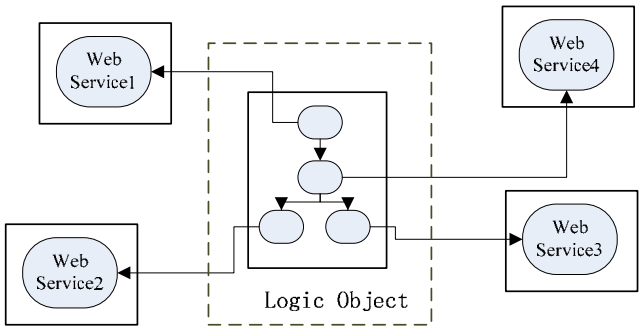


Figure10: Process of Generating CM in CEWS

The run-time environment is that CPU is double kernels, 1.8G; memory is 2GB; operation system is WindowsXP; database is MySQL. Data model of database is consistent with the above the structure of example. All the services of multimedia video conference are stored in database, in which every record represents a service. There are four query requirements. The query condition is as follows:

Table2: Query condition

NO.	Query condition
1	ServiceName LIKE '%conference%'
2	ServiceName LIKE '%conference%', Domain = 'videomedio', FunctionDescription LIKE '% conference %'
3	ServiceName LIKE '%conference%', Domain = 'multimedio', FunctionDescription LIKE '%start conference%'
4	ServiceName LIKE '%conferernce%', Domain= ' videomedia ', FunctionDescription LIKE '% conference %', Precondition = 'none'

The Maximum portfolio is obtained as follows:

Table3: Maximum portfolio

NO.	Maximum portfolio	Portfolio granularity
1	ServiceName LIKE '%conference%'	1
2	ServiceName LIKE '%conference%', FunctionDescription LIKE '% conference %'	2
3	ServiceName LIKE '%conference%', Domain = 'multimedio',	2
4	FunctionDescription LIKE '% conference %', Precondition= 'none'	2

Query result is:

Table4: Query Result

NO.	SNG	SNS	Query Accurate Rate
1	2	2	50%
2	10	2	50%
3	30	2	50%
4	13	4	100%

SNG indicates service number which is obtained in grammar query stage.

SNS indicates service number which is obtained in semantic query stage.

Note: the number of services which completely satisfy user requests is four. The Query Accurate Rate is obtained by the under formula shown as follow:

$$QAR = SNS / SN \times 100\% \quad (6)$$

In which, SN indicates satisfying service number which is four here.

From the data, we can see that maximum portfolio algorithm comes into force. When the number of query condition is one (query condition 1) , although maximum portfolio algorithm is used, there is not any effect. Even though there is misspelling, the algorithm is helpless. In the Table2 and Table3, query condition 2 indicates that in ontology library, condition ServiceName LIKE '%conference%' and FunctionDescription LIKE '% conference %' are maximum portfolio and the PG is 2.

The maximum portfolio algorithm plays an important role. The reason is that condition ServiceName LIKE '%conference%' and FunctionDescription LIKE '% conference %' are in one domain and condition Domain = 'videomedia' is in another domain. Condtion3 represents that in keyword array there is mistake in condition FunctionDescription LIKE '%start conference%'. So PG is 2. The condition 4 indicates there are mistakes of condition2 and condition3. So its PG is 2.

From the Table4, we can see that although sometimes query result set is not better, it is very helpful for the user. User may think about query condition again based on the result set and use accurate condition to obtain satisfying result set.

V. CONCLUSION

This paper provides an available approach to deal with Web service discovery problem. The aim of maximum portfolio is that suitable maximum portfolios are found out from ambiguous input keyword set. We can obtain the best query result set for one time input. So from the principle of the maximum portfolio we can see that it can decrease user's input time and find out suitable compositions from the ambiguous input keyword set. The approach is applied to the SGP and experiments show the algorithm is feasible and effective.

With the rapid development of the Web service technology, we will confront more and more new problems and numerous challenges of Web service technology. Further research about other approach to solve the issue is necessary. We promote the technological development in the process of solving problems about engineering techniques. But there are some problems to solve. For example in this algorithm, the searching method of maximum portfolio are complicated, Better approaches will be created for this question. In addition, the algorithm performance needs to be verified based on huge data. In the future we will collect more data to verify the algorithm performance.

ACKNOWLEDGEMENT

This research is supported by the National 973 Programs (Grant No. 2011CB302704) . National Natural Science Foundation of China (Grant No. 61003067) , Chinese Universities Scientific Fund (Grant No. 2009RC0507) and Key Projects for Science and Technology Development (No. 2011ZX03002-002-01) .

REFERENCES

[1] Tran Vuong Xuan, Puntheeranurak, Hidekazu TSUJI. A new service matching definition and algorithm with SAWSDL.2009 3rd IEEE International Conference on Digital Ecosystems and Technologies, DEST '09, pp.371-376

[2] Qianhui Althea Liang, Herman Lam. Web Service Matching by Ontology Instance Categorization. Proceedings of the 2008 IEEE International Conference on Services Computing, pp. 202 – 209

[3] Hongen Lu. Semantic Web services discovery and ranking. In proceedings of 2005 IEEE/WIC/ACM International Conference on Web Intelligence, pp.157 – 160

[4] Budan Wu, Zhi Jin, Bin Zhao. A Modeling Approach for Service-Oriented Application Based on Extensive Reuse. Proceedings of IEEE International Conference on Web Services (ICWS2008) , Beijing, 2008, pp.754-757

[5] PHui Guo, Anca Andreea Ivan, Rama Akkiraju, PRichard Goodwin. Learning ontologies to improve the quality of automatic web service matching. May. 2007 Proceedings of the 16th international conference on World Wide Web, pp.1241-1242

[6] Unai Aguilera, Joseba Abaitua, Josuka Díaz, David Buján, Diego López de Ipiña. A Semantic Matching Algorithm for Discovery in UDDI .Semantic Computing , 2007. ICSC 2007 .International Conference on17-19 Sept. 2007 pp.751 – 758

[7] Zhi Yang, Junliang Chen, Budan Wu.A New Ontology-based Service Matching Algorithm.2010 IEEE Congress on Services,2010

[8] Sanchez, Christian, Sheremetov, Leonid. A model for semantic service matching with leftover and missing information. Proceedings - 8th International Conference on Hybrid Intelligent Systems, HIS 2008, pp.198-203

[9] Hai Wang, Zengzhi Li. A Semantic Matchmaking Method of Web Services Based on SHOIN^+ (D) *. IEEE Asia-Pacific Conference on Services Computing, APSCC '06, 2006

[10] Pottinger R, Halevy A. MiniCon: a scalable algorithm for answering queries using views. The Int'l Journal on Very Large Data Bases, 2001, 10(2-3) ,pp.182-198

[11] Lin Yue, Song Bao-hua, Duan Hai-bo, Huang Feng-lei. Overview of Semantic Technology and App lications .ComputerApplicationResearch.2005/06

[12] Deng Zhihong, Tang Shiwei, Zhang Ming Yang Dongqing, Chen Jie. Overview of Ontology. Acta Scicentiarum Naturalum Universitis Pekinesis , 2002, 38 (9) ,pp.728- 730

[13] Studer R, Benjamins V R, Fensel D. Knowledge Engineering: Principles and Methods. Data and Knowledge Engineering ,1998 ,25 (122) :161～197

[14] Li Jie. Research On on OWL Ontology Storage Mode. China Science and Technology Information. Nov.2007

[15] J. Fan, B. Ren, L. Xiong. An Approach to Web Service Discovery Based on the Semantics. Proceedings of 2005 International Conference on Fuzzy Systems and Knowledge Discovery, 2005, pp. 1103 – 1106

[16] Zhilei Ma, Frank Leymann. BPEL Fragments for Modularized Reuse in Modeling BPEL Processes. Networking and Services, 2009. ICNS '09. Fifth International Conference on,pp: 63 – 68

[17] OAS IS ,http://www.oasis-open.org/who/

[18] D. Booth and C. K. Liu, eds. Web Services Description Language (WSDL) 2.0, 2006.http://w3.org/TR/2006/CR-wsdl20-primer20060327/

Zhi Yang was born in 1977 and received the M.S. degrees from the School of Computer Science and Technology, North China Electric Power University in 2008. He is currently working toward the Ph.D. degree at State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and

Telecommunications, China. His research interests include service description and discovery, ontology creating and reasoning. Zhi Yang is a student member of ACM and CCF.

Budan Wu received her Ph.D. (2009) degree on Computer Software and theory from Academy of Mathematics and Systems Science, Chinese Academy of Sciences. She joined Beijing University of Posts and Telecommunications in 2009, where she is now Assistant Professor in computer applications. Her research interests include SOA modeling, service-oriented software engineering, requirements engineering and business process modeling. She has published over 15 papers in various conferences and in journals such as ICWS, SCC, COMPSAC, and Chinese Journal of Computers. Budan Wu is a member of ACM and CCF. She is also a secretary for Technical Committee on Services Computing of CCF.

Jun-Liang Chen was born in 1933. He received his associate Ph.D. from Moscow Institute of Telecommunications Engineering in 1961. Now he is a professor at Beijing University of Posts and Telecommunications, China. He is an academician of both Chinese Academy of Sciences and Chinese Academy of Engineering. His current research interests include communication network and communication software, service computing and Web service.

Pingli Gu was born in 1980, and received the M.S. degrees from the School of Computer and Communication, China University Of Petroleum in 2009. She is currently working toward the Ph.D. degree at State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, China. Her research interests include service platform and cloud computing.