WEB SERVICE DISCOVERY BASED ON COMPUTATION OF SEMANTIC SIMILARITY DISTANCE AND QOS NORMALIZATION

Jeberson Retna Raj
Research Scholar
Sathyabama University
Chennai, India
jebersonin@yahoo.co.in

Dr.T.Sasipraba
Prof. & Dean,
Sathyabama University
Chennai, India
tsasipraba@yahoo.com

Abstract

Web service discovery is an important activity in services computing paradigm. Omnipresence of internet and proliferation of web services provides an opportunity to many service providers to create, locate and access the business services. Recommending pertinent web service to the client's requirement is a complex task as numerous functionally similar web services are readily available to meet the demand. Furthermore, the conventional method does not support the client's non functional requirements. To alleviate the difficulties, web service search engine has been introduced to assist the user to search the appropriate web services. The proposed approach consists of functional and nonfunctional evaluation for computing the similarity measurement, and normalizing the QoS values respectively. The WSDL parser extracts the Meta data contents from the WSDL file and it will be preprocessed. Soon after preprocessing, appropriate terms will be identified. To find the similarity between two web services, the semantic distance between terms will be calculated using Normalized Google Distance (NGD). The degree of similarity between web services can be calculated by using bipartite graph algorithm. The score value will be computed based on aggregation of functional and non functional measurement values. For a user request, a list of candidate services will be provided to the client based on degree of matching with the search query from higher to lower order. The proposed system has been tested with real world web services and the test result shows better precision and recall values, which outperforms existing approaches.

Keywords**:** web service discovery, QoS, similarity distance.

1. Introduction

With the principles of Service Oriented Architecture (SOA) and realization of web services attracts the organization and developers for make the use of the technology. In Services computing, service discovery is an important activity to search the pertinent service to the client's requirement. Due to the availability of large number functionally similar web services, QoS driven discovery becomes imperative to differentiate services. The service discovery can be classified into two categories namely syntactic discovery and semantic discovery. Syntactic discovery incorporated the techniques such as UDDI based search, text document based search, schema matching and software component matching. Semantic-based discovery is mainly based on ontology. Generally, UDDI is based on keyword search. Typically, text document search is widely used in search engines and web pages [4]. The functionality of schema matching is based on to capture clues about the semantics of schema and suggests matches based on them [5][6]. Operations are typically much more loosely related to each other than are tables in a schema and each web service in isolation has much less information than a schema. Finally, software component matching has defines the problem of examining signature (data type) and specification (behavior) matching [7]. It is mainly for software reuse. For Semantic discovery, Ontology based discovery is one of the approach for discovering web services [2]. The ontology based discovery approach suffers from performance problems due to the use of ontology reasoners. Furthermore, constructing ontology as a semantic backbone for a large number of distributed web service is really not easy. These are the major setback for ontology based discovery.

2. Overview of Web Service Discovery

As the user's interest of searching a web service is changed often, the need of service discovery becomes imperative. For example, a client requires the service with the response time <100ms and reliability and availability>95%, and cost is nil. Such a kind of service discovery is not possible with the traditional discovery mechanism. So the need of the hour is a system that assists the user to search a pertinent web service which incorporate with these capabilities. Therefore, the proposed QoS driven web service discovery which assists the user to search the pertinent web service for a client's requirements.

2.1. *Search engines and UDDI*

Discovering web services using UDDI have several issues such as seldom updating of registry and complexity in the discovery process. Furthermore, significant parts of information in these registries are out of date and the returned results are effectively inadequate. Generally, UDDI is based on keywords [2]. The consumer has to do view-select-request queries several times. These enormous hardships may hindrance the consumer to locate the desired web service [3]. Discovering web services using search engines is inherently impractical for web services. Search engine is designed mainly devoted for web sites not web services. Furthermore, search engines mainly based on keywords. Furthermore, Search engines do not understand the web service functionality outlined in the description file [8]. So we argue that a web service search engine can able to fill the gap of web service discovery difficulties.

2.2. *Keyword search*

The idea behind the keyword search is that the keyword involved in the search query which matches them with web service description. Since the keyword based searching unable to match the underlying semantics of web service, they may miss the relevant results and returns irrelevant one to the client. User unable describes the search request more precisely than keyword is another limitation of keyword search. Furthermore, keywords do not suffice for accurately specifying user's information needs. These hardships may hinder the consumer to search an appropriate web service.

2.3. *Related work*

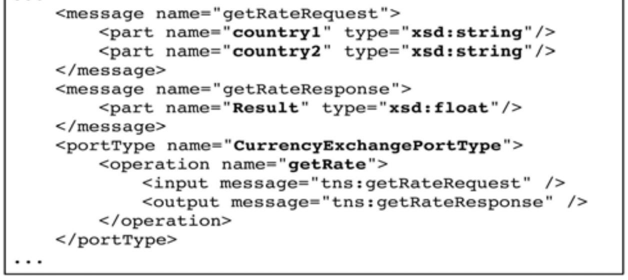
Recently, several works has been proposed in web service discovery. Meng Li et al [1] proposed web service search engine based on functional computation and QoS reputation. Xin Dong Et al., proposed "Similarity Search for Web Services approach for web services discovery" [10] for searching web services. They construct a search engine named as woogle used to search similar services. However, the work is not supported the QoS based search for a user request. Xuanzhe Liu [4] proposed a model for homogeneous web service discovery. The search model is based on input/output operation. In [9], a web service search engine WSExpress has been introduced. The functional value and the non functional values are aggregated to get the overall score value used to index the web services for a search query.

3. Web service discovery based on semantic distance

The proposed web service discovery mechanism consists of four phases. In the first phase, the web service Meta data is extracted and preprocessed. Second phase constitutes of computing the functional evaluation and third phase for non functional QoS computations for normalize the QoS values. Finally, the fourth phase for aggregating the functional and non functional values.

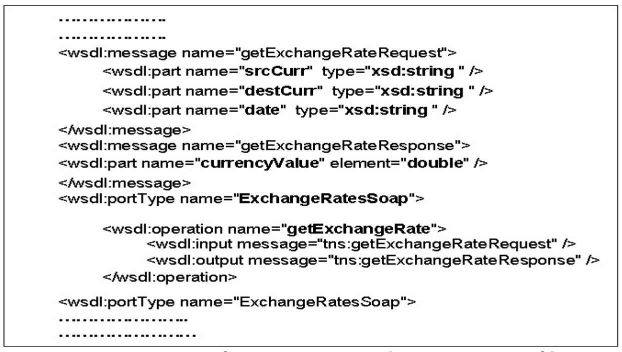
3.1. *Preprocessing*

The web services are collected by a crawler. The Meta data are extracted from the WSDL file. The element contents such as <service name>, <message>, <type>, <portType> are extracted and stored in a service pool.



```
...
    <message name="getRateRequest">
        <part name="country1" type="xsd:string"/>
        <part name="country2" type="xsd:string"/>
    </message>
    <message name="getRateResponse">
        <part name="Result" type="xsd:float"/>
    </message>
    <portType name="CurrencyExchangePortType">
        <operation name="getRate">
            <input message="tns:getRateRequest" />
            <output message="tns:getRateResponse" />
        </operation>
    </portType>
...
```

Figure 2. Excerpts from currency-1 web service WSDL file



```
……………….
……………….
<wsdl:message name="getExchangeRateRequest">
        <wsdl:part name="srcCurr"  type="xsd:string " />
        <wsdl:part name="destCurr"  type="xsd:string " />
        <wsdl:part name="date"  type="xsd:string " />
</wsdl:message>
<wsdl:message name="getExchangeRateResponse">
<wsdl:part name="currencyValue" element="double" />
</wsdl:message>
<wsdl:portType name="ExchangeRatesSoap">

        <wsdl:operation name="getExchangeRate">
                <wsdl:input message="tns:getExchangeRateRequest" />
                <wsdl:output message="tns:getExchangeRateResponse" />
        </wsdl:operation>

<wsdl:portType name="ExchangeRatesSoap">
…………………..
……………………
```

Figure 1. Excerpts from currency-2 web service WSDL file

Figure 1 and 2 shows the excerpts from two different currency web service WSDL. The Meta data contents such as message, portType, input and output parameters are extracted from a WSDL file.

3.2. *Semantic distance of terms*

To calculate the semantic distance of terms as a first step to calculate the semantic distance of terms. The conformity of similarity between two terms can be set as 1 for two terms are same. Semantic distance between the terms available in two different term sets are obtained using Google search API. Google search API returns the number of web results obtained for a term. Semantic distance between two terms are obtained using the following formula (1),

$$term\_distance_{1,2} = \frac{\log\left(\frac{N(k_1 \cap k_2) \times N}{N(k_1) \times N(k_2)}\right)}{\log N} \tag{1}$$

Where, N is the Number of total web pages and usually set to 10^{11}. k_1 is a term available in term set 1. k_2 is a term available in term set 2. $N(k_1 \cap k_2)$, $N(k_1)$, $N(k_2)$ are returned by the Google search API. $N(k_1 \cap k_2)$ denotes the co occurrence of terms k_1 and k_2 in Google search results. $N(k_1)$ denotes the occurrence of term k_1 in the web pages from Google. $N(k_2)$ denotes the occurrence of term k_2 in the web pages from Google. Semantic distance between two terms can also be obtained using Google normalized distance formula [2],

$$NGD(x,y) = \frac{\max\{\log f(x), \log f(y)\} - \log f(x,y)}{\log M - \min\{\log f(x), \log f(y)\}} \tag{2}$$

x is a term available in term set 1 and y is a term available in term set 2. In this formula, $NGD(x,y)$ represents Google semantic distance and M represents the total web pages that Google search engine obtains. $f(x)$ represents the return pages of Google search engine containing term x. $f(y)$ represents the return pages of Google search engine containing term y. $f(x,y)$ represents the return pages of Google search engine containing term x and y. To ensure the precision the semantic distance between two terms set to be 1 if it is same and need not to be calculated.

3.3. *Bipartite graph based similarity measurement*

Once the semantic distance between two different term sets are calculated using Google search API, a bipartite graph is constructed to show the calculated semantic distance between the terms. To find the semantic distance between two terms k_{11} and k_{21}, first k_{11} is sent to Google search API and number of web pages containing term k_{11} is returned as output. Google search API then returns the number of web pages containing the term k_{21}. Then both terms are given to Google search API to calculate the total number of web pages containing both the terms. Once the number of web pages containing term1, term2 and term1 & term2 are calculated, semantic distance between k_{11} and k_{21} can be obtained using the above formulas. Then semantic distance between k_{11} and k_{22} are calculated. The same process repeats for terms k_{12} & k_{21} and terms k_{12} & k_{22} are calculated. The maximum semantic distance between the terms are used to calculate the similarity measure.

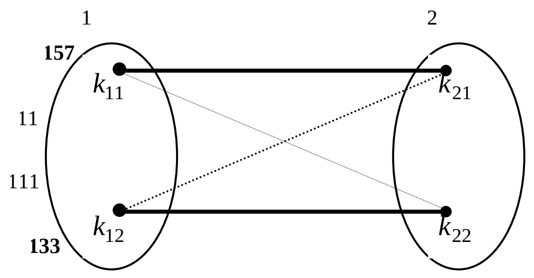


Figure 3. Two services as a balanced bipartite graph

The maximum weight matching of two services in fig is 0.157+0.133=0.29
The Similarity measure based on bipartite graph

$$similarity_{s1,s2} = \frac{\max\_value_{s1,s2}}{|M|} \tag{3}$$

Where $0 \le similarity_{s1,s2} \le 1$. When two services have same cardinality, i.e., s1=s2, the graph is balanced.

3.4. *Similarity measurement algorithm*

For graph $G = (S_1, S_{2,} E)$, let MV and ME represent the vertexes and the edges of the maximum matching M respectively. $\max\_value_{1,2}$ represents the maximal value of M. assume that $|S_1| \le |S_2|$, let U_MV denote unmatched term set and we have $U\_MV = S_2 - MV$. U_ME represents the edges of the left terms, $U\_ME = \{(<k_i, k_j>, w_{i,j}) \mid k_i \in S_1, k_j \in U\_MV\}$.
Input: Maximum semantic distance between the terms.
Output: Similarity measure between two web services.
Steps:

1. Check whether the similarity measure is to be calculated for equal terms or unequal terms.
2. If equal terms then apply the below formula,

$$Similarity_{1,2} = \frac{\max\_value_{1,2}}{|M|} \tag{4}$$

3. If unequal terms then apply the below formula,

$$Similarity_{1,2} = \frac{2\max\_value_{1,2} + \sum_{k_j \varepsilon U\_MV} \max\{w_{i,j}\}}{|S_1| + |S_2|} \tag{5}$$

When two services have different number of operations there have left of operations which are out of consideration. These left of operations to be considered for measuring similarity. The unmatched terms definitely have effect on the similarity services. For balanced graph, each term is included in the maximum matching and corresponding maximal value is able to represent how closely a term set is with the other term set. As to the unbalanced graph, the inner associations between matched terms and unmatched terms have obvious effects on the similarity [11].

3.5. *QoS optimization*

The QoS attribute such as response time, throughput, reliability and availability can be normalized by the following formula (7).

$$Utility_i(q_i) = \begin{cases} \dfrac{q_i - (q_i)_{\min}}{(q_i)_{\max} - (q_i)_{\min}} & if Q_i \in QoS^{Increment} \\ \dfrac{(q_i)_{\max} - q_i}{(q_i)_{\max} - (q_i)_{\min}} & if Q_i \in QoS^{Decrement} \\ 1 & if\ (q_i)_{\max} - (q_i)_{\min} \end{cases} \tag{7}$$

And the final score value can be computed as

$$Score(S_i) = Similarity(S_i) + Utility_i(S_i(q_i)) \tag{8}$$

4. Experimental results

The system has been tested with more than 3000 web services with its operations. The QoS values are measured by using WSRec [13]. When the no of operations are equal in web service s1 and s2, the similarity measurement between two operations has been measured by (4). If the number of operations are unequal, we can use equation (5) can be used for measuring the similarity between web services. We have categorized the web services for currency converter, weather and address validation, E-mail verification and credit card services for validating the proposed system. The precision and recall values can be computed from the following

Table I: performance measures of five clusters

Cluster	Our approach Operation based		Existing approach[11]	
	Precision%	Recall%	Precision%	Recall%
Currency exchange	94.0	100	90.0	94.7
weather	98.0	100	94.1	100
Address validation	92.0	98.0	83.3	93.7

Table 1 shows the performance measurement of the proposed approach. The test results shows that the proposed approach outperforms the existing approach in terms of high precision and recall values.

5. Conclusion

Web service discovery become an integral part of services computing paradigm. Recommending appropriate web service is a key challenge as list of candidate services are ready to fulfill the request. The operation based search with bipartite graph algorithm has been discussed in this paper to ease the discovery process as simple and efficient. The system has been designed such a way that the trained and novice user can discover the services based on keyword and operation based queries. The similarity distance that can be obtained from Google, which provides accurate weight to the respective terms. The test result shows its efficiency in terms of providing the optimal web service for a user request.

References

[1] Meng Li, Junfeng Zhao, Lijie Wang, Sibo Cai and Bing Xie, CoWS: An Internet-Enriched and Quality-Aware Web Services Search Engine", 2011 IEEE International Conference on Web Services, PP:419-427.

[2] M. Brian Blake, "Knowledge Discovery in Services", IEEE Internet Computing, March/April 2009. pp.88-91.

[3] M. Brian Blake, "Knowledge Discovery in Services (KDS):Aggregating Software Services to Provide Complementary Data", IEEE Transactions On Knowledge And Data Engineering

[4] Xuanzhe Liu, GangHuang, "Discovering Homogeneous Web Service Community in the User-Centric Web Environment", ", IEEE Transactions on Services Computing, Vol. 2, No. 2, April-June 2009

[5] E. Rahm and P.A. Bernstein, "A Survey on Approaches to Automatic Schema Matching," VLDB J., vol. 10, no. 4, pp. 334-350, 2001.

[6] H.H. Do and E. Rahm, "COMA: A System for Flexible Combination of Schema Matching Approaches," Proc. Very Large DataBase Conf. (VLDB '02), 2002.

[7] A.M. Zaremski and J.M. Wing, "Specification Matching of Software Components," ACM Trans. Software Eng. and Methodology, vol. 6, pp. 333-369, 1997.

[8] Yin Baocai et.al, "A Framework and QoS Based Web Services Discovery",IEEE International Conference on Software Engineering and Service Sciences (ICSESS), 2010 , PP:755-758.

[9] Yilei Zhang, Zibin Zheng, and Michael R. Lyu, "WSExpress: A QoS-Aware Search Engine for Web Services", 2010 IEEE International Conference on Web Services, PP:91-98

[10] Xin Dong Alon Halevy Jayant Madhavan Ema Nemes Jun Zhang, "Similarity search for web services" VLDB '04 Proceedings of the Thirtieth international conference on Very large data bases - Volume 30

[11] Fangfang Liu, Yuliang Shi, Jie Yu, Tianhong Wang, Jingzhe Wu "Measuring Similarity of Web Services Based on WSDL", 2010 IEEE International Conference on Web Services, PP:155-162

[12] Eyhab Al-Masri and Qusay H. Mahmoud, "Toward Quality-Driven Web Service Discovery", IT Pro May/June 2008, PP:24-28

[13] Z.B. Zheng, H. Ma, M.R. Lyu, and I. King, "WSRec: a collaborative filtering based Web service recommendation system," Proc. 7th International Conference on Web Services (ICWS 2009), 2009, pp.437-444.