IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Volume-6, Issue-1 E-ISSN: 2347-2693

QoS Based web services selection using a Bi-Level model

Lakshmi.H.N[1]

IT Dept., CVR College of Engg, Hyderabad, India

**Corresponding Author: hnlakshmi@gmail.com*

Available online at: www.ijcseonline.org

Received: 24/Dec/2017, Revised: 04/Jan/2018, Accepted: 19/Jan/2018, Published: 31/Jan/2018

Abstract — Service registries and web service engines are the main approaches for discovering web services. UDDI offers limited search functionalities which may return a huge number of irrelevant services. Often consumers may be unaware of precise keywords to retrieve the required services satisfactorily and may be looking for services capable of providing certain outputs. Another critical challenge in web service search and composition is the selection of web services, to be executed or to be composed, from the pool of matching services. Most of the current service selection proposals apply a weighted sum model (WSM) as an evaluation method for selection of services with the same functionality.

In this paper, we propose a new system called Extended Service Registry (ESR) for extended and efficient service search and selection using an object relational database. ESR uses a bi-level service selection approach that selects the most appropriate web services from the pool of matching services that considers both the functional and non-functional requirements for service selection. The functional requirements are provided by the user as a set of input parameters provided for and output parameters desired from the web service. The user also provides a set of desired QoS values and the order of their preference for selection. The experimental results demonstrate the efficiency of service search in our Extended Service Registry (ESR) and the variety of user queries supported.

Keywords - *Service Registries; Service Search; UDDI; I/O Parameters*

I. INTRODUCTION

Web Services are self-contained, self-describing, modular applications that can be published, located, and invoked across the Web. As growing number of services are being available, selecting the most relevant web service fulfilling the requirements of a user query is indeed challenging. Various approaches can be used for service search, such as, searching in UDDI, Web and Service portals.

The shortcomings of UDDI have motivated us to build an Extended Service Registry (ESR) system capable of offering powerful and efficient search operations. We propose the use of Object Relational Database as repository of web services. Information about the web services, extracted from their WSDLs, are stored in tables and relational algebraic operators are used for service search. This work is an extension of our previous work [5], where we proposed a RDBMS approach for service registries.

Often consumers may be unaware of exact service names that are fixed by service providers. Rather consumers being well aware of their requirements would like to search a service based on their commitments (inputs) and expectations (outputs). Based on this concept we have explored the feasibility of I/O based web service search in our proposed ESR system, to support varying requirements of the consumer. Utility of such an I/O based web service search for composition of web services is shown in our previous work[5].

Another critical challenges in the area of service search and composition is to define a service selection approach that selects the most appropriate web services from the pool of services discovered. Most of the current approaches [10, 12, 13, 14, 15], select services based on their QoS values from a set of web services that are functionally similar. These approaches usually apply a weighted sum model (WSM) as an evaluation method.

On the contrary, in order to model both the functional and non-functional requirements of users, we propose a bi-level service selection approach. The functional requirements are provided by the user as a set of input parameters provided for and output parameters desired from the web service. The user also provides a set of desired QoS values and the order of their preference for selection. In first level services matching the functional requirements are shortlisted, which are further filtered in second level based on given QoS requirements, thus providing a list of web services that best matches a given user query. Experiments were conducted using QWS dataset [22] to compare the second level (QoS based selection) of our approach with that of Chen's[14] approach. Various sets of queries were fed for both the approaches and the results were analyzed on the quality of services selected and the execution time taken by both approaches. From the results obtained we can infer that our approach performs better and returns quality web services as compared with Chen's approach.

The rest of the paper is organized as follows. In Section 2 we describe the object relational schema used for our extended service registry. This is an extension of our previous work [5]. Section 3 describes our bi-level model for service selection. Section 4 discusses our experimental results. In Section 5 we essay the related work. We conclude our work in Section 6.

II. Extended Service Registry

In this section we shall describe the object relational schema for storing web services in the registry. The relationship between the various tables in the proposed schema is depicted in the ER diagram in Fig 1. This is an extension of our previous work [5].

1. Each web service is given a unique ID (WSID) and stored with its name (WSName), port address (WSPortAdd) and operation name (OPName) in a web service table (WSTable).

WSTable:{*WSID*,*WSName*,*WSPortAdd*,*OPName*}

2. The parameters table (ParTable) contains all the parameters,that take part as either input or output in any of the web services in the registry, with their names in (PName). Each parameter is given a unique ID,(PID).

$$ParTable:\{PID, PName\}$$

3. The web service input output table(WSInOutTable) lists all web services in the registry with their respective input parameters (InPars) and output parameters (OutPars). InPars and OutPars are of collection type (ParList) and are stored as nested tables.

$$WSInOutTable:\{WSID, InPars, OutPars\}$$

4. The schema also includes a Query Table for storing the user query (QueryT) and a table to store services matching the user query (MWSTable).

$$QueryT:\{PNo, OutPars\}$$

where, OutPars are the PIDs of output parameters specified in user query.

MWSTable: {*WSIDs*}

II.I I/O Parameter based Service Search

A web service, ws, has typically two sets of parameters - set of inputs ws^I and set of outputs ws^O. When a user searches for a service providing a requested set of Outputs and/or accepting a requested set of Inputs, there may be many matching services in the registry. To categorize these matching services, we have defined various degrees of matching for services, based on Input/Output Parameter match, in our previous work [5] as follows:

1.**Exact Match** : ws_i is an Exact match of ws_j if the input/output parameters of ws_i exactly matches all the input/output parameters of ws_j

2.**Partial Match** : ws_i is a Partial match of ws_j if the input/output parameters of ws_i partially matches the input/output parameters of ws_j

3.**Super Match** : ws_i is a Super match of ws_j if the input/output parameters of ws_i is a superset of the input/output parameters of ws_j

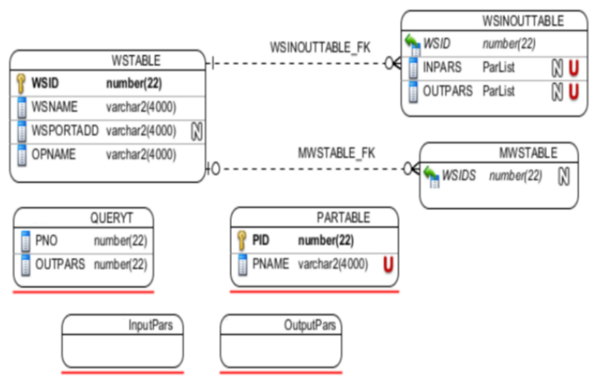


Figure 1: ER Diagram for Service Repository

II.II Process of Parameter based Service Search

To empower the service registries with additional search capabilities, we define algorithms for I/O parameter based service search. Algorithm 1 presents pseudo-code for Output Parameter based Service Search. Q^O represent output parameters specified in user query. Similar procedure is used for Input Parameter based Service search.

Input: Q^O,WSInOutTable:table
Output: MWSTable : table
foreach ParName in Q^O do
Select PID from ParTable where PName = ParName
INSERT PID into the QueryT table
foreach ws in WSInOutTable do
if $ws^O = Q^O$ then
INSERT ws as Exact Match in MWSTable
else if $ws^O \subset Q^O$ then
INSERT ws as Partial Match in MWSTable
else if $ws^O \supset Q^O$ then
INSERT ws as Super Match in MWSTable
Else Continue

Algorithm 1: Output Parameter based Service Search

III. I/O PARAMETER AND QOS BASED SERVICE SELECTION

There are two kinds of requirements that are crucial to web service selection and composition: functional and non-functional requirements. Functional requirements focus on functionality of the selected service, whereas the non-functional requirements are concerned with the quality of service (QoS).

We propose a bi-level model that considers both the functional and non-functional requirements for service selection. The functional requirements are provided by the user as a set of input parameters provided for and output parameters desired from the web service. The user also provides a set of desired QoS values and the order of their preference for selection. In the proposed bi-level model, the two objective functions: functional match and non-functional match, are arranged in two levels according to their order of importance. The first level shortlists a set of web services that optimizes the functional requirements from which services that best matches the QoS requirements are selected in the second level.

In the first level, we propose to compute input and output parameter deviation of a matched web service with respect to query input and output parameters using weighted sum model. This computation is done for all matching web services and is utilized for ranking them on functional match. Web services with lesser deviation values are shortlisted and considered in the second level, where further selection is done based on QoS values of these services. In the second level we consider 4 QoS attributes: response time, reliability, availability and price to rank web services. These attributes are modelled as constraints to be satisfied. For selection of services for composition we propose a ε-constraint method for ranking services. Figure 2 depicts the bi-level service selection approach discussed above. Each of these levels is explained in detail in the following subsections.

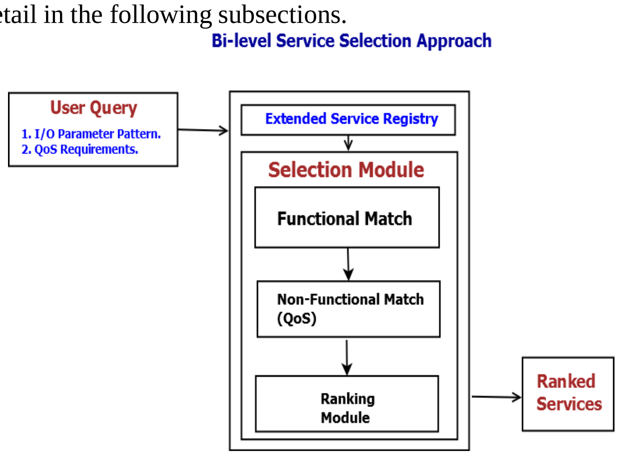


Figure 2: Bi-level service selection approach

III.I. Functional Match

The first objective of our bi-level model is to select web services that best match the given functional requirements of the user. The functional requirements are specified as a set of input parameters (the user is providing) and output parameters (that the user expects). We introduce a deviation measure for both input and output parameters that measures the deviation in the parameter set of the matched web service with respect to those provided by the user. The input and output parameter deviations are combined using weighted sum method and the values obtained is used to rank the available matching web services for further shortlisting. Higher the value of the deviation measure, lesser will be the rank of the matched web service.

Output parameter deviation measure

Here we describe the method to compute the output parameter deviation measure. There are 3 types of output parameter matches: exact, super and partial, as explained earlier. The following notations are used for defining output parameter deviation measure:

- Let WS_D denote web service with desired requirements.
- Let WS_M denote the matched web service.
- Let WS_D^O denote desired output parameter set.
- Let WS_M^O denote output parameter set of matched web service.

The number of non-matched output parameters (NMOP) is given by,

$$NMOP = \left|WS_D^O\right| - \left|WS_M^O\right| \quad (2)$$

Using eqn.2 we determine the type of output parameter match, as follows:

1. $NMOP$=0 for exact and super match.
2. $NMOP$>0 for partial match.

Measuring deviation from WS_D^O :

The output parameter set of the matched web service, WS_M^O, is compared with the desired set, WS_D^O, and by using eqn.2 the type of match is determined. Then the *output parameter deviation measure*, DM_P^O, for the matched web service is calculated, depending on the type of match, as follows :

1.Exact match : For a web service that matches exactly, DM_P^O=0, since there is no deviation and hence no ordering is required.

2.Super match : For a web service that is a super match of the desired web service, deviation is measured in terms of number of parameters that are redundant in the output parameter set of the matched service WS_P^O, and is given by -

$$DM_P^O = \left| WS_M^O - WS_D^O \right| \quad (3)$$

3.Partial match : For a web service that is a partial match of the desired web service, deviation is measured in terms of number of parameters not provided by output parameter set of the matched service, WS_M^O, and is given by -

$$DM_P^O = \left| WS_D^O \right| - \left| WS_M^O \right| \quad (4)$$

Input parameter deviation measure

Here we describe the method used to compute the input parameter deviation measure. There are 3 types of input parameter matches: exact, super and partial, as explained earlier. The following notations are used for defining input parameter deviation measure:

- Let Q^I denote query input parameter set provided by the user.
- Let WS_M^I denote input parameter set required by the matched web service.

The number of non-matched input parameters(NMIP) is given by -

$$NMIP = \left| WS_M^I \right| - \left| Q^I \right| \quad (5)$$

Using eqn.5 we determine the type of input parameter match, as follows:
1. $NMIP=0$ for exact and super match.
2. $NMIP>0$ for partial match.

Measuring deviation from Q^I :

The input parameter set of the matched web service, WS_M^I, is compared with the query input parameter set, Q^I, and by using eqn.5 the type of match is determined. Then the deviation measure, DM_P^I, for the matched web service is calculated, depending on the value of *NMIP*, as follows :

1. Full match: The value of $NMIP=0$ for a full match, which implies that the input parameters required by the matched web service is satisfied by Q^I. This is possible when $Q^I \supseteq WS_M^I$. Hence

   $DM_P^I = NMIP = 0$. (6)
2. Partial match: The value of $NMIP<0$ for a partial match , which implies that the input parameters required by the matched web service is not completely provided by Q^I. This case is encountered when $Q^I \subset WS_M^I$. Hence

   $DM_P^I = NMIP > 0$. (7)

Combining input/output deviation measures

We use *weighted sum method for combining output and input parameter deviation measures*, as follows :

• Let $x_1 = DM_P^O$, the output parameter deviation measure.

• Let $x_2 = DM_P^I$, the input parameter deviation measure.

Then, the total deviation of the matched web service, WS_M, is given by -

$$DM_P^{IO} = w_1 * x_1 + w_2 * x_2, \ \textit{,where,} \quad (8)$$

$$w_1 = \frac{1}{\left| WS_M^O \right|} \ \textit{for exact and super output match} \quad (9)$$

$$w_1 = \frac{1}{\left| WS_D^O \right|} \ \textit{for partial output match} \quad (10)$$

$$w_2 = \frac{1}{\left| WS_M^I \right|} \ \textit{for both full and partial match} \quad (11)$$

The value for DM_P^{IO} is computed for all the matching web services and is utilized for ranking them on their functional match. Since the value represents the amount of deviation in input and output parameter set of the matched web service with respect to queried input and output parameter set, it's obvious that lesser the value higher will be the matching and ence higher the rank. Web services with lesser deviation values are shortlisted and considered in the second level, where further selection is done based on the QoS values of these services.

III.II. QoS based service selection

The second objective of our bi-level model is to select web services that match best the given non-functional requirements of the user. After shortlisting the matched web services considering their functional match, they are now further ranked considering their QoS values. The user is expected to provide a set of desired QoS values which will be considered in this level. The objective of this level is to list a set of web services that best match with the desired QoS values. We consider 4 QoS attributes: response time, reliability, availability and price in our model for ranking web services.

We assume that the service provider provides the values of web services QoS attributes and also update their value often. These values are stored in QoSTable in our extended service registry. QoS attributes are either positive, for which higher values indicates better quality, E.g.: availability, reliability,etc or negative, for which lower values indicate better quality, E.g.: price, response time, etc. We present a QoS model taking into consideration these aspects, for service selection in the next subsection.

QoS model used

Each of the QoS attribute used for service selection is explained briefly below:

1. Response time (q_{RT}): Evaluating a service's response time to a request typically comprises of measurement of the execution time and waiting time of the web service. It is measured as the time between sending a service request and receiving a response.
2. Reliability (q_R): Reliability refers to the service provider's ability to successfully deliver requested service functionality. This ability can be quantified by the probability of success in a service execution, but it is usually evaluated through the service failure rate. This rate is calculated as the ratio of execution time and mean time between failures (MTBF).
3. Availability (q_A): Availability of a web service is the degree to which a service is operational and accessible when it is required for use. This value is defined by the proportion of the service's uptime to downtime, as represented by the mean time between failures (MTBF) and mean time to recovery (MTTR), respectively.
4. Price (q_C): It is the amount of money the requester has to pay for using the service.

Selection method

Most of the current proposals have applied a weighted sum model (WSM) as a uniform evaluation method for selection of services with the same functionality. This is represented as -

$$Score(WS) = \sum \left(q_i^{'} * w_i\right) \quad (13)$$

where q_i is a normalized QoS attribute value and w_i is the weight given to the QoS attribute. Such methods require users to express their preference over different (and sometimes conflicting) quality attributes as numeric weights. The objective function assigns a scalar value to each service based on the QoS attribute values and the weights given by the user. The service that has the highest value for the objective function will be selected and returned to the user.

Such optimization techniques are unable to model user preferences precisely For example, let us assume that the service selection is based on two quality attributes q_1 and q_2 with 0.6 and 0.4 as the associated weights for the objective function. Suppose there are two web services w_i and w_j with QoS values as $\{3,8\}$ and $\{5,5\}$ respectively. The weighted sum model gives a Score of 5 for both w_i and w_j. However, from the weights specified by the user, it is quite clear that q_1 needs to be given a greater preference than q_2 and hence w_2 would be the obvious choice.

The shortlisted web services from the first level can be categorized as those that have a deviation measure of 0 (an exact match) and those having a deviation measure > 0 (a partial or super match). When there are no exact matching services available, then service composition becomes inevitable and services need to be selected in each step of composition process. Hence, in order to model both the qualitative and quantitative preference of users, we propose a ε–constraint model [17]. The four QoS attributes that we consider, as explained in section **Error! Reference source not found.** are modeled as four objectives. Out of these we choose to minimize the cost and the remaining three objectives: response time, reliability and availability, are constrained to be greater/lesser than or equal to given user values. Formally,

$$\min q_C(WS_i) \quad (14)$$

$$q_{RT}(WS_i) \le \varepsilon_{RT} \quad (15)$$

$$q_R(WS_i) \ge \varepsilon_R \quad (16)$$

$$q_A(WS_i) \ge \varepsilon_A \quad (17)$$

where values for $\varepsilon_{RT}, \varepsilon_R$ and ε_A are the desired QoS values for response time, reliability and availability respectively and are provided by the user. The model selects a web service that has minimum cost with the desired (or better) response time, reliability and availability values.

The effectiveness of our QoS based service selection approach is shown by comparing the selected services of our system with the system proposed by Chen and Delnavaz [14]. Experiments were conducted using the QWS dataset[22] as explained in detail in section4.1. From the results obtained, we can infer that our approach outperforms Chen's method both in terms of execution time and the quality of services matched.

IV. EXPERIMENTAL RESULTS

Performance of QoS based service selection

The effectiveness of proposed QoS based service selection approach is shown by comparing the selected services of our system with the system proposed by Chen and Delnavaz[14]. We compare the two approaches with respect to the following:

1. Number of exact/super service matches obtained w.r.t. user-specified QoS ranges.
2. Number of partial service matches obtained w.r.t. user-specified QoS ranges.
3. Performance in terms of average running time of both the algorithms.

Experimental setup

We conducted experiments on QWS Data set[22], which includes WSDLs and QoS information of 2507 web services. We ran our experiments on a 1.3GHz Intel machine with 4 GB memory running Microsoft Windows 7. Our algorithms were implemented using Oracle 10g and JDK 1.6. Each query was run 5 times and the results obtained were averaged, to make the experimental results more sound and reliable.

Quality of service matches

In this section, we analyze the quality of services selected in our algorithm versus those selected in Chen's[14] approach. Chen Ding[14] propose a selection model capable of handling both exact and fuzzy requirements. The model returns two categories of matching web services: super-exact and partial matches, which are ranked based on relaxation orders and then preference orders of the QoS attributes provided by the user, using MIP as the base algorithm. Symbolic dynamic clustering algorithm(SCLUST) is used to cluster services into 3 groups: good, medium, and poor, based on the values of QoS attributes of the web services.

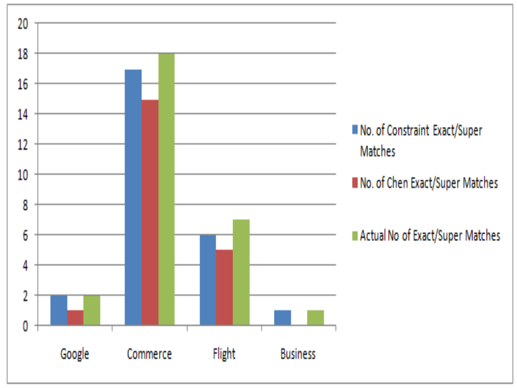
To analyze correctness of both the methods, we count the number of web services that have an exact/super match and partial match with respect to the QoS ranges specified by the user. We check the number of matching web services available in the registry for a given user query manually and compare this with the results of both the methods. Our experimental results shows the web services obtained for 4 different keywords - Google, Commerce, Business and Flight. The QoS requirements fed were as follows:

- Cost below 100.
- Reliability between 50 and 100%.
- Response time below 200ms.
- Availability between 50 and 100%.

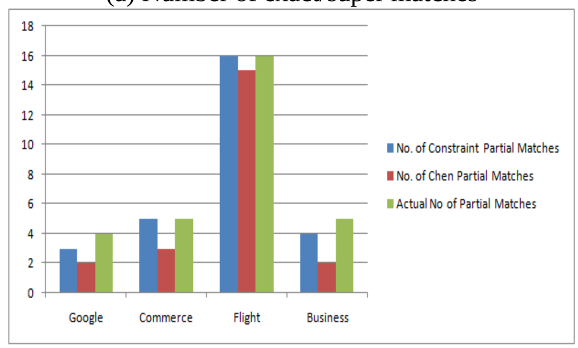
From the results obtained we can infer that our constraint based approach retrieves matching services better than Chen's method, in terms of number of matching services retrieved v/s the number of matching services available in the registry. The accuracy in our method is due to the ORDBMS schema used for storing services and their QoS details, whereas the clustering method adopted in Chen's approach might miss some matching services as seen in results.

Performance comparison

Next, we compare the average execution time taken by our proposed selection method with that of Chen's[14] method. The time taken for the set of queries explained in section.4.4.2 were noted for both the approaches and tabulated as shown in fig.8. From the results obtained, we can infer that our approach outperforms Chen's method both in terms of execution time and the quality of services matched.



(a) Number of exact/super matches



(b) Number of partial matches

Figure 7: Service matches of constraint method v/s Chen's method v/s available services in registry

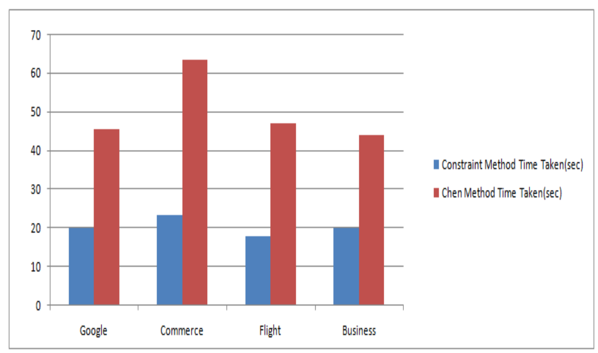


Figure 8: Performance comparison of constraint method v/s Chen's method

V. Related Work

Of the many service selection methods proposed in web service literature, we review here the approaches that model selection as an optimization problem or as a Multi Criteria Decision Making (MCDM) problem. Optimization can be performed at two levels: Local Optimization, for an individual Web Service selection and Global Optimization, for a given business process.

Chia Lin et.al[18] propose a QoS-based service selection (RQSS) algorithm to discover feasible Web Services based on functionalities and QoS criteria of user requirements.

The QoS constraints are classified as relaxable and non-relaxable constraints and the approach not only discovers Web Services fulfilling the functional requirements and non-functional QoS constraints, but also recommends solutions which could satisfy the non-relaxable QoS constraints by relaxing the relaxable QoS constraints.

Karim et.al[19] propose to use an enhanced PROMETHEE model for QoS-based Web Service selection. They take into account the QoS interdependency by using Analytical Network Process (ANP) to calculate the priority associated with each QoS criterion. In their original PROMETHEE model they do not consider user's QoS requirement due to which the model may end up in listing Services that optimizes the overall QoS criteria but fail to satisfy the user requirements. Hence they enhance their approach to rank the Web Services listed in the search to assess how well a Service satisfies the user requirement.

Huang[20] applies multiple criteria decision making (MCDM) with a weighted sum model (WSM) to help Service requesters evaluate Services numerically. QoS-based optimization of Service composition is then transformed into an Integer programming problem by deriving the objective functions of constituent workflow patterns. User needs to provide a workflow of the Service composition and the approach searches for Services that best matches the given workflow and the QoS constraints.

Ronald et.al[15] propose a simple but effective selection approach for finding the most suitable Web Services fitting user's requirements. The user needs to identify the QoS criteria of interest, provide ranking of those criteria, from which constraint satisfaction functions are constructed. They use Lexicographic method for multi criteria decision making: to order the QoS criteria according to the preference provided by the user, this ordering ensures that some QoS criteria must be satisfied before considering the others.

Mohammad et.al[21] propose a hybrid solution that combines global optimization with local selection techniques, visualizing the problem as an instance of multi-dimensional multiple choice knapsack problem(MMKP). The approach selects Web Services for a given composition request from a collection of candidate Services satisfying the specified QoS constraints. They use MIP (Mixed Integer Programming) to find the optimal decomposition of global QOS constraints into local constraints and then distributed local selection is applied to find the best Web Services satisfying these local Constraints.

Chen Ding[14] propose a selection model capable of handling both exact and fuzzy requirements. The model returns two categories of matching Web Services: super-exact and partial matches, which are ranked based on relaxation orders and then preference orders of the QoS attributes provided by the user, using MIP as the base algorithm. Symbolic dynamic clustering Algorithm (SCLUST) is used to cluster services into 3 groups: good, medium, and poor, based on the values of QoS attributes of the Web Services.

VI. Conclusion

One of the critical challenges in web service search and composition is the selection of web services, to be executed or to be composed, from the pool of matching services. Here, we propose a service selection approach, as explained in section3 that selects the most appropriate web services from the pool of matching services based on a bi-level model that considers both the functional and non-functional requirements for service selection. Experiments were conducted and the results of our approach were compared with that of Chen's [14] approach. The experimental results show that our approach outperforms Chen's approach as discussed in section4.1.

REFERENCES

[1] Apache jUDDI - http://juddi.apache.org/index.html

[2] C. Zhou, L. Chia, B. Lee, QoS-Aware and Federated Enhancement for UDDI, Int. J. of Web Services Research (IJWSR), 2004; No. 1,58-85.

[3] H. Mili, R. B. Tamrout, A. Obaid, JRegistry: An Extensible UDDI Registry, Reports of NOTERE, 2005, 115-128.

[4] J. C. Goodwin, D. J. Russomanno, J. Qualls, Survey of Semantic Extensions to UDDI: Implications for Sensor Services, SWWS, 2007; 16- 22.

[5] Lakshmi, H.N.and Mohanty H., RDBMS for service repository and composition,2012 Fourth International Conference on Advanced Computing (ICoAC),13-15 Dec. 2012.

[6] M. B. Juric, A. Sasa, B. Brumen, I. Rozman, WSDL and UDDI Extensions for Version Support in Web Services, J. of Systems and Software, No. 82 2009, 1326-1343.

[7] S. Ran, A Model for Web Services Discovery With QoS, ACM Sigecom exchanges 4.1 2003; 1-10.

[8]UDDI Specifications - http://uddi.org/pubs/uddi_v3.html

[9] The Web Service Challenge (WS-Challenge), http://www.ws-challenge.org/.

[10] Cai Dunbo,A Xu Sheng, Lexical Multicriteria-Based Quality Evaluation Model for Web Service Composition, Knowledge Engineering and Management, Advances in Intelligent Systems and Computing, Springer Berlin Heidelberg, 2014-01-01, 253-258.

[11] Lakshmi, H.N., Mohanty Hrushikesha, Extended Service Registry to Support I/O Parameter-Based Service Search , Proceedings of Intelligent Computing, Communication and Devices 2014,Advances in Intelligent Systems and Computing Series,2015.

[12] El Hadad, J.; Manouvrier, M.; Rukoz, M., "TQoS: Transactional and QoS-Aware Selection Algorithm for Automatic Web Service Composition," Services Computing, IEEE Transactions on , vol.3, no.1, pp.73,85, Jan.-March 2010.

[13] Jun Li, Xiao-Lin Zheng, Song-Tao Chen, William-Wei Song, De-ren Chen, An efficient and reliable approach for quality-of-service-aware service composition, Information Sciences, Volume 269, 10 June 2014, Pages 238-254.

[14] Mobedpour Delnavaz, Ding Chen ,User-centered design of a QoS-based web service selection system,Journal of Service Oriented Computing and pplications, Springer- Verlag, 2013.

[15] Ronald R. Yager, Giray Gumrah, Marek Z. Reformat, Using a web Personal Evaluation Tool PET for lexicographic multi-criteria service selection, Knowledge-Based Systems, Volume 24, Issue 7, October 2011, Pages 929-942.

[16] Marler R.T., Arora, Survey of multi-objective optimization methods for engineering , Journal of Structural and Multidisciplinary Optimization , Springer-Verlag,2004-04-01,369-395.

[17] R.T. Marler and J.S. Arora. Survey of multi-objective optimization methods for engineering. Structural and Multidisciplinary Optimization, 26(6):369-395, 2004.
[18] Chia-Feng Lin, Ruey-Kai Sheu, Yue-Shan Chang, and Shyan-Ming Yuan. A relax-able service selection algorithm for qos-based web service composition. Information and Software Technology, 53:1370-1381, 2011.
[19] R. Karim, Chen Ding, and Chi-Hung Chi. An enhanced promethee model for qos- based web service selection. In Services Computing (SCC), 2011 IEEE Inter-national Conference on, pages 536-543, July 2011.
[20] Angus F.M. Huang, Ci-Wei Lan, and Stephen J.H. Yang. An optimal qos-based web service selection scheme. Information Sciences, 179:3309-3322, 2009.
[21] Mohammad Alrifai, Thomas Risse, and Wolfgang Nejdl. A hybrid approach for ecient web service composition with end-to-end qos constraints. ACM Trans. Web, 6:7:1-7:31,2012.
[22]QWS-WSDL Dataset -http://www.uoguelph.ca/qmahmoud/qws/.

Authors Profile

Dr.Lakshmi.H.N is currently working as a professor in Dept of IT, CVR College of Engg, Hyderabad. She has completed her PhD in Computer Science from University of Hyderabad. She has published 10 papers in various international conferences and journals which are available online. She completed her M.S in Software Systems from BITS Pilani and B.Tech from BMS College of Engg, Bangalore University. She has over 17 years of teaching experience. Her main research work focuses on web service search and composition and has interests in the area of Big Data Analytics, Algorithms and programming.

