Research Paper Engineering

Analyzing And Ranking Web Services Based On Multi-Criteria Matching

* Bhanu Prakash Goud ** S. K. Prashanth

* M.Tech (S.E), VCE, Hyderabad.

** Associate Professor in CSE Dept, VCE, Hyderabad.

ABSTRACT

Searching and retrieval of web services become an important issue in present.web services are software entities that have a well defined interface and perform a specific task. Typical examples include services returning information to the user, such as news or weather forecast services. A web service is formally described in a standardized language (WSDL). The service description may include the parameters associated with web services like input , output and quality of service. As web services and service providers proliferate, there will be a large number of candidate, and likely competing, services for fulfilling a desired task. Hence, effective service discovery mechanisms are required for identifying and retrieving the most appropriate services. The main contributions of our paper are summarized as follows; we propose and implement a method for determining dominance relationships among service advertisements that simultaneously takes into consideration multiple PDM criteria. We introduce a method for prioritization and clustering web services based on similarity measures using efficient algorithms.

Keywords: Web Service ,Symmetric metrics , dominance score ,TKDD.

1. INTRODUCTION

Now days, the world of web has been witnessing the transformation from the web of data to the web of services. The user desired functionality is provided as a service which can be accessed over the web using the web services. Web services are becoming prominent in providing the services over the web. So application of the improved searching techniques and matchmaking approaches can be done in order to increase the efficiency of the web services search.

2. PROBLEM DEFINITION

The problem of ranking web services entails computing the scores of services and returning the top highest ranking ones. All the scores defined satisfy the requirements R1 and R2, and, therefore, qualify as possible ranking scores.

Given that the similarity measures provide only an indication of the actual relevance of the considered service to the given request, interesting services may be missed using the score.

Based on the above discussion, ranking services based on the dominated and dominating scores is chosen. A straightforward algorithm is the following. For each instance u of a service object U, iterate over the instances of all other objects and increase a counter associated with U, if u dominates (resp., is dominated by) the instance examined.

Then, to produce the top-k results, simply sort them according to the score in the counter. However, the applicability of this approach is limited by its large computation cost, as it needs to compute the score for all services, even those which are not in the top-k. Observe that independently of k, the algorithm exhaustively performs all possible dominance checks among instances.

The present problem of multi-criteria Web services matching, the terminology and notation are introduced, and the notion for top dominant Web services is formalized. To abstract way from a particular Web service representation, a Web service operation as a function that receives a number of inputs and returns a number of outputs is modelled. Hence, in the following, the description of a Web service operation corresponds to a vector S containing its I/O parameters. A request R is viewed as the description of a desired service operation, and is therefore represented in the same way.

2.1 PROBLEM STATEMENT

Given a Web service request, the search engine matches registered services against the desired description. For this purpose, it uses a similarity measure f_m to assess the similarity between the parameters in these descriptions. If more than one offered parameters match a requested parameter, the closest match is considered. Thus, the result of matching a pair (R,S) is specified by a vector $U_{R,S}$ such that

$$\forall i \in [0,|R|]\ U_{R,S}[i] = \max_{j=0}^{|S|} f_m(R[i],S[j])$$

Then, for each different similarity measure f_mi , a match vector $U_{R,S}^{mi}$ is produced. Hereafter, we refer to each such individual vector as match instance, denoted by lowercase letters (e.g., u,v), whereas to the set of such vectors for a specific pair (R,S) as match object , denoted by uppercase letters (e.g., U,V).

3. SIMILARITY MEASURES.

3.1 COSINE SIMILARITY

Cosine similarity is a measure of similarity between two vectors by measuring the cosine of the angle between them. The result of the Cosine function is equal to 1 when the angle is 0, and it is less than1 when the angle is of any other value. Calculating the cosine of the angle between two vectors thus determines whether two vectors are pointing in roughly the same direction. This is often used to compare documents in text mining. In addition, it is used to measure cohesion within clusters in the field of Data mining.

Cosine of two vectors can be easily derived by using the Euclidean Dot product formula:

$$a.b = \|a\|\ \|b\|\ cos\theta$$

3.2 JACCARD SIMILARITY
Jaccard Similarity is the extension of the Cosine similarity with co-efficient value. Jaccard Similarity is a simple and very intuitive measure of document to document similarity. It is defined as follows:

$$similarity\ (A,B) = n(A \cap B)/n(A \cup B)$$
$$similarity\ (A,B) = \frac{n(A \cap B)}{n(A)+n(B)-n(A \cap B)}$$

3.2 SIMPLE SIMALARITY
It compares every pair of letters from the adjacent words of a string to the other string and increments the score if there is a match. It is given by:

$$similarity\ (A,B) = \frac{common\ pairs\ matched(A,B)}{n(A)+n(B)}$$

4. DOMINANCE SCORES
The Dominance Scores are used in order to consider the multi-criteria problem and rank the services based on these scores. The three types of dominance scores, namely dominated score, dominating score, and dominant score are used to handle the aforementioned problem caused due to multi-criteria.

4.1 DOMINATED SCORE
Dominated Score: Given an instance u, we define the dominated score of u, denoted by dds, as:

$$u.dds = \sum_{V \neq U} \frac{|\{v \in V | v \succ u\}|}{|V|}$$

Hence , u.dds considers the instances that dominate u. The dominated score of an object U is defined as the (possibly weighted) average of the dominated scores of its instances:

$$U.dds = \sum_{u \in U} \frac{u.dds}{|U|}$$

The dominated score of an object indicates the average number of objects that dominate it. Hence a lower dominated score indicates a better match.

4.2 DOMINATING SCORE
Dominating Score: Given an instance u, we define the dominating score of u, denoted by dgs, as:

$$u.dgs = \sum_{V \neq U} \frac{|\{v \in V | u \succ v\}|}{|V|}$$

Thus, u.dgs considers the instances that u dominates. The dominating score of an object U is defined as the (possibly weighted) average of the dominating scores of its instances:

$$U.dgs = \sum_{u \in U} \frac{u.dgs}{|U|}$$

The dominating score of an object indicates the average number of objects that it dominates. Hence, a higher dominating score indicates a better match

4.3 DOMINANCE SCORE
Dominance Score: Given an instance u, we define the dominance score of u, denoted by ds, as:u.ds=u.dgs-□. u.dds

The dominance score of an object U is defined as the (possibly weighted) average of the dominance scores of its instances:

$$U.ds = \sum_{u \in U} \frac{u.ds}{|U|}$$

5.RANKING CRITERIA.
In the following, the algorithms that address the aforementioned deficiencies by establishing dominated/dominating scores are depicted.

The different algorithms used for ranking the services are.

5.1 RANKING BY DOMINATED SCORE
The first algorithm, hereafter referred to as RDD, computes the top web services according to the dominated score criterion, dds. The goal is to find, for each object, other objects dominating it.

Algorithm RDD:
Input: A set of objects U, each comprising M instances which corresponds to similarity measures.

Output: A ranked list consisting top objects with respect to dds in a sorted set R.

begin

for u $\in$ U do

for measure $u_{mi} \in U_{lm}$ similarity measures
Find DDs scores as follows

for each instance u

$$u.dds = \sum_{V \neq U} \frac{|\{v \in V | v \succ u\}|}{|V|}$$

for each object U

$$U.dds = \sum_{u \in U} \frac{u.dds}{|U|}$$

For each pair (service, measure) of measure instances,

Finding Average DDS

$$Ui.avgdds = \sum_{i \in Uim} \frac{Ui.dds}{|I_m|}$$

For each object U_i

Add the U_i.avgdds to set *R*.

Sort the set R in increasing order of Ui.avgdds .

end.

5.2 RANKING BY DOMINATING SCORE.
This algorithm, hereafter referred to as RDG, computes the top web services according to the dominating score criterion, dgs. The goal is to find, for each object, the other objects that this object dominates.

Algorithm RDG:
Input: A set of objects U, each comprising M instances which corresponds to similarity measures.

Output: A ranked list consisting top objects with respect to dgs in a sorted set R.

begin

for *u* $\in$ *U* do

for measure $u_{mi} \in U_{lm}$ similarity measures

Find DGs scores as follows

for each instance u

$$u.dgs = \sum_{V \neq U} \frac{|\{v \in V | u \succ v\}|}{|V|}$$

for each object U

$$U.dgs = \sum_{u \in U} \frac{u.dgs}{|U|}$$

For each pair (service, measure) of measure instances,

Finding Average DGS

$$Ui.avgdgs = \sum_{i \in Uim} \frac{Ui.dgs}{|I_m|}$$

For each object U_i

Add the U_i .avgdgs to set *R*.

Sort the set *R* in decreasing order of U_i.avgdgs .

end.

5.3 RANKING BY DOMINANCE SCORE.

This algorithm, hereafter referred to as RDS, computes the top web services according to the dominance score ds criterion i.e., calculated using the dgs and dds. The goal is to find the converged result obtained by using both dgs and dds. It is the combined version of the RDG and RDD algorithms.

Algorithm RDS:

Input: A set of objects U, each comprising M instances which corresponds to similarity measures.

Output: A ranked list consisting top objects with respect to ds in a sorted set R.

begin

for $u \in U$ do

for measure $u_{mi} \in U_{lm}$ similarity measures

Find DGs scores as follows

for each instance u

u.ds=u.dgs-. u.dds

for each object U

$$U.ds = \sum_{u \in D} \frac{u.ds}{|U|}$$

For each pair (service, measure) of measure instances,

Finding Average DS

$$Ui.avgds = \sum_{Ui \in Ulm} \frac{Ui.ds}{|U_{lm}|}$$

For each object U_i

Add the U_i.avgds to set *R*.

Sort the set R in decreasing order of U_i.avgds .

end.

6. CONCLUSION AND FUTURE WORK.

The notion of top dominant Web services, specifying three ranking criteria for matching Web service descriptions with service requests using multiple similarity measures is introduced.

The ranking algorithms based the dominance score dds, dgs and ds namely RDD, RDG and RDS algorithms are introduced and implemented.

1. The notion of top dominant Web services, specifying three ranking criteria for matching Web service descriptions with service requests using multiple similarity measures is introduced.
2. Based on this specification, the efficient algorithms for selecting the top-k matches for a service request are presented.

This approach is demonstrated experimentally using the sample data test collection set. All the three ranking algorithms were demonstrated on this set and the results were obtained. These results were compared using the previous single criteria matching of the services using the similarity measures

There are many more advancements in the web services technologies in their implementation, placement and exposing areas.

Presently the publishing of the web services in the public UDDI registry is not used. And there is a need to have a generalized way of publishing, invoking and searching web services for better match making and searching of services. The searching scope must also be reduced in order to minimize the computation cost caused due to exhaustive performing of all possible dominance checks. Another direction of it would be combining the other approaches of general document searching techniques algorithms to solve them respectively.

REFERENCES

1. M. Barhamgi, D. Benslimane, and B. Medjahed. Aquery rewriting approach for web service composition. | 2. IEEE T. Services Computing, 3(3):206{222, 2010. | 3. K. Benouaret, D. Benslimane, and A. Hadjali. On theuse of fuzzy dominance for computing serviceskylinebased on qos. In ICWS, page to appear, 2011 | 4. L. Si and J. Callan, "CLEF 2005: Multilingual Retrieval by CombiningMultiple Multilingual Ranked Lists," in Proceedings of the 6th Workshopof the Cross-Language Evalution Forum, 2005, pp. 121–130. | 5. S. Cetintas and L. Si, "Exploration of the Tradeoff Between Effectivenessand Efficiency for Results Merging in Federated Search," in SIGIR,2007, pp. 707–708. | 6. D. Lillis, F. Toolan, R. W. Collier, and J. Dunnion, "ProbFuse: AProbabilistic Approach to Data Fusion," in SIGIR, 2006, pp. 139–146. | 7. S. Osinski, J. Stefanowski, and D. Weiss, "Lingo: Search ResultsClustering Algorithm Based on Singular Value Decomposition," inIntelligent Information Systems, 2004, pp. 359–368. | 8 S. Papadopoulos, D. Sacharidis, and K. Mouratidis, "Continuous Medoid | Queries over Moving Objects," in SSTD, 2007, pp. 38–56.