Re-ranking Web Service Search Results Under Diverse User Preferences

Dimitrios Skoutas
L3S Research Center
Hanover, Germany
skoutas@L3S.de

Mohammad Alrifai
L3S Research Center
Hanover, Germany
alrifai@L3S.de

Wolfgang Nejdl
L3S Research Center
Hanover, Germany
nejdl@L3S.de

ABSTRACT

Web service discovery aims at finding available services that match a given service description. This involves mainly the matchmaking of the functional parameters of the services, whereas non-functional attributes can also be considered and aggregated in the matching score of a candidate service as additional criteria for ranking the results. In this paper, we address the problem of re-ranking discovered services that include nominal attributes in their descriptions in order to satisfy users with diverse preferences. We present an approach to diversify the search results combining the degree of match on functional parameters with a method to achieve good coverage with respect to the values of nominal attributes. An evaluation on a publicly available dataset of Semantic Web services is also presented.

1. INTRODUCTION

Web service discovery aims at finding services whose description matches that of a desired service. The description of a service contains a functional and a non-functional part. The former provides information about what the service does and how it works. This is basically expressed in terms of the required inputs and generated outputs, as well as any pre-conditions that need to be satisfied in order for the service to be executed and any effects that result from its execution. Several methods exist for the matchmaking of functional service parameters. More traditional techniques include the application of string similarity measures on the parameter names, whereas, in the case of services in the Semantic Web, they mainly rely on logic-based match between concepts in an ontology that annotate service parameters; moreover, combinations thereof have also been proposed. The result is typically a score indicating the degree of match between the service request and the service advertisement, which is used to rank the discovered services.

The non-functional part of a service description may include information about the service provider and quality of service (QoS) parameters, such as price, response time, availability or reputation. These attributes can also be used during the service discovery and selection process as additional criteria to improve the ranking of the match results. A typical case is to use such information to resolve ties. For example, if two services have the same functional degree of match, then the one with the lowest cost or response time is preferred. Alternatively, an aggregate degree of match can be calculated, possibly using different weights on the various parameters. However, such solutions focus on numerical attributes, for which a total ordering exists, since these can be more easily handled and incorporated in the matchmaker. For example, it can be rather safely assumed that all users would prefer services that have cost and response time as low as possible or availability and reputation as high as possible.

However, some of these attributes contained in the service descriptions are nominal attributes, which can not be seamlessly incorporated in the matchmaker, since for them it is not possible to specify an ordering. On the contrary, different users, or even the same user in different contexts, may have different preferences regarding the values of these attributes. Typical examples include the provider of a service, the communication or security protocols a service may support, accepted file formats or different algorithms the service may employ to solve a specific problem. Matching such types of attributes is not straightforward, since the match depends on the user preferences for the values of these attributes. However, gathering explicit or implicit knowledge about the preferences of users on the Web is often very difficult or even impossible.

In this paper, we present a method for the discovery and selection of Web services focusing on attributes for which an ordering of the values is not defined, but instead it depends on the preferences of the user. The main idea underlying our approach is to increase the diversity in the search results in order, consequently, to increase the probability of satisfying users with different preferences. Diversifying search results has already been investigated in the context of document search on the Web [9, 6, 22]. Proposed solutions rely essentially on introducing more dissimilar documents in the result set, finding an appropriate balance between the relevance of the results and their dissimilarity. Our approach follows the same direction, but it proposes a different diversification objective that emphasizes on selecting representative results that provide good coverage of the whole list of matches. Moreover, Web service descriptions are complex objects; hence, simpler models, such as a bag-of-words model, which often work well for documents, are not appropriate for Web services search.

VLDB '10, September 13-17, 2010, Singapore

The main goal and advantage of this approach is to allow for personalized results in a flexible way and minimizing the burden to the user. One possibility for personalized search would be to request each user to create a profile with his/her preferences. However, this is not always desired; moreover, these preferences may change over time or may depend on a particular information need (e.g., a preference for a service provider). Alternatively, the system could ask the user to indicate his/her preferences at query time. However, this is cumbersome, and in addition, the user may not be aware in advance of the possible options. Instead, the described approach constitutes a two-step process to personalizing the results. First, starting from a potentially underspecified request, the system retrieves a diverse set of results aiming at covering the available options as much as possible. Then, once the user indicates a preferred result, a nearest neighbor query can be issued to retrieve more similar services. Hence, the user preferences are indicated implicitly, dynamically, and with minimum overhead.

In summary, our main contributions are listed below.

- We propose a method for re-ranking Web service search results to satisfy users with diverse preferences.
- We propose a diversification objective that favors representative services to achieve good coverage of the result set.
- We show how the diversification objective can be computed based on the different types of attributes in the service descriptions.
- We present an experimental evaluation of our approach on a collection of Semantic Web services.

The rest of the paper is structured as follows. Section 2 presents related work. Section 3 introduces our method for diversifying Web service search results. Section 4 presents our experimental evaluation. Section 5 concludes the paper.

2. RELATED WORK

Service discovery. Several approaches have been proposed for matching a Web service request with available service descriptions. These approaches can be categorized in two main families. The first -more traditional ones, based on WSDL and UDDI standards- follow IR-style search, performing a keyword-based search on the textual descriptions of the services or comparing the parameter names in the service descriptions using some standard string similarity measure [8]. The second family of approaches considers services in the Semantic Web, where service parameters are annotated using concepts from domain ontologies. Then, the matchmaking between service parameters is transformed to logic-based match between the corresponding ontology concepts, i.e., inferring, by means of an ontology reasoner, whether the denoted ontology classes are equivalent, superclass or subclass of each other, or disjoint [17, 16, 7, 20, 4]. To combine the advantages of both categories and to address their limitations, hybrid approaches have also been proposed and implemented. These compute aggregate scores based on both logic-based matching and string-based parameter similarities [14, 12]. Finally, ranking of services based on dominance relationships computed over multiple matching criteria has been proposed in [19].

In addition, there exist some approaches that involve the user in the service discovery process. The approach presented in [3] employs ontologies and user profiles, and uses techniques such as query expansion or relaxation to better satisfy user requests. The work in [23] focuses on QoS-based Web service discovery, proposing a reputation-enhanced model. Reputation scores are assigned to the services by a reputation manager based on user feedback regarding their performance. Then, a discovery agent uses the reputation scores for service matching, ranking and selection. User preferences, expressed in the form of soft constraints, are applied for Web service selection in [13], focusing on the optimization of preference queries. Utility functions are used in [15] to model service configurations and associated user preferences for optimal service selection. In [8], different types of similarity for service parameters are combined using a linear function, with weights being assigned manually. It is mentioned that the weights can be learned from user feedback, but this is not addressed.

Our approach defers from the above works mainly in two aspects. First, the aforementioned approaches deal with numerical QoS parameters, for which a global and total ordering exists; hence, they can more easily be incorporated to and combined with other criteria for service selection and ranking. Instead, we focus in this paper on QoS parameters with non-numeric values (e.g., text or categorical data), for which no ordering can be defined. Second, the approaches above assume that a user profile, preferences or feedback is available for the service requestor. However, collecting such implicit or explicit information for Web users at large is not practical, if not infeasible. Nevertheless, the fact that different users have different preferences and needs still needs to be taken into account. Our approach addresses this issue by re-ranking the match results to increase their diversity, and, hence, to reduce the risk that for a given user there is not *any* result that meets his/her preferences.

Diversification of Web search results. Recently, the problem of diversifying Web search results has received a lot of attentions as a means to satisfy users on the Web with diverse preferences and information needs. The problem is typically formulated as optimizing an objective function that specifies a trade-off between the relevance of the returned documents with respect to the given query and the dissimilarity among these documents [9]. Essentially, this is similar to the idea of the Maximal Marginal Relevance criterion, which has been proposed in [6] for re-ranking the query results and it is also applied often for document summarization. It combines query relevance and novelty of information, by measuring the dissimilarity of a search result with respect to the ones before it in the ranked result list. In a similar direction, [22] addresses the problem of re-ranking search results adopting the idea of Modern Portfolio Theory from the field of finance. It considers documents not individually but in combination with other documents, formulating the problem as a portfolio selection problem. Results are then selected to maximize the relevance, while minimizing the variance, where the notion of variance corresponds, inversely, to that of diversity. The problem of diversifying query results from a database has been considered in [21]. However, that approach assumes that there is a known ordering of the user preferences with respect to the involved attributes. For example, for a query for "cars", the returned tuples should be first diversified with respect

to the car model, then to price and then to color. Moreover, we propose and apply a diversification objective that targets the selection of more representative results in order to achieve better coverage.

The main difference of our approach is that it deals with Web service search results rather than documents. Service descriptions are complex objects, comprising parameters of different types (functional and non-functional, numeric and non-numeric) that need to be handled accordingly.

3. DIVERSIFYING DISCOVERED SERVICES

3.1 Diversification Objective

Assume a service request R and a repository containing a set of service advertisements $\mathcal{S}$. To discover services that match the request R, a matchmaker is invoked, which applies one or more matching criteria to calculate a *degree of match* (*dom*) between R and each available service $S \in \mathcal{S}$ (see Section 2 for more details). The core of this operation is to match the input and output parameters in the two descriptions, although pre-conditions and effects or other parameters can also be taken into account. Then, the found matches are sorted according to their degree of match and the ranked list (or the top-k matches) is returned to the user. However, this list may often contain services that are very similar to each other, hence being biased toward the needs and preferences of only a subset of users. For this reason, the goal is to re-rank the list of results in order to include services that are still relevant to the request but less similar to each other.

Let $sim : \mathcal{S} \times \mathcal{S} \rightarrow [0, 1]$ be a function that computes the similarity between two service descriptions. Notice that the two functions, *dom* and *sim*, serve very different purposes. Function *dom* checks, for example, whether the outputs of the advertised service "fulfill" the outputs of the request, whereas function *sim* checks, for example, whether two services have the same provider or whether they support the same protocols or file formats. Then, the goal is to re-rank the discovered services so that the top-k results have a degree of match to the query that is as high as possible, while at the same time being as dissimilar to each other as possible. However, these two objectives are often contradictory, since a new service that is introduced to make the result list more diverse may have lower degree of match than the one it replaces. Therefore, these two factors need to be balanced using an objective function.

A diversification objective, referred to as MaxMin, has been proposed in [9] for diversifying search results of keyword queries on the Web. Applying this objective would return a set of k services that maximize the minimum degree of match and the minimum dissimilarity in the set. Formally, it selects the set of top-k results that maximizes the function:

$$f(\mathcal{S}_k) = \lambda \cdot \min_{S \in \mathcal{S}_k} dom(R, S) + \min_{S_1, S_2 \in \mathcal{S}_k} dist(S_1, S_2) \quad (1)$$

where distance (i.e., dissimilarity) is measured by $dist(S_1, S_2) = 1 - sim(S_1, S_2)$ and $\lambda > 0$ is a parameter that specifies the relative emphasis between the two factors.

The drawback of this diversification objective is that it measures the degree of match and the dissimilarity only *within* each candidate top-k list of results, ignoring the remaining ones, i.e., the ones below rank k. This has the side effect that it is biased towards more extreme rather than more representative cases. In other words, it might give priority to outliers. Assume, for example, a list of matches containing two services that have a high degree of match to the request and are very dissimilar to each other with respect to other characteristics. Then, this objective would favor these services for the top 2 results. However, these are not necessarily representative of the rest of the matches.

To address this issue, we propose a different diversification objective that aims also at providing good *coverage* of the whole result list, selecting services that are good representatives of the whole result set (and, consequently, also diverse with respect to each other). The main idea is that, in order to achieve good coverage, for each identified match there should be at least one service in the top-k list that is sufficiently similar to it. Moreover, since this apparently can not be achieved for all the services in the result list, priority should be given to those services with higher degree of match to the request. That is, if a service has a very high degree of match to the query, it should be well represented in the top-k results, either by including itself or by including one that is very similar to it. Notice however that this does not mean necessarily a higher representation, in terms of number of services, in the output set for the more relevant services. Indeed, if the services with high degrees of match happen to be very similar to each other, then they can be adequately covered with just a few representatives, or even a single one. Hence, this method is robust with respect to the problem of diminishing returns, as pointed out in [1], which leads to decreased user satisfaction.

We now formalize this diversification objective. Given a subset $\mathcal{S}_k$ of the query results, and a service S, we define the *coverage error* of S with respect to $\mathcal{S}_k$ as the minimum distance between S and any of the services in $\mathcal{S}_k$, i.e.:

$$cerr(S, \mathcal{S}_k) = \min_{S' \in \mathcal{S}_k} dist(S, S') \quad (2)$$

Lower values mean that there exists a selected service that is highly similar to the considered one. Hence, the quality of $\mathcal{S}_k$ is characterized by its maximum error in representing the matched services for the given request. Moreover, the coverage error for a service should be weighted according to its degree of match. If a service has a high degree of match to the request but is poorly covered, i.e., it is neither included in the top-k results nor there exists another sufficiently similar service in the top-k list, this should incur a higher penalty. Hence, this objective selects the subset of query results that *minimizes* the following function:

$$f(\mathcal{S}_k) = \max_{S \in \mathcal{S}} \{dom(R, S)^\lambda \times cerr(S, \mathcal{S}_k)\} \quad (3)$$

where the parameter λ determines, as previously, the trade-off between the two factors, namely the degree of match and the diversity of the results.

We refer to this diversification objective as MaxCov. Notice that the score of the set $\mathcal{S}_k$ in Equation 3 depends on *all* the matched services for the query (max is computed over all the services in $\mathcal{S}$), while in Equation 1 it depends only on the top-k subset (min is computed over $\mathcal{S}_k$).

Example. We present a simple example to illustrate the difference between the MaxMin and the MaxCov objectives. Assume a service request R and a set of 6 relevant services S_1, S_2, ..., S_6, with distances from each other as

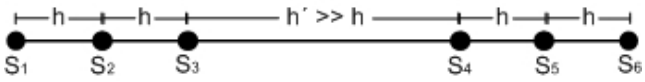

Figure 1: Illustrative example showing the difference between the MaxMin and MaxCov objectives

depicted in Figure 1. Assume also that we want to select as answer to this request a subset comprising $k = 2$ of these services. Disregarding the factor of the degree of match, i.e., assuming that all these services equally match the request, the MaxMin objective, which is driven by the pair-wise distances of the objects within the result set, selects as output the set $M_1 = \{S_1, S_6\}$. These are indeed the most dissimilar services. On the other hand, the MaxCov objective, which considers the distances of the selected objects to the whole result set, selects as output the subset $M_2 = \{S_2, S_5\}$. These can indeed be considered as better representatives.

3.2 Computing DoM and Similarity

In the following we discuss how to compute the degree of match *dom* between a service request R and a set of available service descriptions $\mathcal{S}$ and the similarity between two service descriptions S_1 and S_2.

As presented in Section 2, there exists several methods for matching a service request R with a service advertisement S. This operation is based on matching functional service parameters, typically inputs and outputs. Recently a lot of efforts have focused on the discovery of services in the Semantic Web and three main languages have been proposed to semantically mark up service descriptions, namely WSDL-S [2], OWL-S [5] and WSMO [11]. The main idea underlying all these approaches is to associate service parameters with classes in a domain ontology $\mathcal{O}$. This allows both the service provider and the user searching for a service to describe the intended meaning of the parameters in an unambiguous way, which facilitates the automation of the discovery process and increases the precision w.r.t. plain keyword search, where ambiguity, synonyms and homonyms need to be taken into account. In this setting, a reasoner is employed to infer the relationship between the corresponding ontology classes (i.e., equivalence, subsumption, disjointness) and based on that the type of match is determined accordingly (e.g., exact, plug-in, subsumes, subsumed-by or fail) [17].

For simplicity, we consider here only input and output parameters. A service advertisement S matches a service request R if (a) the outputs requested by R are matched by those offered by S and (b) the inputs required by S are provided in R. To quantify the degree of match, a method that considers the "proximity" of classes in the ontology can be applied [20, 18]. More specifically, the degree of match between two parameters can be measured by means of the common super classes of the corresponding classes C_1 and C_2 in the ontology $\mathcal{O}$, as follows:

$$dom(C_1, C_2) = \frac{|\{C \mid C \sqsubseteq C_1 \wedge C \sqsubseteq C_2\}|}{\texttt{max}(|\{C \mid C \sqsubseteq C_1\}|, |\{C \mid C \sqsubseteq C_2\}|)} \quad (4)$$

Then, the degree of match between the request and the advertisement is computed by aggregating the degrees of match of individual parameters. This can be extended to other parameters, apart from inputs and outputs.

As a metric for computing the similarity between two service descriptions S_1 and S_2 we use the Jaccard similarity, which is also commonly used in information retrieval for measuring the similarity between two documents.

The Jaccard similarity (also known as the Jaccard Coefficient) between two sample sets A and B is defined as the size of the intersection divided by the size of the union of the two sample sets, i.e.:

$$sim(A, B) = \frac{|A \cap B|}{|A \cup B|} \quad (5)$$

For computing the similarity between two services, we compare the non-functional part of the descriptions, and in particular nominal attributes, e.g. the set of supported protocols for transport, security, transactions etc.

Algorithm 1 Approximate algorithm for MaxCov

Input: : A service request R, the available service descriptions $\mathcal{S}$, an integer k
Output: : The re-ranked list $\mathcal{S}_k$ of top-k matches
1: $d = \arg\max_{S \in \mathcal{S}} dom(R, S)$
2: $S_k = \{d\}$
3: **for** $i = 1$ **to** $k - 1$ **do**
4: $\quad S = \arg\max_{S \in \mathcal{S}} \{dom(R, S)^\lambda \times cerr(S, \mathcal{S}_k)\}$
5: $\quad \mathcal{S}_k = \mathcal{S}_k \cup \{S\}$
6: **end for**
7: **return** $\mathcal{S}_k$

3.3 Diversification Algorithm

The MaxMin diversification objective presented in Section 3.1 corresponds to the vertex weighted version of the Minimum k-Center problem, which is known to be NP-hard [10] (for metric distances). Notice that the same also holds for the MaxMin objective specified in Equation 1 which was proposed in [9]. Consequently, one needs to resort to greedy approximation algorithms. For the MaxCov objective, a 2-approximation greedy algorithm can be derived from the Minimum k-Center problem. According to that, the re-ranking of services is done as specified in Algorithm 1. The algorithm works as follows. First, the service with the highest degree of match is selected. Then, the algorithm proceeds in $k-1$ iterations, selecting in each iteration the service with the maximum weighted coverage error with respect to those selected so far.

4. EXPERIMENTAL EVALUATION

In this section we describe our experimental evaluation of the proposed method for diversifying the results of Web service search.

For our experiments, we have used the OWLS-TC v2 collection[1]. This is a publicly available collection of Semantic Web services described in OWL-S and it is often used to evaluate and compare different matchmaking algorithms. It comprises 1007 services retrieved from public UDDI repositories. These services have then been described in OWL-S, using ontologies from 7 different domains to semantically annotate service input and output parameters. This collection provides also a set of 28 service requests and their corresponding relevance sets identified manually, in order to allow different matchmakers to compare the results based on the standard recall and precision measures. However,

[1]http://projects.semwebcentral.org/projects/owls-tc/

Table 1: Non-Functional Attributes

Attribute	Possible Values
Message Encoding	XML
	SOAP 1.1
	SOAP 1.2
	WS-Addressing
Security Protocol	WS-Security SOAPMessageSecurity 1.0
	WS-Security SOAPMessageSecurity 1.1
	WS-Security UsernameTokenProfile 1.0
	WS-Security UsernameTokenProfile 1.1
	WS-Security X.509 Certificate 1.0
	WS-Security X.509 Certificate 1.1
	WS-Security X.509 Kerperos 1.0
	WS-Security X.509 Kerperos 1.1
	WS-SecureConversationLanguage
	WS-TrustLanguage
	WS-ReliableMesseging 1.0
	WS-ReliableMesseging 1.1
Transport Binding Protocol	HTTP 1.1
	SOAP 1.1 HTTP Binding
	SOAP 1.2 HTTP Binding
Transaction Protocol	WS-Coordination 1.0
	WS-Coordination 1.1
	WS-Coordination 1.2
	WS-AtomicTransaction 1.0
	WS-AtomicTransaction 1.1
	WS-AtomicTransaction 1.2
	WS-BusinessActivity 1.0
	WS-BusinessActivity 1.1
	WS-BusinessActivity 1.2

this benchmark is not appropriate for our purpose, since it only indicates services that are relevant to the request w.r.t. input and output parameters without any consideration on the diversity of the results. Therefore, we have used the provided queries for our experiments but we do not conduct any evaluation in terms of recall and precision w.r.t. the provided relevant sets; instead, we want to measure the improvement in terms of the coverage error, as shown below. Moreover, the service descriptions included in this collection contain information only about input and output parameters, since these are the typical criteria taken into consideration by existing matchmakers, as described in Section 2. To overcome this limitation, we have extended the description of each Web service in the dataset with a vector of 4 non-functional attributes: `Message Encoding Schema`, `Security Protocol`, `Transport Binding Protocol` and `Transaction Protocol`. For each one of these attributes, we have identified a set of possible values, as shown in Table 1. Then, we have randomly assigned to each Web service in the collection a value for each of these attributes. For the implementation, we have used the OWL-S API[2] for parsing the OWL-S descriptions of the services in the collection.

In this evaluation, we wanted to measure how diverse are the results obtained by our *MaxCov* diversification method as described in Algorithm 1. As a measure for diversification we computed the objective value based on the coverage error (as defined in Equation 3) of the results set. Notice that lower values indicate lower coverage error and higher diversity in the results. For this purpose, we first computed the degree of match *dom* between each service in the collection and each query. The set of candidate matches for a certain query R consists of all the services that have a non-zero degree of match with R. In this experiment, we considered only queries that have at least 100 candidate matches, since considering coverage and diversity is less important when dealing with queries that have a relatively small result set. We then applied the following two methods for selecting the top-k results of each query:

[2]http://www.mindswap.org/2004/owl-s/api/

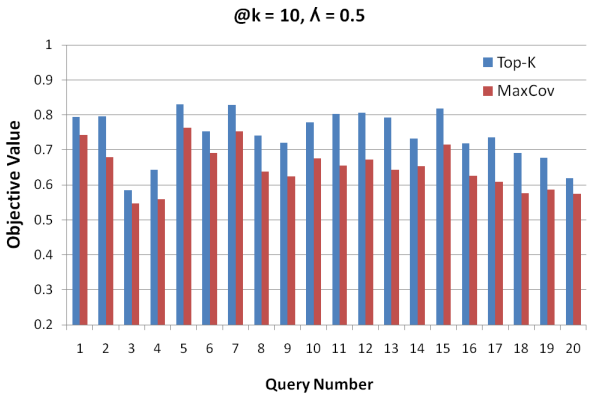

Figure 2: Comparison of Diversity Degree @ k = 10

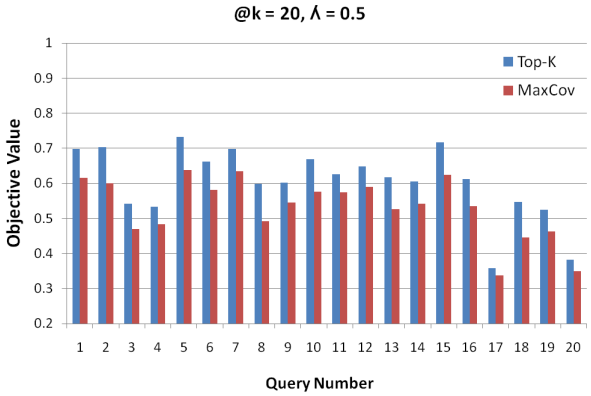

Figure 3: Comparison of Diversity Degree @ k = 20

1. `Top-K`: this is the "default" ranking, i.e., considering only the degree of match without taking into account any non-functional attributes and without applying any diversification method. The set of candidate matches are sorted by the *dom* value in descending order and the top-k services on the list are returned.

2. `MaxCov`: these are the top-k results returned by the method that applies Algorithm 1 for minimizing the objective function given in 3.

The degree of match between the query and the services was computed based on the input and output parameters as described in Section 3.2, while the similarity between the services was computed on the 4 aforementioned attributes, using Jaccard similarity, also described in Section 3.2.

In Figure 2 and Figure 3 we compare the objective value defined by Equation 3 of the selected top-k services for each query with $k = 10$ and $k = 20$, respectively. As discussed earlier in Section 3.1, the parameter λ is used to determine the trade-off between the degree of match and the diversity of the results. Note that in the objective function defined in Equation 3, λ is used as an exponent of the *dom* value and that the *dom* value is between 0 and 1. Therefore, the lower the value of λ, the higher the weight of the *dom* value in the objective function. In our experiments, we set the value of λ to 0.5, which gives the *dom* value more weight than the diversity. The results in Figure 2 and Figure 3 show that the objective value of the `MaxCov` method is lower than the objective value of the `Top-K` method for all queries. This indicates that the results obtained by the `MaxCov` method provide better coverage than those obtained by the other methods w.r.t. the whole set of candidate matches. We also observe that with $k = 20$ the average objective value of both methods is lower than with $k = 10$. The reason for this behavior is that the increase in the number of selected services increases the probability that more relevant services are represented by the top-k services, and hence lowering the coverage error, which in turn leads to a lower value of the objective function.

5. CONCLUSIONS

We have proposed a method to diversify Web service search results in order to deal with users on the Web that have different, but unknown, preferences. Our method focuses specifically on nominal attributes in service descriptions, for which a total ordering can not be defined, since it is dependent on the preferences of each particular user. Such attributes can not be easily incorporated in the matchmaking process, when computing a degree of match to the query. Instead, our approach relies on including diverse and representative services in the results to satisfy different users. We have presented a diversification method and we have evaluated the results on a collection of Semantic Web services.

Directions for future work include mainly the evaluation of our method using larger collections of services and, especially, service descriptions with a larger number of attributes. We would also like to conduct a user study to examine how user satisfaction increases when providing more diversity in the discovered services.

Acknowledgments

This work was partially supported by the FP7 EU Projects LivingKnowledge (contract no. 231126) and Sync3 (contract no. 231854).

6. REFERENCES

[1] R. Agrawal, S. Gollapudi, A. Halverson, and S. Ieong. Diversifying search results. In *WSDM*, pages 5–14, 2009.

[2] R. Akkiraju and et. al. Web Service Semantics - WSDL-S. In *W3C Member Submission*, Nov. 2005.

[3] W.-T. Balke and M. Wagner. Cooperative Discovery for User-Centered Web Service Provisioning. In *ICWS*, pages 191–197, 2003.

[4] U. Bellur and R. Kulkarni. Improved Matchmaking Algorithm for Semantic Web Services Based on Bipartite Graph Matching. In *ICWS*, pages 86–93, 2007.

[5] M. Burstein and et. al. OWL-S: Semantic Markup for Web Services. In *W3C Member Submission*, Nov. 2004.

[6] J. G. Carbonell and J. Goldstein. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *SIGIR*, pages 335–336, 1998.

[7] J. Cardoso. Discovering Semantic Web Services with and without a Common Ontology Commitment. In *IEEE SCW*, pages 183–190, 2006.

[8] X. Dong, A. Y. Halevy, J. Madhavan, E. Nemes, and J. Zhang. Similarity Search for Web Services. In *VLDB*, pages 372–383, 2004.

[9] S. Gollapudi and A. Sharma. An axiomatic approach for result diversification. In *WWW*, pages 381–390, 2009.

[10] T. F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theor. Comput. Sci.*, 38:293–306, 1985.

[11] H. Lausen, A. Polleres, and D. Roman (eds.). Web Service Modeling Ontology (WSMO). In *W3C Member Submission*, June 2005.

[12] F. Kaufer and M. Klusch. WSMO-MX: A Logic Programming Based Hybrid Service Matchmaker. In *ECOWS*, pages 161–170, 2006.

[13] W. Kießling and B. Hafenrichter. Optimizing Preference Queries for Personalized Web Services. In *Communications, Internet, and Information Technology*, pages 461–466, 2002.

[14] M. Klusch, B. Fries, and K. P. Sycara. Automated Semantic Web service discovery with OWLS-MX. In *AAMAS*, pages 915–922, 2006.

[15] S. Lamparter, A. Ankolekar, R. Studer, and S. Grimm. Preference-based selection of highly configurable web services. In *WWW*, pages 1013–1022, 2007.

[16] L. Li and I. Horrocks. A Software Framework for Matchmaking based on Semantic Web Technology. In *WWW*, pages 331–339, 2003.

[17] M. Paolucci, T. Kawamura, T. R. Payne, and K. P. Sycara. Semantic Matching of Web Services Capabilities. In *ISWC*, pages 333–347, 2002.

[18] D. Skoutas, D. Sacharidis, V. Kantere, and T. K. Sellis. Efficient semantic web service discovery in centralized and p2p environments. In *ISWC*, pages 583–598, 2008.

[19] D. Skoutas, D. Sacharidis, A. Simitsis, and T. Sellis. Ranking and clustering web services using multi-criteria dominance relationships. In *IEEE Trans. on Services Computing (to appear)*, 2010.

[20] D. Skoutas, A. Simitsis, and T. K. Sellis. A Ranking Mechanism for Semantic Web Service Discovery. In *IEEE SCW*, pages 41–48, 2007.

[21] E. Vee, U. Srivastava, J. Shanmugasundaram, P. Bhat, and S. Amer-Yahia. Efficient computation of diverse query results. In *ICDE*, pages 228–236, 2008.

[22] J. Wang and J. Zhu. Portfolio theory of information retrieval. In *SIGIR*, pages 115–122, 2009.

[23] Z. Xu, P. Martin, W. Powley, and F. Zulkernine. Reputation-Enhanced QoS-based Web Services Discovery. In *ICWS*, pages 249–256, 2007.