Volume 2, Issue 10, October 2012 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Semantic Web services Discovery and Grading Using Different Matchmaking Values

O.Srinivas[1]	G.prasad[2]	R.Sathyaprakash[3]	K.Shashikanth[4]	S.Naveen Kumar[5]
M.Tech.(CSE)	Assoc. Professor	Asst. Professor	M.Tech.(SE)	Assoc. Professor
SCCE,AP	SCCE,AP	SCITS,KNR,AP	JITS,KNR,AP	SCITS,KNR,AP
India	India	India	India	India

Abstract— ***The web has been increasingly used not only to find answers to specific information needs but also to carry out various tasks, enhancing the capabilities of current web search engines with effective and efficient techniques for web service retrieval and selection becomes an important issue. Existing service matchmakers typically determine the relevance between a web service advertisement and a service request by computing an overall score that aggregates individual matching scores among the various parameters in their descriptions. Two main drawbacks characterize such approaches. First, there is no single matching criterion that is optimal for determining the similarity between parameters. Instead, there are numerous approaches ranging from Information Retrieval similarity measures up to semantic logic-based inference rules. Second, the reduction of individual scores to an overall similarity leads to significant information loss. Determining appropriate weights for these intermediate scores requires knowledge of user preferences, which is often not possible or easy to acquire. Instead, using a typical aggregation function, such as the average or the minimum of the degrees of match across the service parameters, introduces undesired bias, which often reduces the accuracy of the retrieval process. Consequently, several services, e.g., those having a single unmatched parameter, may be excluded from the result set, while being potentially good candidates. In this work, we present two complementary approaches that overcome the aforementioned deficiencies. First, we propose a methodology for grading the relevant services for a given request, introducing objective measures based on dominance relationships defined among the services. Second, we investigate methods for clustering the relevant services in a way that reveals and reflects the different trade-offs between the matched parameters. We demonstrate the effectiveness and the efficiency of our proposed techniques and algorithms through extensive experimental evaluation on both real requests and relevance sets, as well as on synthetic scenarios.***

.

Keywords- Web services matching, grading, clustering, skyline.

I. INTRODUCTION

The web has been increasingly used to find answers to specific information needs but also to carry out various tasks, enhancing the capabilities of current web search engines with effective and efficient techniques for web service retrieval and selection becomes an important issue. Existing service matchmakers typically determine the relevance between a web service advertisement and a service request by computing an overall score that aggregates individual matching scores among the various parameters in their descriptions [2]. Two main drawbacks characterize such approaches. First, there is no single matching criterion that is optimal for determining the similarity between parameters. Instead, there are numerous approaches ranging from Information Retrieval similarity measures up to semantic logic-based inference rules [3]. Second, the reduction of individual scores to an overall similarity leads to significant information loss. 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, or services altering the world state, such as online shopping or booking services. A web service is formally described in a standardized language (WSDL). The service description may include the names and types of input and output parameters, preconditions and effects, as well as Quality of Service (QoS) attributes, such as price, execution time, availability, and reputation. 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. Assume the existence of a repository that contains a large number of advertised service descriptions. In a typical scenario, a user provides a complete definition of the requested service, and issues a discovery query. The repository, then, employs a matchmaking algorithm to identify services relevant to the user's request[2]. Note that perfect matches, i.e., services with the same description as the request, are seldom found. Furthermore, even when a perfect match exists, it may not constitute the most desirable option, e.g., the service is currently unavailable. For these reasons, given a request, the matchmaking algorithm needs to consider a potentially large number of partial matches, and to select the best candidates among them. To effectively deal with the large number of candidates, it's imperative that the identified matches are rovided in auseful form. For this purpose, we examine two distinct methods: grading and clustering. Grading entails assigning a score to each advertisement, quantifying its suitability for the given request. Given

that users typically view only a few top search results, it is important that useful results appear high in the list. Similarly, in fully automated scenarios, where a software agent automatically selects and composes services to achieve a specific task, only the few top ranked results are typically considered. On the other hand, clustering organizes advertisements so that services within a cluster provide similar matches with respect to the request. Since several parameters are involved in the matchmaking process, finding a service that provides a high degree of match for all parameters is difficult; instead, it is often needed to decide between different trade-offs[5].

II. LITERATURE REVIEW.

Given a set *d* of multidimensional points, the *skyline* consists of the points that are not dominated by any other point. specifically, a point *p dominates* another *p'* if the coordinate of *p* is smaller than or equal to that of *p'* on all dimensions, and strictly smaller on at least one dimension. figure 2.1 shows a classical example with a set *d* of 13 points, each capturing two properties of a hotel: its distance to the beach (the horizontal coordinate), and price (the vertical coordinate). the skyline has 8 points *p*1, *p*2, ... , *p*8. skyline retrieval has received considerable attention from the database community, resulting in a large number of interesting results as surveyed. these research efforts reflect the crucial importance of skylines in practice[1]. In particular, it is well-known that there exists an inherent connection between skylines and top-1 queries. Specifically, given a preference function *f*(*p*) which calculates a *score* for each point *p*, a *top*-1 query returns the data point with the lowest score. as long as function *f*(♯) is *monotone*1, the top-1 result is definitely in the skyline. Conversely, every skyline point is guaranteed to be the top-1 result for at least one preference function *f*(♯).the skyline operator is particularly useful in scenarios of multi-criteria optimization where it is difficult, or even impossible, to formulate a good preference function. for example, consider a tourist that wants to choose from the hotels in figure 1 a good one offering a nice tradeoff between price and distance. He may not be sure about the relatively weighting of the two dimensions, or in general, whether the quality of a hotel should be accessed through a linear, quadratic, or other types of preference functions. The concepts are defined in web ontologies, which serve as the key mechanism to globally define and reference concepts. Formal languages enable service composition [10], in which a developer uses automatic or semiautomatic tools to create a integrated business process from a set of independent web services. service composition in a heterogeneous environment immediately raises issues of evaluating the accuracy of the mapping. as an example, consider three real-world web services, as illustrated in fig. 1. the three services distance between zip codes (a), store it contracts (b), and translation into any language (c) share some common concepts, such as the code concept. However, these three services originate from very different domains. service a is concerned with distance calculation and uses the zip codes as input, service b defines currency code as part of the it contract information to be stored, and service c uses a client code as an access key for users. it is unlikely that any of the services will be combined into a meaningful composition. This example illustrates that methods based solely on the concepts mapped to the service's parameters may yield inaccurate results [8]. We aim at analyzing different methods for automatically identifying possible semantic composition. We explore two sources for service analysis: wsdl description files and free textual descriptors, which are commonly used in service repositories.

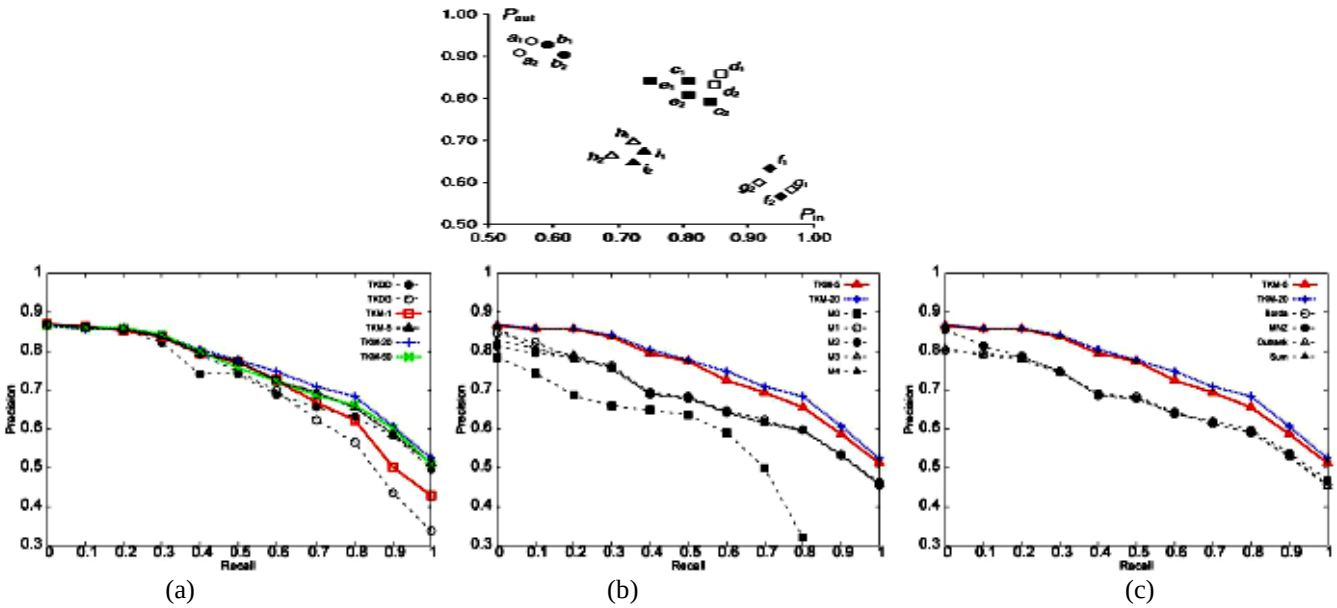

Fig 1 , Fig 2: Grading: Precision-Recallgraphs (a) TKDD, TKDG, and TKM (b) TKM and Measures M0-M4 (c) TKM and Fusion Approaches

We investigate two methods for web service classification for each type of descriptor: term frequency/ inverse document frequency (tf/idf) and context based analysis, and a baseline method. We define contexts as a model of a domain for a given term, which is automatically extracted from a fragment of text. In this work, contexts are created by finding-related terms from the Web.

Service Discovery and Clustering

Services for several domains have been increased, and a large scale service network to simulate the service situation based on complex network. A composition approach has been developed using planning approach on the service network [11. A data-driven service composition approach has been introduced using service data correlation model, which motivates investigation of service discovery. A repository has been created to find proper biomedical services[12]. The repository consists of clusters that have created based on functional similarity by information retrieval technology.

III. PROPOSED APPROACH

Functional Modules:

***Initialization*:**

The problem of grading web services entails computing the scores of services and returning the top-k highest grading ones. All the scores satisfying the requirements qualify as possible grading scores. 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 then the instance examined. 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.

Grading by Dominated Score:

This module computes top-k web services according to the dominated score criterion. The goal is to quickly find, for each object, other objects dominating it, avoiding an exhaustive comparison of each instance to all other instances. The algorithm maintains three lists, Imin, Imax, and I, containing, the minimum bounding instances, the maximum bounding instances, and the actual instances of the objects. The instances inside these lists are sorted and are examined in descending order. The results are maintained in a list R sorted in ascending order of dds. The algorithm uses two variables, ddsMax and min Value, which correspond to an upper bound for dds, and to the minimum value of the current kth object, respectively. for an object U, we are interested in objects that dominate it, we search only for instances that are prior to those of U in I. Since, the top matches are expected to appear in the beginning of I, this significantly reduces the search space.

Grading by Dominating Score:

This module computes the top-k dominant web services with respect to the dominating score, i.e., it retrieves the k match objects that dominate the larger number of other objects. This task is more time-consuming compared to that of above. Furthermore, TKDD allows for more efficient pruning, since it searches among objects and/or instances that have already been examined in a previous iteration, and, therefore, their scores are known. The TKDG algorithm maintains three structures: 1) an I list, 2) a list R of at most k objects of current results, ordered by dominating score descending, and 3) a list L containing objects that have been disqualified from R, used to prune other objects. The lists R and L are initially empty. Similar to TKDD, the algorithm iterates over the objects, in descending order of their maximum bounding instance[9].

Clustering Web Services:

The purpose of clustering is fundamentally different from that of grading. Instead of selecting the k best matches, we organize the matched services into groups that capture different trade-offs among all the considered request parameters. The proposed web service clustering framework effectively summarizes the various trade-offs associated with the multiple matching parameters, while eliminating irrelevant services [4]. The framework is based on dominance relationships, and, thus, satisfies the requirements. More specifically, it comprises the following high-level steps: Select the services that have a skyline probability above a specified threshold [6]. Select representative services from the above set. Form the clusters by assigning each of the remaining services to its closest representative. it uses the distance function between two services defined already. Regarding the step, we choose to set the probability threshold, so that the derived set will contain those matches that have a nonzero probability to belong to the skyline. Thus, intuitively, these are the "most interesting" objects with respect to the existing trade-offs, from which we can then select the seed for forming clusters.

IV. Experiment Results

Main Page:

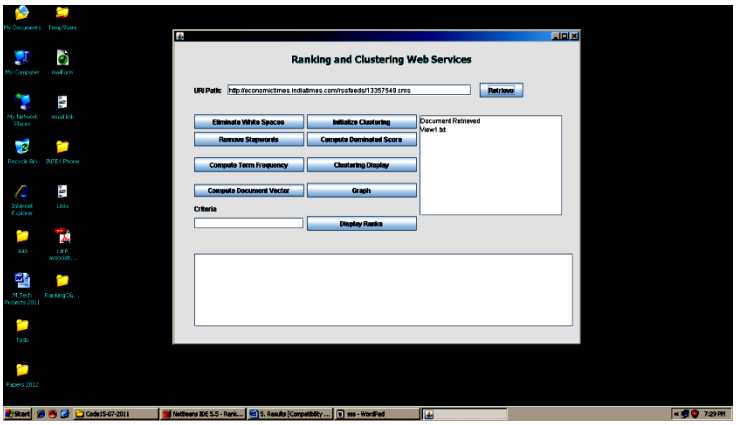

Fig: 2.1

. This is main screen to run the Grading and Clustering web services. This screen will includes step by step procedure for Grading and clustering web services. The Grading will be perform by using Criteria of Textbox of which user wants to display Grading.

The first action starts from top of the textbox when we entering the URL path and when press Retrieve Jbutton, it will retrieve the .xml page details form the web world and that will be stored in the TempCluster folder as View1.txt file. i.e., the document retrieved successfully.

Result:

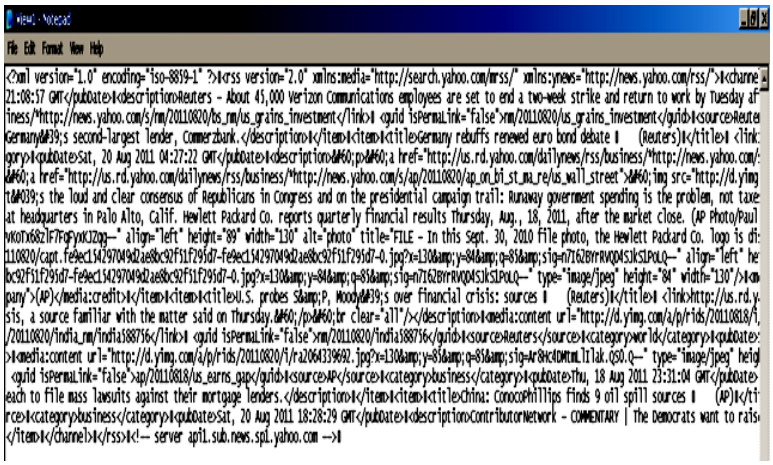

Fig: 2.2

Retrieving xml page 1

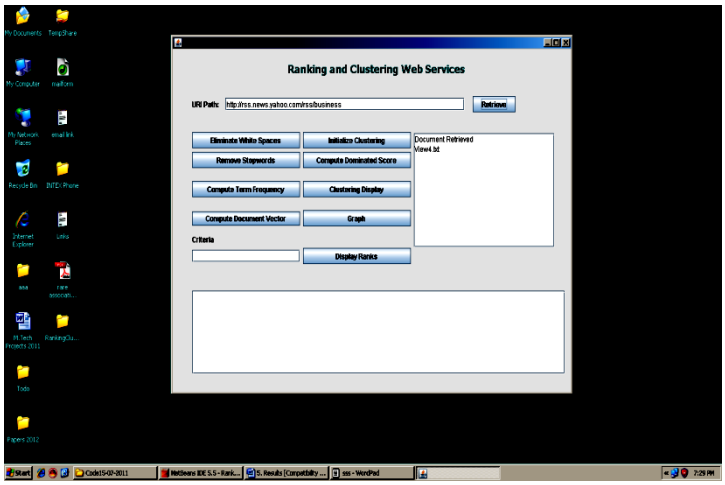

Fig: 2.3

Here we are retrieving different types of web services form web server through the Internet

Compute Term Frequency:

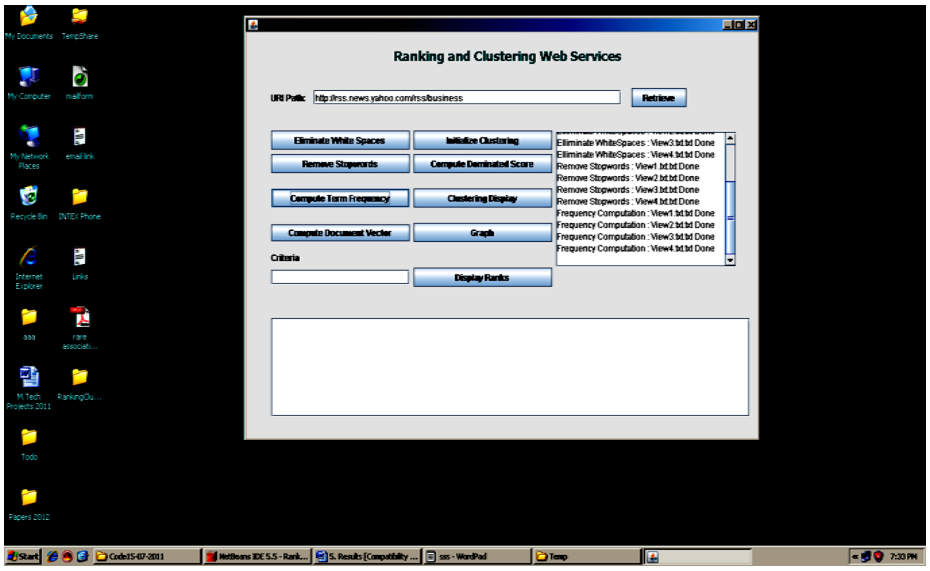

Fig:2.4

In this screen we will "Compute Term Frequency". That means we will calculate that the term of the web service is how many times repeated. The calculated term frequency will be stored in the Temp cluster

Result:

Fig: 2.5

Initialize Clustering:

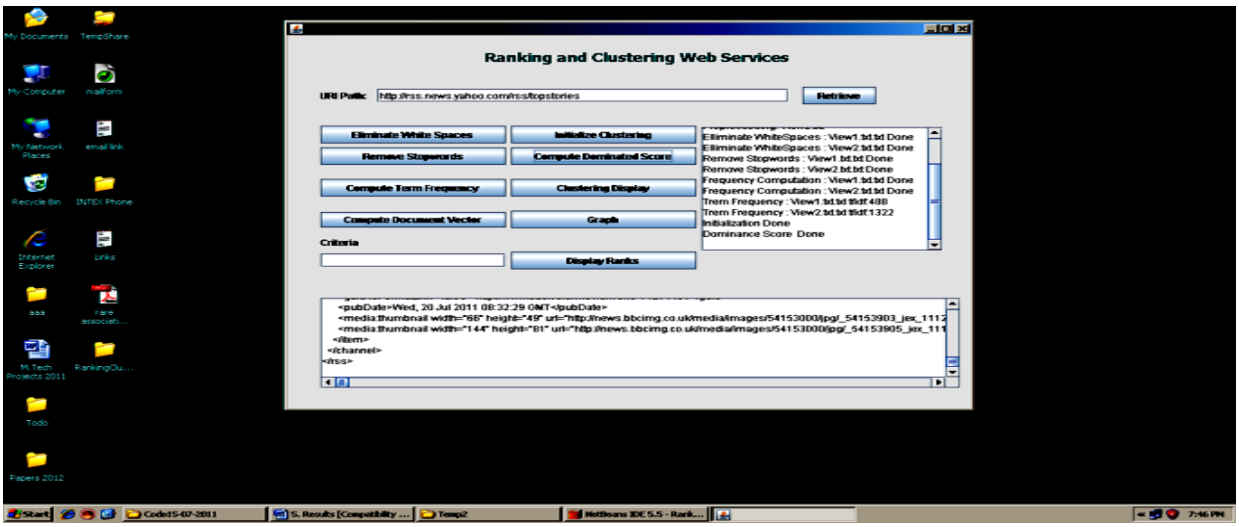

Fig: 2.6

Now, the Clustering will be initialized using above methods. The web service is clustered in to some groups using above actions. When initialization successfully done it gives an message "Initialize Done

Result:

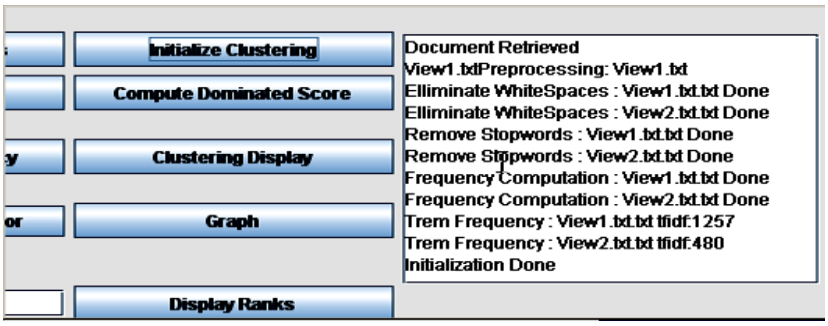

Fig: 2.7

Clustering Display:

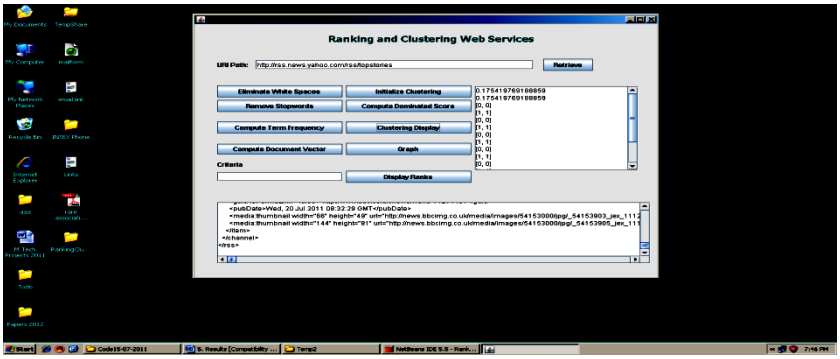

Fig: 2.8

In this screen the Graph will be defined using the Clustering. If the clusters are two then the graph will show two line in the graph chart.

Result:

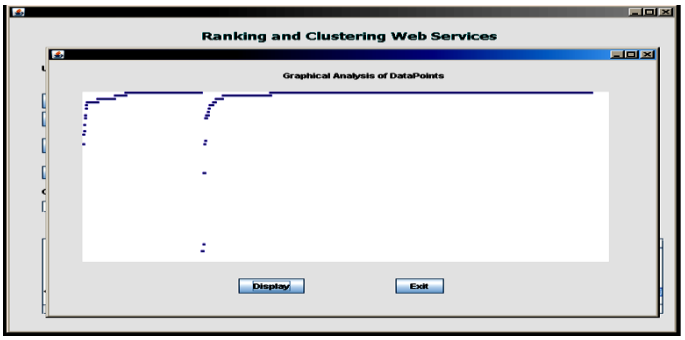

Fig: 2.9

Grading using Criteria:

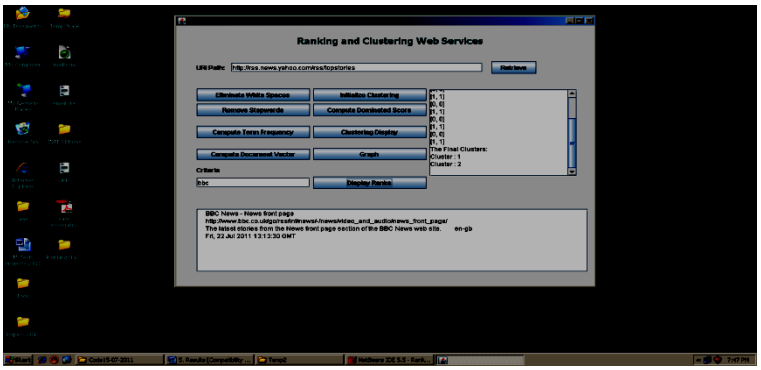

Fig: 3.0

In this screen we display Grading. That means when enter something in the Criteria, by the Criteria the web service will be displayed in the below Text Area.

Result:

Fig: 3.1

V. CONCLUSIONS

In this project, we have addressed grading and clustering of web service search results and proposed methods based on the notion of different match making values, which apply multiple matching criteria without aggregating the match scores of individual service parameters. We have presented three algorithms for grading the search results, and two algorithms for selecting the most representative services for clustering, so that the produced clusters reflect the trade-offs between the matched parameters. An extensive experimental evaluation validates the effectiveness and efficiency of our approach.

REFERENCES

[1] D. Skoutas, D. Sacharidis, A. Simitsis, V. Kantere, and T.K. Sellis, "Top-k Dominant Web Services under Multi-Criteria Matching," Proc. 12th Int'l Conf. Extending Database Technology: Advances in Database Technology (EDBT), pp. 898-909, 2009.
[2]. M. Paolucci, T. Kawamura, T.R. Payne, and K.P. Sycara, "SemanticMatching of Web Services Capabilities," Proc. First Int'l SemanticWeb Conf. (ISWC), pp. 333-347, 2002.

[3] D. Skoutas, A. Simitsis, and T. Sellis, "A Ranking Mechanism for Semantic Web Service Discovery," Proc. IEEE Services Computing Workshops (SCW), pp. 41-48, 2007

[4]. U. Bellur and R. Kulkarni, "Improved Matchmaking Algorithm for Semantic Web Services Based on Bipartite Graph Matching,"Proc. IEEE Int'l Conf. Web Services (ICWS),pp. 86-93, 2007.

[5] J. Ma, Y. Zhang, and J. He, "EfficientlyFinding Web Services Using a Clustering Semantic Approach," Proc. Int'l Workshop Context Enabled Source and ServiceSelection,Integration and Adaptation(CSSSIA), p. 5, 2008

.[6] I. Bartolini, P. Ciaccia, and M. Patella, "Efficient Sort-Based Skyline Evaluation," ACM Trans. Database Systems, vol. 33,o. pp. 1-45, 2008.
[7] J.A.Asalam and M.H Montague ,"Models for Meta Search," Proc ACM SIGIR, pp-275-284, 2001.

[8] F. Kaufer and M. Klusch, "WSMO-MX: A Logic Programming Based Hybrid Service Matchmaker," Proc. European Conf. WebServices (ECOWS), p. 61-170, 2006.

[9] E.A. Fox and J.A. Shaw, "Combination of Multiple Searches," Proc.Second Text REtrieval Conference (TREC), pp. 243- 252, 1993.

[10] S. Oh, D. Lee, and S.R.T. Kumara, "Effective Web Service Composition in Diverse and Large-scale Service Networks" IEEE Transactions on Services Computing, vol. 1, no. 1, pp. 15–32, 2008

[11]W. Tan, et al., "ServiceMap: Providing Map and GPS Assis-tance to Service Composition in Bioinformatics", Proceed-ings of the IEEE International Conference on Web Services (ICWS 2011), Washington D.C., U.S.A, Jul. 2011.

[12] J. Zhang, et al., "Toward Semantics Empowered Biomedical Web Services", Proceedings of the IEEE International Con-ference on Web Services (ICWS 2011), Washington D.C., U.S.A, Jul. 2011.

