Journal of Software Engineering and Applications, **2012, 5, 110-118**
http://dx.doi.org/10.4236/jsea.2012.53018 Published Online March 2012 (http://www.SciRP.org/journal/jsea)

A New Matchmaking Algorithm Based on Multi-Level Matching Mechanism Combined with Fuzzy Set

Adnan I. Al Rabea, Muawiah M. A. Al Fraihat

Al Balqa Applied University, Salt, Jordan.
Email: Adnan_alrabea@yahoo.com, moawya.fraihat@gmail.com

Received December 31st, 2011; revised January 31st, 2012; accepted February 15th, 2012

ABSTRACT

The lack of semantic parts, increasing the number of Web services in the Web, and syntactic-based search operation are the main problems of current Web service technologies, these factors make difficult for clients to find a required web service. This paper shows a matchmaking algorithm to discover Semantic Web Services that are satisfying client requirements. It depends on two factors that distinguish it from any conventional Web service discovery algorithm; the first one is using semantic matching technique to overcome shortcoming of keyword matching techniques, the second one is tying Quality of Service (QoS) metrics of Web Service (WS) with fuzzy words that are used in user's request. At least fifty percent average gain in search relevancy is obtained when our matchmaking algorithm is applied to WSs that are actually matching the chosen fuzzy semantic theme.

Keywords: Semantic Similarity; Syntactic Similarity; Structure-Based Similarity; QoS Metrics; Fuzzy Set; WordNet; Ontology

1. Introduction

One of the crucial steps in an efficient Web service search is to understand what users mean in their request. The search request is usually in the form of natural language. The current popular search engines literally take the search input without much semantic interpretation and attempt to find WS that may contain all or some of the keywords in the input query. This sometimes leads to the inclusion of WS that are not relevant to the user's request in the returned search result. The idea of adding machine-processable semantics to data, that lets computer to understand the information and therefore process it instead of the human user, was behind the evolution of Semantic Web (SW). As expected, SW had also its effect on WS technologies and theory. Thus, some service-oriented SW technologies have also appeared, such as Semantic Markup for Web Services (OWL-S) (formerly DARPA agent markup language for services (DAML-S)) [1], Web Services Semantics (WSDL-S) as extension to the Web Service Description Language (WSDL) [2], Semantic Web Services Ontology (SWSO) [3]/Semantic Web Services Language (SWSL) [4], Web Service Modeling Ontology (WSMO) [5]/Web Service Modeling Language (WSML) [6], and so forth [7]. The services developed with such technologies are called Semantic Web Services (SWS) [8-11] respectively. These technologies provide means of describing in detail the service capabilities, execution flows, policies and other related information. Moreover, these technologies have given a new boost to service discovery and service composition research as new fields for experimentation have emerged [12]. Matchmaking of semantic service descriptions is a key technique for realizing discovery that aims at judging whether a located service is relevant compared to a given request [13].

Many impediments face efficiency of any matchmaking algorithm. From provider's side, the existence of many SWS technologies will lead to that many SWSs will appear without having the same technology, in addition, it is too impractical to expect disparate companies to standardize according to domain-specific (or application-specific) ontologies to define semantics of Web services, Also, application-specific ontologies will be an impediment to automatic discovery of services since the application developer will have to be aware of the specific ontology that has been used to describe the semantics of the service in order to write the query that will be used to search for the service. Here a common universal ontology, such as Word Net [14], should be used by matchmaking algorithm to discover WS. Also every service developer depends on his willing to define his service interface (*i.e.*, input(s) and output(s)) by using words represent abbreviations just understood for him and they may not exist in any thesaurus, in this case no sense for depending only on a semantic similarity to get required WSs therefore in this

paper matchmaking algorithm is used multi-level mechanism(*i.e.*, using three different, matching techniques in matching process, and these techniques are data type, syntactic and semantic technique) to measure similarity of discovered WS. From user's side, user may use fuzzy words in his/her request to describe a level of (QoS) metric for WS such as *high* availability WS or *medium* cost of providing a WS and so forth, also user may use hedges to describe such words (e.g., very ,indeed somewhat) as we will see later. To solve such problem our matchmaking algorithm includes AI mechanism, using fuzzy sets, to convert fuzzy words (*low*, *medium*, *high* or any counterparts for those words) to values that will be used eventually to rank WSs to get relevant of them. If user use hedge in his query that will affect membership value therefore rank of WSs.

The main idea in the proposed algorithm is improving WS search efficiency by filtering search results to extract more relevant ones. Using multiple matching techniques and dealing with fuzzy words that may be used in user request are important factors will contribute to increase the efficiency of matchmaking algorithm by increasing relevance ratio of WSs in the result of search. Our experiments show a clear advantage of the proposed search methodology over the conventional one whereas we gain fifty percent average in search relevancy when our search methodology is applied, therefore improving in the efficiency as much as the double of what conventional search do.

This paper describes a matchmaking algorithm based on multi-level matching mechanism combined with fuzzy set, whereas this paper is structured as follows; Section 2 gives related works, Section 3 represents problem formulation, Section 4 shows QoS in Web Services, Section 5 exhibits using fuzzy set, Section 6 demonstrates a proposed methodology and Section 7 shows simulated result and reminder section gives conclusion and future work.

2. Related Works

One of the first works in the field of SWS discovery is what described in [15], this work takes into account only the inputs and outputs of the service profiles during matchmaking. The degree of match between two outputs or two inputs depends on the relationship between the domain ontology concepts associated with those inputs and outputs. This approach is classified as semantic capabilities matching where WS's capabilities include IOPE (*Inputs*, *Outputs*, *Preconditions and Effects*) set. In this approach QoS factor has not been taken into account and also it doesn't measure all kind of similarities that I used, that is, syntactic and data type. Duygu Celik and Atilla Elci developed a system uses a Semantic Search Agent (SSA) to discover required Web services from Web and selects them according to the client requirements, the SSA takes client's request word(s) and augments them with their synonym and is a related terms by using ontology database. Then these terms are sent to UDDI (Universal Description, Discovery, and Integration) registry server to retrieve all related Web services [16], discovery process of their approach will be performed which equals the number of total such terms multiplied by the number of WSs that are published in UDDI [17], however, in the proposed approach the discovery process will be repeated which equals the number of WSs are published in repository, because only the terms that are used in user request will compute their similarity with each WSs without the need to augment them, thus will give better efficiency than such approach provides.

Zhang proposed a system for WSs composition, in the part which is related to Interface-Matching Automatic Composition technique (IMAC), the possible compositions are obtained by checking semantic similarities between interfaces of individual services and considering the service quality using [18]. In such approach syntactic similarity and data type similarity are not considerable, she used matchmaking algorithm depends on checking semantic similarity (*i.e*., matching). In addition, she depended on tailor-made ontologies which are built for Web services that exist in the repository, therefore, any additional WSs may require modifying the existed ontologies or may create new ontologies, in other words, frequent maintenance for used ontologies. But in this paper Word Net is used as semantic lexicon for English language that includes comprehensive ontologies, therefore, less maintenance may be needed if new WSs will be added to the repository.

One of the powerful concept of Fuzzy Logic is its association of linguistic variables to numeric variable [19]. The advantage of this concept has been taken by Tseng and Vu [20] to enhance the capability of Web search engines in perceiving search queries [21]. Therefore their contribution was in Web search field not in WSs they developed Fuzzy Numeric Semantic (FNS) search. Their experiments show more improvement in the search results than what conventional search shows. Such improvement is the motivation behind a using FNS search methodology to increase relatedness of discovery Web services result, so such search on semantic Web service is applied in the proposed approach but with modifications to deal with QoS of WS. Approach of Tseng and Vu concerning Web pages search not Web Services search so the parameters of their approach will be different. Tseng and vu have focused on the search query in the form of (X, Y); where X is the constraining variable and Y is the constrained variable. Take for example; in the search query *reduce cholesterol*, *reduce* (X) is considered as a variable constraining the variable *cholesterol* (Y). To take advantages of semantic structure in the search string, first they have

submitted the constrained variable Y to the search engine to retrieve Web pages that may contain Y. Then their system would parse through the returned Web pages for possible existence of the constraining variable X; X represents fuzzy set that has been used to compute degree of membership.

We have adopted same form of query (X,Y) that they have followed but in our method we deals with Web services not Web pages in addition X represents fuzzy set that deals with QoS metrics values of WS and these values can't be used in web pages search, QoS is considered as the universe of discourse Q, and (*high*, *medium* and *low*) are linguistic variable that is determined by fuzzy sets on such universe of discourse (*i.e.*, Q). And Gaussian membership function has been chosen to represent these fuzzy sets. In our algorithm represents inputs and outputs of web services that user requests according them our algorithm conducts three matching techniques (syntactic, data type and semantic matching technique) but Y, in Tseng and Vu, represents submitted keyword to the search engine and obtain the set, **U**, of the search result web pages and not all such techniques are used in matching process to get desired pages that fit to user request.

3. Problem Formulation

Conventional Web services discovery, that uses pure syntax-based matching, has shortcomings and limitations, the pure syntax-based matching of service capabilities cannot give quality results, because a service request matches an advertisement only if their keywords match. This prevents what is known as inexact matching (*i.e.*, the matching between two services even if they are described with different keywords).

Most service descriptions are provided in natural language form. In addition, inference cannot be performed on UDDI business and service catalogs. This further hinders the matching process, if the provider and requestor do not use common vocabulary.

Also ignoring data type in matching process leads to get WS in the results which user can't use (*i.e.*, passing inputs to this WS and can't get the result) because of type mismatch. Inability of Service matching process to handle the presence of fuzzy words that are contained in user's request shape another hinder to enhance the intelligence. We proposed our algorithm, so called Sohaib (**Appendix 1**), that is just not build on syntax-based matching but it involves two other matching techniques (semantic, data type) in addition we used fuzzy sets to enhance intelligence in our matchmaking therefore overcoming all aforementioned hinders.

4. QoS in Web Services

Selecting suitable Web services regarding the QoS provision is a determinant factor to ensure customer satisfaction. Different users may have different requirements and preferences regarding QoS. For example, a user may require minimizing the response time (RT) while satisfying certain constraints in terms of price and reputation, while another user may give more importance to the price than to the response time [22].

In the context of Web services, different classes of service may be considered, either depending on a classification of users in occasional buyers, regular clients, and VIP clients, or depending on a service charge which the user is willing to pay. It is therefore appropriate to provide different classes of services [23].

A metric defines a qualitative or quantitative property that users want to evaluate. It is characterized by its name, data type, type of element to be measured, and computation logic [23]. In Web services environments, metrics are not static, and they should always be modified to handle the change. As example of QoS metrics the response time, cost and reputation etc. **Table 1** describes examples of Web services classes implementing different QoS parameters or metrics.

Each of metric, as we assumed in our experience, has value between 0 and 1. This value we use to get membership value of fuzzy sets, **Figure 1**, that represents words that user may use to express the level of QoS metric he needs, as example, high availability, low cost.

5. Using Fuzzy Set

Our algorithm uses fuzzy set as type of AI techniques, our algorithm is designed to deal with fuzziness based on terms such as high, medium, low or any their synonyms, such words admit of degrees and all come on a sliding scale. Such a sliding scale often makes it impossible to distinguish members of a class from non-members [24]. When WS has 0.7 as a value of availability ,one of QoS metrics, we can't determine accurately either such WS has a medium value of availability or a high value of availability in case user requests a high availability WS. So we represented each of such terms as fuzzy set, set that calibrates vagueness, by which we can determine the degree of membership of each published WS according its value of QoS metric. In addition our algorithm deals with the hedges that may be used in user query (e.g., little, slightly, very, extremely, etc) ,these hedges will affect the shape of fuzzy sets [24].

6. Proposed Methodology of Solution

Sohaib algorithm depends on two main steps [25]. The first one is using multi-level matching mechanism through three categories of matching methods; structure-based matching technique, syntactic matching technique and semantic matching technique. These categories are used to

Table 1. Differentiated classes of services [14].

Qos Parameters \ Class of web serives	Class 1	Class 2	Class 3	...	Class n
Response Time	N/A	0.7 ms	0.5 ms		0.1 ms
Latency	N/A	N/A	0.1 ms		0.01ms
Availability	N/A	N/A	0.8		1(100%)
Reputation	N/A	N/A	N/A		5/5
Service charge	0.10 $	0.2 $	0.25 $		0.35 $

N/A: not applicable.

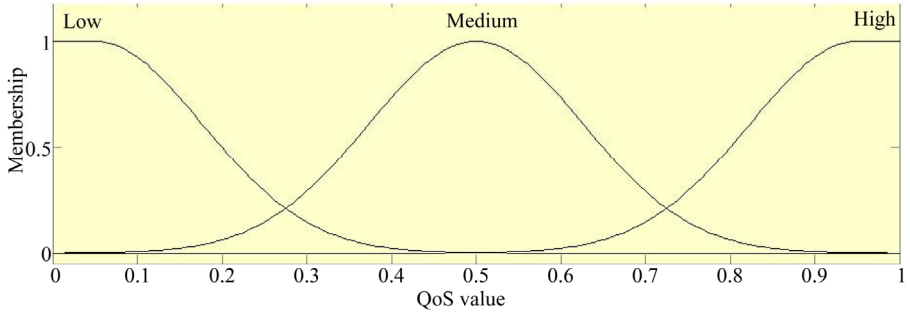

Figure 1. Membership functions that are used in Sohaib algorithm.

measure data type similarity, Syntactic similarity and Semantic similarity respectively between what user wants and WSs that are published in registry. The second step is using QoS as a value that we want to compute the membership of fuzzy set that represents linguistic variable to describe what quality of SWSs which user wants.

The reason behind of using first step is to get accurate similarity value and therefore increases relevance of WSs in the search results. For each published WS, the proposed algorithm will compute similarity between the inputs of such WS and the inputs of desired WS in user's request then it will transfer to compute similarity for the outputs of both according to threshold value. To increase efficiency of running of proposed algorithm thresholds θ_1 and θ_2 are used (see **Appendix 1**), thereby the cost of matching process will decrease. θ_1 is resorted to in the proposed algorithm in two cases, in the first one to decide if syntactic and semantic similarity can be measured after a data type similarity has been measured; if data type similarity is greater or equal θ_1 then we can go forward, otherwise another WS will be chosen to examine, in the second case if three similarity types are measured, an amalgamation equation will be used as the following:

$$S_T = S_D \times \text{Max}\,(S_{Sem}, S_{Syn}) \quad (1)$$

where S_T is total similarity value, S_D is data type similarity value, Max is maximum value between two values between two brackets, S_{sem} is semantic similarity value, S_{syn} is syntactic similarity value.

If one or two words that we need to catch semantic similarity is/are not exist in the WordNet, measuring of such similarity will be skipped. Such equation will performed twice; when we compute similarity for inputs and when we compute similarity for outputs. We see that this equation combines the result of the individual similarity value of each types in one value then it will be compared by θ_1, if we found such value less than θ_1 value we will not transfer to outputs of WS that is being examined to catch it's similarity with existed outputs in user's request, then we will go to next WS in the repository to conduct matching process. We resorted to θ_2 to filter WSs to more related ones, that's will be done after catching similarity of inputs and also similarity of outputs and then get the minimum value between both, such value will be compared with θ_2, if it equals or greater than θ_2 WS will be stored in the list of related WSs (LRWS). We know that the order of the inputs and outputs of WS in its description; that is published in registry, will not be always identical with the order of inputs and outputs of desired WS that user requests, to overcome this problem used Kuhn-Munker algorithm has been used (also known as the Hun-garian method) [26]. Hungarian method is used here to solve the maximum total weight of bipartite matching problem of two resulted matrices; one related to inputs and the other one related to outputs. Ultimate similarity value will be average of assigned similarity values using such method. So we have two average values; one represent similarity of inputs and the other for similarity of outputs, but we need one to represent ultimate similarity value here minimum value of both is the solution for that.

The second step in the proposed methodology is used mainly to get WSs, in LRWS, that are match with user's request in terms of the level of QoS he/she wants. The

proposed algorithm has come to handle the problem of using fuzzy words (*i.e.*, Fuzzy Linguistic Variables(FLV)) in user's request such as *low*, *medium*, *high* or any counterparts for them, in this context Gaussian membership function has been used, **Figure 1**, to represent each fuzzy set (*i.e.*, *low*, *medium*, *high* or any counterparts for them). The reason behind using Gaussian membership function Tseng and Vu, in their experiments, show improvement in the search results than what conventional search shows. Fifty percent average gain in search relevancy is obtained when their search methodology is applied to websites matching the chosen fuzzy semantic theme. They have used Gaussian membership function to represent each fuzzy set have been used in their experiments, so we used Gaussian membership function to represent each fuzzy set we used to exploit its ability to improve search relevancy.

The value of QoS metric of WS in LRWS will be used, that user requests, to compute membership value (μ) then such value will be used in the following general equation to rank WSs in LRWS

$$\begin{aligned} \mathrm{WS_R} &= \mu_{\mathrm{FLV}}\left(\mathrm{QoS}\right) \\ &\times \mathrm{Min}\left(\mathrm{S_T\ of\ Ws\ input}\left(\mathrm{s}\right),\ \mathrm{S_T\ of\ Ws\ output}\left(\mathrm{s}\right)\right) \end{aligned} \tag{2}$$

where $\mathrm{WS_R}$ is Rank of WS, μ_{FLV} is membership degree that we get when we pass QoS metric value of WS to fuzzy set that represents Fuzzy Linguistic Variable (FLV), where FLV (*high, medium*, *low* or any their synonyms), Min is minimum value between two values between two brackets, $\mathrm{S_T}$ is Total similarity value that we get from applying equation No. 1.

We have deduced our equation from Tseng and Vu equation ($R = S^*\mu_{\mathrm{FNS}}(z)$) but the operands ,in our equation, is different of what they have used. We adopt this formula because we may found many WSs have same input(s) and output(s) and they have suitable similarity value with user's request but the QoS is different because we know there are many vendors to the same service therefore same services may not have same QoS value so we consider QoS as factor that effects directly the similarity value then the existence of multiplication operation between two operands in our equation does make since.

If user resorts to more than one QoS metric in his/her query, here we will resort to operations of fuzzy sets (union, intersection and so forth) to get aggregate value of all membership values of used Fuzzy sets, for example when user request *high* reputation AND *low* cost, AND (*i.e.*, intersection) operation will be used such that the result of such operation is minimum value μ_{High} (value of QoS of reputation) and μ_{Low} (value of QoS of cost) but if user used OR (*i.e.*, union) operation here we will use maximum value. We can't also forget that the cases that user may use hedges such as (*a little*, *slightly*, *very*, *extremely*, *somewhat* and etc.) and we know hedges will affect the shape of any fuzzy set, as example of using hedge, when user request *very high* availability WS, here membership function of such request is [μ_{High} (value of QoS of availability)]^2.

7. Simulated Results

In the experimental side, it assumed input(s), output(s) and QoS values are extracted from description file of each WS in registry, to explain how the proposed algorithm is work we will use dataset that contains WSs in the domain of books and we will perform one request .Before doing that, we should explain followed way to judge how efficient the proposed algorithm is, in this side two indicators are depended on; the first one is ratio of relevance that represents the number of related WSs in search result divided by number of actual related WSs that are supposed to be in the result, and the second one is ratio of irrelevance that represents the number of unrelated WSs that appeared in the result divided by the number of actual unrelated WSs that should not be in the result.

All queries that are used in the experiments of this paper in the form of (X, Y) same a form that has been followed by [17]; where X is the constraining variable and Y is the constrained variable; **Figure 2** represents followed form of user's query when user used just one FLV with one hedge.

Let us take query that will be applied on the WSs in the domain of book's services. In this query user wants *high* availability WS where *title* of book represents the input with string data type and *writer* of book as the output with string data type, here X is *high* reputation and Y is all WSs that have input is *title* and its output is *writer*. All WSs in the domain of books will be examined one by one to get WSs that fit to Y part, three matching types (multi-level) will be performed from inputs to outputs of each, any WS has similarity value is equal or greater than θ_2 will be stored in LRWS then they will be ranked according level of QoS metric(s) that user needs (X part).also any used hedge should be taken into account. The dataset that will be discovered has 64 WSs in books domain, the Actual Related WSs (ARWs) that match with the Y part semantically or syntactically or both and they should be appeared in search result equal 16 WSs, thus the rest of WSs (*i.e.*, 48) will be Actual Unrelated WSs (AUWs) that should not be in the result of Y part, both values will be used to determine relevance and irrelevance ratio respectively as the following:

$$\begin{aligned} &\text{Relevance ratio} \\ &= \left(\text{no. of related WSs} / \text{no. of ARWs}\right) \times 100\%, \\ &\text{Irrelevance ratio} \\ &= \left(\text{no. of unrelated WSs} / \text{no. of AUWs}\right) \times 100\%. \end{aligned}$$

Table 2 shows the results of using different values of

Table 2. Ratio of relevance and irrelevance for different values of θ_2.

	$\theta_2 = 0.5$	$\theta_2 = 0.6$	$\theta_2 = 0.7$	$\theta_2 = 0.8$	$\theta_2 = 0.9$	$\theta_2 = 1$
Relevance ratio	(16/16) × 100% = 100%	(16/16) × 100% = 100%	(16/16) × 100% = 100%	(16/16) × 100% = 100%	(8/16) × 100% = 50%	(8/16) × 100% = 50%
Irrelevance ratio	(16/48) × 100% ≈ 33%	(16/48) × 100% ≈ 33%	(16/48) × 100% ≈ 33%	(8/48) × 100% ≈ 16.7%	(0/48) × 100% = 0%	(0/48) × 100% = 0%

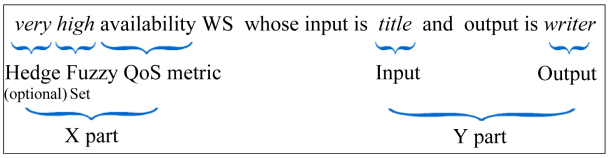

Figure 2. Followed form of user's query.

θ_2, and we can see that when θ_2 increases the ratio of relevance and ratio of irrelevance decreases.

To compare our algorithm with any conventional WS discovery algorithms based keyword matching technique (*i.e.*, syntactic-based matchmaking) in terms of the ratio of relevance we will use Y part to do that and data type similarity will be ignored, it has been found that the ratio of relevance of such algorithm is 25% to the same previous query but the proposed algorithm was the double of its ratio (*i.e.*, 50%) and that was in the worst case when $\theta_2 = 1$ (WSs in the result is identical to desired WS in user's query).

Table 3 represents all WSs that have appeared for the same previous user's request example when we used syntactic-based matchmaking algorithm and we got 4 out of 16 WSs should appear in the search result which is mean we get only 25% as Ratio of relevance. **Table 4** represents all WSs that have appeared, for the same request, when we used our proposed matchmaking algorithm and we got 16 out of 16 WSs should appear in the search result which is mean we get only 100%, in the best case, as Ratio of relevance. We have conducted this comparison only on Y part of query not on both X and Y because the traditional matchmaking algorithm don't enhance intelligence specially dealing with fuzzy words which describe a level of QoS that user requests and the matching process of traditional matchmaking settles for numeric values to get WS that fits to user's request but our algorithm deals with fuzzy words so we can't conduct comparison that involves part X of user's query that has form (X, Y).

After getting LRWS we want to rank these WSs according to QoS metric(s) that user wants, this is the role that X part plays in the proposed algorithm in which user uses FLV(s), according that we can chose fuzzy set that is tied with required QoS metric to get membership value of each WS in LRWS, such value will be multiplied by similarity of WS with desired WS to rank it. **Figure 3** shows rank value of each WSs in LRWS when $\theta_2 = 0.7$ and **Figure 4** shows rank value of each WSs in LRWS when

Table 3. Syntax-based matchmaking algorithm results.

WS Name	Syntax of Input	Syntax of Output
a21	Title	Writer
a22	Title	Writer
a23	Title	Writer
a24	Title	Writer
Total	4	

Table 4. Our algorithm results.

WS Name	Syntax of Input	Syntax of Output
a17	Title	Author
a18	Title	Author
a19	Title	Author
a20	Title	Author
a21	Title	Writer
a22	Title	Writer
a23	Title	Writer
a24	Title	Writer
a49	Name	Author
a50	Name	Author
a51	Name	Author
a52	Name	Author
a53	Name	Writer
a54	Name	Writer
a55	Name	Writer
a56	Name	Writer
Total		16

$\theta_2 = 1$ of the same user's request and u can see when θ_2 increases the number of WSs that have high score (over 0.5) decrease that denotes that θ_2 has influence on ranking of WSs.

Let us add a hedge to used user's request in this paper to know if it can affect or not therefore know is it worth to take hedges into account in our algorithm, we will chose very hedge to add it to used request and such request will be as the following very high availability WS that has title as input and writer as output. **Figure 5** and **Figure 6** show applying hedge on the result of opposite figures **Figure 3** and **Figure 4** respectively and you can see there is a difference in the rank value of the same WS between these figures, that's why we consider hedges in our algorithm.

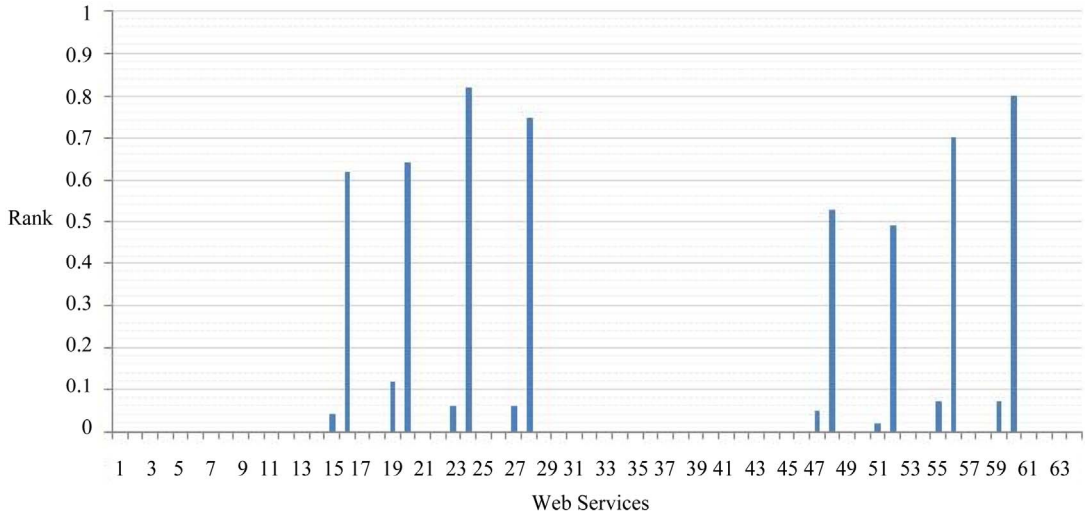

Figure 3. Rank value of each WS in LRWS when θ_2 = 0.7.

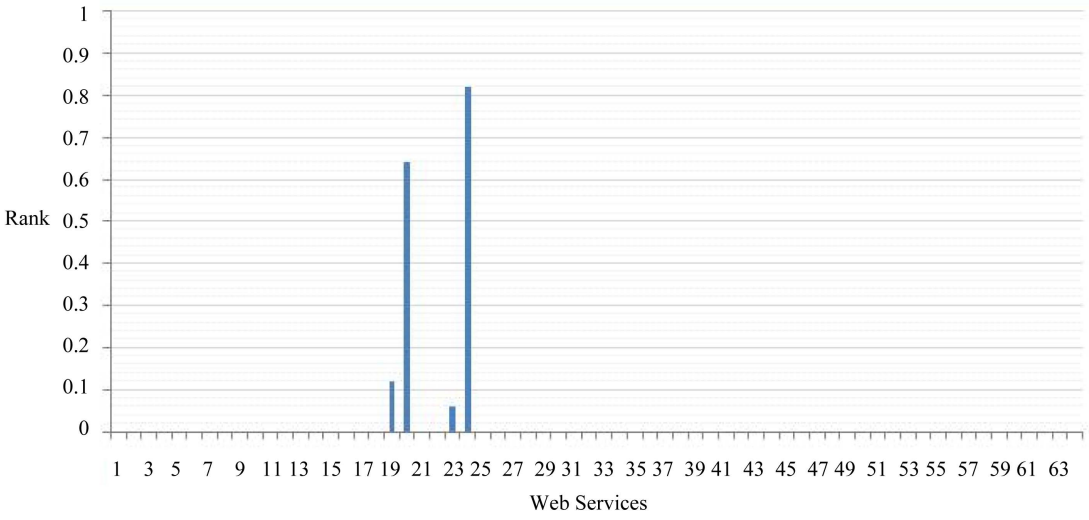

Figure 4. Rank value of each WS in LRWS when θ_2 = 1.

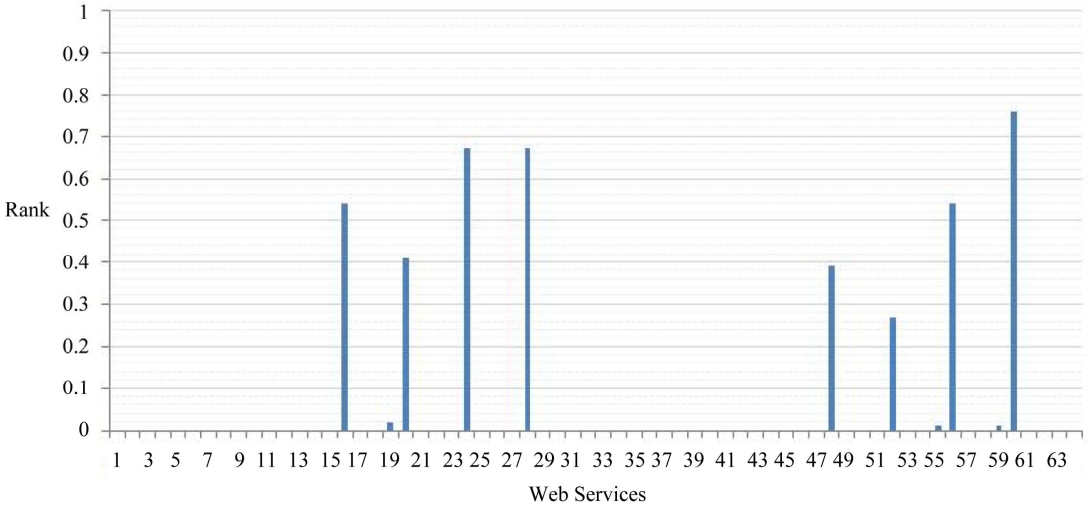

Figure 5. Rank value of each WS in LRWS when θ_2 = 0.7 with using hedge (very).

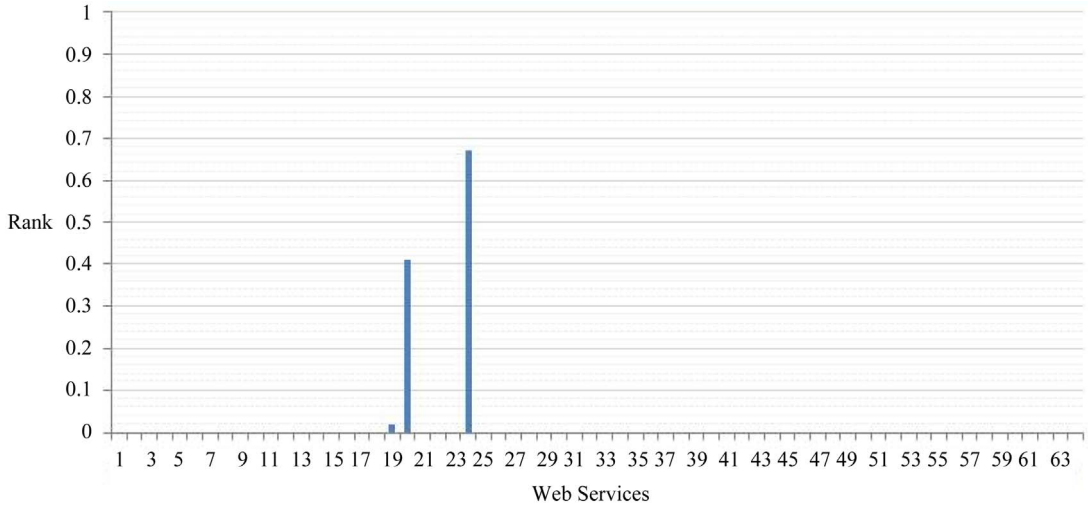

Figure 6. Rank value of each WS in LRWS when $\theta_2 = 1$ with using hedge (very).

8. Conclusion and Future Work

Satisfying user's request behind proposing many algorithms to discover WS in the last years and getting accurate result represents efficiency of any of them, in this context, the proposed algorithm in this paper come to enhance accuracy of WS search results by using multi measurements to compute similarity between examined WS and user's request and with taking into account dealing with FLVs that maybe used in user's request. Fifty percent average, in the worst case when $\theta_2 = 1$, gain in search relevancy is obtained when our search methodology is applied to WSs and this shows the importance of semantic in addition using fuzzy set in improving WS search efficiency. In future work, we see that the choosing suitable value of θ_1 and θ_2 to make the ratio of relevance optimal and ratio of irrelevance minimal is optimization problem that needs AI to solve , to improve results of proposed algorithm we use more similarity measurements (such as security, region of WS, effect and etc.) than that are used here will has a significant leverage to increase accuracy level of search result, in addition finding the weight of how each measurement will influence on relevance and irrelevance ratio is optimization problem needs AI to solve.

REFERENCES

[1] DAML-S, http://www.daml.org/services/owl-s/ http://www.w3.org/Submission/OWL-S/

[2] WSDL-S, http://www.w3.org/Submission/WSDL-S/

[3] SWSO, http://www.daml.org/services/swsf/1.0/swso/

[4] SWSL, http://www.daml.org/services/swsl/

[5] WSMO, http://www.w3.org/Submission/WSMO/

[6] WSML, http://www.w3.org/Submission/WSML/

[7] J. Cardoso and A. Sheth, "Introduction to Semantic Web Services and Web Process Composition," In: J. Cardoso and A. Sheth, Eds., *A Semantic Web Process*: *Powering Next Generation of Processes with Semantics and Web Services*, Springer-Verlag, Heidelberg, 2005, pp. 1-13.

[8] R. Lara, H. Lausen, S. Arroyo, J. de Bruijn and D. Fensel, "Semantic Web Services: Description Requirements and Current Technologies," Paper Presented at the Semantic Web Services for Enterprise Application Integration and E-Commerce Workshop, in Conjunction with ICEC 2003, Pittsburgh.

[9] S. McIlraith, T. C. Son and H. Zeng, "Semantic Web Services," *IEEE Intelligent Systems*, Vol. 16, No. 2, 2001, pp. 46-53. doi:10.1109/5254.920599

[10] L. Nixon and E. Paslaru, "State of the Art of Current Semantic Web Services Initiatives," Technical Report No. D2.4. ID1, Knowledge Web Project, 2005.

[11] H. Wang, J. Z. Huang, Y. Qu and J. Xie, "Web Services: Problems and Future Directions," *Journal of Web Semantics*, Vol. 1, No. 3, 2004, pp. 309-320. doi:10.1016/j.websem.2004.02.001

[12] J. Cardoso, "Semantic Web Services: Theory, Tools and Applications," Electronic Edition, 2007. http://www.ebooksclub.org

[13] S. Grimm, "Intersection-Based Matchmaking for Semantic Web Service Discovery," *Second International Conference on Internet and Web Applications and Services*, Germany, 13-19 May 2007.

[14] WordNet, "An Electronic Lexical Database for English," Massachusetts Institute of Technology Press, Cambridge, 1998. http://wordnet.princeton.edu

[15] M. Paolucci, T. Kawamura, T. R. Payne and K. P. Sycara, "Semantic Matching of Web Services Capabilities," In: I. Horrocks and J. Hendler, Eds., *First International Semantic Web Conference on the Semantic Web*, Springer-Verlag, Sardinia, 2002, pp. 333-347.

[16] D. Celik and A. Elci, 2006. "Discovery and Scoring of Semantic Web Services Based on Client Requirement(s) through a Semantic Search Agent," *Proceedings of the* 30*th Annual International Computer Software and Ap-*

plications Conference, Chicago, 17-21 September 2006, pp. 273-278. doi:10.1109/COMPSAC.2006.127

[17] UDDI Universal Description, Discovery and Integration. http://www.uddi.org/

[18] R. Zhang, "Ontology-Driven Web Services Composition Techniques," Master Thesis, University of Georgia, Georgia, 2004

[19] L. Zadeh, "From Computing with Numbers to Computing with Words, from Manipulation of Measurements to Manipulations," *Annals of the New York Academy of Sciences*, Vol. 929, 2001, pp. 221-152. doi:10.1111/j.1749-6632.2001.tb05718.x

[20] C. Tseng and T. Vu, "A Perception-based Web Search with Fuzzy Semantic," In: E. Sanchez, Ed., *Fuzzy Logic and the Semantic Web*, Elsevier, Amsterdam, 2006, pp. 211- 230.

[21] L. Zadeh, "Toward a Perception-Based Theory of Probabilistic Reasoning with Imprecise Probabilities," *Journal of Statistical Planning and Inference*, Vol. 105, No. 1, 2002, pp. 233-264. doi:10.1016/S0378-3758(01)00212-9

[22] E. Badidi, L. Esmahi and M. A. Serhani, "A Queuing Model for Service Selection of Multi-Classes QoS-Aware Web Services," *Proceedings of the 3rd IEEE European Conference on Web Services*, Sweden, 14-16 November, 2005, pp. 204-212. doi:10.1109/ECOWS.2005.3

[23] M. Radaideh and H. Al-Ameed, "Architecture of Reliable Web Applications Software," Electronic Edition, 2007. http://www.ebooksclub.org

[24] M. Negnevitsky, "Artificial Intelligence: A Guide to intelligent Systems," 2nd Edition, Addison Wesley, Boston, 2005.

[25] M. Fraihat, "Using Artificial Intelligence Techniques (Fuzzy Set) for Enhancing Semantic Web Services Discovery," Master Thesis, Al-Balqa` Applied University, Salt, 2009.

[26] J. Munkres, "Algorithms for the Assignment and Transportation Problems," *SIAM Journal on Applied Mathematics*, Vol. 5, No. 1, 1957, pp. 32-38. doi:10.1137/0105003

Appendix 1