IJICT International Journal of Information & Communication Technology

ITRC

Volume 2- Number3 - November 2010 (pp.1- 8)

Fuzzy Clustering for Semantic Web Services Discovery based on Ontology

Nayereh Gholamzadeh
School of Electrical and Computer Engineering
College of Engineering, University of Tehran
Tehran, Iran
n.gholamzade@gmail.com

Fattaneh Taghiyareh
School of Electrical and Computer Engineering
College of Engineering, University of Tehran
Tehran, Iran
ftaghiyar@ut.ac.ir

Azadeh Shakery
School of Electrical and Computer Engineering
College of Engineering, University of Tehran
Tehran, Iran
Shakery@ut.ac.ir

Received: September 25, 2010 – Accepted: November 26, 2010

Abstract— **Web services as the most important event in distributed computing, have achieved great popularity among software developers today. A critical step in the process of developing service-oriented applications is web service discovery, i.e., the identification of existing relevant web services that can potentially be used in the context of a new web application.**

In this paper, we have proposed a novel method based on data mining techniques to assist and improve the web service discovery process as well as the development of service-oriented applications. Our assistant discovery approach is based on automatic finding of semantic similarity between web services through the application of clustering methods. We have introduced a new fuzzy semantic clustering algorithm which assists web service consumers in discovering a group of similar web services through an individual query. This objective is attained by way of a search space reduction mechanism which adds to the efficiency of the approach. Our proposed approach provides dynamic and flexible clusters which can be changed at discovery process.

We have conducted an experimental study on a data set of tagged web services with ontology. The ontology supports the semantic analysis. Preliminary results from clustering indicate the possibility of retrieving web services at the discovery process with reasonable precision by applying the proposed similarity model. From these promising results, we conclude that web service discovery process could be performing in a reasonable time because of reduced search space.

Keywords- Web Service; Fuzzy Clustering; Ontology; Data Mining; Semantic; Web Service Discovery.

I. INTRODUCTION

The web was conceptualized for human use; this infrastructure has evolved to allow computer programs become a key player in this role. Recently, the trend in software development has been converging towards reusing and composing loosely coupled functionality accessible by the web, commonly known as the service. A web service is a loosely coupled software component that can be published, located and invoked by using the standard web infrastructure [1]. The objective of the service-oriented computing methodology is to develop specific software components with a combination of service discovery, selection and invoking [2] that has acquired enormous popularity.

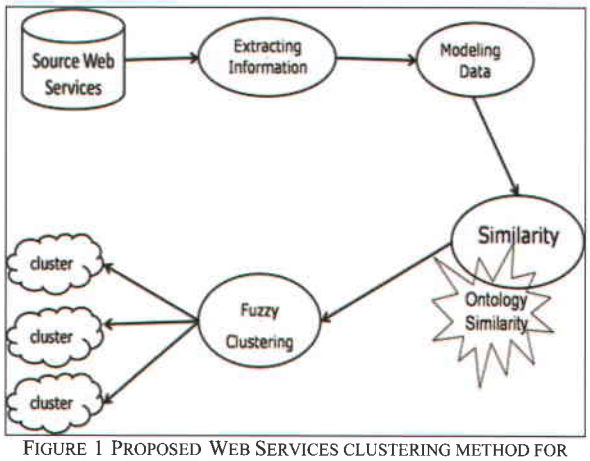

FIGURE 1 PROPOSED WEB SERVICES CLUSTERING METHOD FOR ASSISTING DISCOVERY PROCESS

degree than points near the center of cluster.

The vector will be giving useful data in retrieving more accurate ranked list of web services as results for a user's query without consuming any computing time at the response time. In the clustering algorithm, we find representative objects near the calculated center, called medoids, in clusters. For do this, we model fuzzy k-medoid clustering by k-means clustering algorithm. As the result, there are some clusters with semantic similar web service members.

Ontologies are proposed as means to address semantic heterogeneity among web services. They are used to provide meta-data for effective manipulation of available information including discovering information sources and reasoning about their capabilities. In the context of Web services, ontologies promise to take interoperability a step further by providing rich description and modeling of services properties. As mentioned above, we offer involving semantic in calculating similarity measure of clustering. For do this, before comparing two ontology-base services and calculating similarity between them, we consider their used ontologies. For this purpose, we have profited an ontology matching algorithm [21] to achieve relations between the ontology elements and we assign a score to each similar ontology elements which denote the degree of similarity between them. The similarity between two ontology is calculated by Equation (1, 2, 3) using the score. Then the overall similarity of two ontologies is computed by integrating result of the proposed equation.

$$similarity(O_1,O_2)=\frac{\sum(score * w)}{\sum obj\ O_1+\sum obj\ O_2} \quad (1)$$

$$similarity(O_1,O_2)= \frac{\sum(score * w)}{(\sum obj\ O_1+\sum obj\ O_2)-(\sum obj\ O_1\cap\sum obj\ O_2)} \quad (2)$$

$$similarity(O_1,O_2)=\frac{\sum(score_c * w)}{\sum concept\ O_1+\sum concept\ O_2}+\frac{\sum(score_p * w)}{\sum property\ O_1+\sum property\ O_2}+\frac{\sum(score_i * w)}{\sum individual\ O_1+\sum individual\ O_2} \quad (3)$$

At these equations, w denotes the weight of different objects at ontology. Also *object*, *concept*, *property* and *individual* respectively is the number of all objects, concepts, properties and individuals in a defined ontology. One of the main benefits of describing services with ontology is that discoverers can have access to an unambiguous definition of each part of a web service (e.g., inputs, outputs, operations, etc.). Additionally, in ontology-based semantic web, the software can automate service discovery, verification and monitoring service properties due to the determination of its purpose.

With each web service, there is an associated WSDL file describing its functionality and interface. We have selected the following sections as important parts of our approach:

- Name and text description,
- Annotations: A web service may be annotated by ontology files or ontology elements,
- Operation descriptions, and
- Input/output descriptions

Fig. 2 shows an example of WSDL file and its parts.

Since different developers implement different services, and may use different coding conventions, it is necessary to preprocess WSDL documents before carrying out some operations. So each web service description is assumed to be similar to a document.

In the preprocessing stage, we remove the stop words and then obtain the word stem. We have a richer stop word collection that is specialized for web service domain. This means that the collection contains words that are solely stop words in the web service domain. This specialization makes it possible to achieve more refined words and better results. Another element in this step is word resolution which is performed by separating distinct words in each phrase (for example, phrase "BookAuthorService" converts to "Book-Author-Service"). This type of phrase is seen more in web service descriptions so the word resolution preprocessing step is expected to have a favorable effect on the final results.

In the next step, we extract preprocessed words from web service description as published. The word extraction process is more different from a usual document. The main difference is that we have extracted words with their position in the web service description instead of a bag of identical words. We

combination of text mining techniques for bridging syntactic differences of Web service descriptions uses the classification approach that supports WSQBE search space reduction and techniques for easing query generation. The classifier exploits machine learning techniques to learn from available services, which have been previously categorized and published in UDDI registries. Also this method provides a query language for representing user needs. Namely, it is based on easing query specification and returning a short list of web services. In essence, by applying the idea of Query-by-example, they propose to make easier the task of defining a query for discovering Web services. Their discovery approach cornerstone is that consumers partially know the descriptions of the web services they want. Also, it uses no semantic for web service description and its efficiency depends on the web service developers who adopt best practices for describing and documenting services.

[7] Has described the algorithms underlying the Woogle search engine for web services. Woogle supports similarity search for web services, such as finding similar web service operations and finding operations that compose with a given one. The underlying algorithm is based on clustering algorithm that groups parameters names in the collection of web services operations into semantically meaningful concepts. These concepts are leveraged to determine similarity of inputs or outputs of web service operations. This method uses no annotation for web services and they attempt to provide semantic for parameters by relying on the information of WSDL and UDDI. So the clustering is accomplished at the operation level. This clustering method have just focused on the operation matching and input/output matching. Their clustering algorithm is a refinement of agglomerative clustering. The concept clustering is based on association rule by exploiting conditional probabilities of occurrence. This supplement semantic in proposed method for finding similarity between web services has improved the precision and recall. Their engine, however, does not adequately consider data types, which usually reveal important information about the functionalities of Web services.

[19][20] Have proposed a novel technique to mine WSDL documents and cluster them into functionally similar web service groups. A search engine's crawler crawls WSDL documents from the internet and applies off-line the proposed clustering approach to group similar functionally services. WDSL documents are not downloaded; instead, the contents are read directly from the WSDL document URI. So all words have extracted from WSDL document to the vectors. These vectors are clustered into two groups by using the Yahoo search engine to distinguish between function words and content words. For measuring similarity of services, four and five features have been extracted from WSDL document respectively by [19], [20] and special measures are applied for each one and the final similarity is achieved by integrating the partial similarity of each feature.

[19] Has utilized the tree-traversing ant algorithm for clustering the web services similar to its content words clustering during features mining.

[20] Has used the Quality Threshold (QT) clustering algorithm to cluster similar Web services based on the similarity features. It is necessary to mention that they perform a manual classification of the WSDL documents to serve as a comparison point for the clustering algorithms, so the assessment depend on who does the classification. Because of the similar functionalities, [19] name such service clusters as homogeneous service communities. Theoretically, these papers introduced the use of text mining techniques to effectively separate content words from function words, and a spreading activation inspired algorithm for cluster selection. Also the difference between the papers is similarity measures used for various parts.

We try to extract and integrate different aspects of web service description which have useful information and the proposed approach clusters web services in order to reduce the search space and improve query matching.

III. Proposed Approach

Web service discovery is the process of finding suitable services for a request by applying a matchmaker. The basic requirements of this process are: service description, service publishing, and a description of the requester's needs. A suitable discovery approach is one that retrieves required services at a reasonable time and with accepted quality.

It is possible to discover web services based on the parameters entered in the UDDI registry at the publishing phase. However, the registry does not facilitate the matching of web services which provide similar functionality.

Essentially, the idea is to improve the precision of retrieved services in response to user query and to perform web service discovery process in desirable time. Therefore, our objective is to reduce search space in the run time without degrading the precision of the results.

Our current research effort goes in two directions. On the one hand, in the process of intelligent grouping of web services, clustering has been used to create a more relevant set of services due to reducing search space. On the other hand, semantic technologies which are gaining wider use in web applications, typically has been involved in processing of services supported by ontologies. Fig. 1 shows the main steps of our fuzzy clustering approach.

By applying a fuzzy clustering technique in our proposed approach, it brings about some clusters as well as a belonging probability vector assigned for each web service. For each point x, the vector includes coefficient giving the degree of belonging to clusters per a cluster, as in fuzzy logic, rather than belonging completely to just one cluster. Thus, points on the edge of a cluster may be in the cluster to a lesser

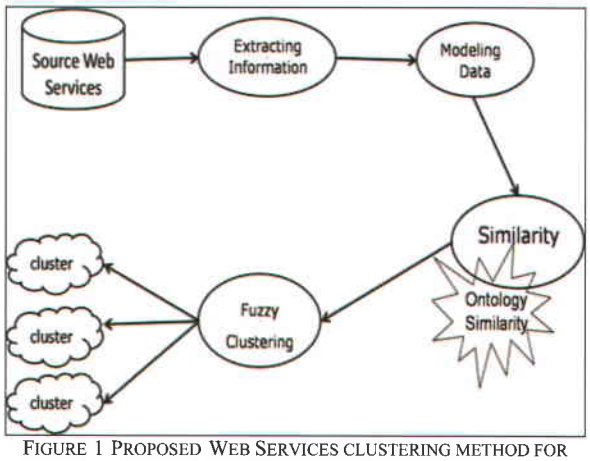

FIGURE 1 PROPOSED WEB SERVICES CLUSTERING METHOD FOR ASSISTING DISCOVERY PROCESS

degree than points near the center of cluster.

The vector will be giving useful data in retrieving more accurate ranked list of web services as results for a user's query without consuming any computing time at the response time. In the clustering algorithm, we find representative objects near the calculated center, called medoids, in clusters. For do this, we model fuzzy k-medoid clustering by k-means clustering algorithm. As the result, there are some clusters with semantic similar web service members.

Ontologies are proposed as means to address semantic heterogeneity among web services. They are used to provide meta-data for effective manipulation of available information including discovering information sources and reasoning about their capabilities. In the context of Web services, ontologies promise to take interoperability a step further by providing rich description and modeling of services properties. As mentioned above, we offer involving semantic in calculating similarity measure of clustering. For do this, before comparing two ontology-base services and calculating similarity between them, we consider their used ontologies. For this purpose, we have profited an ontology matching algorithm [21] to achieve relations between the ontology elements and we assign a score to each similar ontology elements which denote the degree of similarity between them. The similarity between two ontology is calculated by Equation (1, 2, 3) using the score. Then the overall similarity of two ontologies is computed by integrating result of the proposed equation.

$$similarity(O_1,O_2)=\frac{\sum(score * w)}{\sum obj\ O_1+\sum obj\ O_2} \quad (1)$$

$$similarity(O_1,O_2)= \frac{\sum(score * w)}{(\sum obj\ O_1+\sum obj\ O_2)-(\sum obj\ O_1\cap\sum obj\ O_2)} \quad (2)$$

$$similarity(O_1,O_2)=\frac{\sum(score_c * w)}{\sum concept\ O_1+\sum concept\ O_2}+\frac{\sum(score_p * w)}{\sum property\ O_1+\sum property\ O_2}+\frac{\sum(score_i * w)}{\sum individual\ O_1+\sum individual\ O_2} \quad (3)$$

At these equations, w denotes the weight of different objects at ontology. Also *object*, *concept*, *property* and *individual* respectively is the number of all objects, concepts, properties and individuals in a defined ontology. One of the main benefits of describing services with ontology is that discoverers can have access to an unambiguous definition of each part of a web service (e.g., inputs, outputs, operations, etc.). Additionally, in ontology-based semantic web, the software can automate service discovery, verification and monitoring service properties due to the determination of its purpose.

With each web service, there is an associated WSDL file describing its functionality and interface. We have selected the following sections as important parts of our approach:

- Name and text description,
- Annotations: A web service may be annotated by ontology files or ontology elements,
- Operation descriptions, and
- Input/output descriptions

Fig. 2 shows an example of WSDL file and its parts.

Since different developers implement different services, and may use different coding conventions, it is necessary to preprocess WSDL documents before carrying out some operations. So each web service description is assumed to be similar to a document.

In the preprocessing stage, we remove the stop words and then obtain the word stem. We have a richer stop word collection that is specialized for web service domain. This means that the collection contains words that are solely stop words in the web service domain. This specialization makes it possible to achieve more refined words and better results. Another element in this step is word resolution which is performed by separating distinct words in each phrase (for example, phrase "BookAuthorService" converts to "Book-Author-Service"). This type of phrase is seen more in web service descriptions so the word resolution preprocessing step is expected to have a favorable effect on the final results.

In the next step, we extract preprocessed words from web service description as published. The word extraction process is more different from a usual document. The main difference is that we have extracted words with their position in the web service description instead of a bag of identical words. We

FIGURE 2 WEB SERVICE DESCRIPTION DOCUMENT

have exploited the XML-structure of web service description [22] and the position is determined based on the tree structure. This means that we obtain some sub trees for each word and the words are only meaningful in their assigned sub trees. This structure causes differentiation between "book" in the input parameter and "book" in the output parameter. So we call each word and its position pair as positional-term and denote it by *pt(p; t)*.

At this stage, the input data is prepared for computing the weight. The weight of each word determines the importance of it within the test data set. We have used the TF-IDF measure because some studies show that it is more suitable for weighting words in web service descriptions and that this method outperforms other approaches in most cases. Formally:

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,i}} \tag{4}$$

$$idf_i = \log \frac{|WS|}{|\{ws : pt_i \in ws\}|} \tag{5}$$

$$weight(ws_j, t, pt_i) = tf_{i,j} * idf_i \tag{6}$$

Given a vector space model, the service vectors are presented by (ws1;ws2; …;wsn) where xi = (xi1; xi2; …; xid) represents the weighted positional-terms for wsi, n denotes the total number of services and xij is the normalized frequency of the jth positional-term in the service.

Clustering methods identify objects with similar characteristics in a data set and form groups of similar objects as a cluster. There are other similarity measures for finding the nearest neighbors. Cosine measure is widely used and it is superior to other similarity metrics in terms of retrieval effectiveness [23]. We have used a variant of the cosine measure (Equation 7) based on the XML-tree structure [22] for computing final score similarity. Formally:

$$similarity(ws_1, ws_2) = \sum_{pt_k \in B} \sum_{pt_l \in B} match(pt_k, pt_l) \sum_{t \in V} \frac{weight(ws_1, t, pt_k) * weight(ws_2, t, pt_l)}{|WS_1||WS_2|} \tag{7}$$

Where

$$|WS| = \sqrt{\sum_{pt \in B, t \in V} weight^2(ws, t, pt)} \tag{8}$$

$$match(pt_k, pt_l) = \begin{cases} 0 & \text{if } pt_k \text{ does not match } pt_l \\ \alpha \frac{1+|pt_k|}{1+|pt_l|} & \text{if } pt_k \text{ matches } pt_l \end{cases} \tag{9}$$

B: positional-terms set
Ptk: a positional-term in ws1
Ptl: a positional-term in ws2
V: terms set

Equation (9) compares two positional-terms. When they have no matching, it returns 0 but if they have any matching, the returned number is based on α factor which indicates the extent of matching, i.e. if they match completely, α is equal to 1.

By applying Equation (7) to our web service vector space model, semantic similarity between web services is obtained. And In the calculating similarity, where vector feature is about ontology, ontology matching equations are used.

Next, the fuzzy clustering method divides the space into subspaces, known as clusters which {C_1; C_2; …; C_k} denote the *k* clusters of services. A cluster is centered on an average vector, known as the center. Our fuzzy clustering method has a different centroid meaning. Due to our special modeling of web services, we find that avoiding a dummy center may lead to an increase in the result.

Our proposed clustering method has obtained clusters with highly correlation because the similarity measure is based on composition of service structure and semantics. Fuzzy k-means algorithm minimizes intra-cluster variance as well, but has the same problems as k-means; the algorithm does not ensure that it converges to an optimal solution; because the initial centroids (the initial values for matrix U) are randomly initialized so the minimum is a local minimum. It should mention that the number of clusters reported by FCM is less or equal to k (there is a possibility to obtain empty clusters). Also fuzzy clustering organizes spherical clusters of approximately the same size.

IV. Experimental Result

Web service clustering is about discovering novel, interesting and useful patterns from structural-term which have extracted from services. Web services are approximately large dimensional data elements which are not the same as documents and web service clustering requires a preprocessing task to convert the raw data into valuable ones.

We have implemented the modules of our described algorithm in Java. The experimental results of clustering are provided for data sets with 20, 50, 100 and 200 semantic web services which have been selected randomly from the SAWSDL-TC1 [24].

The collection is intended to support the evaluation of the performance of SAWSDL service matchmaking algorithms. SAWSDL-TC has been semi-automatically derived from OWLS-TC2.2. SAWSDL-TC1 provides 894 semantic web services written in SAWSDL (for WSDL1.1) from 7 domains (education, medical care, food, travel, communication, economy and weapon). The concepts used in the service input/output parts refer to 26 defined ontologies (.owl files) in the collection that all of them are included in our data set.

Here, the performance of the proposed algorithm has been evaluated in terms of the Minkowski score [25], V_{PE} and V_{PC} cluster validity index [26]. We show result of proposed clustering method for a set of n web services as an $n * n$ matrix Fm, where:

$$Fm_{i,j}=\begin{cases}1 & \text{if service } i \text{ and } j \text{ are in the same cluster}\\ 0 & \text{otherwise}\end{cases} \quad (10)$$

Matrix Jm is the reference clustering result and Minkowski score (MS) is defined as:

$$MS(Jm,Fm)=\frac{|Jm-Fm|}{|Jm|} \quad (11)$$

Where

$$|Jm|=\sqrt{\sum_i\sum_j Jm_{i,j}} \quad (12)$$

The score is the normalized distance between the two matrices. Lower Minkowski score implies better clustering solution, and a perfect solution will have a score zero.

Table 1 shows the MS for the algorithm. As can be inferred from the table, the clustering algorithm is performed approximately well at the data sets and whatever the number of services increase, the proposed clustering method performs better.

V_{PC} Bezdek Partition Coefficient and V_{PE} Partition Entropy are defined as below:

$$V_{PC}(U)=\frac{\sum_{i=1}^{n}\sum_{j=1}^{k}\mu_{i,j}^2}{n} \quad (13)$$

$$V_{PE}(U)=\frac{-1}{n}\sum_{i=1}^{n}\sum_{j=1}^{k}\mu_{i,j}\log\mu_{i,j} \quad (14)$$

$\mu_{i,j}$ is belonging degree of service i to cluster j. Table 2 shows the V_{PC} and V_{PE} index for the clustering algorithm. The result shows that the proposed clustering method in k = 5 has better behavior. These two indexes essentially measure the distance U from being crisp. The results demonstrate the proposed clustering approach is self-proved.

One of the objectives of our proposed approach is grouping similar web services by applying clustering technique as well as reducing the search space. Although this reduction is more profitable in discovery relevant services to user query but it would deduct accuracy of result by searching in a candidate cluster (one or two cluster). So we have decided to apply fuzzy clustering because objects on the boundaries between several classes are not forced to fully belong to one of the clusters. Membership degrees between zero and one are used in fuzzy clustering instead of crisp assignments of the data to clusters that indicate their partial membership.

Therefore, by applying the clustering method on the web service data set, we reduce search space to one or two cluster space at the discovery process on run time. While a cluster space is smaller than the whole space, search time is reduced significantly and this would be a valuable phase at web service discovery process.

While web services technology has clearly influenced positively the potential of the web infrastructure by providing programmatic access to information and services. Semantic web services are emerging as a promising technology for the effective automation of services discovery, combination, and management. So this approach by inclusion ontology provides automation support to facilitate effective discovery, combination, and management of services. Furthermore, the possibility to calculate the semantic distance between two sets of concepts (two ontologies) finds a natural application in the agent field, in particular for improving the agents that serve their human owners, by exploring the semantic web and looking for useful services.

Table 1 Evaluation of Fuzzy Clustering by Minkowski Score

Data Set	*# of Services*	*# of Unique Terms*	*Avg. Service. Length*	*MS*
DS1	20	1072	61	0.375604
DS2	50	1959	64	0.381854
DS3	100	2843	72	0.409645
DS4	200	5278	74	0.302017

Table 2 Evaluation of Fuzzy Clustering by Validity Index

Number of cluster	*VPC*	*VPE*
3	0.67	0.21
4	0.71	0.18
5	0.75	0.15
6	0.63	0.20
7	0.52	0.25
8	0.49	0.28
9	0.42	0.31
10	0.40	0.36

V. Conclusion and Future Work

As the number of web services grows, the problem of searching for relevant services becomes more critical in the discovery process. This paper proposes a new algorithm for fuzzy K-medoids web services clustering which runs like the K-means. The proposed algorithm calculates the distance matrix once and uses it for finding new medoids at every iterative step. Also we present a new semantic similarity measure in the fuzzy clustering method for web services and describe the algorithm in detail. Our algorithm exploits the XML-structure of WSDL file and ontology for supporting semantic in our proposed measure.

One of our contributions is compatibility with the current UDDI registry infrastructure and it imposes no modifications. In the other words, we have discussed how to introduce the method of embodying ontologies into unsupervised learning in order to consider the term semantics in the preview of linguistics. So the fuzzy web services clustering scheme is enhanced with semantic analysis mechanism.

We test our algorithm on a data set with 20, 50, 100 and 200 web services on seven domains due to time limit for preprocessing. However, we intend to extend our sample by experiment our algorithm on a example, our algorithm detects web services which sound diverse but are semantically similar. Fuzzy clustering is representative for the method of overlapping clustering. It uses fuzzy sets to cluster data, so each point may belong to two or more clusters with different degrees of membership. It appears that the proposed method would improve the web services larger data set with services on variant domains. The experimental results show that our method significantly improves finding semantic similarity between services and perform well overall. For discovery process by reducing search space and time and increasing the precision of results. On the other hand, at the run time it needs not to be investigating all of data. The resulting web service semantic-based partition improves services understanding and browsing and reveals its internal structure.

Since developing web service technology, the main trend is moving toward when without human intervention; agent can discover and select required services to configure systems. Our proposed approach by exploring semantics and automatically looking for useful services is applicable in particular for improving agent. Furthermore, one of the applications of this method is web service mashups because of grouping relevant services based on semantics.

It also provides a good starting point for the development of practical service search engines. The web service clusters of our approach provide systems with access to redundant services in the case of a failure. So fault-tolerant capabilities support the search engine.

A future extension to our work also includes expanding our approach to an automatic web service discovery. We project to develop a two-step web service discovery which first step is done by exploiting proposed clustering method. On the other hand, after semantically organize potential similar services into clusters; it is possible to retrieve automatically relevant services with user query.

As a future work, it has the potential to evaluate the proposed method of calculating the mutual information similarity of objects in the ontology to create the ontology in different field or a complete common ontology. In the other word, it would help in the ontology integration.

Acknowledgment

The authors would like to grateful Iran Telecom Research Center (ITRC) for its financial support.

References

[1] S. Vaughan-Nichols, "Web services: Beyond the hype," Computer, vol. 35, no. 2, pp. 18–21, 2002.

[2] M. Huhns and M. Singh, "Service-oriented computing: Key concepts and principles," Internet Computing, IEEE, vol. 9, no. 1, pp. 75–81, 2005.

[3] J. Han and M. Kamber, Data mining: concepts and techniques. Morgan Kaufmann, 2006.

[4] "Grand central. http://www.grandcentral.com/directory/.,"

[5] "Salcentral. http://www.salcentral.com/.,"

[6] "Web service list. http://www.webservicelist.com/."

[7] X. Dong, A. Halevy, J. Madhavan, E. Nemes, and J. Zhang, "Similarity search for web services," in Proceedings of the Thirtieth international conference on Very large data bases-Volume 30. VLDB Endowment, 2004, p. 383.

[8] Y. Wang and E. Stroulia, "Semantic structure matching for assessing web-service similarity," Service-Oriented Computing- ICSOC 2003, pp. 194–207, 2003.

[9] K. Lee, M. Lee, Y. Hwang, and K. Lee, "A Framework for XML Web Services Retrieval with Ranking," in Multimedia and Ubiquitous Engineering, 2007. MUE'07. International Conference on. IEEE, 2007, pp. 773–778.

[10] G. Salton, A. Wong, and C. Yang, "A vector space model for automatic indexing," Communications of the ACM, vol. 18, no. 11, p. 620, 1975.

[11] N. Shadbolt, W. Hall, and T. Berners-Lee, "The semantic web revisited," Intelligent Systems, IEEE, vol. 21, no. 3, pp. 96-101, 2006.

[12] S. McIlraith and D. Martin, "Bringing semantics to web services," Intelligent Systems, IEEE, vol. 18, no. 1, pp. 90-93, 2005.

[13] R. Karimpour and F. Taghiyareh, "Conceptual discovery of Web services using WordNet," in Services Computing Conference, 2009. APSCC 2009. IEEE Asia-Pacific. IEEE, 2010, pp. 440–444.

[14] J. Shen, G. Grossmann, Y. Yang, M. Stumptner, M. Schrefl, and T. Reiter, "Analysis of business process integration in Web service context," Future Generation Computer Systems, vol. 23, no. 3, pp. 283–294, 2007.

[15] M. Paolucci and K. Sycara, "Autonomous semantic web services," IEEE Internet Computing, vol. 7, no. 5, pp. 34–41, 2003.

[16] C. Mateos, M. Crasso, A. Zunino, and M. Campo, "Supporting ontology-based semantic matching of Web services in MoviLog," Advances in Artificial Intelligence-IBERAMIASBIA 2006, pp. 390–399, 2006.

[17] G. Vega-Gorgojo, M. Bote-Lorenzo, E. G´omez-S´anchez, Y. Dimitriadis, and J. Asensio-P´erez, "A semantic approach to discovering learning services in grid-based collaborative systems," Future Generation Computer Systems, vol. 22, no. 6, pp. 709–719, 2006.

[18] M. Crasso, A. Zunino, and M. Campo, "Easy web service discovery: A query-by-example approach," Science of Computer Programming, vol. 71, no. 2, pp. 144–164, 2008.

[19] W. Liu and W. Wong, "Web service clustering using text mining techniques," International Journal of Agent-Oriented Software Engineering, vol. 3, no. 1, pp. 6–26, 2009.

[20] K. Elgazzar, A. Hassan, and P. Martin, "Clustering WSDL Documents to Bootstrap the Discovery of Web Services," in 2010 IEEE International Conference on Web Services. IEEE, 2010, pp. 147–154.

[21] M. Ehrig, S. Staab, and Y. Sure, "Framework for Ontology Alignment and Mapping," 2005.

[22] C. Manning, P. Raghavan, and H. Schtze, An introduction to information retrieval. Cambridge University Press, 2008.

[23] M. Kim and K. Choi, "A comparison of collocation-based similarity measures in query expansion," Information Processing & Management, vol. 35, no. 1, pp. 19–30, 1999.

[24] "http://projects.semwebcentral.org/projects/sawsdl-tc/."

[25] A. Ben-Hur and I. Guyon, "Detecting stable clusters using principal component analysis in methods in molecular biology," Humana Press, MJ, pp. 159–182, 2003.

[26] J. Bezdek, "Cluster validity with fuzzy sets," Cybernetics and Systems, vol. 3, no. 3, pp. 58–73, 1973.

Nayereh Gholamzadeh is currently a M.Sc. student of Software Computer Engineering at the Electrical and Computer Engineering Dept. University of Tehran, Tehran, Iran. She received her B.Sc. in Computer Engineering - Software field from Ferdowsi University of Mashhad in 2006. Her research interests include Semantic Web services, Web intelligence and Data Mining Technique.

Fattaneh Taghiyareh is Assistant Professor of Computer Engineering-Software & Information Technology- at the University of Tehran, where she has served since 2001. She received a Ph.D. in Computer Engineering- Parallel Algorithm Processing- from the Tokyo Institute of Technology in 2000. Her research interests include Human-Centered Computing applied to Learning Management Systems and based on Multi-Agent Systems. She is currently working on "Group Collaborative Learning" as well as "Educational Data Mining (EDM)" with a pedagogical approach. In addition, Web Services are also another concern of her research to solve the composition problem from an ontology-based perspective. She was ex manager of the Department of Information Technology at the School of Electrical & Computer Engineering, and former Director of Technology Incubator, the University of Tehran. She is a member of Editorial Board of International Journal of Information & Communication Technology. Currently she is serving both Software Engineering and Information Technology departments as a member and she is supervising more than 10 post-graduate students of the Multi-Agent System Lab and HCC Lab has founded by her. Moreover, she is supervising IT Foundation Lab which was established also by her 5 years ago. She is the author of more than 50 conference and journal papers.

Azadeh Shakery is an Assistant Professor of Electrical and Computer Engineering at the College of Engineering, University of Tehran. She received her Ph.D. in Computer Science from University of Illinois at Urbana-Champaign in 2008. Her research interests include text information management, information retrieval, and text and data mining.