An Approach to Computation of Similarity, Inter-Cluster Distance and Selection of Threshold for Service Discovery using Clusters

Chellammal Surianarayanan and Gopinath Ganapathy, *Member, IEEE*

Abstract— Meeting out similarity demands of clients, selection of threshold and computation of inter-cluster distance (ICD) are difficult while clustering. Hierarchical agglomerative clustering based approach is proposed for service discovery including two similarity models viz., Output Similarity Model (OSM) and Total Similarity Model (TSM) with additional levels for Degree of Match (DoM). The OSM which computes similarity between services using solely the outputs of services is proposed while clustering services to eliminate irrelevancy completely. The TSM which computes similarity between services using both inputs and outputs of services is proposed while discovering matched services of a given query. The work justifies the 'complete linkage' as suitable method for computing ICD. It selects *threshold-ICD* in terms of DoM without altering the similarity demands of clients and ensures rightness of clusters. The computation time of discovery using clusters is found to be faster (7.32 against 170.59 seconds) than that of sequential method.

Index Terms— similarity models, clustering of services, service discovery using clusters, optimization of discovery, semantic service discovery.

1 Introduction

Implementation of complex business requirements needs automatic discovery of multiple services from different domains and their composition in short time[1-2]. Explicit semantic descriptions of services [3-7] have brought maximal automation and dynamism with sufficient accuracy into service discovery [8]. Semantics based discovery is time consuming due to the usage of semantic reasoning which detects semantic relations through different Degree of Match (DoM)s, namely, *exact, plug-in, subsumes and fail* [9] that might exist between queried and available concepts. It is certainly essential to reduce the time taken for discovery as crucial business processes involve many services from different domains. Clustering of services will reduce the time [10] by grouping similar services into clusters such as *banking, education, weather*, *travel, financial*, etc. For the query 'findTemperature', it searches only *weather* cluster, ignoring other clusters.

Similarity computation which can be syntactic or semantic is pre-requisite for clustering. Semantics based similarity is promising for bringing accuracy, automation and dynamism into discovery. The first aspect of this work is to introduce additional levels of DoMs while computing semantic similarity. Existing semantic approaches [8-9] compute similarity between a queried concept and an available concept through 4 different levels of DoM, namely, *exact, plugin, subsumes and fail*. But these 4 levels detect semantic relatedness of a concept only with respect to its super/sub classes whereas additional levels of DoM such as sibling, grandparent, common-children, etc., are required to meet out most of the similarity demands of clients.

Further, the similarity demands of clients are disparate. Some applications have strict similarity needs than the others. Suppose a need of a client with 'head injury' is *emergency-ambulance*. This query can only be fulfilled by *exact* match viz., *Emergency-ambulance* class(Fig. 1) as the client has strict similarity demand. Suppose a need of a client is *Car* to visit a physician for general checkup. This query can be fulfilled by *exact, plugin,* and *sibling(i.e. Ford, Hyundai, Bronco, Explorer, Ambulance, Emergency-ambulance, Patient-transport-ambulance and Bus*) matches. If *Car (i.e. exact match)* is not available, sibling or plugin matches will satisfy the need. Here similarity demand is not strict as in the previous case. Hence, there is a need to introduce additional levels of DoM such as *indirect-subsumes, indirect-plugin, sibling, partial-parent, grandparent, common-children* and *common-grandchildren*.

The second aspect of this work is to enhance relevancy by prioritizing the service features while computing similarity. Firstly, during discovery, ensure that the published services have the queried outputs. Secondly, the inputs of services can be taken for further matching. But, some approaches such as [10-11] do not completely eliminate irrelevancy because they compute service similarity for clustering as weighted sum of similarities by different

- *Chellammal Surianarayanan is with the Department of Computer Science, Bharathidasan University Constituent College for women, Orathanadu, 614625, India. E-mail:chelsrsd@rediffmail.com.*
- *Gopinath Ganapathy is with the School of Computer Science and Engineering, Bharathidasan University, Tiruchirappalli, 620024, India. E-mail: gganapathy@gmail.com.*

Manuscript received (insert date of submission if desired). Please note that all acknowledgments should be placed at the end of the paper, before the bibliography.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2015.2399301, IEEE Transactions on Services Computing

features viz., inputs, outputs and description without any prioritization.

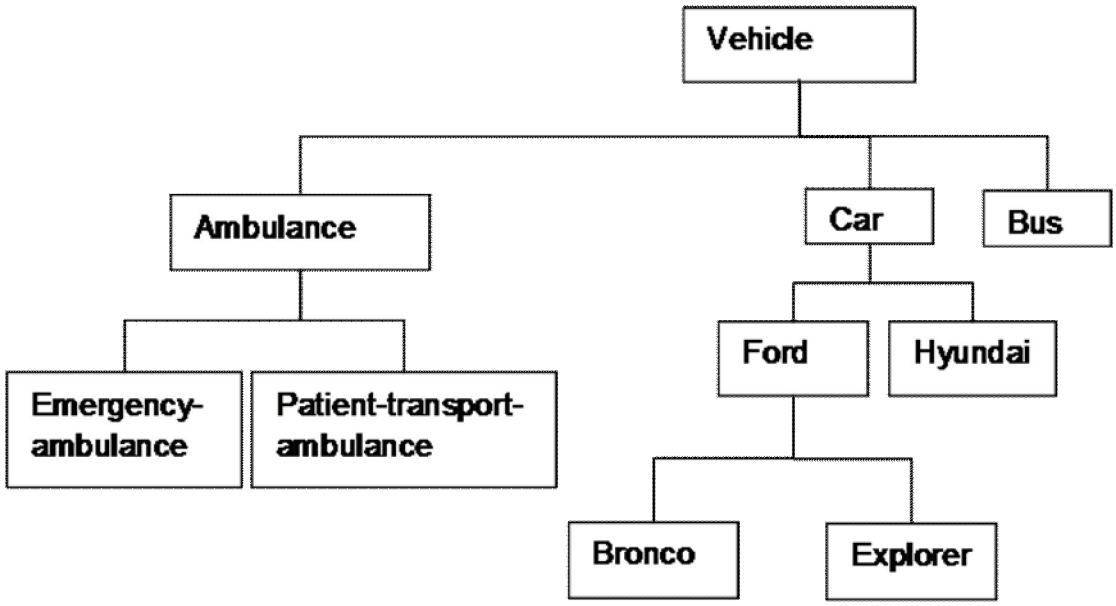

Fig.1 Fragment of sample ontology

In these approaches the clustering solution may become incorrect as per [12] from the perspective of eliminating irrelevancy. In order to eliminate irrelevancy completely, it is essential to cluster services through output similarity rather than the total similarity. Consider two services, *A* and *B* as in Fig. 2.

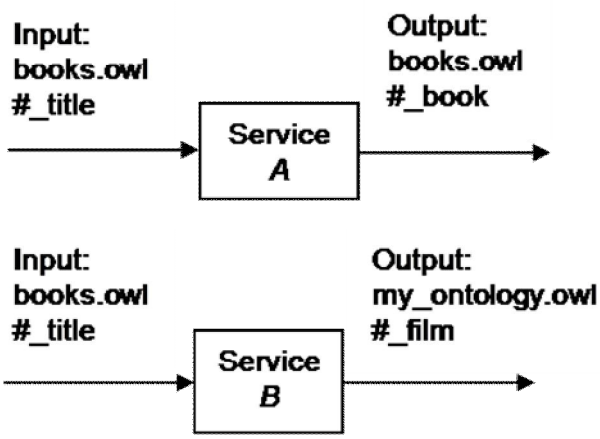

Fig.2 Service having same inputs but different outputs

The services have same input but different output. When clustering is performed based on total similarity, then the services *A* and *B* will be clustered together via input similarity. But when clustering is based purely on output similarity, then services *A* and *B* will not be clustered as their outputs are different. Similarly, services having different outputs may get clustered via similar description. When services are clustered by total similarity, the irrelevancy of a query is not removed completely as the *relevant cluster* (the cluster which is most similar to the query) may contain irrelevant services (services having same inputs/description but different outputs). Alternately, clustering services purely by outputs will efficiently eliminate irrelevancy.

The third aspect of the work is to identify alternate clustering algorithm that produces spontaneous clustering solution. Existing approaches such as [13] partition services into known number of categories (K), where the rightness of clustering is altered by the value of K. Though there are a few approaches such as [14] uses hierarchical clustering it does not focus on on two essential aspects of practical importance, namely, choosing a suitable *method of computing inter-cluster distance(ICD)* and selecting a right value for *threshold inter-cluster distance(threshold-ICD)* which is used as *stopping criterion for clustering. Also, it is essential to ensure that the given similarity demands of clients are satisfied while computing ICD and selecting threshold-ICD.*

1.1 Objectives

This work addresses the aspects of cluster based service discovery with the following objectives.

- To introduce additional levels of DoM in order to satisfy many of the similarity demands of clients.
- To propose two different similarity models, one for clustering viz., Output Similarity Model (OSM) and the other for discovery, viz., Total Similarity Model (TSM). OSM computes similarity between services based only on the outputs of services using the additionally defined levels of DoM. TSM computes similarity between services based on both inputs and outputs of services using the additionally defined levels of DoM.
- To identify a suitable method for computing Inter-Cluster Distance (ICD) from the three standard methods, single, average and complete linkage while using hierarchical agglomerative clustering.
- To devise a method for selecting threshold-ICD without conflicting the similarity demands of clients.

- To study and compare the performance of service discovery using clusters against standard methods.

1.2 Contributions

This work contributes a few methodologies for service discovery using clusters. It employs hierarchical agglomerative clustering to cluster services as it provides more structural information about the clusters.

From methodological perspective two similarity models, OSM and TSM have been introduced. The work contributes additional levels of DoM as mentioned in Section 3.1 for computing semantic relations among service concepts. The advantage of additional levels of DoM is that it provides a better control for clients while specifying their required similarity demands. The work introduces a method for labeling clusters to identify the relevant cluster of the query easily. Also, it suggests a method to discover matched services for a query using clusters. It computes performance optimization achieved using clustering and evaluates its accuracy against standard method.

From evaluation perspective, the work performs an extensive series of experimentation to analyze the clustering solutions produced by different linkage metrics and evaluates the results using various evaluation measures.

Also, the work serves as a strategic model for developers working in service clustering.

Rest of the paper is organized as follows. Section 2 highlights the research works which handle cluster based discovery. Section 3 describes similarity models, clustering of services and discovery using those clusters. Section 4 describes experimental setup, test data, evaluation measures. Section 5 presents the results obtained using different methods of computing ICD and analyzes the results using different evaluation measures. Suitable method for computing ICD and method for selecting threshold-ICD are presented. Section 6 concludes the paper.

2 Related Work

A survey on service discovery using clusters is made. Web Service Description Language (WSDL) based approaches such as [15-17] are syntactic and have no intended semantics of services. This limits the accuracy of discovery. For example, a syntactic based method cannot relate meaningfully *zipcode* and *postalcode*. Also, they are inadequate for automatic discovery whereas the proposed work is semantics based one. Though the work [18] maps WSDL into a richer semantic representation and [19] uses WordNet similarity for discovery, these methods are applicable to only to WSDL services

Though semantics based approaches [8-9] facilitate automation and dynamism into service discovery, the levels of DoM defined by these approaches are insufficient to meet out the diversity in similarity demands of clients. The proposed work handles the diversity in similarity demands using additional levels of DoM (Section 3.1)

Further, features which are considered for computing similarity and their weights vary among approaches. For example, [11] considers 5 features, viz., description, OWL-S profile, WSDL, OWL-S process and OWL-S grounding with weights 0.1, 0.3, 0.2, 0.1 and 0.1 respectively. The work [18] considers description, OWL-S profile, and WSDL similarities with weights 0.5, 0.3 and 0.2 respectively whereas the relevancy of a query is primarily decided by the outputs. If a service does not produce outputs as required by the query, it becomes irrelevant to the query irrespective of any other feature. Approaches such as [11], [14], [18] may not completely eliminate irrelevancy as they do not prioritize the ouputs at first place. Another approach [10] also gives equal weight to inputs and outputs while computing semantic similarity. The uniqueness of proposed work is its prioritization of service features while computing similarity.

Approaches such as [20-23] are K-Means based where the value of K is mandatory. Also, in K-Means, the resulting clustering pattern is influenced by K. Specifying a right value for K is very difficult. Also, in K-Means, there is no provision to specify the required similarity among services within a single cluster. Alternately, the proposed work uses hierarchical agglomerative clustering which does not require the value of K to be specified.

In [24], the profile of most general service of a cluster is used as its label. This method may not find matched services if the query does not contain generic service terms. An alternate labeling method is proposed in this work.

In [25], graph theory based algorithm and in [26] genetic algorithm are used for cluster based discovery. The work[27]uses Chameleon algorithm to build K-Nearest Neighbour (K-NN) graph but it focusses on filtering functionally chosen services by non-functional properties of services. The work [28] provides support for visualization of clusters using spatial clustering. The work presented in [29] uses clustering technique for organizing the results returned by a search engine and not for service retrieval. In [30], a non-distance based self organizing clustering algorithm is proposed to overcome the problem of sample selection and sub-optimal choice of threshold. Alternately, the proposed approach handles the sub-optimal choice of threshold according to the similarity demands of service clients.

3 Proposed Approach

This section proposes an approach for service discovery using clusters with its basic concepts. This approach would cluster services having similar outputs using OSM. Each cluster is labeled with a string which is a concatenation of all outputs contained in it. Clustering and labeling are done offline to querying. For a query, the outputs of query are matched with label of each cluster and the cluster which is most similar to the query is identified as its relevant cluster. Matched services of the query are found out by comparing the inputs and outputs of the query with that of each service present in relevant cluster using TSM. Both similarity models find similarity between services with the help of additionally defined levels of DoM. In this work hierarchical clustering algorithm which gives more structural information about the hierarchy of sub clusters[31] is used.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2015.2399301, IEEE Transactions on Services Computing

The work suggests the following strategies for service discovery using clusters. The strategies include (i) Introduction of additional levels of DoM while finding semantic relations between a pair of concepts to meet out similarity demands of clients (ii) Two similarity computing models, namely, OSM which computes similarity between services based on outputs using additional levels of DoM for clustering services and TSM which computes similarity between services based on both inputs and outputs using additional levels of DoM for discovering matched services of a given query. It is proposed to employ OSM for clustering services as it ensures complete removal of irrelevancy. (iii) Selection of suitable method for computing ICD among three methods, single, average and complete linkage. (iv) Introduction of a new method for selecting threshold-ICD using levels of DoM.

Hierarchical agglomerative clustering starts by considering each service as a cluster and proceeds iteratively by merging the most similar pair of clusters. The clusters are merged using major inter-cluster linkage metrics, namely, single, average and complete linkage[27]. Of the three methods, complete linkage merges the most similar pair of clusters based on the similarity between farthest pair of services. In this case, if the farthest pair satisfies the given threshold-ICD it is implied that the similarity between any pair of services would also satisfy the given threshold-ICD. With this kind of merging, within a single cluster, the similarity between any two services will meet the given threshold-ICD. Hence, this feature of complete linkage is purposefully employed to meet the similarity demands of service clients.

Further, in order to ensure that the similarity demands of clients are preserved while specifying threshold-ICD, it is proposed to specify the threshold-ICD according to similarity demands of clients, using various levels of DoMs. For example, if the client's similarity demand(called *similarity_demand*) is *exact*, the client will specify its threshold_ICD as 1. Now, consider two clusters, C_1 and C_2 for merging. Let s_1 denote any service from C_1. Let s_2 denote any service from C_2. Let m and n denote the number of output parameters of s_1 and s_2 respectively. The clusters will be merged only when the similarity between s_1 and s_2 is at least the threshold_ICD. Here, threshold_ICD has to be calculated for a given pair of services according to

$$threshold\_ICD = \frac{(m+n) \times similarity\_demand}{2mn} \qquad (1)$$

In (1), numeral 2 is introduced in the denominator because the similarity between s_1 and s_2 is computed as the average of $OutSim(s_1, s_2)$ and $OutSim(s_2, s_1)$ as given in (4). Thus, this model recommends specifying threshold-ICD in terms of DoM and complete linkage for computing ICD as these two will ensure that the similarity demands of client are satisfied.

After having been clustered, each cluster is labeled. Next stage is querying against the clusters for discovering matched services where the outputs of the query are matched with the label of each cluster. The cluster which is most similar to the query is identified as the *relevant cluster*. As mentioned earlier, the matched services are discovered by matching each service present in the relevant cluster (*candidate service)* with the query using TSM.

The models introduced for computing similarity, clustering and discovering matched services are described in the subsequent sections.

3.1 Output Similarity Model

Let us consider two services, *A* and *B*. Let *A* contain *m* number of output parameters, denoted by $o_1^a, o_2^a, o_3^a ..., o_m^a$ and *B* contain *n* number of output parameters denoted by $o_1^b, o_2^b, o_3^b ..., o_n^b$ as in Fig. 3. To compute output similarity between *A* and *B*, each output parameter of *A* is matched with every output parameter of *B*. While matching, the parameters are tested for having any semantic relation between them. The semantic relation between parameters is expressed using various levels of DoM. We extend the conventional levels of DoM, namely, *exact, plug-in, subsumes* and *fail* with finer split as described below.

Let us consider two parameters, an i^{th} output parameter of service *A* denoted by o_i^a and j^{th} output parameter of service *B* denoted by o_j^b. Let $DoM(o_i^a, o_j^b)$ denote the DoM between o_i^a and o_j^b.

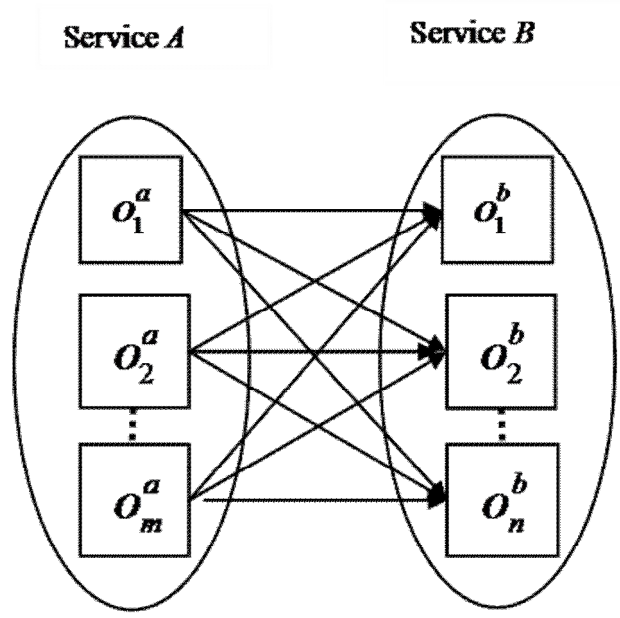

Fig. 3 Example services and their outputs

Exact: If o_i^a is equivalent to o_j^b, then the $DoM(o_i^a, o_j^b)$ is defined as exact.

Direct-plugin: If o_i^a is direct super class of o_j^b, then the $DoM(o_i^a, o_j^b)$ is defined as direct-plugin match.

Indirect-Plugin: If o_i^a is indirect super class of o_j^b, then the $DoM(o_i^a, o_j^b)$ is defined as indirect-plugin match.

Direct-subsumes: If o_i^a is direct sub class of o_j^b then the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2015.2399301, IEEE Transactions on Services Computing

$DoM(o_i^a, o_j^b)$ is defined as direct-subsumes match.

Indirect-subsumes: If o_i^a is an indirect sub class of o_j^b then $DoM(o_i^a, o_j^b)$ is defined as indirect-subsumes match.

Common-parent/Sibling: If all parents of o_i^a are same as that of o_j^b then the $DoM(o_i^a, o_j^b)$ is common-parent/sibling.

Partial-parent: If at least one parent of o_i^a is same as at least one parent of o_j^b then $DoM(o_i^a, o_j^b)$ is partial-parent.

Grandparent: If both o_i^a and o_j^b have at least one grandparent in common or if at least one grandparent of o_i^a is same as at least one parent of o_j^b or if at least one parent of o_i^a is same as at least one grandparent of o_j^b then the $DoM(o_i^a, o_j^b)$ is defined as grandparent.

Common-children: If both o_i^a and o_j^b have at least one child in common, then $DoM(o_i^a, o_j^b)$ is common-children.

Common-grandchildren: If both o_i^a and o_j^b have at least one grandchild in common, then $DoM(o_i^a, o_j^b)$ is defined as common-grandchildren.

Fail: If both o_i^a and o_j^b have none of the above semantic relations, then $DoM(o_i^a, o_j^b)$ is defined as fail.

In order to obtain useful information about similarity the levels of DoM are assigned with numerical equivalents by heuristics similar to Duygu Celik et al.[32], Mehdi Bayat et al.[33] and Syeda-Mahmood T et al.[34]. The levels of DoM proposed in this work along with numeric equivalents are compared with [32-34] as given in Table 1.

Let $Sim(A,B)_{OSM}$ denote the similarity between *A* and *B*. To compute the value of $Sim(A,B)_{OSM}$, each i^{th} output parameter in *A*, say, o_i^a is matched with every output parameter of *B* and DoM is computed for each pair of parameters formed by o_i^a with parameters of *B*. The maximum of DoMs of all possible pairs formed by o_i^a with all parameters of *B* will represent the individual parameter score contributed by o_i^a. The individual parameter score is computed for all outputs in *A* and the sum of all individual parameter scores represents the total output similarity score between *A* and *B*.

TABLE 1
Levels of DoM and their numerical equivalents

DoM levels as per [32-34]	DoM values as per [32]	DoM values as per [33]	DoM values as per [34]	DoM levels as per proposed method	DoM values as per proposed method
Exact	1	1	1	Exact	1
Plugin	0.75	0.75	0.5	Direct-plugin	0.9
Subsumes	0.5	0.5	0.5	Indirect-plugin	0.8
Dissimilar	0	0	0	Direct-subsume	0.7
				Indirect-subsume	0.6
				Common-parent	0.5
				Partial parent	0.4
				Grandparent	0.3
				Common-children	0.2
				Common-grandchildren	0.1
				Fail	0

In the above example *A* has *m* number of output parameters, $o_1^a, o_2^a, o_3^a ..., o_m^a$ and *B* has *n* number of output parameters $o_1^b, o_2^b, o_3^b, ..., o_n^b$. The possible pairs formed by o_1^a are $(o_1^a, o_1^b), (o_1^a, o_2^b), (o_1^a, o_3^b), ..., (o_1^a, o_n^b)$. The possible pairs formed by o_2^a are $(o_2^a, o_1^b), (o_2^a, o_2^b), (o_2^a, o_3^b), ..., (o_2^a, o_n^b)$. Similarly, the possible pairs formed by o_m^a are $(o_m^a, o_1^b), (o_m^a, o_2^b), (o_m^a, o_3^b), ..., (o_m^a, o_n^b)$. Now, the normalized output similarity between *A* and *B*, denoted by $OutSim(A,B)$ is given as

$$OutSim(A,B) = \frac{1}{m} \times [\max(Dom(o_1^a, o_1^b), Dom(o_1^a, o_2^b), ..., Dom(o_1^a, o_n^b)) + \max(Dom(o_2^a, o_1^b), Dom(o_2^a, o_2^b), ..., Dom(o_2^a, o_n^b)) + ... + \max(Dom(o_m^a, o_1^b), Dom(o_m^a, o_2^b), ..., Dom(o_m^a, o_n^b))] \quad (2)$$

Similarly, the normalized output similarity between *B* and *A* denoted by $OutSim(B,A)$ is given as:

$$OutSim(B,A) = \frac{1}{n} \times [\max(Dom(o_1^b, o_1^a), Dom(o_1^b, o_2^a), ..., Dom(o_1^b, o_m^1)) + \max(Dom(o_2^b, o_1^a), Dom(o_2^b, o_2^a), ..., Dom(o_2^b, o_m^a)) + ... + \max(Dom(o_n^b, o_1^a), Dom(o_n^b, o_2^a), ..., Dom(o_n^b, o_m^a))] \quad (3)$$

Now, the value of $Sim(A,B)_{OSM}$ is given as

$$Sim(A,B)_{OSM} = \frac{1}{2} \times (OutSim(A,B) + OutSim(B,A)) \quad (4)$$

3.2 Total Similarity Model

The similarity between two services, say, A and *B* denoted by $Sim(A,B)_{TSM}$ is given as

$$Sim(A,B)_{TSM} = 0.5 \times (Sim(A,B)_{OSM} + InputSim(A,B)) \quad (5)$$

In the above equation, $Sim(A,B)_{OSM}$ represents normalized output similarity between services which is computed using (4) and $InputSim(A,B)$ represents normalized input similarity between services. The values of $InputSim(A,B)$ is computed using

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2015.2399301, IEEE Transactions on Services Computing

$$InputSim(A,B) = 0.5 \times InSim(A,B) + InSim(B,A)) \quad (6)$$

The computation of $InSim(A,B)$ and $InSim(B,A)$ is similar to that of $OutSim(A,B)$ and $OutSim(B,A)$ respectively which is explained in the previous subsection.

3.3 Clustering of Services using OSM

Consider a service repository of *N* services. The similarity score among all possible pairs of services is computed using OSM. After computing pair-wise similarity, the values of dissimilarity (or distance) among services are computed using the following formula,

$$Dissim(s_1,s_2) = (1 - Sim(s_1,s_2)) \quad (7)$$

In (7), $Dissim(s_1,s_2)$ represents dissimilarity between s_1 and s_2. A $N \times N$ dissimilarity matrix is constructed to be used as input for hierarchical clustering algorithm. The algorithm starts by assigning each service to a cluster and initially each cluster contains just one service. The algorithm finds the most similar pair of clusters and merges them into a single cluster. It recomputes the similarities between new cluster and each one of the old clusters. The merging of clusters continues till the similarity between the clusters of most similar pair satisfies the given threshold-ICD.

Consider two clusters, C_1 and C_2. The formula for computing similarity between C_1 and C_2 denoted by $Sim(C_1,C_2)$ according to the standard methods, viz, single, complete and average linkage methods is given through (8), (9), and (10).

$$Sim(C_1,C_2) = \max\{sim(a,b) : a \in C_1, b \in C_2\} \quad (8)$$

$$Sim(C_1,C_2) = \min\{sim(a,b) : a \in C_1, b \in C_2\} \quad (9)$$

$$Sim(C_1,C_2) = \frac{1}{|C_1||C_2|} \sum_{a \in C_1} \sum_{b \in C_2} sim(a,b) \quad (10)$$

In the equations (8)-(10), a and b denote services that belong to clusters C_1 and C_2 respectively. Also in (10), $|C_1|$ and $|C_2|$ denote the number of services present in C_1 and C_2 respectively. In the proposed approach, clusters obtained using single, average and complete linkage for various threshold-ICDs are analyzed and a suitable method for computing ICD is suggested.

3.4 Service Discovery using TSM

Each cluster is labeled with all output parameters that are contained in that cluster. During discovery the cluster whose label (or characteristic representation) is most similar to the query is found out as the *relevant cluster* of the query.

Let *q* be the query containing a set of output parameters denoted by $\{o_i^q, 1 \le i \le p\}$. Let $\{(CR)_{C_i} | 1 \le i \le k\}$ be the characteristic representation of all clusters $C_1, C_2, C_3, ..., C_k$. The sub steps used for finding the relevant cluster of *q* denoted by $C(q)$ are given in Fig. 4. The services which are present in the relevant cluster are called *candidate services*. The input and output parameters of each candidate service are matched with that of query using TSM in order to discover matched services of the query.

Inputs: $\{(CR)_{C_i} | 1 \le i \le k\}$, $\{o_i^q, 1 \le i \le p\}$;
Output: $C(q)$
Sub step 1: $\max Sim = 0; i = 1; C(q) = \phi$
Sub step 2: *Get* $(CR)_{C_i}$; $score = sim(\{o_i^q, 1 \le i \le p\}, (CR)_{C_i})$
if $(score > \max Sim)\{\max Sim = score; C(q) = C_i\}$
Sub step 3: $i = i + 1;$
Sub step 4: While $(i \le k)$ Repeat Sub steps 2 & 3
Sub step 5: Return $C(q)$

Fig. 4 Steps for finding relevant cluster of the query

4 EXPERIMENTATION

4.1 Aims

The first aim is to cluster services using OSM and analyze the influence of method of computing ICD (single, complete and average linkage) on the rightness of clustering solution. Variations in clustering solution for various threshold-ICDs obtained using the three methods of computing ICD has to be analyzed and a better method has to be advised for computing ICD. The method of selecting threshold-ICD has to be discussed.

The second aim is to find the performance optimization achieved using clustering. Here different test queries are chosen. An experiment will be carried out to find the average reduction in computation time of discovery using clusters and it will be compared with the computation time of sequential method without clustering.

The third aim is that the accuracy of the proposed approach should be compared with that of sequential approach.

The fourth aim is to compare the results of proposed approach with that of K-Means clustering and other semantic service discovery techniques for justification.

4.2 Experimental Setup

An experimental setup as in Fig.5 is constructed. As in Fig. 5, similarity among all possible pairs of services present in the test data is computed using the OSM, implemented in Java. OSM finds various levels of DoM among outputs of services with the help of Jena API and Pellet reasoner. Dissimilarity matrix is constructed and given as input to clustering tool, available in the link, http://www.cs.umb.edu/~smimarog/agnes/agnes.html.

The clusters thus produced are labeled using representation module. The details of clusters are archived along with their labels. When a query is submitted, the outputs of the query are matched with label of each cluster using textual similarity (simmetrics.jar) and relevant cluster of the query is found out. From the relevant cluster, matched services of the query are found out using TSM.

Experiments are performed on a Laptop with Intel Pentium(R) Dual-Core, 2.20GHz CPU, 3.0 GB memory and Windows 7 Ultimate Operating System using JDK 1.6.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2015.2399301, IEEE Transactions on Services Computing

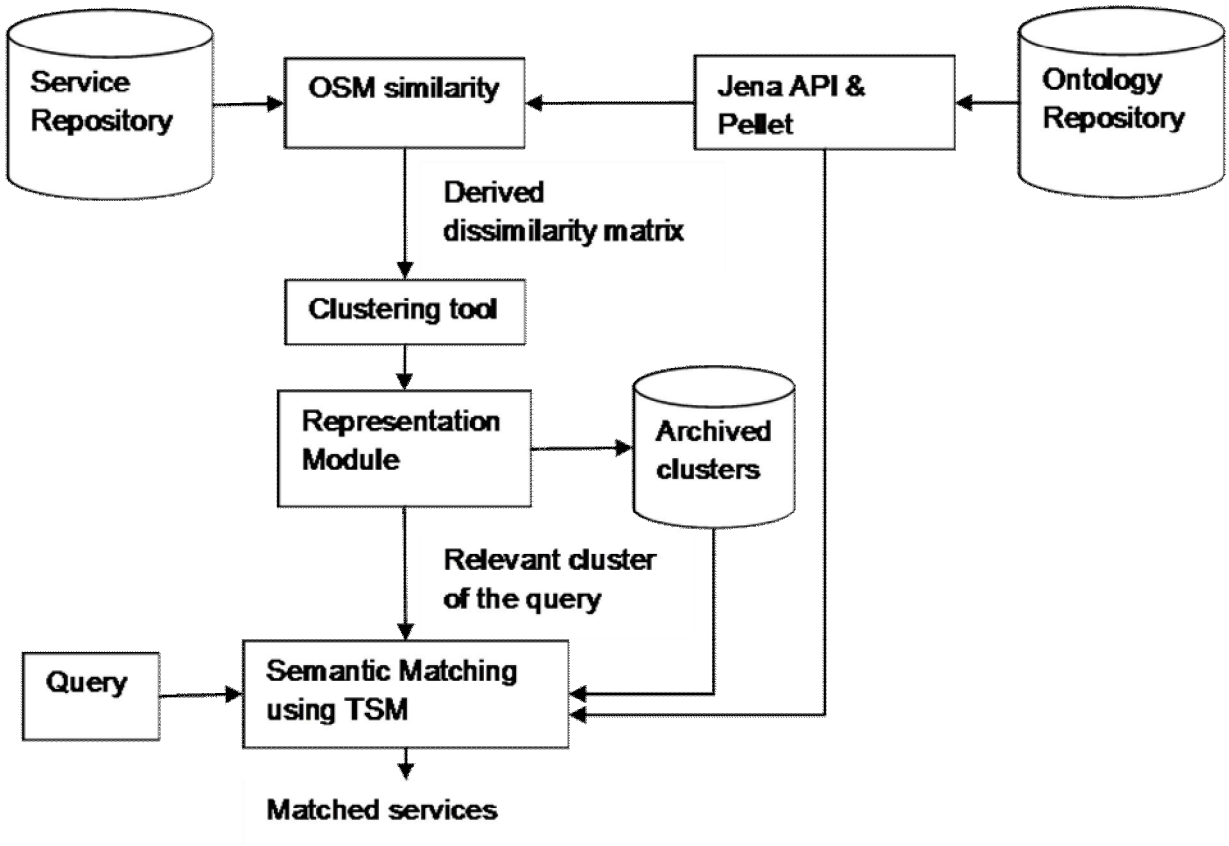

Fig. 5 Experimental setup

4.3 Test Data

A collection of 100 services from publicly available *hRESTS Service Retrieval Test Collection Version 3.0*(http://www.semwebcentral.org/projects/hrests-tc/) given in Appendix-A(Table A.1) is constructed as test collection. The test collection is constructed in such a way that it contains internal groups of similar services from different domains such as education, food, travel, film, etc. Each service in the test collection is given an *ID*. Let s_1, s_2, s_3, …, s_{100} denote the 1st, 2nd, 3rd,…,100th service. The test collection contains 13 internal groups of services. These groups together contain 92 services. Details of internal groups are given in Appendix-A (Table A.2). Remaining 8 services are included in the Test Collection with different purposes as given in Appendix-A (Table A.3).

4.4 Evaluation Measures

Four evaluation measures, viz., intra-cluster similarity of partitions, inter-cluster similarity of partitions, Silhouette Width and Purity of clustering solution are used for evaluation.

Intra-cluster similarity Intra-cluster similarity of a cluster partition is computed as the average of intra-cluster similarity of all services present in that cluster. Let $\text{int}ra-sim(s)$ denote the intra cluster similarity of a service s. Let $\text{int}ra-sim(C_i)$ denote the intra-cluster similarity of i^{th} cluster C_i. The value of $\text{int}ra-sim(C_i)$ is computed using computed using

$$\text{int}ra-sim(C_i)=\frac{1}{|C_i|}\times\sum_{s\in C_i}\text{int}ra-sim(s) \quad (11)$$

In (11), $|C_i|$ denote the number of services present in C_i.

Inter-cluster similarity Let N denote the number of clusters obtained using a clustering algorithm. Let $\text{int}er-sim(C_i)$ denote the inter-cluster similarity of ith cluster and it is computed as

$$\text{int}er-sim(C_i)=\frac{1}{N-1}\sum_{j=1}^{N}\text{int}er-sim(C_i,C_j)|i\neq j \quad (12)$$

In (12), $\text{int}er-sim(C_i,C_j)$ denotes the inter-cluster similarity between two clusters, C_i and C_j and the value of $\text{int}er-sim(C_i,C_j)$ is computed as

$$\text{int}er-sim(C_i,C_j)=\frac{1}{|C_i||C_j|}\sum_{a\in C_i}\sum_{b\in C_j}sim(a,b) \quad (13)$$

In (13), $sim(a,b)$ denotes the similarity between services a and b where a and b belong to C_i and C_j respectively. Also, $|C_i|$ and $|C_j|$ denote the number of services present in C_i and C_j respectively.

Silhouette Width Silhouette Width of a particular cluster partition is computed as the average Silhouette value of all services of that partition. Silhouette width of an individual service denoted by $S(s)$ which reflects the confidence of assigning s to a particular partition, is computed using

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2015.2399301, IEEE Transactions on Services Computing

$$S(s) = b_s - a_s \Big/ \max(b_s, a_s) \quad (14)$$

In (14), a_s denotes the average dissimilarity between s and all other services in the same cluster and b_s denotes the average dissimilarity between s and all services in the closest other cluster (which is defined as the one yielding minimal b_s). The Silhouette width is limited to the interval [-1, 1]. If the silhouette width of an object is near to one, the object is said to be correctly clustered. If silhouette width of an object is around zero then the object will lie between two clusters. Negative value of Silhouette width of an object indicates that the object might be placed in an incorrect cluster. Let τ be the set of all clusters of services. Let s be a service and $C(s)$ be a set of all services in a cluster which contains s. Now the values of a_s and b_s are computed using the following formulae

$$a_s = \frac{1}{|C(s)|-1} \sum_{s' \in C(s)-\{s\}} Dissim(s,s') \quad (15)$$

$$b_s = \min\{\frac{1}{|C|} \sum_{s' \in C} Dissim(s,s') \Big| C \in \tau - \{C(s)\}\} \quad (16)$$

Purity To compute Purity, each cluster is assigned with a class which is most frequent in that cluster. The *most frequent class* of a cluster is found manually. Let N be number of services partitioned into K number of clusters. Let n_{imfc} be the number of services of the *most frequent class* of i^{th} cluster. The Purity of clustering solution should be maximized. The value of Purity of a clustering solution, denoted by P is computed using

$$P = 1/N \sum_{i=1}^{K} n_{imfc} \quad (17)$$

5 Results and Discussion

5.1 Analysis

Towards the first aim, three experiments are carried out, one for each method of computing ICD. In each experiment, clustering solutions are obtained by varying the threshold-ICD from 0 to 1 in steps of 0.1 and compared against the internal groups of test data given in Appendix-A(Table A.2 and Table A.3). From analysis, the clustering methods are found to produce partial clustering of services up to a certain minimum value of threshold-ICD. The methods produce complete clustering of services when the threshold-ICD is relaxed to comparatively higher values. For the given test collection, the number of services that is expected to be partitioned is 96 (because three services are having no outputs and one service is having different output from all other services). The number of partitions produced and the number of services that got clustered by the three methods for various threshold-ICDs are given in Table 2. Further, in Table 2, the partial clustering details are given in normal font whereas the complete clustering details are given in bold and italics.

TABLE 2
Details about number of clusters produced and number of services that have got clustered by the three methods for various values of threshold-ICD

Threshold-ICD	Single linkage method		Average linkage method		Complete linkage method	
	# of partitions	# of services got clustered	# of partitions	# of services got clustered	# of partitions	# of services got clustered
0	18	75	18	75	34	68
0.1	No change		No change		18	75
0.2	16	87	18	84	19	85
0.3	12	94	17	91	19	91
0.4	11	95	16	93	18	91
0.5	***11***	***96***	15	95	15	93
0.6	***No change***		No change		No change	
0.7	***10***	***96***	***13***	***96***	***14***	***96***
0.8	***No change***		***12***	***96***	***No change***	
0.9	***No change***		***11***	***96***	***13***	***96***

From Table 2, single linkage method is found to produce partial clustering for threshold-ICD ranges from 0 to 0.4. The method produces complete clustering with 96 services clustered into 11 clusters when threshold-ICD is relaxed to 0.5. When threshold-ICD is increased to 0.6, no change is found in the clustering solution. When threshold-ICD is increased to 0.7, the 96 services are clustered into 10 clusters. There is no further change in the clustering solution when threshold-ICD is relaxed to 0.8 or 0.9.

Similarly, the average and complete linkage methods produce partial clustering of services when threshold-ICD is up to 0.6. These methods produce complete clustering of services when the threshold-ICD is relaxed to 0.7 and above.

When threshold-ICD is fixed at 0.7, 0.8 and 0.9, the average linkage method partitions the 96 services into 13, 12 and 11 clusters respectively and complete linkage method clusters the services into 14, 14 and 13 clusters respectively.

Please note: *Details of complete clustering solutions produced by the three methods which are considered for analysis are given in Appendix-B. Partial clustering solutions though not considered for analysis are given in Appendix-C for reference. Individual service-level data such as intra-cluster similarity of individual services, inter-cluster similarity of individual services and silhouette width of individual services for various values of threshold-ICDs, are given in Appendix-D.*

At first, the clustering solution obtained using single

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2015.2399301, IEEE Transactions on Services Computing

linkage is considered for analysis. For simplicity, we denote intra-cluster similarity, inter-cluster similarity and Silhouette Width as intra-sim, inter-sim and SW respectively. The values of intra-sim, inter sim and SW of all partitions produced by single linkage for different threshold-ICDs are given in Table 3.

TABLE 3
Intra-sim, inter-sim and SW of clusters obtained using single linkage for threshold ICD set at 0.5/0.6 (96 services are partitioned into 11 clusters) and 0.7/0.8/0.9 (96 services are partitioned into 10 clusters)

Threshold-ICD set at 0.5/0.6					Threshold-ICD set at 0.7/0.8/0.9				
Cluster ID	# of Services	Intra-sim	Inter-sim	SW	Cluster ID	# of services	Intra-sim	Inter-sim	SW
0	45	0.2759	0	0.2759	0	45	0.2759	0	0.2759
1	3	1.0000	0	1.0000	1	3	1.0000	0	1.0000
2	7	0.8857	0	0.8857	2	7	0.8857	0	0.8857
3	8	1.0000	0	1.0000	3	8	1.0000	0	1.0000
4	5	0.6100	0.012	0.5436	4	7	0.3810	0	0.3810
5	9	0.7333	0	0.7333	5	9	0.7333	0	0.7333
6	7	0.8310	0	0.8310	6	7	0.8310	0	0.8310
7	4	0.9000	0	0.9000	7	4	0.9000	0	0.9000
8	4	1.0000	0	1.0000	8	4	1.0000	0	1.0000
9	2	0.8	0	0.8	9	2	0.8	0	0.8
10	2	0.7	0.012	0.6591					

For any good clustering solution, the basic criterion to be fulfilled is that the intra-cluster similarity of each partition of a clustering solution must be greater than the inter-cluster similarity of that partition. From Table 3, it is observed that the above criterion is fulfilled by each partition produced by single linkage method. Further, the Silhouette Width of each partition is found to be greater than zero which indicates that the services are appropriately placed in their clusters. Further, the cluster partitions produced by single linkage are compared against the known internal groups(Appendix-A)

The details of clusters, the number of services, service IDs, *most frequent class* and number of services present in the *most frequent class* obtained using single linkage for threshold-ICD 0.5 or 0.6 are given in Appendix-B(Table B.1). The clustering solution is compared with test services given in Appendix-A(Table A.2 and Table A.3). It is found that there are 45 services in Cluster-0. It is observed that the three groups namely, 'film', 'books-author' and 'books-book' are merged with 'price' group by single linkage. This is because single linkage merges clusters based on similarity between the nearest pair of services. The film group is merged to price group by the service s_{50} which contains two output parameters, namely, *my_ontology.owl#_VideoMedia* and *concept.owl#_RecommendedPrice*. The service s_{50} has similarity with each service which has output parameter related to *concept.owl#price*. Similarly, 'books-author' group is merged with 'price' group by the services, s_{22} and s_{27}. These services have two output parameters, namely, *concept.owl#price* and *books.owl#author*. Also, 'books-book' group is merged with 'price' group by service, s_{22}.

The merging of above clusters occurs thus because single linkage method merges clusters based on a single service pair which is constituted by the nearest services with one service from each cluster. Further, when the threshold-ICD is relaxed to 0.7, the 'researcher' group is merged with 'academic' group(i.e. Cluster-10 is merged with Cluster-4). The values of Purity and average SW of clustering solution produced by single linkage method for different threshold-ICDs from 0.5 to 0.9 are given in Table 4.

TABLE 4
Purity and Average SW of single linkage

Threshold ICD	Purity	Average SW
0.5/0.6	0.7604	0.7844
0.7/0.8/0.9	0.7604	0.7807

From Table 4, the Purity of clustering solution is found to be 0.7604 whereas the expected purity is 1. This is because in Cluster-0, there are 23 services which do not belong to the *most frequent class*. Though the intra-cluster similarity, inter-cluster similarity and Silhouette Width of partitions are good, the Purity of clustering solution obtained using single linkage is low.

Secondly, clustering solution produced by average linkage is taken for discussion. The clustering solution produced by average linkage when threshold-ICD fixed at 0.7 is given in Appendix-B(Table B.2). The clustering solution is compared against the internal groups given in Appendix-A(Table A.2 and Table A.3). The average linkage is found to cluster 96 services into 13 clusters and clustering solution produced by average linkage is found to be matched with expected groups when the threshold-ICD is fixed at 0.7. When the threshold-ICD is relaxed to 0.8, the Cluster-3 is merged with Cluster-0 by the service s_{14}. Though in average linkage, the ICD is measured using all service pairs constituted by one service from each

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2015.2399301, IEEE Transactions on Services Computing

cluster, this single service s_{14} is related to all the services in Cluster-0 leading to produce inter-cluster similarity required for merging with average linkage. When threshold-ICD is relaxed to 0.9, Cluster-5 is merged with Cluster-0. This is because the services s_{22} and s_{27} produce the inter-cluster similarity as required by average linkage. Further, the values of intra-sim, inter-sim and SW of all cluster partitions produced by average linkage method are given in Table 5(for threshold-ICD set at 0.7 and 0.8) and Table 6(for threshold-ICD set at 0.9).

TABLE 5
Intra-sim, Inter-sim and SW of cluster partitions obtained using average linkage with threshold-ICD set at 0.7 (96 services grouped into 13 clusters) and 0.8 (96 services grouped into 12 clusters)

Threshold ICD set at 0.7					Threshold ICD set at 0.8				
Cluster ID	# of Services	Intra-sim	Inter-sim	SW	Cluster ID	# of services	Intra-sim	Inter-sim	SW
0	17	0.9059	0.0356	0.8811	0	20	0.7251	0.0154	0.6936
1	16	0.5748	0.0045	0.5305	1	16	0.5748	0.0035	0.5365
2	3	1.0000	0.0000	1.0000	2	3	1.0000	0.0000	1.0000
3	3	0.8333	0.0238	0.6290	3	7	0.8857	0.0000	0.8857
4	7	0.8857	0.0000	0.8857	4	9	0.8472	0.0129	0.7581
5	9	0.8472	0.0164	0.7202	5	8	1.0000	0.0000	1.0000
6	8	1.0000	0.0000	1.0000	6	7	0.3810	0.0000	0.3810
7	7	0.3810	0.0000	0.3810	7	9	0.7333	0.0000	0.7333
8	9	0.7333	0.0000	0.7333	8	7	0.8310	0.0000	0.8310
9	7	0.8310	0.0000	0.8310	9	4	0.9000	0.0000	0.9000
10	4	0.9000	0.0000	0.9000	10	4	1.0000	0.0000	1.0000
11	4	1.0000	0.0000	1.0000	11	2	0.8000	0.0000	0.8000
12	2	0.8000	0.0000	0.8000					

From Table 5 and Table 6, intra-cluster similarity of cluster partitions obtained by average linkage is found to be better when compared to that of partitions produced by single linkage. The average Silhouette Width of the clustering solution produced with threshold-ICD fixed at 0.7, 0.8 and 0.9 are found to be close to 1. The value of Purity of clustering solution obtained using average linkage with threshold-ICD fixed at 0.7, 0.8 and 0.9 are computed and given in the Table 7. From Table 7, it is seen that, average linkage clustering produces best clustering solution when the threshold-ICD is fixed at 0.7. It produces clustering results which are same as expected solution. But, when the threshold-ICD is relaxed from 0.7 to 0.8 and 0.9, the Purity of clustering solution decreases from 100% to 97.9% and 90.6% respectively.

Thirdly, the clustering solution produced by complete linkage with threshold-ICD fixed at 0.7, given in Appendix-B(Table B.3) is considered for analysis. When the clustering solution produced by complete linkage is compared with the internal group of services in Appendix-A(TableA.2 and Table A.3), it is observed that the complete linkage method partitions the given test data as expected except that Cluster-6 should be merged with Cluster-3. When the threshold-ICD is increased to 0.8, there is no change in the clustering solution. Further, when the threshold-ICD is increased to 0.9, Cluster-6 is merged with Cluster-3 and the clustering solution exactly matches with the expected results. Further, the values of intra-cluster similarity, inter-cluster similarity and Silhouette Width of all cluster partitions produced by complete linkage method when the threshold-ICD is fixed at 0.7/0.8 and 0.9 are given in Table 8.

As average clustering solution, complete linkage also produces better values for intra-cluster similarity and inter-cluster similarity and silhouette width of cluster partitions. The average silhouette width of clustering solution is near to 1 which indicates the presence of a well-defined clustering structure. Purity of solution produced by complete linkage for the threshold-ICDs 0.7, 0.8 and 0.9 are computed and given in Table 9. When we compare average silhouette width of clustering solutions produced by the three methods using the data given in Table 4, Table 7 and Table 9, all the methods are found to produce a value near to 1 for average Silhouette Width which indicates the presence of well-defined clusters in the given data. Also, the Purity of clustering solution produced by single linkage is found to be very low and hence single linkage clustering is not considered for further analysis.

Further, the Purity of clustering solution obtained using complete linkage is better than that of clustering solution produced using average linkage. From Table 7 it is understood that the Purity of average linkage is decreasing when the threshold-ICD is increased from 0.7 to 0.8/0.9. This indicates that services which do not belong to *most frequent class* of a partition might be grouped in that partition. As mentioned earlier, the average linkage merges the Cluster-3 with Cluster-0 (Appendix-B) when the threshold-ICD is increased to 0.8. Here cluster-3 contains three services s_{12}, s_{21}, and s_{14}. Out of these three

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2015.2399301, IEEE Transactions on Services Computing

services, the first two services are totally different from 'price' group. But still as the service s_{14} is related to each service in 'price' group, it generates inter-cluster similarity which is more than the threshold-ICD.

Similarly, when threshold-ICD is relaxed to 0.9, it is observed that Cluster-5 is merged with Cluster-0. Services present in Cluster-5 are s_{22}, s_{26}, s_{27}, s_{32}, s_{33}, s_{34}, s_{37}, s_{38} and s_{39}. Only two services of Cluster-5 namely, s_{22} and s_{27} have similarity with services present in Cluster-0. Other services in Cluster-5 have output parameter *books.owl#author* which is totally different from *concept.owl#price*. But services s_{22} and s_{27} could produce enough inter-cluster similarity as required by average linkage to merge the clusters. Thus, though the average linkage computes inter-cluster similarity between two clusters based on all possible service pairs, the Purity of clustering solution is found to decrease when the threshold-ICD is relaxed to 0.8 or 0.9.

TABLE 6
Intra-sim, inter-sim and SW of clusters produced by average linkage with threshold-ICD set at 0.9

Cluster ID	# of services	Intra-sim	Inter-sim	SW
0	29	0.4751	0.0024	0.4650
1	16	0.5748	0.0024	0.5500
2	3	1.0000	0.0000	1.0000
3	7	0.8857	0.0000	0.8857
4	8	1.0000	0.0000	1.0000
5	7	0.3810	0.0000	0.3810
6	9	0.7333	0.0000	0.7333
7	7	0.8310	0.0000	0.8310
8	4	0.9000	0.0000	0.9000
9	4	1.0000	0.0000	1.0000
10	2	0.8000	0.0000	0.8000

TABLE 7
Purity and Average SW of average linkage

Threshold ICD	Purity	Average SW
0.7	1	0.7917
0.8	0.979	0.7933
0.9	0.906	0.7769

When the purity of clustering solution obtained using complete linkage is analyzed, from Table 9, the value is found to be 100% for all threshold-ICDs 0.7, 0.8 and 0.9.

To find out suitable method for clustering, the clustering solutions produced by average and complete linkage are analyzed using silhouette width of individual services. The silhouette width of one service namely, s_{50} is found to be a special case for analysis because the silhouette width produced by average and complete linkage is negative. The intra-cluster similarity, inter-cluster similarity and silhouette width of this service, s_{50} obtained using average and complete linkage methods for different threshold-ICDs are given in Appendix-D(Table D.8). When the results produced by average linkage are analyzed, the silhouette width of the service s_{50} is found to be negative for all threshold-ICDs, 0.7, 0.8 and 0.9. The service s_{50} contains two output parameters, namely, *my_ontology.owl#videomedia* and *concept.owl#price*. At first this service is merged with other services which have *videomedia* as one of their outputs (i.e. with services s_{51}, s_{52}, s_{53} and s_{54}).

When videomedia group is not merged with the film group the silhouette width of the service is positive. But when this group is merged with film group, the intra-cluster similarity of the service with film and media services is found to be less as the service has more inter-cluster similarity with services in the price group. Hence, when the videomedia group is merged with film group, the silhouette width of the service becomes negative. But in the case of complete clustering, the merging of videomedia and price groups is taken place only when the threshold-ICD is kept at 0.9. When both purity and silhouette width of individual services are concerned, complete linkage method is found to yield better results than average linkage.

Finally, hierarchical clustering with complete linkage criterion is suggested for clustering of services for the following reasons. Firstly, complete linkage merges clusters based on farthest pair of services. If the farthest pair satisfies the given threshold-ICD, then any other pair in the concerned clusters will satisfy the given threshold-ICD. This kind of merging is very suitable even in applications having strict similarity requirements. Secondly, in the case of complete linkage as the merging of clusters is decided by the farthest pair of services, the purity measure is found to be better than the average linkage. Thirdly, it is observed that in the case of complete linkage when threshold-ICD is relaxed to 0.9, services such as s_{50} is found to have negative Silhouette Width. Though merging by farthest pair of services does not warrant for positive silhouette width, fixing threshold-ICD slightly higher than 0.1 will prevent the unwanted merging of clusters, leading to positive silhouette width.

5.1.1 Selection of Threshold-ICD

As part of the first aim, further, in general the threshold-ICD is selected according to the similarity demands of client through the predefined similarity scores assigned to various levels of DoM(Table 1). The value is application specific and it is a variable. The value can range from 0.1 to 1. Any value can be chosen for threshold-ICD in this range but according to similarity demands of clients. The selection of threshold-ICD is illustrated for examples discussed in Section 1. Suppose a need of a client with 'head injury' is *emergency-ambulance*. In this case, the client can accept only exact matches. Hence, the numeric value corresponding to exact i.e. 1 is chosen as threshold-ICD. Suppose a need of a client is *Car* to visit a physician for general checkup. Here the client can accept matches having DoM right from exact to grandparent. Hence, the numeric value corresponding to grandparent, i.e. 0.3 is chosen as threshold-ICD.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2015.2399301, IEEE Transactions on Services Computing

TABLE 8
Intra-sim, Inter-sim and SW of clusters produced by complete clustering with threshold-ICD set at 0.7/0.8 and 0.9

Threshold-ICD set at 0.7/0.8					Threshold-ICD set at 0.9				
Cluster ID	# of services	Intra-sim	Inter-sim	SW	Cluster ID	# of Services	Intra-sim	Inter-sim	SW
0	3	1.0000	0.0000	1.0000	0	3	1.0000	0.0000	1.0000
1	3	0.8333	0.0239	0.6290	1	3	0.8333	0.0238	0.6290
2	7	0.8857	0.0000	0.8857	2	7	0.8857	0.0000	0.8857
3	11	0.7836	0.0262	0.6485	3	16	0.5748	0.0045	0.5305
4	9	0.8472	0.0161	0.7202	4	9	0.8472	0.0164	0.7202
5	17	0.9059	0.0391	0.8811	5	17	0.9059	0.0356	0.8811
6	5	0.7175	0.0393	0.5011	6	8	1.0000	0.0000	1.0000
7	8	1.0000	0.0000	1.0000	7	7	0.3810	0.0000	0.3810
8	7	0.3810	0.0000	0.3810	8	9	0.7333	0.0000	0.7333
9	9	0.7333	0.0000	0.7333	9	7	0.8310	0.0000	0.8310
10	7	0.8310	0.0000	0.8310	10	4	0.9000	0.0000	0.9000
11	4	0.9000	0.0000	0.9000	11	4	1.0000	0.0000	1.0000
12	4	1.0000	0.0000	1.0000	12	2	0.8000	0.0000	0.8000
13	2	0.8000	0.0000	0.8000					

TABLE 9
Purity and average SW of complete linkage

Threshold ICD	Purity	Average SW
0.7	1	0.7793
0.8	1	0.7793
0.9	1	0.7917

To summarise, the computation of ICD using complete linkage and selection of threshold ICD in terms of DoM in this proposed approach are validated extensively with a typical collection of 100 services. The proposed approach is generic and it works with any number of services. Further to strengthen this argument, the experiment is extended with another data set of 200 services with threshold-ICD set at 0.2. The data set is split into 10 clusters as given in Appendix-E(Table E.1). The purity of the clustering solution is found to be 1 which shows that the services are rightly clustered and it works for any number of services.

5.2 Performance Optimization and Testing Accuracy

Towards the second aim, to assess the performance improvement achieved using clustering approach, test queries corresponding to different cluster partitions have been chosen. The time taken for finding matched services for test queries using sequential (i.e. no clustering) and proposed methods is given in Appendix-F(Table F.1). With a test collection of 100 services, the average time taken to find matched services of a test query using sequential and proposed methods are found to be 170.59 seconds and 7.32 seconds respectively.(Table F.1 in Appendix-F).

Towards the third aim, to test the accuracy of proposed approach the number of relevant services retrieved by the recommended method is compared in with that of sequential method.(Table F.1 in Appendix-F). It is found that clustering of services does not affect the accuracy of results but helps in reducing the computation time of discovery.

5.3 Comparison with K-Means clustering

Towards the fourth aim, to compare the results with that of K-Means clustering in order to highlight that hierarchical agglomerative clustering with complete linkage is better for service discovery clustering results are obtained using K-Means (See Appendix-G) for different values of K from 13 to 21(as the test collection contains 13 internal groups). The results are compared with K-Means using Purity(Table 10) as purity is an external criterion.

TABLE 10
Purity of clustering solutions obtained used K-Means algorithm for different values of K

Value of K	Purity
13	0.83
14	0.84
15	0.88
16	0.94
17	0.95
18	0.93
19	0.93
20	0.97
21	0.93

From Table 9 and Table 10, the value of Purity obtained using complete linkage method(100%) is higher than that(83% to 97%) obtained using K-Means. In K-Means, there is no provision for specifying the required similarity among services. Further, within a single cluster the value of average intra-cluster similarity is small which may not meet the similarity demands of clients(Table

G.10 to Table G.12 in Appendix G) whereas in complete linkage, one can specify the required similarity demands of clients through threshold-ICD.

According to complete linkage criterion, the clusters will be merged only when the similarity of fartherest pair satisfies the threshold-ICD. (See Table 8) This aspect guarantees that within a cluster the intra-cluster similarity between any pair of services is at least equal to the threshold-ICD.

5.4 Towards Comparison with other Discovery Techniques

As part of the fourth aim, the proposed approach is compared with other semantic service discovery approaches [10, 14, 35]. Each work has its own method for discovery with its own specific focus. For example, in [35], a service recommendation approach which considers a number of user criteria and many context dimensions while recommending service to end users is presented. This approach is end user oriented. But the proposed approach, a developer oriented one presents a strategic model to technicians working in service clustering. The proposed work is also unique with its own method, data set and specific focus and hence the need does not arise for one to one comparison. However, one of its unique aspects, namely, prioritization of features(Section 1) while computing similarity is compared with Viorica Rozina Chifu et al.,'s method [10] and Peng Liu et al.,'s method [14]. The Peng Liu et al.,'s method [14] uses three features of services, viz., description, inputs and outputs with weights, 0.5, 0.2 and 0.3 respectively. The Viorica Rozina Chife et al.,'s approach[10] considers inputs and outputs with 0.5 weightage whereas the proposed considers only outputs for clustering. The impact of weight factor is analyzed with a set of 6 services, s_1, s_2, s_3, s_4, s_5 and s_6. Three clustering patterns are obtained by varying weights according to [14] [10] and the proposed work. The comparative study is elaborated and the resulting patterns are given in Appendix-H(from Fig. H.1 to Fig. H.6). Further, sample queries(Appendix-H, Table H.2) are chosen in order to find out the percentage of removal of irrelevancy. The proposed approach is found to eliminate irrelevancy more efficiently when compared with other approaches(Appendix-H, Table H.3) For further reading refer to Appendix-H.

6 Conclusion

In this work, a few methodologies have been proposed for service discovery using clusters. The methodologies include two similarity models for computing similarity between services, called OSM and TSM. The similarity models use newly defined levels of DoM in order to satisfy the similarity demands of clients effectively. Computing similarity between services based only on their outputs using OSM is proposed for clustering services in order to completely eliminate irrelevancy. Specifically, hierarchical agglomerative clustering is emphasized as it is more structured. Of the major inter-cluster linkage metrics, viz., single, average and complete linkage, complete linkage warrants that within a cluster the similarity between any two services is at least the threshold-ICD. Keeping this as base, specifying threshold-ICD in terms of similarity demands using different levels of DoM is suggested. Computing similarity between services based on both inputs and outputs of services using TSM is employed for discovery of matched services of a query.

The proposed approach is a generalized one. It is not domain specific or application specific or implementation specific. The concept has been tested and proved with OWL-S services and hRESTful service colletions (*http://www.semwebcentral.org*). However services are described in different formats. A generalized interface could be constructed which would extract outputs (with their ontologies) and feed them as input to similarity computation.

To mention few limitations, for convenience, the values of m and n in (1) are set as 1. The proposed approach is tested with services having single output parameter. I.e. $m = n = 1$ So, in future, it is proposed to alleviate this limitation by developing a hierarchical agglomerative algorithm in complete linkage mode which dynamically computes threshold-ICD for a given pair of services according to (1). Further, the proposed approach assigns numeric values to DoMs arbitrarily with implicit pre-ordering. In future it is proposed to assign numeric equivalent values of DoM automatically using some learning algorithms like linear regression model as such assignment leads to continuous & fine-grained similarity assessment.

References

[1] Sonia Ben Mokhtar, Anupam Kaul Nikolaos Georgantas and Valerie Issarny, "Towards Efficient Matching of Semantic Web Service Capabilities", International Workshop on Web Services Modeling and Testing (WS-MaTe 2006).

[2] Sonia Ben Mokhtar, Davy Preuveneers, Nikolaos Georgantas, Valerie Issarny, Yolande Berbers, "EASY", Efficient semantic Service discovery in pervasive computing environments with QoS and context support", The Journal of Systems and Software, 2008, Vol. 81, pp. 785-808.

[3] Ruiqiang Guo, Jiajin Le, XiaoLIng Xia, "Capability Matching of Web Service Based on OWL-S", Proceedings of the 16th International Workshop on Database and Expert Systems Applications, IEEE Computer Society, 2005.

[4] Jacek Kopecky, Tomas Vitvar, Carine Bournez, Joel Farrell, "SAWSDL: Semantic Annotations for WSDL and XML Schema", IEEE Internet Computing, IEEE Computer Society, 2007, Vol. 11, No. 6, pp. 60-67.

[5] Matthias Klusch, Frank Kaufer, "WSMO-MX: A hybrid Semantic Web service matchmaker", Journal of Web Intelligence and Agent Systems, vol. 7, Issue 1, 2009, pp. 23-42.

[6] Uwe Keller, Ruben Lara, Holger Lausen, Axel Polleres, Livia Predoiu, Ioan Toma, "WSML Deliverable D5.2 v0.1 WSMO Discovery Engine", 2004.

[7] Haller A., Cimpian, E., Mocan, A., Oren, E., Bussler, C., "WSMX - a semantic service-oriented architecture," Proc. of IEEE International Conference on Web Services, , vol. 1, pp.321-328, 2005.

[8] Katia Sycara, Massimo Paolucci, Anupriya Ankolekar, Naveen

Srinivasan, "Automated discovery, interaction and composition of Semantic Web Services", Journal of Web Semantics, Elsevier, December 2003, Vol. 1, No.1, pp. 27-46.

[9] Massimo Paolucci, Takahiro Kawamura, Terry R Payne and Katia Sycara, "Semantic Matching of Web Services Capabilities", International Semantic Web Conference, Springer Verlag, LNCS, Vol. 2342, pp. 333-347, 2002.

[10] Viorica Rozina Chifu, Cristina Bianca Pop, Ioan Salomic, Mihae Dinsoreanu, Vlad Acretoaie, Tudor David, "An Ant-inspired Approach for Semantic Web Service Clustering", Roedunet 9th International Conference (RoEduNet), pp.145-150, 2010.

[11] Richi Nayak, Bryan Lee, Web Service Discovery with additional Semantics and Clustering", IEEE/WIC/ACM International Conference on Web Intelligence, pp. 555-558. 2007.

[12] Dasgupta Sourish, Satish Bhat, and Yugyung Lee. "STC: Semantic Taxonomical Clustering for Service Category Learning." arXiv preprint arXiv:1303.5926 (2013).

[13] Huiying Gao, Wolffried Stucky, Lei Liu, "Web Services Classification Based on Intelligent Clustering Techniques", International Forum on Information Technology and Applications (IFITA 2009), vol. 3, pp.242-245, 2009.

[14] Peng Liu, Jingyu Zhang, Xueli Yu, "Clustering-Based Web Service Matchmaking with Automated Knowledge Acquisition", W. Liu et al., (Eds), Springer-Verlag Berlin Heidelberg, LNCS, Vol.5854, pp. 261-270, 2009.

[15] Qianhui Liang, Peipei Li, Patrick C. K. Hung, Xindong Wu, "Clustering web services for automatic categorization", in proceedings of IEEE International Conference on Services Computing, pp. 380-387, 2009.

[16] Zhiliang Zhu, Haitao Yuan, Jie Sonet, WS-SCAN: A Effective Approach for Web Services Clustering, International conference on Computer Application and System Modeling, Volume 5, pp.618-622, 2010.

[17] Elgazzar, K.; Hassan, A.E.; Martin, P., "Clustering WSDL Documents to Bootstrap the Discovery of Web Services," Web Services (ICWS), 2010 IEEE International Conference on, vol., no., pp.147,154, 5-10 July 2010.

[18] Jingyu Zhang, Xueli Yu, Peng Liu and Zhen Wang, "Research on improving Performance of Semantic Search in UDDI", Proceedings of the WRI Global Congress on Intelligent Systems, IEEE Computer Society, Vol. 4, pp. 572-576, 2009.

[19] Xianyang Qu, Hailong Sun, Xiang Li, Xudong Liu, Wei Lin, "WSSM: A WordNet-Based Web Services Similarity Mining Mechanism", Computation World: Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns, IEEE Computer Society, Greece, pp.339-345, 2009.

[20] Huiying Gao, Wolffried Stucky, Lei Liu, "Web Services Classification Based on Intelligent Clustering Techniques", International Forum on Information Technology and Applications (IFITA 2009), vol. 3, pp.242-245, 2009

[21] Jiangang Ma, Yanchun Zhang, Jing He, "Efficiently Finding Web Services Using a Clustering Semantic Approach", CSSSIA '08 Proceedings of the International Workshop on Context Enabled Source and Service Selection, Integration and Adaptation: organized with the 17th International World Wide Web Conference (WWW 2008), Beijing, China, pp. 1-8, 2008.

[22] Wang Lei, SI Jing, HU Xiao-bo, "Research on the clustering and composition of P2P-based web services", 2nd International on BioMedical Engineering and Informatics, pp.1-5. 2009.

[23] Shun Hang, Haiyang Wang, Lizhen Cui, "A user experience-oriented service discovery method with clustering technology", Proceedings of the Second International Symposium on Computational Intelligence and Design ,Volume 02, pp. 64-67, 2009.

[24] Witold Abramowicz, Konstanty Haniewics, Monika Kaczmarek, Dominik Zyskowski, "Architecture for Web services filtering and clustering", ICIW '07 Proceedings of the Second International Conference on Internet and Web Applications and Service, pp. 18, May 13-19, 2007.

[25] Li Ying, "Algorithm for Semantic Web Services Clustering and Discovery", International Conference on Communications and Mobile Computing (CMC), pp.532-536, 2010.

[26] Juan Zhou, Shuyu Li, "Semantic web service discovery approach using service clustering", International Conference on Information Engineering and Computer Science, pp.1-5,2009.

[27] Shun Hang, Haiyang Wang, Lizhen Cui, "A user experience-oriented service discovery method with clustering technology", Proceedings of the Second International Symposium on Computational Intelligence and Design ,Volume 02, pp. 64-67, 2009.

[28] Banage T.G.S. Kumara, Yuichi Yaguchi, Incheon Paik, Wuhui Chen, "Clustering and Spherical Visualization of Web Services,", IEEE International Conference on Services Computing, 2013, pp.89-96.

[29] Dong Shou, Chi-Hung Chi, "Effective Web Service Retrieval Based on Clustering", Fourth IEEE International Conference on Semantics, Knowledge and Grid (SKG '08), pp.469-472, 2008.

[30] Dasgupta, S.; Bhat, S.; Yugyung Lee, "Taxonomic Clustering and Query Matching for Efficient Service Discovery," IEEE International Conference on Web Services, 2011, pp.363-370.

[31] Berkhin, Pavel, "A survey of clustering data mining techniques", Grouping multidimensional data, Springer Berlin Heidelberg, 2006, pp. 25-71.

[32] Duygu Celik, Atilla Elci "A broker-based semantic agent for discovering semantic web services through process similarity matching and equivalence considering quality of service, Science China Information Sciences, vol. 56, No. 1, 2013, 012102(:24).

[33] Mehdi Bayat, Morteza Analoui, "A new approach for flexible matching of grid resources", World of Computer Science and Information Technology Journal, Vol. 2, No. 7, 220-224, 2012.

[34] Syeda-Mahmood, T., Shah, G., Akkiraji, R., Ivan, A.A., Goodwin, R., "Searchig service repositories by combining semantic and ontological matching", In Proc. of IEEE International Conferences on Web Services, pp. 13-20,2005.

[35] Liwei Liu; Lecue, F., Mehandjiev, N., Ling Xu, "Using Context Similarity for Service Recommendation", Fourth IEEE International Conference on Semantic Computing, pp.277-284, 2010.

Chellammal Surianarayanan has 10 years of R&D experience as in IGCAR, Kalpakkam, India. She has been working as an Assistant Professor in computer science since 2009.

Gopinath Ganapathy has 26 years of experience including International experience in the U.S and U.K. with R&D and consultancy expertise in industries that include IBM, Toyota, Lucent and other US/UK based firms. He is Professional Member in IEEE and ACM. Currently he is the Chair, School of Computer Science and Engineering, Bharathidasan University, Tiruchirappalli, India.