Journal of Theoretical and Applied Information Technology
30th November 2014. Vol. 69 No.3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

I/O MATCH (I/O MAT) AND BEHAVIORAL MATCH (BEH MAT) BASED SEMANTIC WEB SERVICE DISCOVERY

[1]J.UMA MAHESWARI, [2] Dr.G.R.KARPAGAM
[1]Assistant Professor (Senior Grade), Dept of CSE,PSG College of Technology, Coimbatore, India
[2] Professor, Dept of CSE, PSG College of Technology, Coimbatore, India
E-mail: [1]uma_ftt@yahoo.co.in , [2]grkarpagam@gmail.com

ABSTRACT

Discovery is one of the central reasoning tasks in SOA systems, concerned with the detection of usable Web services for a specific request or application con-text. Aiming at the automation of this task, most existing works on semantically enabled Web service discovery focus on the degree of match of two services based merely on their I/O pairs. Another approach for matchmaking in Semantic Web Services (SWS) that considers each service as a sub-graph of the semantic network of the ontology formed by inputs, outputs, pre and post-conditions with contribution of syntactical information such as keywords and textual descriptions. The similarity between services is defined as the similarity between these graphs (Behavior Graphs). This paper presents the detailed description of both approaches and it also analyses the advantages, disadvantages and retrieval effectiveness of these two matchmaking systems (I/O Mat, BEH Mat) and proposes new algorithm for semantic match making.

Keywords: *Semantic Web, Web Services, Matchmaking, Bipartite Graph, Behaviourally Correct Path, Critical Elements*

1. INTRODUCTION

The popularity of the service oriented computing and web services, attracts organizations to use the web to sell their own services. Web services are advertised in a central repository, later it can be invoked and used by the consumers. In central repository the web services are described by the description language called WSDL. A Web service based on Web Service Description Language (WSDL)[1] is termed as syntactic based web services. WSDL based description allows keyword based processing. This limitation prevents fully automatic discovery, composition, invocation, and monitoring of web services. The reason for this shortcoming is the lack of semantic understanding. To overcome this problem, Web services require a method to incorporate semantics. Just as the Semantic Web [2] is an extension of the current World Wide Web, a semantic Web service [3][4] is an extension of Web services. It overcomes Web service limitations by using knowledge representation technology from the semantic Web. Specifically, it uses ontology[5][6] to describe its service instead of using WSDL. Such ontology can be understood by machines. This allows a fully automatic discovery, composition, invocation, and monitoring in Web services.

Due to the increase of web services, finding most appropriate services among list of services becomes difficult task. Web service discovery is the vital part in web service model because for all types of research like service composition, service selection discovering the suitable service according to the user requirement is the first phase.

WS discovery is performed with the aid of the UDDI[7][8]]registries that support keyword based matching between the textual descriptions of the user request and the published/advertised services. However, according to the Semantic Web vision, WS will eventually be replaced by Semantic Web Services (SWS). SWS are, essentially, a metadata layer that allows for more expressive description of service capabilities, used both for service advertisements (formed by the service providers) and requests (formed by the service requestors). Such metadata is represented through semantic Web technologies like ontology and rules.

Semantic service discovery is the process of locating existing Web services based on the description of their functional and non-functional semantics. Most current approaches measure the

Journal of Theoretical and Applied Information Technology
30th November 2014. Vol. 69 No.3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

degree of match of two services based merely on their I/O pairs. Another approach[9] for matchmaking in Semantic Web Services (SWS) that considers each service as a sub-graph of the semantic network of the ontology formed by inputs, outputs, pre and post-conditions with contribution of syntactical information such as keywords and textual descriptions is also called as behavioral match. This paper analyses and presents the retrieval effectiveness of Input-Output match and behavioral match and proposes new algorithm for semantic matchmaking.

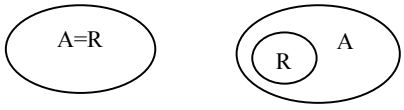
2. SEMANTIC MATCH MAKING BASED ON INPUT – OUTPUT PARAMETERS

Each web service contains service profiles that has enough information for a match maker to determine whether this service is suitable for user requirement. In fact, several matchmaking algorithms [10][11] rely only on the matching of Inputs and Outputs of the Service Profiles. In this matchmaking algorithm [12], the input and output of services are represented in ontology. Both requested service and advertised service utilize the same domain ontology. The match degree between advertised and requested services are determined through their levels in ontology. The detailed algorithm is given in Figure 1(a) and the illustration is given in Figure 1(b). In this outR stands for an output of a requested service and outA stands for an output of an advertised service.

The following algorithm can also be applied to find the match between inR and inA. These four degrees [13] as ranked as: Exact > Plugin > Subsumes > Fail.

```
degreeOfMatch(outR,outA)
    if(outA==outR)
        return exact;
    if(outR is a subClassOf outA)
                    return plugin;
    if(outA subsumes outR)
                    return subsume;
    else return fail
```

Figure 1(a): Matchmaking Algorithm



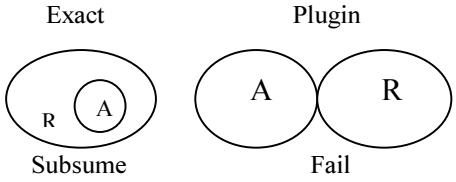


Figure 1(b): Matchmaking Algorithm Illustration

. Here, x > y indicates that x is ranked higher (is a more desirable match) than y. To overcome certain false positives and false negatives, the same algorithm is applied for a bipartite graph which is created using inputs and outputs of requested service and advertised service.

Bipartite Graph: A Bipartite Graph is a graph G = (V, E) in which the vertex set can be partitioned into two disjoint sets, V = V0 ∪ V1,such that every edge e □ E has one vertex in V0 and another in V1.

Matching: A matching of a bipartite graph G = (V,E) is subgraph G′ = (V,E′), E′ □ E, such that no two edges e1, e2 □ E′ share the same vertex. A vertex v is matched if it is incident to an edge in the matching. Given a bipartite graph G = (V0 + V1,E) and its matching G′, the matching is complete if and only if, all vertices in V0 are matched.

A numerical weight is assigned to every edge in the bipartite graph. The weight of an edge, e = (a, b), is a function of the degree of match between concepts a and b. In G = (V0 + V1, E), the values of the edge weights are computed as follows:

Table 1: Weights and categories

Degree of Match	Weight of edge
Exact	w_1
Plugin	w_2
Subsume	w_3

also $w_1<w_2<w_3$

Table 2: Computation of Edge Weights

Journal of Theoretical and Applied Information Technology
30th November 2014. Vol. 69 No.3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Degree of Match	Weight
Exact	$w_1=1$
Plugin	$w_2=(w_1*\|V_0\|)+1$
Subsume	$w_3=(w_2*\|V_0\|)+1$

$|V_0|$ =Cardinality of set V_0

2.1. Input Output Matching Algorithm

Figure 2 defines the search() procedure. It accepts a Query as input and tries to match it with each advertisement in the repository. A match is computed for both, output and input concepts. If the match is not a Fail, it appends the advertisement to the result set. Finally the sorted result set is returned to the client. The match() procedure in Figure 3 accepts two concept-lists as inputs and constructs a bipartite graph using them. It then invokes a hungarian algorithm [14][15]to compute a complete matching on the graph. The match() procedure is invoked twice in search(). The order of Query and Advertisement in each call is however swapped. The computeWeights() function computes the values of w1,w2,w3, depending on the number of concepts in V0. It uses the formulae presented in Fig.3. The degreeOfMatch() function is a call to the reasoner in order to determine the relationship between the two concepts a and b.

```
Result = Empty List
for each Advt in Repository do
    outputMatch=match(Query_out,Advt_out)
    if(outputMatch = Fail) then
        Skip and take next Advt
    end if
    inputMatch=match(Advt_in,Query_in)
    if(inputMatch=Fail) then
        Skip and take next Advt
    end if
    Add (Advt,outputMatch, inputMatch) to the
Result
end for
return sort(Result)
```

Figure2: search (Query)

3. SEMANTIC MATCH MAKING BASED ON BEHAVIOUR

Although an appropriate measurement of degree of match is difficult to define, it is that the result of matching should agree with human intuition. Inputs and outputs sometimes may not provide sufficient information about service's behavior, and relying on them may lead to false results. Every service is described as 4-tuple (T; I;O;Q), [9] where:

```
match(List1,List2)
Initialize Graph G
compute weights(w1,w2,w3) for List1
for each concept a in List1 do
    for each concept b in List2
        degree = degreeOfMatch(a,b)
        if degree ≠ Fail then
```

Figure 3: match ($List_1$,$List_2$)

S(T) is syntactical information of the service which has been taken from service description and service name.

S(I) is a set of input concepts.

S(O) is a set of output concepts.

S(Q) is a set of post-conditions or effects

The preconditions are omitted because before the execution of services, these conditions are checked and it does not supply any information for determining behavior of the service.

3.1. Service Behavioral Graph (SBG)

The ontology can be represented by a multi relational graph where each vertex denotes a concept and each edge denotes a relationship between concepts. This feature of ontology encourages the researcher to view the service as sub graph of ontology.

As per the definition of [9] the Service Behavioral Graph is termed as, Let G be an ontology in its graph representation, G = (V, E) where V is the set of concepts and E is the set of relations of heterogeneous types, where each relation is represented using a pair < L, (V×V) >, where L is the label of the relation (e.g. hasBirthday). A service S is denoted as G_S = (V', E') where V' contained in V and E' contained in E. Elements of V' and E' are identified using the service description. This sub-graph of the ontology is referred as Service Behavioral Graph (SBG). Consider the following two services s1 and s2.

Journal of Theoretical and Applied Information Technology
30th November 2014. Vol. 69 No.3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

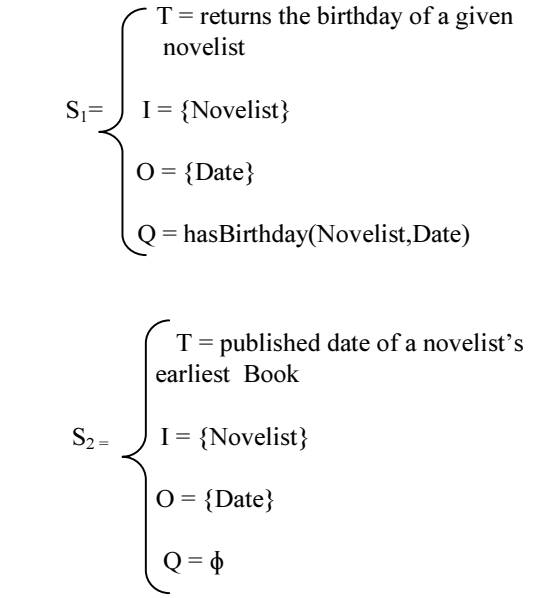


Figure 4: SBGs of S1 and S2

The above figure shows the SBGs of S1 and S2. These graphs can be discovered from the ontology graph using critical elements and behaviorally correct paths, which are defined in the following sections.

3.2. Critical Elements

The mapping from input to output of a service is done through finding several paths between input and output concepts. There may exist multiple paths between pair of I/O concepts. Such paths are determined by several components in the ontology which can be concepts or relations, and are referred as critical elements. Q (Post-condition) and T (Syntactical information) of service descriptions may provide certain information to determine these critical elements.

The syntactical information about the services illuminates the service discovery because it can contribute to find the service behavior by removing unnecessary words using traditional information retrieval technique (TF-IDF)[16]. After removal of unwanted words the core words are extracted and those words are considered as critical elements.

The post conditions are the set of rules that should be true after the execution of the services. It supply some information regarding the service behavior by connecting input to the output of a service.

Figure 5 shows how the weights of the critical elements are computed. The syntactic information (T) about the service is tokenized, stemmed and then using [16] weights are assigned to each token. The weight assigned to each token, specifies the importance of that token. The ontological element corresponding to this token considered as critical elements. Any post condition which its domain and range are from inputs and outputs separately is considered to be critical elements with weight 1.

3.3. Behaviorally Correct Path (BCP)

The path has to be found between inputs to output in the ontology for services. The path can have multiple critical elements. There can be multiple paths that can exist from input to output for a service, but the path which has the maximum weight has been considered as a behavioral path for a particular service. Therefore the behavioral correct path can be defined as maximum weighted path that connects input to the output of a service including critical elements.

```
function CriticalElements(O,S)
    for all o ∈ O do
        o.weight = 0
    end for
    for all S(Q) with domain and range is equal to
input and output respectively
        Set Property weight(O) =1
    end for
    T← Tokenize(S(T))
    T← Stem(T)
    for all t ∈ T do
        w← Tfidf(t)
        E← OntologyElements(O,t) ∩ S(I,O)
            for all e ∈ E do
            e.weight =w
            end for
    end for
    return E
end function
```

Figure 5: Algorithm for determining and weighting the Critical elements

```
function SBG(O,S)
  SBG=Empty Set
  CE← Critical Elements(O,S)
  if |CE| > 0 then
      for all o ∈ S(O) do
          paths← ϕ
```

Journal of Theoretical and Applied Information Technology
30th November 2014. Vol. 69 No.3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Figure 6: SBG Discovery

The above algorithm shows how the behaviorally correct path is discovered. If there are critical elements, for each pair if input and output BCP is determined. The service behavioral graph is thus a set containing these paths. If no critical elements can be identified, SBG will simply be a union of input and output concepts.

4. CASE STUDY

The following table describes the services and their inputs, outputs, syntactical information and effects.The requested service from the user is a service which returns the published date of novelist earliest book. Therefore the requested service is represented as

S= { T= returns the published date of novelist earliest book; I = Novelist; O= Date; Q= φ }

Bipartite Graph is constructed between inputs of requested service and advertised service and outputs of requested service and advertised service. By applying algorithm in Figure 2 and 3 the below results have been obtained. The service which scores lowest is the nearest to the user requirement. In this algorithm, relevant services according to the user need are s1, s2, s3, s5, s4. The service s1 and s2 scores the same. It means that both services are same but in real these two services are different. The input output matchmaking algorithm fails in differentiating these two services.

Services	I/P	O/P	Syntactic Info	Effects
S1	Novelist	Date	Returns the birthday of a given novelist	hasBirthday (novelist, date)
S2	Novelist	Date	Published date of novelist earliest book	φ
S3	Book	Date	Returns the published date of a book	datePublished (Book, Date)
S4	Novel	Writer	Returns the writer of the novel	φ
S5	Novelist	Novel	Returns the novel written by novelist	writtenBy (novel, novelist)

Table 4: I/O Match

S	I/P	O/P	I/P rankg	O/P ranking	Tot score
S1	Novelist – Novelist	Date – Date	Exact	Exact	2
S2	Novelist – Novelist	Date – Date	Exact	Exact	2
S3	Novelist – Book	Date – Date	Plugin	Exact	3
S4	Novelist – Novel	Date – writer	Plugin	Subsume	5
S5	Novelist – Novelist	Date – novel	Exact	Subsume	4

From the service description by applying the second algorithm given in Figure 5, the critical elements of each service have been found and the corresponding scores are calculated using TF-IDF[16] technique. The service behavioural graph is constructed using the algorithm given in Figure 6 and ontology given in[9][17].

Table 3: Example Web Services

Table 5: Weight Calculation

Journal of Theoretical and Applied Information Technology
30th November 2014. Vol. 69 No.3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Service	Critical Elements	Score
S1	Birthday	0.087371
	Given	0.087371
	Novelist	0.083193
	Date	0.055462
	hasBirthday	0.087371
S2	Published	0.056849
	Date	0.063385
	Novelist	0.063385
	Earliest	0.099853
	Book	0.056849
S3	Published	0.049743
	Date	0.083193
	Book	0.149228
	datePublished	0.087371
S4	Writer	0.349485
	Novel	0.19897
S5	Novel	0.149228
	Written	0.087371
	Novelist	0.083193
	writtenBy	0.087371

4.1 Service Behavioural Graph

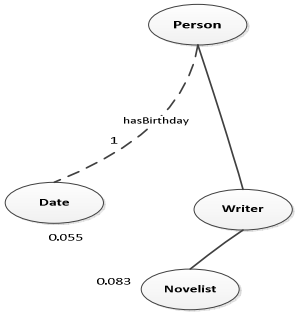


Figure 7(a): SBG of S1

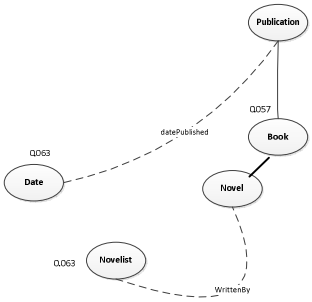


Figure 7(b): SBG of S2

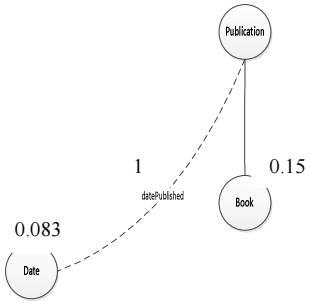


Figure 7(c): SBG of S3

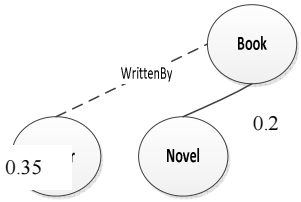


Figure 7(d): SBG of S4

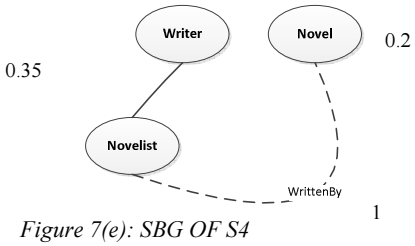


Figure 7(e): SBG OF S4

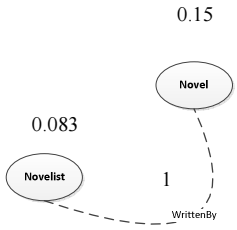


Figure 7(f): SBG OF S5

From the above Figure 7(d) and 7(e), the inferred information is that the SBG for the web service s4 is different. According to the total weight between the critical elements, the first SBG is preferred than the second one. In behaviour match making among several SBGs of the same web service, there is a provision to choose the relevant SBG by applying the score for critical elements. In Input Output Based matchmaking two or more web services can falls in to the same category. The SBG for S1 and S2 are different even though their inputs and outputs are same. Further S1 and S2 can be differentiated using the similarity matching technique given in [18][19][20][21]. Therefore choosing the relevant web services among these

Journal of Theoretical and Applied Information Technology
30th November 2014. Vol. 69 No.3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

services is very difficult. In this paper the two techniques of retrieving web services based on the user specification is analysed. The results of the analysis are given below.

Table 6: Hash Table 1 for IOMAT

Parameters	IOMAT	BEHMAT
Precision	Low(0.5)	High (0.7)
Recall	Same	Same
Relevance	Low (50%)	High(70%)
Grade for the retrieved services	Only categorization is possible	The individual score is calculated
No of services Retrieved according to the user query	One or more relevant web services	The exact result of the query
Execution Time	Low (534ms)	High(741ms)

From the above results, even though the execution time for the BEHMAT is higher compared to IOMAT the BEHMAT performance is high when compared to IOMAT. When the user wants the correct results for their requested service the BEHMAT is highly preferable to retrieve the exact result as the user wants. The base for the semantic match making is input and output of the services. The merit of the I/O match is less execution time. Therefore the I/O match can be used as a filter to reduce the number of relevant services. Based on the merits of these two match making algorithms a new method is proposed to improve the retrieval effectiveness.

5. EXPERIMENTAL EVALUATION

5.1 Experimental Setup:

This approach has been evaluated on data sets. To simulate a real-world scenario, the OWL-S service retrieval test collection OWLS-TC [22] version 3.0 revision 1.0 is considered. This collection contains services retrieved mainly from public IBM UDDI registries, and semi-automatically transformed from WSDL to OWL-S. More specifically, it comprises: (a) a set of ontologies, derived from 7 different domains (education, medical care, food, travel, communication, economy and weapons), comprising a total of 3500 concepts, used to semantically annotate the service parameters, (b) a set of 1007 OWL-S services, (c) a set of 29 sample requests, and (d) the relevance set for each request. For this paper only the domain food and communication is considered.

Based on OWLS-TC V3, data set was synthetically generated to maintain the properties of real-world service descriptions. In particular, a set of approximately 100 services are created, by doing the variations of the services of the original data set. For each original service, randomly one or more input or output parameters are selected, and created a new service description by exchanging and replacing them to have set of services that satisfy IOMAT rules and also set of services that falls under BEHMAT category. A set of 10 requests was generated following the same process, based on the original 29 requests.

After applying IOMAT and BEHMAT, some of the services are identified as identical based on input output parameters and some of the services are behaviourally same. Those services are maintained in a hash table for future reference. The table 7 & 8 shows the structure of the hash table.

Table7: Hash Table 1 for IOMAT

S1	0	S2	1	S6	0	S8	1	S10	0
S7	0	S11	1	S13	0	S15	1	S17	0
S14	0	S23	0						
S18	0	S20	1	S22	0	S30	1	S35	0
S19	0	S40	1	S50	0	S55	1	S60	0
S24	0	S25	0	S95	1	S100	0		

Table 8: Hash Table 2 for BEHMAT

S1	0	S8	1	S10	0
S3	0	S15	1	S20	0
S28	0	S30	1	S35	0
S40	0	S41	1	S50	0

The structure of the hash table is defined as the first field followed by all the fields represents the services that are identical based on input and output parameters. Meaning is that service S1 is similar to S2, S3, S4, S5, S6, S8, S9, S10. Similarly Service S2 is having the same input and output as S1, S3, S4, S5, S6, S8, S9, and S10 and so on. Each service field of the hash table is associated with one more flag field that gives the information about whether the next service is considered as an individual

Journal of Theoretical and Applied Information Technology
30th November 2014. Vol. 69 No.3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

service or continuation of previous service. In the sense in S1 the flag field 0 represents it is an individual service. In S2 the flag 1 represents the next service is the continuation that is from service S2 to S6 is same as service S1. The same principle applies to Hashtable2 also. There the services are behaviourally similar.

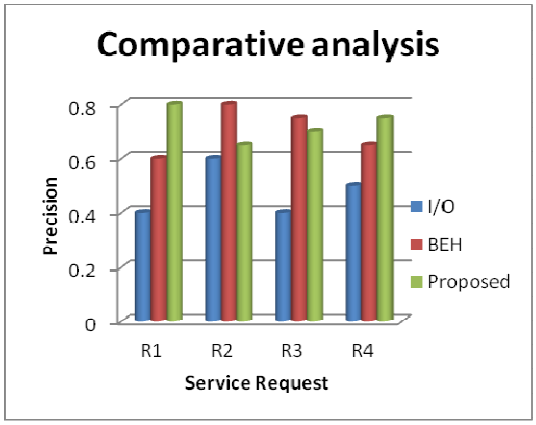


Figure 8: Comparative Analysis

The graph shows that the precision values are higher in BEHMAT and proposed method when compared to IOMAT. In all the above three methods the recall value is same (all the needed web services are retrieved) therefore the separate graph is not provided for recall. The precision plays an important role in web service discovery because precision is defined as the fraction of retrieved documents that are relevant to the search. By analysing the above results the categories of IOMAT algorithms are re formulated to achieve the performance of BEHMAT. The algorithm is given below.

Degreeofmatch(ReqService, Hashtable1,Hashtable 2)

```
Start
If (ReqService in Hashtable1 and Hashtable 2)
   Then Degree=Exact       // alternate services are
available
If(ReqService in Hashtable2)
  Then Degree=Plugin
If(ReqService in Hashtable1)
  Then Degree=Subsume
If(ReqService Not in (Hashtable1 and Hashtable 2))
  Then Degree = Fail                  // Service composition
is necessary
   If(Any Partial Input output Matching)
      then do service composition
   Else Inform to the provider to create new service
End
```

Figure 9: Proposed Algorithm

6. CONCLUSION

This paper analyses the two methods I/OMAT and BEHMAT based on service input, outputs and service Input, Output, Precondition, effect and Syntactical Information respectively. The above two mentioned methods have same recall value that is number of services that are retrieved is same. Behaviour based matchmaking has the highest precision value; it means the number of relevant services that are retrieved is high. But I/O match has the less execution time. Even though the precision of I/OMAT is less compared to BEHMAT, it can be used as a filter to reduce the number of irrelevant services that are retrieved because of its less execution time nature. Based on the observations regarding the merits and demerits of these algorithms new method for match making is proposed. Further this work can be extended to apply similarity based matchmaking on the above mentioned algorithm and also it can be extended to cloud based service discovery. The limitation of the proposed algorithm is , the processing time is more for constructing hash tables based on I/O and behaviour of the services.

REFERENCES

[1] Web Services Description Language (WSDL). http://www.w3.org/TR/wsdl.

[2] Ding Y, Fensel D, Klein MCA , Omelayenko B,The SemanticWeb: yet another hip? Data Knowl Eng ,2002,205–227.

[3] McIlraith S, Son TC, Zeng H , (2001) Semantic Web services. IEEE Intell Sys Special Issue Semantic Web 16(2):46–53.

[4] Bussler C , Fensel D, MaedcheA , Semantic Web enabled Web services. In: Proceedings of the international Semantic Web conference, Sardinia, Italy, June 2002, pp 1–2.

[5] Grigoris Antoniou and Frank Van Harmelen, A Semantic Web Primer.

[6] OWL-S: Semantic Markup for Web Services. http://www.w3.org/Submission/OWL-S/

[7] OASIS. UDDI v3.0,http://uddi.org/pubs/uddiv3.0.1-20031014.htm,2002.

[8] UDDI: The UDDI Technical white paper http://uddi.org/, (2000).

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[9] Alberto Fernández And Zijie Cong, Behavioral Matchmaking of Semantic Web Services.

[10] Amit Gupta , Harin Vadodaria , Umesh Bellur,Semantic Matchmaking Algorithms, Open Access Database www.intechweb.org.

[11] C. Bartolini , J. Gonzalez-Castillo, D. Trastour ,A semantic web approach to service description for matchmaking of services. In Proc. of SWWS,2001

[12]Roshan Kulkarni, Umesh Bellur,(2007),Improved Matchmaking Algorithm for Semantic Web Services Based on Bipartite Graph Matching, Web Services, ICWS 2007.

[13] T. Kawamura, M. Paolucci, , T. Payne, and K. Sycara, Semantic matching of web services capabilities.,The Semantic Web (ISWC 2002), pages 333-347.

[14] H. Kuhn. The Hungarian Method for the Assignment Problem,Naval Research Logistic Quarterly,1955.

[15] K. Nedas. Implementation of Munkres-Kuhn (Hungarian) Algorithm.,http://www.spatial.maine.edu/ Kostas,2005.

[16] http://en.wikipedia.org/wiki/Tf%E2%80%93idf

[17] Protege: Ontology Editor and Knowledge-base framework. http://protege.stanford.edu/.

[18] M. Bouzeghoub J.C. Corrales and D. Grigori, Behavioral matchmaking for service retrieval: Application to conversation protocols. Information Systems, 2008, 33(7-8):681_698.

[19] M. Bouzeghoub J. Corrales, and D. Grigori, BPEL processes matchmaking for service discovery. On the Move to Meaningful Internet Systems 2006: CoopIS, DOA, GADA, and ODBASE, pages 237_254.

[20] Wu, J., Wu, Z, Similarity-based web service matching. In: Proc. of IEEE International Conference on Services Computing, 2005.

[21] X. Dong, A.Y. Halevy, J. Madhavan, E. Nemes, and J. Zhang, Similarity Search for Web Services,VLDB,2004

[22] Owls-tc version 2.2 revision 2. http://projects.semwebcentral.org/projects/owlstc/.