Web Service Discovery Using Combined Bi-term Topic Model and WDAG Similarity

Andreyan Rizky Baskara[1], Riyanarto Sarno[2]
Informatics Department, Faculty of Information Technology
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
andreyan09@mhs.if.its.ac.id[1], riyanarto@if.its.ac.id[2]

***Abstract*— In recent years, many web services had been published by service providers. Finding similar web services to replace existing web services that a business actor owned has become a challenging task. This issue is identified as web service discovery problem. Two approaches to address this problem are measuring the semantic and structural similarity of web services. These approaches are performed by utilizing information in Web Service Definition Language document. This paper proposed a method which combined semantic and structural similarity of web services using Bi-term Topic Model (BTM) and WDAG similarity. In the proposed method, web service structure is modelled into Weighted Directed Acyclic Graph (WDAG). Then BTM is used to mine topic on the modelled WDAG. Jenson-Shannon divergence is used to calculate topic similarity and WDAG similarity is used to calculate the structure similarity of WDAG. The result of experiment shows that the proposed method is applicable for web service discovery with average precision 83.78% and average recall 91.79%.**

Keywords*— BTM; semantic; structural; WDAG; web service discovery

I. INTRODUCTION

Web service can be defined as a software component which is designed to support machine-to-machine interoperability in networks. Web service is used by business actors to support their business process. Business actors can elevate their functionality scale over web and allow them to utilize their resources effectively. Over time, many web service providers that published their web services over network had been rapidly increasing. This condition gives business actors a flexibility to change their used web service depends on their condition like, when they change their business partner or when their current web service goes offline. Though, it also arises an issue on how to find web services that similar to their existing web service among many available web services [1]. This issue identified as Web Service Discovery problem.

Web service discovery has become a popular topic in recent years because of the advance development on Software-Oriented Architecture (SOA) system that make web services become a foundational building blocks for SOA system. Various methods had been applied for web service discovery cases. One of them is web service discovery based on keywords [2]. However, discovery based on keyword similarity is not really effective because of many web services textual description is incomplete and also used of synonym words or too many keywords assigned to a web service. Approaches for web service discovery by measuring similarity between services can be categorized into 3 aspects of measurement which are measuring similarity of semantics, similarity of structure, and similarity of behavior [3].

Web service discovery based on semantic similarity approach is conducted by calculating the similarity degree of contextual in web services. This approach usually used Information Retrieval (IR) techniques i.e. Vector Space Model (VSM) and Latent Semantic Analysis (LSA) [4], [5], [6], [7]. VSM and LSA utilize frequency of words that occurring in document. These methods perform poorly if many synonym words occurred in the documents. Another advanced method on IR domain is topic modelling called Latent Dirichlet Allocation (LDA) [8]. The basic idea of LDA is document consists of many topics and a topic is built from a set words with the same co-occurrence pattern. Some researches had already applied LDA to discovery problem not limited on web services [9], [10], [11]. However LDA has a limitation if it is used on short documents [12].

From case study observation, web services are described using Web Service Definition Language (WSDL). These documents are used as input for the LDA method. WSDL of atomic web services often contain short information on operation name, input and output message, input and output type when it is extracted. If LDA applied using these data as input, then it will perform poorly because of sparsity of word co-occurrence in short documents.

Another aspect for web service discovery is measuring the similarity of structure. WSDL explicitly contain information that can represent web service structure i.e. operation name, input and output message, and types. This information is important in order to differ web services functionalities by understanding their operation input and output. Some researches had been conducted to measure similarity of WSDL structure by using tree matching [13] and Jaccard Structural Similarity [14].

978-1-5386-2827-0/17/$31.00 ©2017 IEEE

Considering the aforementioned background, this paper proposes a method which combines a semantic and structure similarity using Bi-term Topic Model (BTM) and WDAG similarity. BTM is used to overcome the limitation of LDA when applied on short documents such as WSDL. Then WDAG similarity is used to measure the similarity of WSDL structure.

This paper is organized as follows: Section 1 presents an introduction to the background of why using BTM and WDAG similarity as proposed method, Section 2 presents a more detailed explanation on the proposed method, Section 3 presents the experiment result and analysis of the research, and finally a conclusion is given in Section 4.

II. Proposed Method

The proposed method in this paper are consisted of three steps, i.e. (1) WDAG Development, (2) Topic Extraction using BTM, and (3) WDAG similarity calculation. The input of the proposed method is WSDL file as query and WSDL file that stored in repository. The output of the proposed method is a list of names of retrieved web services from repository whose similarity value are higher than given threshold. Fig 1 illustrate the flow of proposed method.

A. WDAG Development

First step, important information i.e. operation name, input message, output message, part name, part type, element name, element type and types, are extracted from WSDL. Then, text pre-processing methods, like tokenization, stop word elimination, stemming and lemmatization is used on the extracted information. After text pre-processing, structure of WSDL is modelled into WDAG.

Weighted Directed Acyclic Graph (WDAG) is defined as a node labeled, arc labeled and arc weighted graph G constructed from a 6 tuple (N, V, L_n, L_v, L_w, r) of a set of nodes N, a set of arcs V, a set of node labels L_n, a set of arc labels L_v, a set of arc weights $L_w = [0,1]$ and element $r \in N$, respectively.

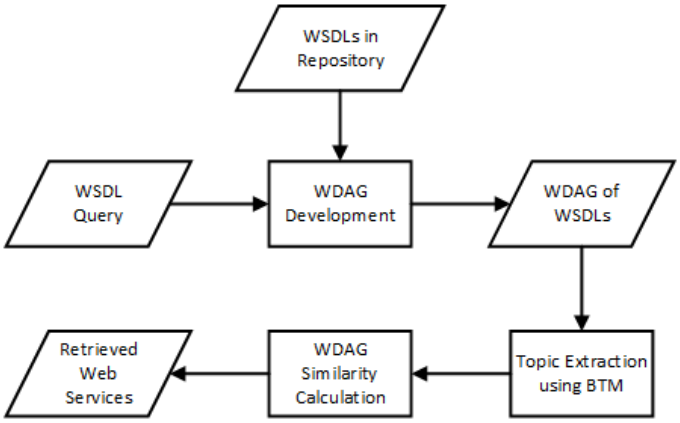

Fig. 1. Flow diagram of proposed method

A schema as showed in Fig. 2 is used to develop WDAG. Then it is serialized into RuleML. RuleML is used as metadata that stores information about WSDL structure in form of WDAG. The pointed arrow line in the schema denotes "has a" relation and regular line denotes "is a" relation.

From schema in Fig. 2, WSDL structure can be defined as follows: (1) an interface has many operations; (2) an operation has one input and one output (3) input and output has one message; (4) a message can have many parts and/or many elements; (5) each part and element have one type; (6) type is either simple type or complex type. (7) complex type can have one or many elements.

WSDL named book_price_service.wsdl taken from dataset as showed in Fig. 3 is used as an example to models its WSDL structure as WDAG. The result of modelled WSDL structure into WDAG is showed in Fig. 4. The weight on a WDAG node arc is 1 divided equally with the total number of its arc because every arc has same importance.

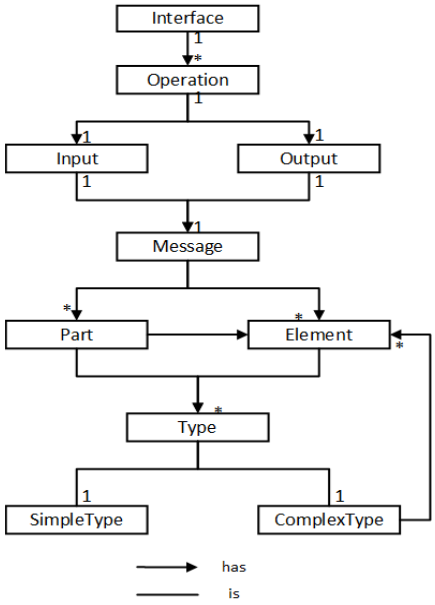

Fig. 2. Schema for WSDL structure

```
            </xsd:simpleType>
            <xsd:simpleType name="Once" sawsdl:modelReference="http://127.0.0.1/ontology/boo
                <xsd:restriction base="xsd:string"/>
            </xsd:simpleType>
            <xsd:simpleType name="Author" sawsdl:modelReference="http://127.0.0.1/ontology/b
                <xsd:restriction base="xsd:string"/>
            </xsd:simpleType>
            <xsd:simpleType name="Title" sawsdl:modelReference="http://127.0.0.1/ontology/bo
                <xsd:restriction base="xsd:string"/>
            </xsd:simpleType>
            <xsd:simpleType name="Publisher" sawsdl:modelReference="http://127.0.0.1/ontolog
                <xsd:restriction base="xsd:string"/>
            </xsd:simpleType>
            <xsd:simpleType name="Book-Type" sawsdl:modelReference="http://127.0.0.1/ontolog
                <xsd:restriction base="xsd:string"/>
            </xsd:simpleType>
        </xsd:schema>
    </wsdl:types>
    <wsdl:message name="get_PriceResponse">
        <wsdl:part name="_Price" type="PriceType">
        </wsdl:part>
    </wsdl:message>
    <wsdl:message name="get_PriceRequest">
        <wsdl:part name="_Book" type="BookType">
        </wsdl:part>
    </wsdl:message>
    <wsdl:portType name="BookPriceSoap">
        <wsdl:operation name="get_Price">
            <wsdl:input message="get_PriceRequest">
            </wsdl:input>
            <wsdl:output message="get_PriceResponse">
            </wsdl:output>
        </wsdl:operation>
    </wsdl:portType>
```

Fig. 3. Book_price_service.wsdl example

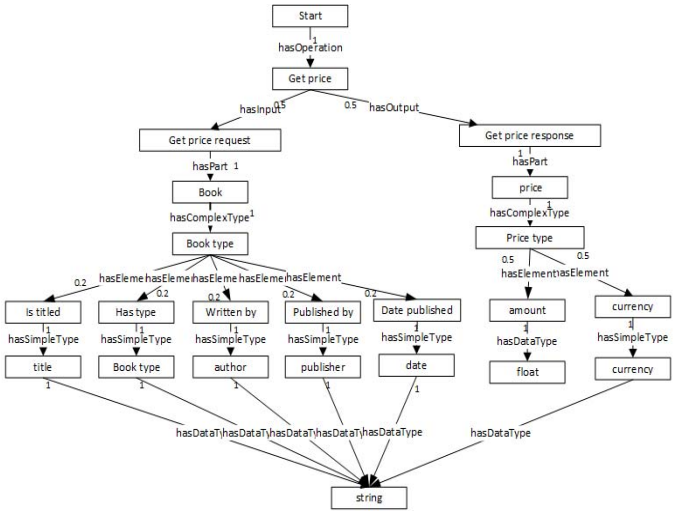

Fig. 4. Illustration of modelled WDAG from example

B. Topic Extraction using BTM

Second step of the proposed method is extracting topic of WDAG node label using Bi-term Topic Model (BTM). BTM is applied to get topics over documents probability distribution. This probability distribution is used to measure contextual similarity of WDAG node label. The generative process of BTM according to [12] are as follows:

1. Initialize topic randomly for all bi-term
2. For *iter* = 1 to N_{iter} do
   a. For $b \in$ set of bi-term B do
      i. Assign topic z_b based on $P(z_i = k|z_{-i}, B)$
      ii. Update n_z, $n_{w_{i|z}}$, dan $n_{w_{j|z}}$
3. Calculate parameter θ dan φ

Bi-term is defined as un-ordered word pair which is co-occurring in a short context. For example, if a short text has word1, word2, and word3 then the created bi-term corpus is {word1, word2}, {word2, word3}, {word1, word3}.

Gibbs sampling is used on BTM to estimate topic assignment of a bi-term. New topic is assigned to a bi-term if the random sampling probability of that bi-term satisfies the conditional probability as in (1).

$$P(z_i = k|z_{-i}, B) \propto (n_{-i,k} + \alpha) \frac{(n_{-i,w_i,1|k} + \beta)(n_{-i,w_i,2|k} + \beta)}{(n_{-i,\cdot|k} + W\beta)^2} \quad (1)$$

After the estimation process using Gibbs Sampling is finished, then probability distribution of word over topic (φ) and probability distribution of topic over document (θ) is calculated using (2) and (3) respectively.

$$\varphi_{k,w} = \frac{n_{w|k} + \beta}{n_{\cdot|k} + W\beta} \quad (2)$$

$$\theta_k = \frac{n_k + \alpha}{N_B + K\alpha} \quad (3)$$

Where n_k total number of bi-term on topic k, $n_{w|k}$ is number of word w given topic k, and $n_{\cdot|k}$ is number of words on topic k. In the proposed method, only probability distribution of topic over document (θ) will be used as input to calculate contextual similarity of WDAG node label. Illustration of generated probability distribution of topic over document with 30 topics is showed in TABLE I.

C. WDAG Similarity Calculation

The last step of the proposed method is WDAG similarity calculation. WDAG query modelled from previous step is compared to WDAG of services in repository using WDAG similarity method. If the similarity value is greater than a given threshold then the web service associated with the WDAG is returned to user as a list of retrieved web services.

WDAG similarity is a method to measure similarity between two Weighted Directed Acyclic Graph (WDAG). WDAG Similarity algorithm started by traversing the most top root node between two WDAG which has identical label and then traverse its child node by left-right depth first traversal method. If the root node label is not identical, then WDAG similarity = 0. Else, the algorithm travers the WDAG until leaf node found.

To determine whether two traversed WDAG node label is identical or not, we used the probability distribution of topic over document (θ) generated from previous step as input to calculate the similarity of WDAG node label. Jenson-Shannon (JS) divergence is used to calculate the distance between two topic probability distribution as in (4).

TABLE I. ILLUSTRATION OF GENERATED TOPIC PROBABILITY DISTRIBUTION USING BTM

Node Label	Topic 1	Topic 2	Topic 3	Topic 4	…..	Topic 30
get price	2.81E-10	4.06E-11	8.09E-10	4.65E-11	…..	6.76E-06
get price request	5.38E-10	7.77E-11	1.55E-09	8.90E-11	…..	0.0875319
get price response	0.1021	0.2001	0.3768	0.321	…..	0.0820828
car	0.4897	0.2541	0.1739	0.0823	…..	0.000261379
get price	2.81E-10	4.06E-11	8.09E-10	4.65E-11	…..	6.76E-06
get price request	5.38E-10	7.77E-11	1.55E-09	8.90E-11	…..	0.0875319
….	….	….	….	….	….	….
once	0.0333544	0.0332695	0.0335392	0.0332716	…..	0.0333059

$$D_{JS}(q,d) = \frac{1}{2}D_{KL}\left(q,\frac{q+d}{2}\right) + \frac{1}{2}D_{KL}\left(d,\frac{q+d}{2}\right) \quad (4)$$

The distance score is normalized into similarity score then it is compared to a threshold. If the similarity score is higher than threshold then the compared WDAG node label is identical.

WDAG similarity can be done recursively and the base of recursive is when both of the compared nodes are leaf node. If the leaf node label is identical then the WDAG similarity = 1 else WDAG similarity = 0. Later, this similarity is used to calculate the parent node similarity. If there is an arc missing in one of the compared WDAG then WDAG simplicity calculation is used to measure the missing node similarity. Then the similarity of the missing node is calculated as WDAG simplicity times 0.5. WDAG similarity [15] for each node traversal can be calculated using (5) and WDAG Simplicity for each missing arc can be calculated using (6).

$$\begin{cases} 0.0, & root\ node label\ g_i\ and\ g_j^{'}\ is\ not\ identical \\ 1.0, & g_i\ and\ g_j^{'}\ is\ leaf\ node \\ \sum \begin{cases} wDAGsim(g_i,g_j^{'}) \bullet \dfrac{(w_i + w_j^{'})}{2}, & if\ g_i\ and\ g_j^{'}\ is\ exist \\ wDAGsim(g_i,\varepsilon) \bullet \dfrac{(w_i + 0)}{2}, & if\ g_j^{'}\ is\ not\ exist \\ wDAGsim(\varepsilon,g_j^{'}) \bullet \dfrac{(0 + w_j^{'})}{2}, & if\ g_i\ is\ not\ exist \end{cases} \\ \sum_{j=1}^{breadth\_of\_g^{'}} wDAGsim(\varepsilon_i,g_j^{'}) \bullet \dfrac{(0 + w_j^{'})}{2}, & if\ node\ g_i\ is\ leaf\ node \\ \sum_{i=1}^{breadth\_of\_g} wDAGsim(g_i,\varepsilon) \bullet \dfrac{(w_i + 0)}{2}, & if\ node\ g_j^{'}\ is\ leaf\ node \end{cases} \quad (5)$$

$$wDAGplicity(g) = \begin{cases} D_1, & if\ g\ is\ leaf\ node \\ \dfrac{D_f}{m}\sum_{j=1}^{m} w_j \bullet wDAGplicity(g_j) \end{cases} \quad (6)$$

III. EXPERIMENT AND RESULT

The proposed method is applied to a public available dataset called SAWSDL-TC3 test collection. This dataset is widely used by previous researchers such as in [16] and [17]. In this research, the semantic annotations on SAWSDL-TC3 dataset are ignored and only the WSDL interfaces are used. The experiment used 154 WSDL files which are taken from SAWSDL-TC3 dataset. Five WSDL files are selected randomly and used as query from set of available queries in dataset to evaluate the performance of proposed method. TABLE III shows the list of WSDL used as query. The threshold given for similarity is 0.7 based on experiment observation.

TABLE III. LIST OF WSDL FILE AS QUERY

No.	WSDL file name
1.	1personbicyclecar_price_service.wsdl
2.	book_price_service.wsdl
3.	bookpersoncreditcardaccount__service.wsdl
4.	title_comedyfilm_service.wsdl
5.	title_videomedia_service.wsdl

The proposed method performance is evaluated using two metrics which are precision and recall. Precision and recall are two most used metrics to measure the performance of a system or method that is used on information retrieval domain. Precision is how many relevant information is retrieved from all of retrieved information by system. Recall is how many relevant information is retrieved by the system from all relevant information stored in repository. Precision can be calculated using (7) and Recall can be calculated using (8).

$$Precision = \frac{\sum_i \#(Relevant_i \wedge Retrieved_i)}{\sum_i \#Retrieved_i}\% \quad (7)$$

$$Recall = \frac{\sum_i \#(Relevant_i \wedge Retrieved_i)}{\sum_i \#Relevant_i}\% \quad (8)$$

The proposed method is compared to each one of the method separately. Experiment result is showed in TABLE II. Semantic only web service discovery using BTM method perform poorly because many web services in dataset have same contextual topic. For example, book_price_service.wsdl and bookpersoncreditcardaccount_service.wsdl have same context. It is about book service and the important information inside those WSDLs are using book word. So, their similarity is high. However, they have different functionality. So, they will become a false candidate if one of them become the query. Structure only web service discovery using WDAG similarity method also perform poorly than proposed method because WDAG similarity method used exact string matching to measure the node label. If an operation in web service used

TABLE II. PRECISION AND RECALL OF EXPERIMENT RESULT

No.	Semantic only using BTM		Structure only using WDAG Similarity		Proposed Method	
	Precision	*Recall*	*Precision*	*Recall*	*Precision*	*Recall*
Query 1	16.90	100	75	23.07	80	92.31
Query 2	28.07	100	66.67	20	80	100
Query 3	13.51	100	66.67	40	71.43	100
Query 4	11.48	100	100	33.33	87.5	100
Query 5	8.20	83.33	100	42.86	100	66.67
Average	15.63	96.66	81.68	31.85	83.78	91.79

different word but it has same context with the compared web service then the matching result will return false. The proposed method can cover these weaknesses.

IV. Conclusion

This paper had proposed a method to discover web services by utilizing semantic and structure information of WSDL. The proposed method combined topic modelling and WDAG similarity measurement to discover services. Weighted Directed Acyclic Graph (WDAG) is used to model structure of web service interface. Then WDAG similarity is used to measure the similarity between two web service structure. Bi-term Topic Model (BTM) is used to mine underlying topic on WDAG node label. Then the generated topic is used to measure the similarity of compared WDAG node label while traversing the WDAG. The experiment result shows that the proposed method perform better than separated method with average precision 83.78 % and average recall 91.79 %.

References

[1] J. U. Maheswari, "Comparison of Web Service Similarity- Assessment Methods," *International Journal of Computer Application*, vol. 98, no. 22, pp. 18–23, 2014.

[2] Z. Qin, P. Li, Q. Zhu, and C. Tian, "SWEE : Approximately searching web service with keywords effectively and efficiently," in *2nd International Conference on Advanced Computer Control*, 2010, pp. 569–574.

[3] M. Bravo, "SIMILARITY MEASURES FOR WEB SERVICE," *International Journal on Web Service Computing*, vol. 5, no. 1, pp. 1–16, 2014.

[4] A. Halevy, E. Nemes, X. Dong, J. Madhavan, and J. Zhang, "Similarity Search for Web Services," in *Proceedings of the 30th VLDB Conference*, 2004, pp. 372–383.

[5] C. Platzer and S. Dustdar, "A Vector Space Search Engine for Web Services," in *Proceedings of the Third European Conference on Web Services*, 2005, pp. 62–71.

[6] J. Hou, J. Zhang, R. Nayak, and A. Bose, "Semantics-Based Web Service Discovery Using," in *International Workshop of the Initiative for the Evaluation of XML Retrieval*, 2010, pp. 336–346.

[7] C. Wu, V. Potdar, and E. Chang, "Latent Semantic Analysis – The Dynamics of Semantics Web Services Discovery," *Lecture Notes in Computer Science on Advances Web Semantic*, vol. 4891, pp. 346–373, 2008.

[8] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet Allocation," *Journal of Machine Learning Research*, vol. 3, no. 4–5, pp. 993–1022, 2003.

[9] A. R. Baskara, R. Sarno, and A. Solichah, "Discovering Traceability between Business Process and Software Component using Latent Dirichlet Allocation," in *International Conference on Informatics and Computing*, 2016, no. 1.

[10] C. Li, R. Zhang, J. Huai, X. Guo, and H. Sun, "A Probabilistic Approach for Web Service Discovery," in *IEEE International Conference on Services Computing*, 2013, pp. 49–56.

[11] N. Zhang, J. Wang, K. He, and Z. Li, "An Approach of Service Discovery based on Service Goal Clustering," in *International Conference on Services Computing*, 2016, pp. 114–121.

[12] X. Yan, J. Guo, Y. Lan, and X. Cheng, "A Biterm Topic Model for Short Texts," in *Proceedings of the 22nd international conference on World Wide Web*, 2013, pp. 1445–1455.

[13] I. Zinnikus, H.-J. Rupp, and Kl. Fischer, "Detecting Similarities between Web Service Interfaces: the WSDL Analyzer," in *Interoperability for Enterprise Software and Applications: Proceedings of the Workshops and the Doctorial Symposium of the Second IFAC/IFIP I-ESA International Conference*, 2010.

[14] R. Sarno, E. W. Pamungkas, D. Sunaryono, and Sarwosri, "Workflow Common Fragments Extraction Based on WSDL Similarity and Graph Dependency," in *International Seminar on Intelligent Technology and Its Applications Workflow (ISITIA)*, 2015, pp. 309–314.

[15] I. G. Anugrah and R. Sarno, "Business Process Model Similarity Analysis Using Hybrid PLSA and WDAG Methods," in *International Conference on Information, Communication Technology and System (ICTS)*, 2016, pp. 231–236.

[16] M. Aznag, M. Quafafou, and Z. Jarir, "Correlated Topic Model for Web Services Ranking," *Int. J. Adv. Comput. Sci. Appl.*, vol. 4, no. 6, pp. 283–291, 2013.

[17] Y. Lei and P. S. Yu, "Service Topic Model with Probability Distance," in *9th International Conference on Utility and Cloud Computing*, 2016, pp. 202–207.

This page intentionally left blank