A Hybrid Similarity Measure for Dynamic Service Discovery and Composition based on Mobile Agents

Naoufal EL ALLALI, Mourad FARISS, Hakima ASAIDI, Mohamed BELLOUKI
Department of Computer Science, Mohammed First University
FPD Nador Laboratory MASI
Nador, Morocco

***Abstract*—With the ever-present competition among companies, the prevalence of web services (WSs) is increasing dramatically. This leads to the diversity of the similar services and their developed nature, which makes the discovery of a relevant service during the composition phase a complex task. Since most of the competition companies aim to discover high-quality services with minimum charges in order to increase the number of customers and their profit. The semantic WSs allow performing dynamic service discovery through the entities software and intelligent agents. However, the solutions provided to the discovery process are limited to their performance in terms of the quickness to respond to the request in real-time, without considering the constraints such as the accuracy in the discovery phase and the quality of the similarity mechanism evaluation. They usually are based on the similarity measure of distance between concepts in the ontology instead of taking into consideration the relationships semantically and the strength of the semantic relationship between concepts in the context. In this paper, we proposed a novel hybrid semantic similarity method to improve the service discovery process. The hybrid method is applied to an architecture based on mobile agents, where cooperative agents are integrated to facilitate and speed up the discovery process. In the first hybrid method, we defined the Latent Semantic Analysis (LSA) with a semantic relatedness measure to avoid the ambiguity of the terms and obtain a purely semantic relatedness at level of the service description. The second one is defined to analyze the relationships at the level of the I/O service based on the subsumption reasoning, called IO-MATCHING. Experimental results on a real data set demonstrate that our solution outperforms the state-of-the-art approaches in terms of precision, recall, F-measure, and consumed time of the service discovery.**

Keywords*—IO-MATCHING; latent semantic analysis; mobile agents; OWL-S; semantic web services; semantic similarity; semantic relatedness

I. INTRODUCTION

Over the last years has become widely popular as the number of Web services deployed in the world is rapidly increasing owing to their low-cost and cross-organizational construction of distributed applications in heterogeneous environments [1]. In another term, as the number of WSs increases, the discovery of web services needed by the user becomes more and more critical [2, 3]. However, the requested information and knowledge from the data remain difficult to obtain precisely. Since there are some conventional approaches based on WSDL [4] as the description of Web Service, it provides limited results due to lack of semantic service description. Contrary to other service descriptions such as OWL-S [5], WSMO [6] and SAWSDL [7], which are based on the semantic description of web services. Thus, The Semantic Web Services (SWS) concept is the result of integrating Web services and Semantic Web technologies [8].

The key point behind integrating Web Service and Web semantic is developing intelligent service-based applications and carrying out high-precision semantic discovery and automated service composition based on formal ontology-based service semantics representations [9, 10]. These service-based applications can reason based on such formal service semantics. This can support not only semantic interoperability between services, but also planning of their logic-based automated composition and more precision service discovery [11]. Thus, the process of service discovery and composition is generally based on service description, increasingly beyond syntactic descriptions to incorporate the semantics of the service to enable more accurate analysis.

With the advancement of semantic technology in web services has become more attractive to researchers in recent years due to the importance of existing web services on the Internet [12]. However, that does not mean there are no complex challenges confronting researchers to improve web service discovery in real-time. Since some solutions [13–15] aim to minimize the execution time of web service discovery but generally lead to low productivity with marginal performance, they do not target semantic analysis of the request to achieve an accurate solution, making it challenging to realize the semantic web discovery process. Most of these solutions are based on the distance between two concepts of the ontology to measure the degree of similarity rather than to consider the semantic relatedness existing between these two concepts in a contextual way.

The crucial issue in the discovery process is that consists on the way to measure the correspondence ratio between the request and the service concepts, and also the semantic correlation strength between both. So that the semantic similarity and the semantic relatedness are two different concepts, because the semantic relatedness includes the strength of relationship between two concepts in a context, while the concept of semantic similarity is more specific than the semantic relatedness [16]. The semantic similarity is done by evaluating two concepts in a taxonomy or ontology, which are constructed only by "is a" relations. For example, "book" is similar to "novel", but is also related to "author" and "publication". Thus, more the similarity between two concepts

is higher, more the relatedness is increased in the given context [17]. For this reason, there are some semantic discovery methods based on simple matching of the concepts annotated to services and requests, without considering the relationship of these concepts to the desired service context, rather than a simple semantic matching of terms that are related to I/O. This is considered to be insufficient to improve performance either in the semantic discovery process or during composition, producing results according to the similarity ratio between terms without taking into account the semantic relatedness of the terms to the desired service.

In this paper, we proposed a novel approach that addresses the service discovery problem based on a cooperative system by mobile agents developed in [18]. This approach aims to analyze the semantic services in a contextual way. The provided approach takes into account all the constraints discussed in the above paragraphs. In particular, the novelties of our proposal are:

- The use of the parallelism of the agent technology to make the service discovery process more efficient and dynamic.
- The cooperative agents enable the semantic analysis of the request autonomously to improve the accuracy rate.
- The integration of a semantic analysis agent in order to facilitate the retrieval of ontology relationships between concepts and to enhance the performance of the discovery process. This integration provided to the proposed module by [18], which allows reinforcing in a robust and more flexible in responding to any point of the execution, enables to achieve better performance and lower memory consumption to select the composition of the services dynamically.
- The support of a secondary database to improve the quickness and reliability of semantic analysis without reproducing the extraction of ontological relations between concepts.
- The semantic analysis agent targets to extract the semantic relatedness strength between the wanted keywords and the service description, in order to return the service in context for responding the desired request to be realized.
- The use of a hybrid similarity method proposed to maximize the matching process between the query and service.
- The first hybrid similarity measurement method is a classical tool to retrieve the description services similarities automatically; through dimensionality compression, it is known as Latent Semantic Analysis [19]. In addition, the semantic relatedness between the concepts and the desired service is performed to support the LSA.
- The second hybrid method is based on the relationship between the input/output (I/O) concepts in their OWL ontologies; it is known as IO-MATCHING [20].

The experimental results on a real dataset demonstrate that our solution outperforms the state-of-the-art approaches in terms of precision, recall, F-measure and consumed the time of the service discovery.

The remainder of this paper is organized as follows: Section 2 introduces the related works, Section 3 presents our proposal approach, Section 4 demonstrates the experiment results and discussion, and the final section concludes the paper and the future work.

II. Related Work

With the radical proliferation occurring in web services technology, it is becoming difficult to discover a service that is adequate to the user's requirements. For that reason, there are many solutions to reinforce the service discovery problem in terms of functionality and QoS. In this regard, we present only related works to achieve a better understanding of the advantages that can be obtained and put our contributions in context.

The technique suggested in [18] provided a method for discovering and composing SWSs in a distributed environment. This technique is based on a mobile agent, which has the characteristics of self-reliance, social capability, self-learning, and, most importantly, mobility. It is a technology suitable for autonomously exploiting SWSs to provide end-user applications. The mobile agent aims to discover the SWS desired from different locations and the generated graphs to perform the composition process. Despite a sufficient result provided by the discovery process, it may provide relevant services, but it does mean that there is no accurate measure to find a service that satisfies the user's requirement.

The authors [21] suggested an approach to automatically compose web services based on multi-agent systems and an algorithm to dynamically select an optimal solution as a service that responds to the customer's requirements. This composition is based on the quality and composition-capacity of the participating services. They aim to design, deploy and manage distributed systems more efficiently by combining, reorganizing, and adapting the services. Despite the efficiency feasibility provided by their proposed module, it does not cover some evaluation metrics and the performance of the similarity semantic method during composition.

The work of [22] proposed a new WSs discovery method based on semantic matching and service clustering for effective and practical web services discovery, which integrates functional similarity with process similarity. Their suggested approach is based on the knowledge available from the semantic description model, based on improving Lin's [23] semantic similarity measure to include opposition or degree of contrast as specified in [24]. Their vision is to develop a practical WS discovery approach based on pre-clustering that enable them to perform semantic and scalable WS discovery in a short period and thus minimize the search space. However, their method similarity proposed is not accurate to describe semantically due to the ignorance of the two concepts' antisense relationships.

A new semantic similarity method is proposed by the authors [25] that may be performed on both the textual

description and the interface of WSs. Their proposed semantic similarity method incorporates multi-conceptual relationships for service discovery. It is based on the relational semantic distance between concepts in WordNet and other ontologies. This method provides a more accurate estimation of the similarity between the terms, the web services and the query. Although the experimental results are promising in terms of precision, recall and f-measure, but it is limited to the semantic similarity of the terms based on the generic WordNet ontology.

In [26], a proposed method for discovering and selecting WSs that use OWL-S to represent web services, quality of service, and customer demand. This architecture is built on system-multi-agent approaches that make use of semantic web services. Their proposed technique discovers services similar to the consumer request based on functional and QoS parallels and reputation computing. Their model is based on four-layers: the web service and request description layer, the functional match layer, the QoS computing layer, and the reputation computing layer. Their Future work includes combining several Web services into an atomic service (service composite) and composes Web services based on customer preferences and QoS.

The authors in [27] suggested an automated approach to discovering semantic Web services. It is characterized by an ontology-based service preprocessor, a reasoning-based service filter, and a parameter-based matcher of the service. The first uses the ontology defined by services and requests to reduce the number of candidate services. The second one consists basically on a reasoning-based service filter to extract the concepts tagged to the input and output parameters of the selected services from the SAWSDL set of documents. Consequently, it logically deduces the concepts and filters out the services that are insufficient to satisfy the user's parameter needs. Finally, the third one is a parameter-based service matcher based on the measure of semantic similarity in the matching algorithm (PBSM_R). This semantic similarity measure is mainly based on the relationship between the concepts of the domain ontology. Lastly, it returns services adequate to the requirements of the user. Although the results achieved in performance of the runtime through the narrowing the search space, but it lacks on one side precision and recall.

The authors [28] proposed a novel service discovery scheme based on a combination of similarity methods using the WSDL specification and ontology to make the service discovery process more automated, discover the best match rapidly, and improve the Hungarian algorithm [29] is used. This method combination includes the structural similarity, the semantic similarity and the concept similarity based on bipartite matchmaking techniques used to discover web services. This suggested scheme includes two phases to discover the most suitable services to the request. In the first phase, measuring similarity between the requested service and a set of advertised services. In the second phase, a bipartite graph of nodes defined based on the ontology is used to describe semantic Web service matching. The obtained experimental results are better than other existing schemes using the Hungarian algorithm in terms of precision, recall and f-measure, but it is lacked parallelizing some steps in the discovery process.

III. The Proposed Approach

This section presents an approach for supporting the discovery process during web service composition using the cooperative agents, which targets the minimization of the discovery performance overhead without requiring the memory pre-loading of service registries. Moreover, the maximization of the matching algorithm by the hybrid method proposed. This method consists in retrieving the context of the terms in the service description, where the service context can provide more accurate information regarding the services that are relevant to the request. The main novelty of our proposed discovery system aims to be self-adapting to unexpected variations, particularly in a heterogeneous environment.

Service discovery is a crucial issue to accomplish at each major step of the composition generation process. However, the increasing number of services on the Internet, the dynamic and unstable nature of these entities makes the composition tasks more difficult. Therefore, it considers the process of discovering a service during composition most important to ensure the system has the ability to semantically parse, respond to a request quickly and accurately in real-time. In order to have an efficient system for identifying the best solutions, we adopted the architecture developed by the authors [18], but with the incorporation of our hybrid semantic similarity measurement method and an agent to analyze ontological relationships as illustrated in Fig. 1, it can lead to successful results as detailed in the next section. This architecture is based on system multi-agent (SMA) for the discovery of Web services. Our contribution to this architecture is to improve the semantic similarity method's efficiency in the service discovery process.

Fig. 1 represents an improvement of the author's architecture [18], based on a primary agent in the distributed environment called a Mobile agent. This agent is used to discover the desired SWS in different locations and graphs to accomplish the composition process. This architecture includes the main entities to support the discovery process through the exploitation of the ontology domain used and facilitate the search process, which is corresponding between the request and the service offers. In the following, we will explain more details what happens in each entity.

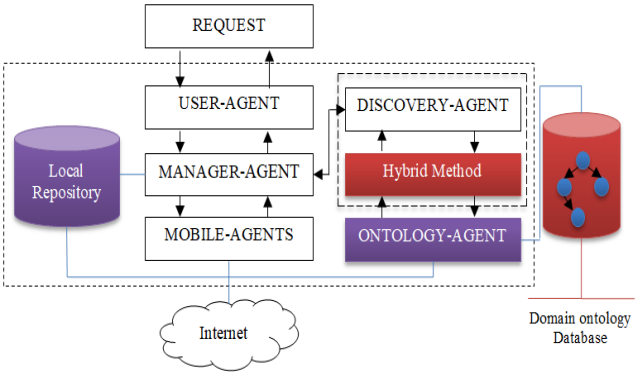


Fig. 1. Integration of the Proposed Solution for Mobile Agent Architecture to Discover and Compose SWS.

- USER-AGENT: is a point to interact between the request and the service discovery system. This agent is responsible for providing the user with an OWL-S standard semantic module [5] to express the request. The user's request is composed of inputs/outputs, a reference to the domain ontology to be used, and after the processing returns the desired results to the user.
- MANAGER-AGENT: Checks the availability of the desired service index in the local repository or whether the service can be composed of the local repository services. In the absence of the desired service, it is up to a set of mobile agents to search in different locations on the web to find the desired service.
- MOBILE-AGENTS: are responsible for retrieving the semantic web services from different websites instead of using a crawler due to their speed performance and low network overhead. It satisfies the needs of "MANAGER-MOBILE" to reinforce these services during the composition.
- ONTOLOGY-AGENT: is designed to facilitate the semantic analysis of the I/O of the service required by the discoverer agent (as illustrated in Fig. 2). It is considered as a cooperative agent. It analyzes the ontology domain that corresponds to the request of the discovery agent. Thus, it extracts the classes and their links to deduce the generalization relations between the concepts, which means a concept is more general than another in the arborescence (as shown in Fig. 3). These domain ontologies are stored in the domain ontology database.

This agent can exploit the relations already deducted in advance that are stored in the secondary database. This database is developed as a memory cache to avoid spending more interaction and extract these relations quickly without reproducing the operation of processing analysis.

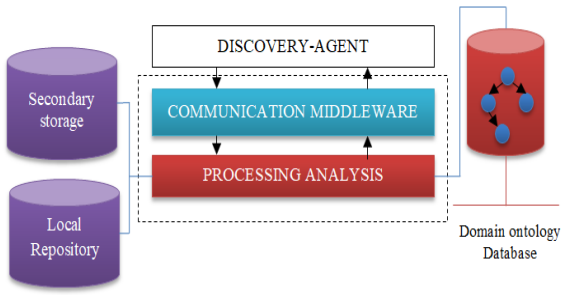


Fig. 2. Components of "ONTOLOGY-AGENT".

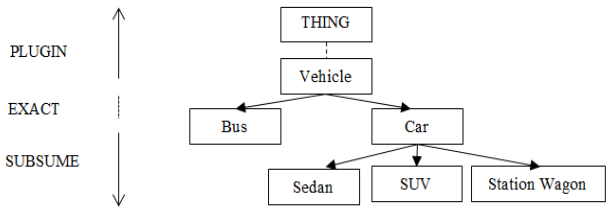


Fig. 3. A Vehicle Ontology Fragment [20].

- DISCOVERY-AGENT: allows discovering semantic web services that fulfills the requirements of the manager agent. The discovery process is based on the mobilization of the hybrid measurement similarity method and the "ontology-agent". The hybrid method integrated into the discovery agent are the following: The first method, based on the LSA, aims to investigate the relationships between a set of service descriptions and the terms embedded, by producing a set of concepts related to the service descriptions and the terms. In addition, to analyze the semantic relatedness between the concepts and the desired service. The second method, based on IO-MATCHING, intends to describe the degree of matching between two I/O concepts using the ontology agent.

To maximize the contextual similarity of service discovery, we propose a hybrid method which focuses on the semantic similarity between the input/output concepts of services, and to find the semantic relatedness between the service description terms in a contextual way. This is intended to facilitate the performance of "ontology-agents" to cooperate in a more intelligent and explicit sense with other agents. In the following, we will describe in more detail the different definitions to clarify the mechanism of similarity provided.

Definition 1 (Request): the request of the user is defined as $R = \langle R_{in} | R_{out} | R_{des} \rangle$, where R_{in} denotes the set of required input parameters, R_{out} denotes the set of required output parameters , and R_{des} denotes the required service description.

Definition 2 (Web Service): A web service is described by the OWL-S ontology. The service defined as a 3-tuple: $S = \langle S_{in} | S_{out} | S_{des} \rangle$, where S_{in} and S_{out} are the input and output concepts respectively, S_{des} is the description of the service.

Definition 3 (LSA): Latent Semantic Analysis is used to discover the hidden and subjacent (latent) semantics of words in a corpus of documents by constructing "concepts" related to documents and terms. It is a standard technique to extract automatically similarities between documents, by reducing the dimensionality. A word-document matrix is packed with weights according to the extent of the word in the specific document and is then reduced by singular value decomposition to a reduced dimensional space called conceptual space. The LSA process includes four steps illustrated in Fig. 4 as follows.

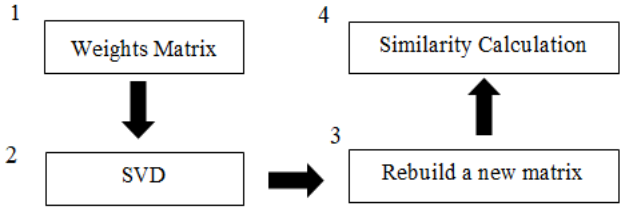


Fig. 4. The LSA Algorithm Processing Steps.

Step 1: Before building a weights matrix, the short text of the service description must be treated in the pre-processing phase as normalization of the service description to reduce the information ambiguity. The next phase is the tokenization task,

which consists of transforming the short text existing in S_{des} to a set of separate terms as a set of tokens. After finishing the tokenization task, it will be the stemming task to convert different forms of terms into a similar canonical form. Before finishing the pre-processing task, the terms must be sorted alphabetically. This step can be completed by building a weigh matrix as illustrated in equation (1) bellow.

$$A=\begin{pmatrix} & t_1 & \cdots & t_j \\ s_1 & w_{1,1} & \cdots & w_{1,j} \\ \vdots & \vdots & \ddots & \vdots \\ s_i & w_{i,1} & \cdots & w_{i,j} \end{pmatrix} \quad (1)$$

where $W_{i,j}$ represents the weight of the term i in the service j. The Term Frequency-Inverse Document Frequency (TF-IDF) can be calculated as:

$$W_{ij} = TF_{i,j}(T,S) \times IDF(T) = \frac{n_{i,j}}{\sum_k n_{k,j}} \times \log\left(1+\frac{N}{df_i}\right) \quad (2)$$

where $n_{i,j}$ is the number of occurrences of the term t in the service, N is the total number of services in the corpus, and df_i is the number of services where the term T_i occurs.

Step 2: After the generation of weight matrix, it follows the step of decomposition of matrix A by SVD as illustrated in equation (2),

$$A=U\sum V^T \quad (3)$$

where,

U Term matrix.

$\sum$ Descriptions service matrix.

V^T Singular value matrix.

Step 3: Once the matrix A is decomposed by SVD, we have to reduce the vector space to an approximation with a rank of $k=4$, it becomes to find a service description closest to the request. A request is represented in the k-dimensional vector space as a service. A request (R) can be represented as follows:

$$R_{des} = R_{des}{}^T U_k \sum\nolimits_K^{-1} \quad (4)$$

Step 4: Then, we need to measure the cosine similarity to evaluate the similarity between the query description and the service description. The cosine similarity measure is defined as follows.

$$sim(R_{des}, S_{des}) = \frac{R_{des} \cdot S_{des}}{\| R_{des} \| \| S_{des} \|} \in [0,1] \quad (5)$$

Definition 4 (Description Similarity): A service description S_{des} is a short text which describes the typical properties of a service. The service descriptions include rich information to be evaluated semantically. To calculate the description similarity (DS) will be evaluated by the similarity of the hidden topics using the LSA method as mentioned in definition 3, additionally evaluating the correlation rate to the I/O concepts with the service description. These concepts are considered essential keywords to improve the precision rate. The description semantic similarity is defined as follows.

$$sim_{DS}(R_{des}, S_{des}) = \delta \times sim(R_{des}, S_{des}) + (1-\delta) \times Relatedness(R_{des}, S_{des}) \quad (6)$$

where $\delta \in [0,1]$ is weight factor of the LSA similarity and $Relatedness(R_{des}, S_{des})$ is the semantic relatedness.

To infer the semantic relatedness between the wanted keywords and the service description, our ontology method mentioned in definition five correlates the I/O concepts to get the diversification of the service description related to these concepts in a semantically precise way. We define the semantic relatedness method as follows.

$$Relatedness(R_{des}, S_{des}) = \begin{cases} e^{-((1-\frac{\ln(1+n_k)}{1+\ln(n)})\times(\frac{T(T_R,T_S)}{\max(W_{T_R},W_{T_S})}))} & \text{if } T(T_R,T_S) \neq 0 \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

where $Relatedness(R_{des}, S_{des})$ is normalized in the range [0,1], n_k is the number of occurrences that a combination of terms appears in the service descriptions, and n is the number of services in the corpus. $T_R = R_{in} \cup R_{out}$ is a set of I/O concepts of the request, and $T_s = \{t_1, t_2, ..., t_n\}: t_1, t_2, ..., t_n \in S_{des}$ is a set of terms of the web service description that are semantically related to concepts of the T_R request. Also, w_{T_R} and w_{T_S} are the term weights of the wanted keywords T_R and T_s respectively.

To analyze the semantic compatibility between the wanted keywords T_R and the concepts of T_s, the semantic matchmaking method is used, which is in charge to assess the degree of compatibility between the concepts included in the keywords with the concepts of the S_{des}. This method uses semantic reasoning (subsumption reasoning) to analyze the relationship between concepts. These ontology relationships allow extracting the concepts compatible to the T_s concepts, and this allows improving the performance of the LSA method semantically. So, to retrieve the web service which is related to the wanted keywords T_R, we determine the subsumption relationship as follows.

$$T(T_R, T_S) = \begin{cases} 1 & if\ T_R = T_s \\ \frac{1}{2} \times \left(\frac{\sum_{c \in T_R} max_{t \in T_S}\{Match(t,c)\}}{|q_{T_R}|} + \frac{\sum_{t \in T_S} max_{c \in T_R}\{Match(c,t)\}}{|q_{T_s}|}\right) & otherwise \end{cases} \in [0,1] \quad (8)$$

where c and t are any concepts of wanted keywords T_R and the service description T_s respectively. Moreover, $|q_{T_R}|$ and $|q_{T_S}|$ are the total number of the ontology relationships that have the maximum value and greater than zero.

As illustrated in the formula (8), if the all concepts of T_R appear in T_s, at this point the value of $T(T_R, T_S)$ is 1. In otherwise, the concepts of T_R are adjusted by other related

concepts using the ontology-agent, these concepts are different than the initial concepts T_R, in order to identify of the different keywords that are closer to the desired service description. It leads to avoid the ambiguity of the terms frequency and to enrich the terms which have a purely semantic relatedness with the service description. Thus, more the value of $T(T_R,T_S)$ is higher, more the concepts of keywords T_R are suitable to the concepts of T_s, that's means the value of semantic relatedness will be maximized.

Definition 5 (Interface Similarity): Interface similarity (IS) is determined by the semantic compatibility between S_{in} and S_{out}. This compatibility is evaluated by the degree of semantic matching between concepts, which is called IO-MATCHING. The relations of these concepts are deduced by the "ontology-agent" analyzer. This degree of semantic matching uses 4 types of matching score: Exact, Plugin, Subsume, and Fail to measure the matching between two S_{in} / S_{out} concepts as follows:

$$Match(C_i,C_j)=\begin{cases}\text{EXACT } if\ C_i \equiv C_j \\ PLUGIN\ if\ C_i \sqsubseteq C_j \\ SUBSUME\ if\ C_i \sqsupseteq C_j \\ FAIL\ otherwise\end{cases} \in [0,1] \quad (9)$$

where C_i and C_j are the request and service concepts respectively. The interface similarity between the request *R* and the service *S* is calculated as shown in the equation below.

$$sim_{IS}(R,S)=\frac{\sum Match(C_i,C_j)}{max\{Card(S_{in}\cup S_{out}),Card(R_{in}\cup R_{out})\}} \in [0,1] \quad (10)$$

In the literature [20], the different degrees of matching that are often considered are as follows:

- EXACT ($C_i \equiv C_j$): if the concepts C_i and C_j belong to the same ontology class.
- PLUGIN ($C_i \sqsubseteq C_j$) : where the C_j concept in the ontology is a sub-class of C_i, the concept C_i is more specific than the desired concept C_j
- SUBSUME ($C_i \sqsupseteq C_j$) : if the class of C_i is more general than the class of C_j, it indicates that the class of C_j is a sub-class of C_i.
- FAIL ($C_i \perp C_j$): when there is no subsumption relationship in ontology between C_i and C_j.

Definition 6 (Functionality semantic similarity): The functionality semantic similarity measure (FSM) includes two main components: description similarity and interface similarity. Functionality semantic similarity is defined as follows.

$$sim_{FS}(R,S)=\alpha\times sim_{IS}(R,S)+\beta\times sim_{DS}(R,S) \in [0,1] \quad (11)$$

where α and β are the interface similarity weight and the description similarity weight, respectively.

Table I represents the best-desired services to fulfill the request. As the desired service which should return a book price. It demonstrated the semantic relatedness/ similarity performance, which reinforces the LSA and IO-MATCHING similarity method to identify the hidden relationships in the service description rather than to focus the semantic analysis of the I/O. Although the similarity at the input/output level is similar, it provides different functionalities than expected. For example, the similarity at the interface level in the service "Cheapest Book Service" provides different request requirements. On the contrary, "BookPrice" and "BookPriceService" services respond to the users' same needs. As a result, it is crucial to measure similarity at the level of service description to extract hidden semantic relations and increase accuracy. This experiment is done by the weight of the interface similarity $\alpha=0.5$ and the description similarity $\beta=0.5$.

TABLE I. AN ILLUSTRATION OF THE RESULTS OBTAINED FROM THE SIMILARITIES BETWEEN THE SERVICES AND THE REQUEST

Service name	Inputs	Outputs	Text description	FSM
Cheapest Book Service	#_BOOK	#_PRICE	A Service that searchest the cheapest Price for a book	0.94
BookPriceService	#_BOOK	#_PRICE	Return price of a book	0.98
Bamzon RecommendedPriceService	#_BOOK	#_RECOMMENDERPRICEINDOLLAR	Bamzon is a popular service to return recommended price of a book	0.87
BookPrice	#_BOOK	#_PRICE	Uses the ISBN to return price of a book	0.96
BDe RecommendedPriceService	#_MONOGRAPH	#_RECOMMENDERPRICEINEURO	BDe is a competitor web service to return recommended price of a monograph in Euro	0.70
BookPriceTaxedPriceService	#_BOOK	#_TAXEDPRICE,#_PRICE	This service informs the taxed price of a book	0.88

IV. Experiment Results

In this section, we present our analysis that includes two main parts: In the first part, an overview of the experimental settings. The second part discusses the experimental results obtained by comparing the performance provided to another work [27].

A. Experimental Setup

To improve the performance mentioned in the last Section, we have been implemented our proposed approach in JADE Platform and OpenNLP Framework [30], which are based on the java language using an Intel® Core (TM) i7-4770 processor with 8 GB of main memory running Windows 10. Our experimental data is from the OWLS-TC version 3.0 dataset, which contains 1007 indexed OWL-S services, most of which were collected from public IBM UDDI registries semi-automatically transformed from WSDL to OWL-S. Table II below summarizes the features of the experimental environment.

To analyze the correctness and performance as discussed in our contribution, we carried out two experiments in different weights to prioritize each aspect of similarity (interface similarity and description similarity) as shown in the Table III. These parameter weights are scaled according to the importance of the similarity parameter in two different scenarios, these two scenarios will be experimented in order to understand the value added in our solution will be illuminated in the next Sub-section. Furthermore, the value of the weight factor δ=0.3 , which indicates a high importance of relatedness semantic than the LSA similarity, to reinforce the relatedness to cover the limitations of the LSA similarity.

TABLE II. The Experimental Environment

Environment	Description
Operating System	Windows 10
CPU	Core (TM) i7-4770
RAM	8 GB
Software Framework	JADE
NLP Toolkit	OpenNLP
Programming Language	JAVA
Dataset	OWL-S TC Version 3.0

TABLE III. The Weights of different Methods (Hybrid and IO-MATCHING Method)

Experiments	Method	Weight name	Parameter	Value
1	IO-MATCHING	Interface Similarity	α	0.50
	Hybrid Method LSA-IO	Description Similarity	β	0.50
2	IO-MATCHING	Interface Similarity	α	1
	Hybrid Method LSA-IO	Description Similarity	β	0

B. Results and Discussion

In order to carry out a standard and comparable analysis, we selected a set of 29 test queries (OWLS-TC3) associated with pertinence sets to lead performance evaluation experiments. These experiments are analyzed in more detail by comparing the precision, recall, and F-measure of the services retrieved by the two experiments, as illustrated in Table III. Then, the processing time is evaluated in function of the rising number of services, which are varied in each test (from 50 to 1007 services). It allows us to measure scalability according to the average speed to fulfill the query's requirements. For more analysis, we compared our solution with another method [27] to evaluate the system's performance in terms of scalability to clarify our system's success in dealing with all these constraints, as mentioned previously.

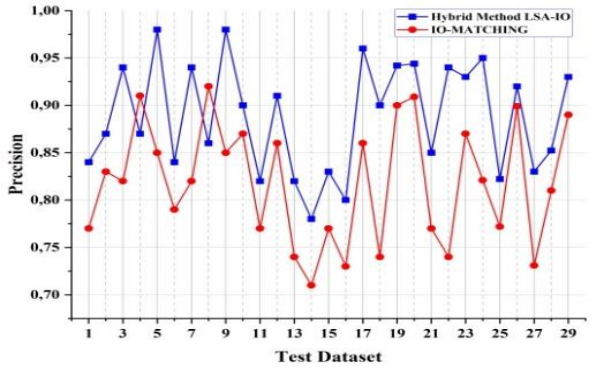
1) Evaluation metrics: As mentioned earlier, the experimental results should be analyzed in advance regarding the precision, recall and f-measure of the services retrieved by the hybrid and IO-MATCHING method. Precision is the ability to retrieve the most precise services. Higher precision means better relevance and more precise results but may imply fewer results returned. Recall means the ability to retrieve as many services as possible that match or are related to a query. F-Measure evaluates a weighted harmonic mean of precision and recall. As we used it for the evaluation process, it is then defined as follows.

$$precision = \frac{A_{Relevant} \cap B_{Retrieved}}{B_{Retrieved}} \quad (12)$$

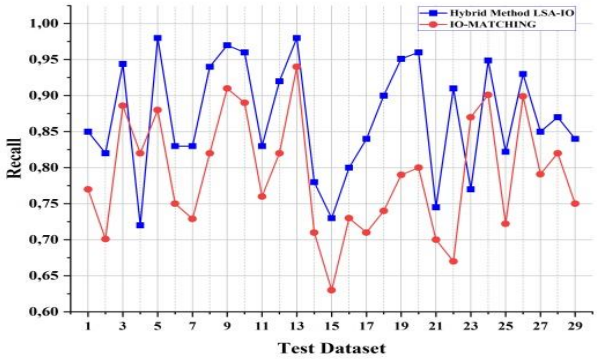
$$recall = \frac{A_{Relevant} \cap B_{Retrieved}}{A_{Relevant}} \quad (13)$$

$$F - measure = 2 \times \frac{precision \times recall}{precision + recall} \quad (14)$$

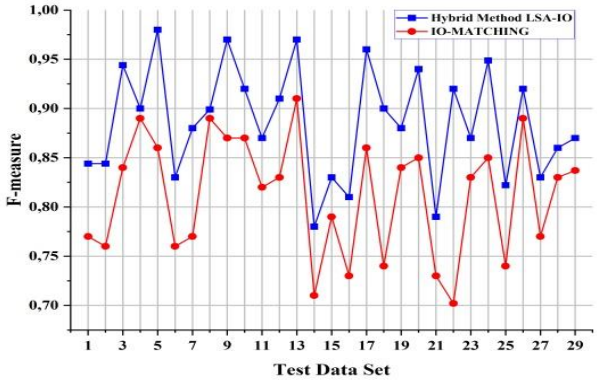
Where $A_{Relevant}$ is the set of relevant services, and $B_{Retrieved}$ is the number of relevant services retrieved. As indicated in the experimental results below, we run 29 test queries (OWLS TC-3) simultaneously to measure precision, recall, and F-measure in each experiment in Table III. The Fig. 5 demonstrates the efficiency of the interface similarity over description similarity as well as the performance provided by just the similarity measure at the I/O interface level. This proves that the hybrid method has a high value of precision, recall, and F-measure in all the query tests as compared to the traditional method (IO-MATCHING), which is purely based on the ontological relationships at the I/O level. With the exception of the query test 4 and 8, which record higher precision rate in concerning the IO MATCHING method. Relative to the query test 4 and 23, it also shows higher value of recall than the hybrid method.



(a) Precision.



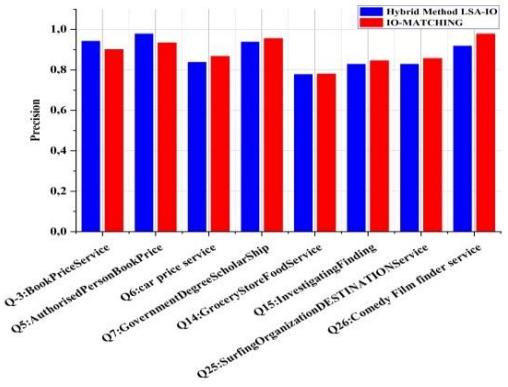
(b) Recall.



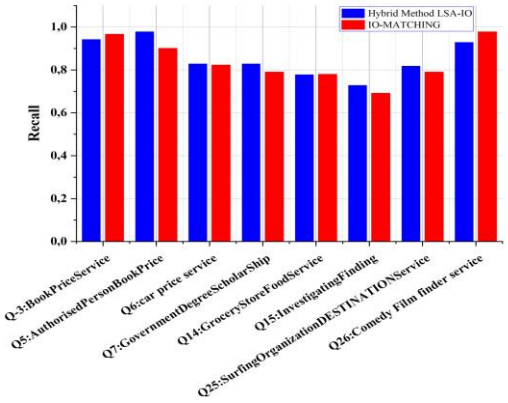
(c) F-measure.

Fig. 5. The Performance Measures for each Query in the Testing Dataset OWLS TC-3 for Both IO-MATCHING and Hybrid Method.

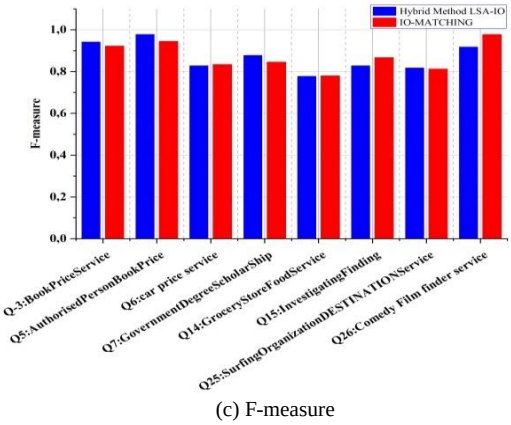
For more explicit the results obtained above, the Fig. 6 represents more in details by the selected test queries (OWLS TC-3), to clarify well the powerful hybrid method proposed in terms of the different evaluation metrics. It is the same criteria that we mentioned previously concerning the query tests, which are the identical to those shown in the Axis-x Fig. 6. These selected query tests indicate the challenges chosen to understand our hybrid method dominated by the IO-MATCHING method regarding precision, recall and F-measure. It was analyzed in Table III, the " Q3:BookPriceService" query can retrieve services similar but not as meaningful to the query in the case of experience 2. We deduced that it is not sufficient to just rely on the I/O interface instead of relating the concepts contained in the inputs/outputs with the description. But with Experimental 1, the hybrid method leads to relevant services to the query, due to the reinforcement provided to the LSA method by the semantic relatedness, which allows to improving the correlation ratio between the terms and the desired service instead of related to the frequency of terms. For example, the query "Q7:Government Degree ScholarShip " which requires the information on scholarships offered by a given government according to degree and government. Thus, the concept "#degree" can be related semantically to "#Academic-Degree" or "#Award", while the concept "#Government" can be related to the concept "#GovernmentOrganization". This is why the hybrid method is robust in terms of correlating these concepts with the service requested in the context.



(a) Precision



(b)Recall



(c) F-measure

Fig. 6. Evaluation of different Queries of OWL-S Dataset in Terms of Precision, Recall and F-measure.

Fig. 7 indicates the number of services retrieved via the hybrid or IO-MATCHING method for each query test (OWL-TC 3). These query tests vary in the number of inputs/outputs and related functionality desired to be achieved, in order to make a real challenge to the discovery systems concerned. As illustrated below, the significant number of services retrieved to satisfy the request set by our proposed system. Moreover, due to the performance provided by cooperative agents such as mobile agent and ontology-agent, that is proved in terms of different evaluation measures, to find a relevant service.

In addition, the comparison illustrated in Fig. 8 indicates the average precision-recall curve between the hybrid method and the IO-MATCHING method in order to expand the evaluation measures of the performance system proposed in retrieving a relevant service. In this experiment, we run 29 test queries in order to compute the average precision and recall. This demonstrates in the average precision-recall curve that the hybrid method excels considerably in the retrieval accuracy for relevant services based on the semantic correlations between the query concepts and the desired service. Consequently, the hybrid method has a higher precision value in service retrieval than the IO-MATCHING method.

2) Runtime performance comparison: To validate and evaluate the speed up performance for the proposed architecture, we compared our system with other works in the same scope [27] . To perform this comparison, we computed the average runtime according to each service set for different test queries. These sets vary in number of services to provide the scalability with a growing set of services (from 50 to 1007 services) and the response of the system in real time. This is demonstrated in Fig. 9, the results obtained in comparison with another work. The work [27] suggested to be compared, based on Ontology filtering and parameter matching relied on the discovery of function-oriented Web services to reduce the space of matching preprocessor and filter. While, we focused on the semantic analysis and the inter-relatedness between the concepts and the desired service, combined with the privileges provided by the agent ontology, which allows to exploit the pre-existing solutions in the second database, as discussed in Section 3, that facilitate the discovery of semantic relations and to reduce the consumption of reproducing the analysis of the operations required. This makes the discovery more flexible and rapid.

According to the results shown in Fig. 9, both approaches do not spend more time to the first sets of services (between 50 ms and 110 ms), while the OFPM method spends more run-time than the hybrid method when the number of services is scaled up. This is due to the cooperation of agents in the environment in order to make the discovery task and to respond in a real-time. This provides for a more flexible process of composition to accomplish its tasks.

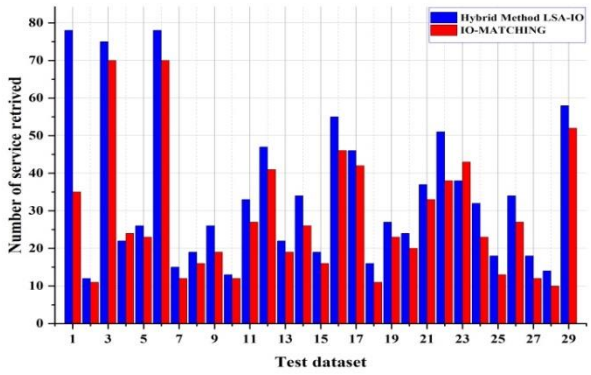


Fig. 7. The Number of Services Retrieved by the Hybrid Similarity Method and IO-MATCHING.

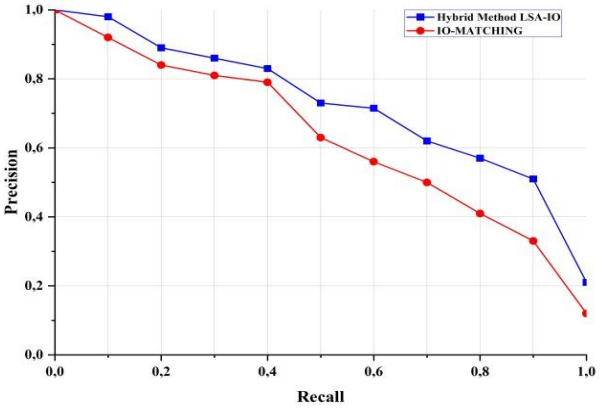


Fig. 8. Average 11-Points Precision-Recall Curve Across 29 Test Queries for the Hybrid Method and IO-MATCHING.

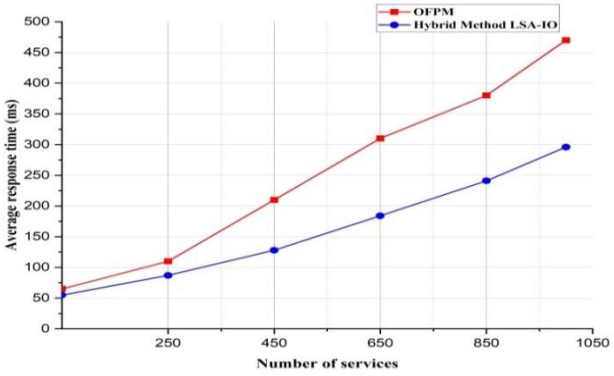


Fig. 9. Runtime Performance Comparison for Both Hybrid Method and OFPM [27].

V. Conclusion

This paper demonstrates the performance dynamicity provided to the architecture proposed. This allows handling the service composition more flexibly and quickly, with autonomous to find services accurately. Thus, it is proved in terms of the scalability and flexibility to respond in a real-time. This is due to the integration of the proposed hybrid method and the ontology analysis agent, which makes the architecture to be more dynamic in terms of autonomy, reliability and robustness. In addition, the proposed hybrid method makes the

system to meet the requirements of the query in the context of the semantic relatedness between the requested concepts and services. This leads to the high retrieval precision, recall and F-measure discovery process.

As future work, we will focus on integrating Micro-services and multi-agent systems (MAS) to reduce the time and complexity of composite semantic web services. Furthermore, we intend to enrich our system with the semantic descriptions of other functional aspects such as pre-conditions/post-conditions.

REFERENCES

[1] N. Niknejad, W. Ismail, I. Ghani, B. Nazari, M. Bahari, and A. R. B. C. Hussin, "Understanding Service-Oriented Architecture (SOA): A systematic literature review and directions for further investigation," Inf. Syst., vol. 91, p. 101491, Jul. 2020, doi: 10.1016/j.is.2020.101491.

[2] T. Aditya Sai Srinivas, S. Ramasubbareddy, and K. Govinda, "Discovery of Web Services Using Mobile Agents in Cloud Environment," in Innovations in Computer Science and Engineering, Springer, 2019, pp. 465–471. doi: 10.1007/978-981-13-7082-3_53.

[3] I. Ghani, W. M.N., and A. Mustafa, "Web Service Testing Techniques: A Systematic Literature Review," Int. J. Adv. Comput. Sci. Appl., vol. 10, no. 8, 2019, doi: 10.14569/IJACSA.2019.0100858.

[4] D. Booth and C. K. Liu, "Web services description language (WSDL) version 2.0 part 0: Primer," W3C Recomm., vol. 26, pp. 39–41, 2007.

[5] D. Martin et al., "OWL-S: Semantic markup for web services," W3C Memb. Submiss., vol. 22, no. 4, 2004.

[6] J. De Bruijn et al., "The Web Service Modeling Ontology," in Modeling Semantic Web Services, vol. 5, no. 1, Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 23–28. doi: 10.1007/978-3-540-68172-4_3.

[7] J. Kopecký, T. Vitvar, C. Bournez, and J. Farrell, "Sawsdl: Semantic annotations for wsdl and xml schema," IEEE Internet Comput., vol. 11, no. 6, pp. 60–67, 2007.

[8] R. Studer, S. Grimm, and A. Abecker, Semantic web services. Springer, 2007.

[9] M. Klusch, P. Kapahnke, S. Schulte, F. Lecue, and A. Bernstein, "Semantic Web Service Search: A Brief Survey," KI - Künstliche Intelligenz, vol. 30, no. 2, pp. 139–147, Jun. 2016, doi: 10.1007/s13218-015-0415-7.

[10] F. Fakhar, "Semantic Constraints Satisfaction Based Improved Quality of Ontology Alignment," Bull. Electr. Eng. Informatics, vol. 2, no. 3, pp. 182–189, 2013, doi: https://doi.org/10.11591/eei.v2i3.202.

[11] M. Fariss, N. El Allali, H. Asaidi, and M. Bellouki, "Review of Ontology Based Approaches for Web Service Discovery," vol. 66, F. Khoukhi, M. Bahaj, and M. Ezziyyani, Eds. Cham: Springer International Publishing, 2019, pp. 78–87. doi: 10.1007/978-3-030-11914-0_8.

[12] R. Hammami, H. Bellaaj, and A. H. Kacem, "Semantic Web Services Discovery: A Survey and Research Challenges," Int. J. Semant. Web Inf. Syst., vol. 14, no. 4, pp. 57–72, Oct. 2018, doi: 10.4018/IJSWIS.2018100103.

[13] Z. Cong, A. Fernandez, H. Billhardt, and M. Lujak, "Service discovery acceleration with hierarchical clustering," Inf. Syst. Front., vol. 17, no. 4, pp. 799–808, Aug. 2015, doi: 10.1007/s10796-014-9525-2.

[14] A. Bukhari and X. Liu, "A Web service search engine for large-scale Web service discovery based on the probabilistic topic modeling and clustering," Serv. Oriented Comput. Appl., vol. 12, no. 2, pp. 169–182, Jun. 2018, doi: 10.1007/s11761-018-0232-6.

[15] K. Belmabrouk, F. Bendella, and M. Bouzid, "Multi-Agent Based Model for Web Service Composition," Int. J. Adv. Comput. Sci. Appl., vol. 7, no. 3, 2016, doi: 10.14569/IJACSA.2016.070320.

[16] A. Gupta, A. Kumar, and J. Gautam, "A survey on semantic similarity measures," Int. J. Innov. Res. Sci. Technol., vol. 3, no. 12, pp. 243–247, 2017, [Online]. Available: http://www.ijirst.org/articles/IJIRSTV3I12083.pdf

[17] Xiaojie Zheng, Jie Hu, and Lin Zhang, "Measuring semantic relatedness based on ontology," in International Conference on Automatic Control and Artificial Intelligence (ACAI 2012), 2012, pp. 1335–1338. doi: 10.1049/cp.2012.1226.

[18] Latrache, "A MOBILE AGENT BASED APPROACH FOR AUTOMATING 'DISCOVER-COMPOSE' PROCESS OF SEMANTIC WEB SERVICES," J. Comput. Sci., vol. 10, no. 9, pp. 1628–1641, Sep. 2014, doi: 10.3844/jcssp.2014.1628.1641.

[19] S. T. Dumais, "Latent semantic analysis," Annu. Rev. Inf. Sci. Technol., vol. 38, no. 1, pp. 188–230, 2004.

[20] M. Paolucci, T. Kawamura, T. R. Payne, and K. Sycara, "Semantic Matching of Web Services Capabilities," in International Semantic Web Conference, Springer, 2002, pp. 333–347. doi: 10.1007/3-540-48005-6_26.

[21] G. Vadivelou, E. Ilavarasan, and S. Prasanna, "Algorithm for web service composition using multi-agents," Int. J. Comput. Appl., vol. 13, no. 8, 2011.

[22] F. Chen, M. Li, H. Wu, and L. Xie, "Web service discovery among large service pools utilising semantic similarity and clustering," Enterp. Inf. Syst., vol. 11, no. 3, pp. 452–469, Mar. 2017, doi: 10.1080/17517575.2015.1081987.

[23] D. Lin, "An information-theoretic definition of similarity.," in Icml, 1998, vol. 98, no. 1998, pp. 296–304.

[24] S. M. Mohammed, B. J. Dorr, G. Hirst, and P. D. Turney, "Measuring degrees of semantic opposition," Tech. Rep., 2011, doi: http://dx.doi.org/10.4224/19040608.

[25] F. Chen, C. Lu, H. Wu, and M. Li, "A semantic similarity measure integrating multiple conceptual relationships for web service discovery," Expert Syst. Appl., vol. 67, pp. 19–31, Jan. 2017, doi: 10.1016/j.eswa.2016.09.028.

[26] R. Benaboud, R. Maamri, and Z. Sahnoun, "Agents and OWL-S Based Semantic Web Service Discovery With User Preference Support," Int. J. Web Semant. Technol., vol. 4, no. 2, pp. 57–75, Apr. 2013, doi: 10.5121/ijwest.2013.4206.

[27] M. Fang, D. Wang, Z. Mi, and M. S. Obaidat, "Web service discovery utilizing logical reasoning and semantic similarity," Int. J. Commun. Syst., vol. 31, no. 10, p. e3561, Jul. 2018, doi: 10.1002/dac.3561.

[28] S. A. Khanam and H. Y. Youn, "A Web Service Discovery Scheme Based on Structural and Semantic Similarity.," J. Inf. Sci. Eng., vol. 32, no. 1, pp. 153–176, 2016.

[29] R. Jonker and T. Volgenant, "Improving the Hungarian assignment algorithm," Oper. Res. Lett., vol. 5, no. 4, pp. 171–175, Oct. 1986, doi: 10.1016/0167-6377(86)90073-8.

[30] A. OpenNLP, "a Machine Learning Based Toolkit for the Processing of Natural Language Text," URL http//opennlp. apache. org (Last accessed 2013-06-18).