Probabilistic Approach for Diversifying Web Services Discovery and Composition

Hafida Naim, Mustapha Aznag, Mohamed Quafafou, Nicolas Durand

Probabilistic Approach for Diversifying Web Services Discovery and Composition

Hafida NAIM, Mustapha AZNAG, Mohamed QUAFAFOU and Nicolas DURAND
Aix-Marseille University, LSIS CNRS UMR 7296, 13397 Marseille, France.
hafida.naim@etu.univ-amu.fr, {mustapha.aznag,mohamed.quafafou,nicolas.durand}@univ-amu.fr

***Abstract*—Due to the increasing number of available web services, discovering the best service that matches a user requirement is still a challenge. In most cases the discovery system returns a set of very similar services and sometimes it is unable to find results for some complex queries. Therefore, integrating web service discovery and composition, taking into account the diversity of discovered results, in a unified way is still a big issue for web services. In this paper, we propose a novel service ranking algorithm for diversifying web services discovery results in order to minimize the redundancy in the search results. This algorithm chooses a set of selected web services based on relevancy, service diversity and service density. We also propose a new method to generate service dependency network using the Formal Concept Analysis (FCA) framework. The generated graph is used to select the composition of discovered web services set. Experimental results show that our method performs better than others baseline approaches.**

***Index Terms*—Web services, Discovery and ranking, Composition, Service Dependency Network, Diversity, Topic Models, Formal Concept Analysis.**

I. Introduction

Recently, many activities in web services have drawn a great deal of attention. As one key motive for the increasingly rapid evolution of web service is the ability to help users to discover the best services that match their requirements. In this regard, service discovery and composition have been intensively studied [1], [7]. Web Service Discovery (WSD) is the mechanism of finding the best web services that match service users' needs. Several approches have been developed to improve the general web services management process. Some of them have been developed to search for both semantic and non-semantic web services discovery [2]. There exist logic-based approaches, non-logic based approaches, and hybrid approaches. The key problem of the logic-based and hybrid approaches is the complexity of the discovery process which makes it an intractable process over large service datasets [5]. The non-logic-based semantic approaches [6] aim to reduce the complexity of the discovery process by analysing the frequency of occurrence of some concepts and determine semantics which are implicit in service descriptions. These approaches generally use techniques from different research fields such as information retrieval (IR), data mining and linguistic analysis [5].

Recently, some web service portals and search engines as Biocatalogue[1] and Seekda![2], WS-Portal[3] [8] offer different features to facilitate the WSD task. In general, the search engines discover several very similar services and sometimes fail to answer some complex queries. To deal with this problem, two important issues should be considered: (1) how can one select matching services that satisfy a given request q when the returned web services are very similar, and (2) how to compose multiple services to satisfy q when selected services don't fulfill this query.

Since the discovered web service set may contain redundant and useless services, thereby decreasing the degree of the user's satisfaction. To avoid redundancy and at the same time maintain the quality of the discovered services, the diversity should be taken into account in discovery system. In this paper, we present a novel service ranking method for diversifying web services discovery results. Diversifying search results has already been investigated in the context of document search on the Web [10]. Our system selects dissimilar web services from a number of top-K ranked web services considering the relevance, diversity and density measures.

In case of complex request, there is sometimes no single web service that can fulfill this request. So, we need to combine multiple web services to build a new service that satisfies the individual users' needs. In this paper, we adress the problem of web services composition (WSC) as a service network search problem. In this regard, many techniques have been proposed taking advantage of graph structures [12], [11]. To begin with, the work presented in [12] introduces an approach that uses a particular formalism called interface automata. The authors store the repository of web services as a graph that contains services information from a particular point of view. In [11], Oh et al. proposed a novel web service benchmark tool to adress the composition problems. First, they formulated the WSC problem in terms of AI planning and network optimization problems to investigate its complexity. Then, they analyzed publicly available Web service sets using complex network analysis techniques. In our work, we model the dependencies between a set of web services as a network where nodes representing services and edges representing invocations in-between.

The main contributions of this paper are listed below:

[1]https://www.biocatalogue.org/
[2]http://webservices.seekda.com/ (the portal is no longer available.)
[3]http://wsportal.aznag.net/

- A method for re-ranking discovered web service search results using diversification measures to minimize the redundancy.
- A new algorithm to construct the dependency web service network based on the Formal Concept Analysis (FCA) formalism.
- A method to find and select the optimal composition of discovered web services set.

The rest of the paper is organized as follows: Section II presents the overview of the proposed system. Section III introduces our probabilistic service discovery and ranking. Section IV discusses how we build the dependency web service network and how we generate the compositions of the discovered web service set. Section V is devoted to the experimental evaluation. Finally, the conclusion and future work can be found in Section VI.

II. Overview of the Proposed System

In our approach, we assume all service descriptions were written in the WSDL and/or SAWSDL. Generally, every web service has a WSDL (Web Service Description Language) document that contains the description of the service. The WSDL document is an XML-based language provides the specifications necessary to use the web service by describing the communication protocol, the message format required to communicate with the service, the operations that the client can invoke and the service location. To manage efficiently web service descriptions, we extract all features that describe a web service from the WSDL document. We also extract from WSDL documents all operations and their input/output parameters. These parameters are used to generate the web services network that describe the relationships between web services. The generated network is used to identify the composition of selected services (see Section IV for more details).

In [9] we investigated the use of probabilistic topic models to extract topics from service descriptions. Topics (or latent factors) are a concept introduced by Probabilistic Topic Models [15]. These are a family of generative probabilistic models based on the assumption that documents are generated by a mixture of topics where topics are probability distributions on words. In our work, a topic t is associated with a group of textual concepts (i.e. words) and/or semantic concepts that can appear in service descriptions and can be expressed as a probability distribution over words. Topic models are used, in our context, as efficient dimension reduction techniques, which are able to capture semantic relationships between word-topic and topic-service interpreted in terms of probability distributions. Based on the extracted topics, a set of matched services can be returned by comparing the similarity between the query and the services related topic.

Figure 1 shows the overview of our approach with the different steps involved. The service discovery phase is responsible for retrieving the Top-R ranked web service set. First, we represent the query in topic space to discover the set of top-K implicit topics that are semantically associated to the query. Based on the selected topics, a set of web services can be returned by comparing the similarity between the query and the related web services to these topics. Then, we use the similarity scores to rank and select the top web services. In order to minimize the redundancy in the search results, we re-rank the top-ranked web service taking into account relevance, diversity and density (see section III). Next, we perform a search over the service dependency network to find the optimal composition. Before this phase, the operation dependency network is constructed using the Formal Concept Analysis (FCA) formalism. Next, we deduct the service dependency network (see section IV-A). This step is done one times. The service dependency network takes as input the Top-R ranked web services set and returns a sub-graph containing all possible service compositions. Then, a selection process is performed over the graph to find the optimal composition for the results of the query (see Section IV-B). Finally, the Top-R ranked web service and their optimised composition are returned.

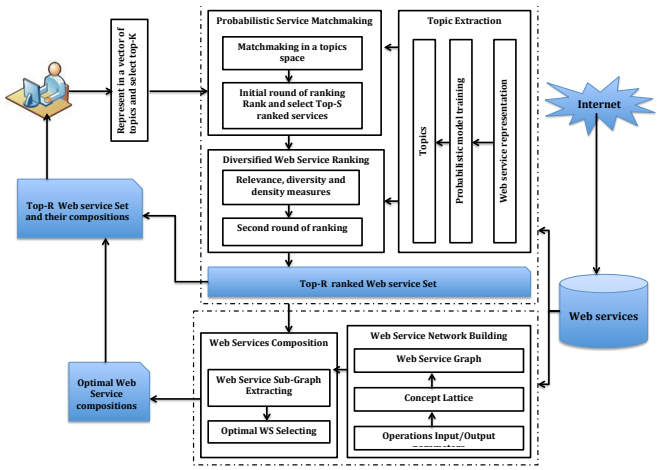

Fig. 1. An overview of our proposed system.

III. Probabilistic Web Service Matchmaking and Ranking

Service discovery and selection process aim to find web services with user required functionalities. Our service discovery mechanism works by computing the degree of similarity between a query and a service descriptions in the topic space.

The work described in this section extends our previous work on probabilistic web services discovery and ranking based on probabilistic topic models [13], [14]. The main idea underlying the work proposed in this paper is to increase the diversity of all discovered web services in order to minimize the redundancy in the search results. Indeed, several existing approaches do not take into account the redundancy which results from very similar and near duplicated web services. Diversifying search results has already been investigated in the context of document search on the Web [10].

In this paper, we propose a novel method for re-ranking web services taking into account relevance, diversity and density of selected web services obtained from the matchmaking and selection process. The main steps of our probabilistic

service matchmaking and ranking are described as follow: (1) discovering implicit topics related to the user query, (2) discovering web services related to the selected topics, (3) ranking discovered web services set, and (4) re-ranking the top-ranked web services.

After training our probabilistic model, the distribution of each topic is known and all the services in the registry can be described as a distribution of topics. Let θ_s refers to the multinomial distribution over topics in the service description s and ϕ_t refers to the multinomial distribution over words for the topic t [9].

Let $q = \{w_1, w_2, \ldots, w_n\}$ be a user query that contains a set of words w_i. First, we compute the query-likelihood model for q; denoted θ_q; by representing it as distribution over topics. Thus, for each topic t we compute the semantic similarity between query q and topic t based on the probability distribution ϕ_t using Equation 1. The purpose of this step is to select the set of top-K implicit topics that are semantically associated to q. The most related topics are the ones that maximize the θ_q^t.

$$\theta_q^t = P(q|t) = \prod_{w_i \in q} P(w_i|t) \tag{1}$$

where $P(w_i|t)$ is the probability of having word w_i given the topic t.

Based on the related topics retrieved for q, we can discover easily the set of web services associated to these topics. After discovering the web services related to user query q, we can rank them by determining their relevances to query q. We use the generated probabilities θ and ϕ as the base criteria for computing the similarity between retrieved services and a user's query q. To this end, we model information retrieval as a probabilistic query to the topic model. We note this as $P(q|s_i)$ where q is the set of words contained in the query. Thus, using the assumptions of the topic model, $P(q|s_i)$ can be calculated by equation 2.

$$P(q|s_i) = \prod_{w_k \in q} P(w_k|s_i) = \prod_{w_k \in q} \sum_{j=1}^{T} P(w_k|t_j) P(t_j|s_i) \tag{2}$$

Where $P(w_k|t_j)$ is the probability of having word w_k given the topic t_j and $P(t_j|s_i)$ is the probability that the topic t_j describes the service s_i.

The most relevant services are the ones that maximize the conditional probability of the query $P(q|s_i)$. Consequently, relevant services are ranked according to their similarity score to the query. The set of relevant services returned in this step, essentially contains very similar services that match with the user query. However, the ranking criteria applied in this process considers only the relevance information and often suffers from returning too many redundant web services. As we have already mentioned, we need to capture the diversity of selected web services to minimize the redundancy in the retrieved set. Therefore we seek to increase the diversity of the returned services that match with the user query by explicitly maximizing the distance between new services and already selected services. Selecting relevant web services with high density will retrieve more relevant services in the subsequent round, which leads to a better discovery performance.

In our approach, the service ranking procedure first takes a number of top-K ranked web services $S_q = \{s_1, \ldots, s_K\}$ retrieved in the previous step. Then the system iteratively selects web services from S_q to form a representative set of selected services S_q^* by simultaneously considering the relevance, diversity and density measures. More specifically, each candidate service s_i is associated with a ranking score RS that is a linear combination of these measures, as expressed in equation 3. Thus, we obtain automatically an efficient ranking of the retrieved services.

$$RS = \underset{s_i \in S_q}{\operatorname{argmax}} [\alpha . M_{rel}(q|s_i) + (1-\alpha)(M_{den}(s_i) + M_{div}(s_i, S_q^*))] \tag{3}$$

Where $M_{rel}(q|s_i) = P(q|s_i)$, $M_{den}(s_i) = Density(s_i)$ and $M_{div}(s_i, S_q^*) = Diversity(s_i, S_q^*)$ are measures of relevance (i.e. semantic similarity between the user's query q and a selected web service s_i), density and diversity of each web service in initial top ranked set S_q where $\alpha \in [0, 1]$ a weighting parameter. More specifically, the selection process described in Equation 3 will be executed iteratively until S_q^* contains a predefined K^* web services. It is worth mentioning that our method bears a close resemblance in spirit to the Maximal Marginal Relevance (MMR) ranking algorithm [16] which was originally proposed for extractive document summarization. The MMR ranking algorithm [16] is a greedy algorithm for ranking documents based on relevance ranking score and at the same time avoiding redundancy. Our algorithm extends the MMR algorithm by adding an extra term, which reflects the web service density and diversity.

In the following, we will explain how we calculate the density and diversity measures in our context. Density and diversity are one of the important factors in the defined selection process. There are several metrics to compute the distance between the semantic of two distributions θ_x and θ_y (i.e. in our case; θ_x and θ_y are the probability distributions over topics). In this paper, we consider the *Kullback Leibler (KL)* Divergence [15] defined as follows:

$$KL(\theta_x, \theta_y) = \sum_{i=1}^{T} \theta_x^i \log_2 \frac{\theta_x^i}{\theta_y^i} \tag{4}$$

Where $\theta_x^i = P(i \mid x)$ represents the proprability of the topic i given the service x. KL is not symetric and KL = 0 when $\theta_x^i = \theta_y^i$ for all i.

Distance between individual web services is measured by symmetric-KL divergence. Symmetric-KL divergence measures the degree of overlap between a given service and all other selected services.

We measure the density performance of the top K ranked web services in the initial retrieval results obtained in the matching/ranking process. In our work, we approximate the density in a region around a particular web service by measuring the average distance from that service to all the other services. Generally, large average Symmetric-KL divergence indicates that the web service is in a low service density region. Thus we use negative average Symmetric-KL divergence as described in equation 5 to approximate service density performance measure, which reflects the closeness of this service to the other services.

$$Density(s_i) = \frac{-1}{|S_q|} \sum_{s_j \in S_q} [KL(\theta_{s_i}, \theta_{s_j}) + KL(\theta_{s_j}, \theta_{s_i})] \tag{5}$$

where $|S_q|$ is the number of selected web services in the initial retrieval results. A web service having a higher value of $Density(s)$ is deemed to be closer to the others services in S_q and thus to be more representative and less likely to be an outlier.

In order to diversify the selected web services, we compute the diversity measure of a candidate service with respect to the set S_q^* of already selected services, which is expressed as follows (i.e. Equation 6).

$$Diversity(s_i, S_q^*) = \underset{s_j \in S_q^*}{\operatorname{argmax}} \frac{1}{2}[KL(\theta_{s_i}, \theta_{s_j}) + KL(\theta_{s_j}, \theta_{s_i})] \quad (6)$$

IV. Lattice based web services compostion

The web service composition is the fact of composing a set of available web services to fully satisfy a given request that cannot be satisfied by individual services. Formally, each web service ws can be defined as ws = (S_{in}, S_{out}) where $S_{in} = \{in_1, in_2, \ldots\}$ is its input parameters wheras $S_{out} = \{out_1, out_2, \ldots\}$ represents its set output parameters. We suppose that the whole set of input parameters S_{in} must be provided in order to invoke ws which in turn has to return the output parameters S_{out}. Given a request $q = \{q_{in}, q_{out}\}$ defined by its initial input parameters q_{in} and its desired output parameters q_{out}. When q can be satisfied just by composing a set of available web services as no individual web service can fulfill it. The Web Service Composition (WSC) problem is then defined as finding a set of web services $\{ws_1, ws_2, \ldots, ws_n\}$ and compose them in a sequential or parallel way, such that: (1) ws_i is invoked sequentially from 1 to n and (2) $q_{out} \subseteq (q_{in} \cup S^1_{out} \cup S^2_{out} \cup \ldots \cup S^n_{out})$.

On one hand, The real-world composition problems are far more complex and hard to solve as the growth of web services. On the other hand, a composite Web service has to collaborate and communicate in order to fulfill a given request. This coordination between services requires also taking into account their dependencies. Thus, the real challenge has become to make easy investigating the real-world Web services composition. Networks constitute a convenient way to represent a collection of WS, allowing visualizing, analyzing and taking advantage of the relationships of dependency or similarity observed between them. They can be considered as complex networks and some authors previously used this approach to model web service collections in other contexts [11]. The composition problem therefore can be considered as a search problem in Service Dependency Network.

To model the dependencies between a set of web services in the form of networks, different entities can be used as nodes and links. First, nodes can be defined from the three levels of granularity (parameters, operations and web services) [11]. Second, links can indicate the parameter matching and operation invocation.

Definition IV-.1: **Operation dependency network:** An operation dependency network consists of a set of nodes representing operations and directed edges representing invocations in-between. There are two modes of invocation to indicate the possible interaction between two operations. The invocations are respectively denoted by **full** and **partial-invocation**. An operation f_i partially invokes the operation f_j if there is at least one output parameter of f_i similar to an input parameter of f_j. An operation f_i fully invokes the operation f_j if and only if for each input parameter of f_j, there exists a similar output parameter of f_i.

Definition IV-.2: **Service Dependency Network:** A web services dependency network is a directed graph $G = (S, V)$ where: S is a set of services: $S = \{s_1, s_2, \ldots, s_n\}$ and $V = \{(s_i, s_j) \in S * S), \exists f_i \in s_i, \exists f_j \in s_j : f_i \rightarrow f_j\}$. $f_i \in s_i$ indicates that the service s_i offers the operation f_i. $f_i \rightarrow f_j$ indicates that the operation f_j depends on the operation f_i. A web service dependency network can be defined from operation dependency network. Service networks are also distinguished according to the mode of invocation (full or partial invocation). In **partial invocation**, a directed edge is created between two services s_1 and s_2 if there is at least an operation of s_1 that partially invokes an operation of s_2. In the case of **full invocation**, an arc (or directed edge) is created between two services s_1 and s_2 if there is at least an operation of s_1 which fully invokes an operation of s_2.

A. *Lattice based web service network building*

In order to construct the web service dependency network, we have to build the operation dependency network. Therefore, we use the *Formal Concept Analysis (FCA)* formalism [17]. Our goal is to organize the web services operations into concept lattices according to their input/output parameters. In our approach, FCA takes as input a set of web services operations and its input or output parameter represented as a formal context and produces the set of all the formal concepts which form a concept lattice. A formal context is denoted by $\mathbf{K} = (O, P, I)$ where O is a set of objects (i.e. operations), P is a set of attributes (i.e. I/O parameters), and I is a binary relation between O and P ($I \subseteq O \times P$). $(o, p) \in I$ denotes the fact that object $o \in O$ is in relation through I with attribute $p \in P$ (i.e. Also read as o has p). In our context, a concept lattice defines a hierarchical representation of objects and attributes, in which a certain concept inherits all the extents (objects) of its descendants and all the intents (attributes) of its ascendants.

In our work, we use CHARM-L [18] to build a concept lattice corresponding to web services and their topics. CHARM-L is an efficient algorithm for generating the concept lattice. The operation dependency network process consists of several steps:

- Construct binary contexts BC_{IN} and BC_{OUT} using dependencies between operations and their input/output parameters.
- Build Concept Lattices CL_{IN} and CL_{OUT} from BC_{IN} and BC_{OUT} respectively.
- Construct operation dependency network by performing the operation search in CL_{IN} and CL_{OUT} (i.e. Algorithm 1).

Algorithm 1 Operation dependency network building

Require:
- $O = \{o_1, \ldots, o_M\}$ Set of service operations and their I/O parameters.
- CL_{IN}: Concept Lattice operation/input parameters.
- CL_{OUT}: Concept Lattice operation/output parameters.
- SearchType: Exact or approximate search.

Ensure:
- A set of directed edges retrieved.

1: **for** each op $o_m \in O = \{o_1, \ldots, o_M\}$ **do**
2: $paramSet = GetInputParameters(op)$
3: **if** $paramSet \neq \emptyset$ **then**
4: Perform search on Concept Lattice CL_{OUT}
5: $opSet = LatticeSearch(SearchType, paramSet)$
6: Create a directed edge from each operation in $opSet$ to selected operation op
7: **else**
8: $paramSet = GetOutputParameters(op)$
9: Perform search on Concept Lattice CL_{IN}
10: $opSet = LatticeSearch(SearchType, paramSet)$
11: Create directed edge from selected operation op to each operation in $opSet$
12: **end if**
13: **end for**
14: **return** Set of directed edges constructed.

Based on the operation dependency network, we use the directed edge to build the web service dependency network as follow: For partial invocation case, a directed edge is created between two services s_1 and s_2 if there is at least an operation of s_1 that partially invokes an operation of s_2. Similarly, for full invocation case, a directed edge is created between two services s_1 and s_2 if there is at least an operation of s_1 that fully invokes an operation of s_2.

B. *Finding optimal k web service compositions*

The first step of the composition graph construction is the calculation of the relevant services. Based on the Top-K diversified ranked web services retrieved by our probabilistic service matchmaking and ranking (i.e. Section III), a set of discovered web services and their compositions can be returned by finding the service dependency sub-graph containing all possible service compositions. Once the sub-graph is retrieved, the next step is to reduce the graph size in order to improve the optimal composition. In [19], Mei et al. proposed a new diversified ranking measure (i.e. DivRank) on large graphs, which captures both relevance and diversity based on the MMR ranking algorithm [16]. Similary, we propose to extend DivRank algorithm by adding an extra term, which reflects the web services diversity and density as described in Section III.

We propose to use the expansion ratio to evaluate the diversity of Web services on service dependency sub-graph generated for the initial top-K ranked services. We consider the following definitions.

Definition IV-B.1: **Expanded Set:** Let S be the subset of nodes in the web service dependency network $G = (V, E)$, the expanded set of S is $N(S)$ such that $N(S) = S \bigcup v \in (V - S) \exists u \in S, (u, v) \in E$.

Definition IV-B.2: **Expanded Ratio:** The expansion of set S is defined as $|N(S)|$ where $N(S)$ is the Expanded Set and $|N(S)|$ is its cardinality of $N(S)$. The expansion ratio is defined as $\sigma = \frac{|N(S)|}{|V|}$.

We aim to find the L nodes (denoted by S) in a generated web service sub-graph that have the highest ranking scores RS (i.e. Equation 3) and the largest expansion ratio $\frac{|N(S)|}{|V|}$. Formally, our goal is to maximize the following comprehensive ranking measure:

$$H(S) = (1 - \lambda) \sum_{s \in S} RS(s) + \lambda \frac{|N(S)|}{|V|} \quad (7)$$

Where $RS(s)$ denotes the score of node s (i.e. Equation 3), and $\lambda \in [0, 1]$ is a tunable parameter that is used to tradeoff the score and the diversity in graph. The first term in Equation 7 is the sum of scores over the selected nodes, which reflects the relevance, diversity and density of selected services. The second term is the expansion ratio of the selected nodes. In general, larger expansion ratio indicates better diversity. Consequently, the Top-k diversified and optimal web services on a web service graph are the ones that maximize Equation 7.

V. Evaluation

A. *Web services corpus*

Our experiments are performed out based on real-world web services obtained from the WSDL service retrieval test collection called *SAWSDL-TC3*[4]. The WSDL corpus consists of 1088 semantically annotated WSDL 1.1-based Web services which cover 9 different application domains. Each web service belongs to one out of nine service domains named as: Communication, Education, Economy, Food, Geography, Medical, Military, Travel and Simulation. The dataset contains different queries (i.e. 42 requests). A service request is defined as a service that would perfectly match the request. Furthermore, a binary and graded relevance set for each query is provided which can be used in order to compute Information Retrieval (IR) metrics. Table I lists the number of services and requests from each domain.

#	Domain	Services	Queries
1	Communication	58	2
2	Economy	358	12
3	Education	285	6
4	Food	34	1
5	Geography	60	10
6	Medical	73	1
7	Military	40	1
8	Simulation	16	3
9	Travel	164	6
	Total	**1088**	**42**

TABLE I
NUMBER OF SERVICES AND QUERIES FOR EACH DOMAIN

Before applying the proposed approach, we deal the WSDL corpus. The objective of this preprocessing is to identify the

[4]http://www.semwebcentral.org/projects/sawsdl-tc

textual concepts of services, which describe the semantics of their functionalities. WSDL corpus processing consists of several steps: *Features extraction*, *Tokenization*, *Tag and stop words removal*, *Word stemming* and *Service Transaction Matrix construction* (see [13] for more details). The observed textual concepts are represented in a Service Transaction Matrix (STM). We use the STM as training data for our implementation of the CTM model (based on the Blei's implementation[5], which is a C implementation of CTM using Variational EM for Parameter Estimation and Inference).

B. Web services discovery and ranking evaluation

	Topic-RDD			MMR		
Alpha	**Pr@5**	**Pr@10**	**Pr@15**	**Pr@5**	**Pr@10**	**Pr@15**
0.00	0.748	0.750	0.726	0.686	0.695	0.710
0.10	0.748	0.750	0.726	0.686	0.695	0.710
0.25	0.748	0.750	0.726	0.686	0.698	0.712
0.50	0.748	0.750	0.729	0.705	0.717	0.715
0.75	0.748	0.752	0.731	0.705	0.719	0.712
0.90	0.748	0.750	0.726	0.700	0.714	0.712
1.00	0.724	0.714	0.699	0.724	0.714	0.699

TABLE II
COMPARISON OF AVERAGE PR@N VALUES OF OUR METHOD TOPIC-RDD AND BASELINE MMR ALGORITM[16] OVER ALL 42 QUERIES.

In order to evaluate the accuracy of our approach, we compute two standard measures used in *Information Retrieval*: *Precision at n* (*Precision@n*) and *Normalised Discounted Cumulative Gain* ($NDCG_n$). These are standard evaluation techniques used in IR to measure the accuracy of a search and matchmaking mechanism. For this experiment, we use the service descriptions collected from the test collection. As described previously, the services are divided into nine domains and some service requests are provided together with a relevant response set for each query. The relevance sets for each query consists of a set of relevant services and each service s has a graded relevance value $relevance(s) \in \{1, 2, 3\}$ where 3 denotes *high relevance* to the query and 1 denotes a *low relevance*.

1) ***Precision@n****:* In our context, $Precision@n$ is a measure of the precision of the service discovery system taking into account the first n retrieved services. Therefore, $Precision@n$ reflects the number of services which are relevant to the user query. The precision@n for a list of retrieved services is given by Equation 8:

$$Precision@n = \frac{|RelevantServices \cap RetrievedServices|}{|RetrievedServices|} \quad (8)$$

Where the list of relevant services to a given query is defined in the test collection.

2) ***Normalized Discounted Cumulative Gain****:* $NDCG_n$ uses a graded relevance scale of each retrieved service from the result set to evaluate the gain, or usefulness, of a service based on its position in the result list. This measure is particularly useful in Information Retrieval for evaluating ranking results. The $NDCG_n$ for n retrieved services is given by Equation 9.

$$NDCG_n = \frac{DCG_n}{IDCG_n} \quad (9)$$

[5]http://www.cs.princeton.edu/~blei/ctm-c/index.html

Where DCG_n is the Discounted Cumulative Gain and $IDCG_n$ is the Ideal Discounted Cumulative Gain. The $IDCG_n$ is found by calculating the DCG_n of the first n returned services. The DCG_n is given by Equation 10.

$$DCG_n = \sum_{i=1}^{n} \frac{2^{relevance(i)} - 1}{log_2(1+i)} \quad (10)$$

Where n is the number of retrieved services and $relevance(s)$ is the graded relevance of the service in the *i*th position in the ranked list. The $NDCG_n$ values for all queries can be averaged to obtain a measure of the average performance of a ranking algorithm. $NDCG_n$ values vary from 0 to 1.

We evaluated the effectiveness of our probabilistic service matchmaking and ranking based on the relevance, diversity and density measures (labelled *Topic-RDD*) by calculating the $Precision@n$ and $NDCG_n$. The sample queries (i.e. 42 queries) provided in the dataset are all in the form of SAWSDL documents and contain the semantic requirements together with a text description of the queried functionality. Our method Topic-RDD proposed in this paper is compared with a syntax-based approach powered by *Apache Lucene*[6] and also method from the *SAWSDL-MX2 Matchmaker*[7] hybrid semantic matchmaker for SAWSDL services[20]. Our method is also compared with a baseline ranking algorithm Maximal Marginal Relevance (MMR) [16] (labelled *MMR*) which takes into account the diversity in search results set in the context of feedback documents retrieving. MMR measure is defined as follow:

$$MMR \stackrel{def}{=} \underset{s_i \in R \setminus S}{\operatorname{argmax}} [\alpha . Sim(q|s_i) - (1-\alpha) \max_{s_i \in R} Sim(s_i, s_j)] \quad (11)$$

Where R is the initial ranked web services set and S is the subset of services in R already selected; $R \setminus S$ is the set difference. α is weighting parameter. $Sim(q, s) = P(q|s)$ (i.e. Equation 2) is the similarity metric used in service retrieval and relevance ranking between services and a query.

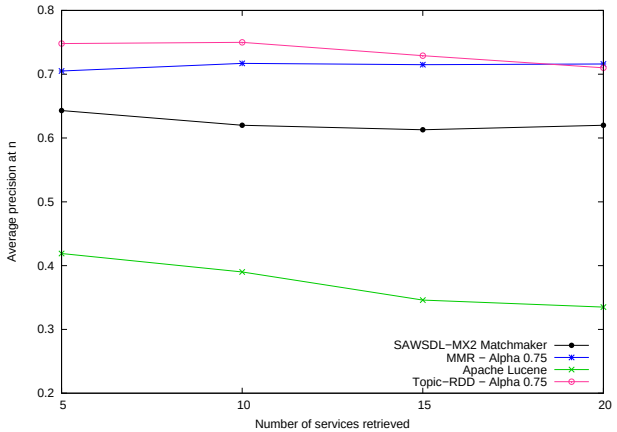

Fig. 2. Comparison of average $Precision@n$ values over 42 queries.

Generally, the top most relevant retrieved services are the main results which are selected and used by the user. The

[6]http://lucene.apache.org/
[7]http://projects.semwebcentral.org/projects/sawsdl-mx

Alpha	Topic-RDD			MMR		
	NDCG@5	NDCG@10	NDCG@15	NDCG@5	NDCG@10	NDCG@15
0.00	0.537	0.578	0.599	0.435	0.478	0.580
0.10	0.540	0.579	0.599	0.435	0.478	0.580
0.25	0.540	0.579	0.598	0.435	0.479	0.581
0.50	0.540	0.580	0.599	0.440	0.487	0.586
0.75	0.544	0.582	0.598	0.437	0.484	0.581
0.90	0.525	0.565	0.598	0.443	0.484	0.584
1.00	0.542	0.597	0.642	0.542	0.597	0.642

TABLE III
COMPARISON OF AVERAGE NDCG@N VALUES OF OUR METHOD TOPIC-RDD AND BASELINE MMR ALGORITHM[16] OVER ALL 42 QUERIES.

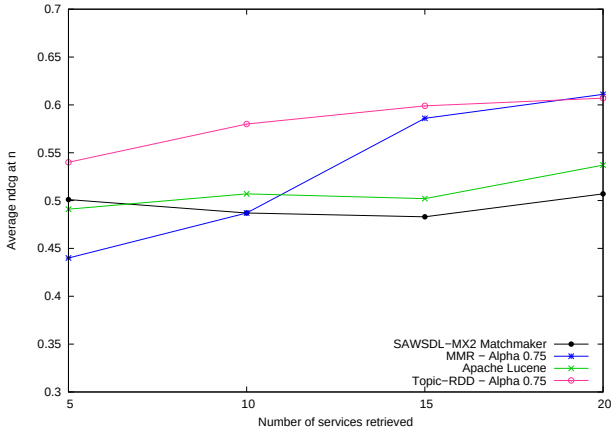

Fig. 3. Comparison of average $NDCG_n$ values over 42 queries.

averaged $Precision@n$ and $NDCG_n$ were measured for up to the first 15 retrieved services from the complete list of results. In Information Retrieval, $NDCG_n$ gives higher scores to systems which rank a search result list with higher relevance first and penalizes systems which return services with low relevance.

For our method Topic-RDD and MMR algorithm, the 42 queries are represented in a vector of topics using Equation 1 and then matched to the services in topic space using a mechanism described in Section III. For SAWSDL-MX2, the search queries are submitted using the SAWSDL-MX user interface. For the ApacheLucene approach, the text descriptions taken from the query template are used as the query string.

In our experiments, we analyse the impacts of the parameter $\alpha \in [0, 1]$ (i.e. Equation 3) which is leveraged to tradeoff the web servicres relevance, density and diversity. The average $Precision@n$ and $NDCG@n$ values are obtained over all 42 queries for Topic-RDD and MMR taking into account different values of α. Setting $\alpha = 1$, the system uses only the relevance measure to select the web services that match with user's query; if $\alpha = 0$, the algorithm selects web services only based on its diversity and density performance measures. The results are shown in Tables II and III respectively and show that our method gives a higher precision when the three measures (relevance, diversity and density) are combined to select best web services that match with user's query. The comparison of average $Precision@n$ (i.e. Table II) shows that our probabilistic service discovery and ranking performs better than the MMR algorithm for all values of α. As can be seen from these results, our method gives a higher precision when $\alpha = 0.75$.

The average $Precision@n$ and $NDCG@n$ values are also obtained over all 42 queries for *ApacheLucene*, *SAWSDL-MX2 Matchmaker*. The results are shown in Figure 2 and 3 respectively. In both cases, the results show that our method Topic-RDD gives a higher average $Precision@n$ and $NDCG_n$ for all 42 queries. In fact, our method perform better than all methods. The results show that ApacheLucene and SAWSDL-MX2 were unable to find some of the relevant web services that were not directly related to some of the requests through keywords or logic descriptions. This reflects that the retrieved services obtained by our method are specific to the user's query. ApacheLucene and SAWSDL-MX2 have a low $NDCG_n$ because, as shown in the $Precision@n$ results, both approaches are unable to find some of the highly relevant services. Our method based on the probabilistic model used the information captured in the latent factors to match web services based on the service relevance, density and diversity. The results obtained for our method reflect the accuracy of our ranking system.

C. *Service dependency network structure evaluation*

Networks properties	Full invocation	Partial invocation
size	1088 nodes	1088 nodes
Number of links	5776 arcs	6729 arcs
Density	0.0052	0.0060
The average degree	10.9914	12.8049
Transitivity	0.0123	0.0162
Diameter	8	7

TABLE IV
PROPERTIES OF WEB SERVICE DEPENDENCY NETWORKS EXTRACTED FROM SAWSDL-TC3 COLLECTION

In order to evaluate the structure of web services dependency networks extracted from the SAWSDL-TC3 collection, we compute several properties defined as follow:

- **Density:** is defined as the number of links divided by the number possible. If self-loops are excluded, then the number possible is n(n-1)/2. If self-loops are allowed, then the number possible is n(n+1)/2.
- **Diameter**: the maximum distance between any pair of nodes in the graph.
- **Degree**: is the number of links connected to a node. The average degree represent the average value of degrees in the whole network.

- **Transitivity**: a network transitivity corresponds to its density of triangles, where a triangle is a structure of three completely connected nodes.

Figure 4 show the exact service dependency network from SAWSDL-TC3. The different values obtained for principal network properties are shown in Table IV. The exact service dependency network exhibits the same global structure as the approximate service dependency network (i.e. table IV). We have the same web service nodes whether we use partial or full invocation mode. The number of vertices is 1088; this value corresponds to the number of wsdl files in the collection test SAWSDL-TC3. Network nodes are distributed among large, small components, and a set of isolated nodes. In the exact service dependency network, there is less directed edges (or arcs) between web service nodes than in the approximate service dependency network. Indeed, directed edges represent the existence of fully or partially invocable operations between web service nodes. Partially invocable operations include a set of fully invocable operations. We can observe also that the approximate service dependency network has the highest transitivity. This is due to the fact that this network has the highest number of directed edges. The exact service dependency network is characterized by its lower density. This is way it can be efficient to be minded first to search for composition. Note also that the diameter tells us about the number of operations required in large compositions. It is defined as the greatest distance between any pair of nodes in a graph.

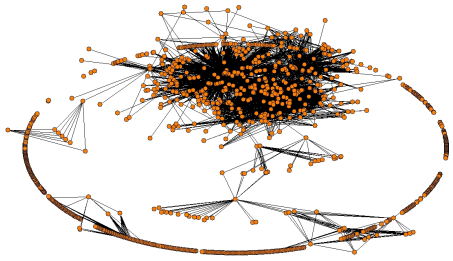

Fig. 4. Exact service dependency network (Full invocation)

VI. Conclusion

In this paper, we proposed a probabilistic approach for discovery of web services and their compositions in order to minimize the redundancy and increase the user's satisfaction. A diversified web service ranking algorithm is proposed to find the top-k diversified web service ranked set based on their relevance, diversity and density. In our approach, we use the probabilistic topic models to extract topics from semantic service descriptions. The generated topics are used as the base criteria for computing the relevance, service diversity and service density measures. We also proposed a method to construct the web service dependency network based on the Formal Concept Analysis formalism. The service dependency network is used to select the composition of discovered web services set. An algorithm for finding top-k optimal web services compositions is also proposed in this paper. Experimental results on a real world web service dataset show that the proposed approach improves the web service recommendation performance in terms of precision. The results show that our method gives a higher precision when the three measures (i.e. relevance, diversity and density) are combined to select best web services that match with user's query.

In future work, we will propose a method to evaluate and rank community detection algorithms. To achieve this goal we will consider a probabilistic topic based approach to define a new measure called semantic divergence of communities. The proposed method for generating web services network will be used to construct the services graph that will be used as input to communities construction.

References

[1] P. Rodriguez Mier, C. Pedrinaci, M. Lama, and M. Mucientes: An Integrated Semantic Web Service Discovery and Composition Framework, IEEE TSC 2015 (DOI 10.1109/TSC.2015.2402679).

[2] Klusch, M.; Kapahnke, P. and Zinnikus, I.: Hybrid Adaptive Web Service Selection with SAWSDL-MX and WSDL-Analyzer, The Semantic Web Research and Applications (2009), pp. 550-564.

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

[4] J. Wu, L. Chen, Z. Zheng, M. R. Lyu, and Z. Wu, Clustering web services to facilitate service discovery, KAIS'2014 vol. 38(1) , pp. 207-229,

[5] Mohebbi, K., Ibrahim, S., Khezrian, M., Munusamy, K., and Tabatabaei, S. G. H.: A comparative evaluation of semantic web service discovery approaches. In iiWAS, ACM 2010. pp. 33-39.

[6] Cassar, G., Barnaghi, P. and Moessner, K.: Probabilistic Matchmaking Methods for Automated Service Discovery. IEEE TSC'2013.

[7] W. Lin, W. Dou, X. Luo, J. Chen. "A history record-based service optimization method for QoS-aware service composition", IEEE ICWS 2011 pp. 666-673.

[8] Aznag, M., Quafafou, M. and Jarir, Z.: WS-Portal; An Enriched Web Services Search Engine. In 12th ICSOC'2014, Paris, France.

[9] Aznag, M., Quafafou, M., Rochd, El M., and Jarir, Z.: Probabilistic Topic Models for Web Services Clustering and Discovery. In ESOCC'2013, LNCS 8135, pp 19-33.

[10] Bache, K.; Newman, D. & Smyth, P.: Text-based measures of document diversity., in ACM KDD'2013, pp. 23-31.

[11] Oh,Seog-Chan, Lee, Dongwon and Kumara, Soundar R. T. "Effective Web Service Composition in Diverse and Large-Scale Service Networks" IEEE Trans. Services Computing 2008: 15-32.

[12] Hashemian, S. V.Mavaddat, F. A Graph-Based Approach to Web Services Composition., In SAINT'2005, pp. 183-189

[13] Aznag, M., Quafafou, M. and Jarir, Z.: Correlated Topic Model for Web Services Ranking. In IJACSA, vol. 4, no. 6, pp. 283–291, July 2013.

[14] Aznag, M., Quafafou, M. and Jarir, Z.: Leveraging Formal Concept Analysis with Topic Correlation for Service Clustering and Discovery. In 21th IEEE ICWS 2014. Alaska, USA.

[15] Steyvers, M. and Griffiths, T.: Probabilistic topic models. In Latent Semantic Analysis: A Road to Meaning, 2007.

[16] J. G. Carbonell and J. Goldstein. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In SIGIR, pages 335-336, 1998.

[17] Rokia Missaoui, Léonard Kwuida, Mohamed Quafafou and Jean Vaillancourt : Algebraic operators for querying pattern bases. In ICFCA 2009.

[18] Zaki, M.J., Hsiao, C.-J.: Efficient Algorithms for Mining Closed Itemsets and Their Lattice Structure. IEEE Trans. on KDE 2005. 17(4), pp. 462-478.

[19] Rong-Hua Li; Yu, J.X: Scalable diversified ranking on large graphs. IEEE Trans. on KDE, vol. 25(9), pp. 2133-2146, 2013.

[20] Klusch, M.; Kapahnke, P. and Zinnikus, I.: Hybrid Adaptive Web Service Selection with SAWSDL-MX and WSDL-Analyzer. In ESWC'09, pp. 550-564.