Social network-based Semantic web services discovery

Huicong Ning, Xiuli Shao

1 College of Information Technology and Science, Nankai University, Tianjin, 300071, china, nhc@mail.nankai.edu.cn
2 College of Information Technology and Science, Nankai University, Tianjin, 300071, china shaoxl@nankai.edu.cn

Abstract

Discovering precise and effective semantic Web services is a fundamental construction for Service Oriented Architecture. There are two important aspects of semantic Web services discovery: The similarity and the interactions between semantic Web services. This paper, firstly, systematically investigates the similarity and the interaction relationships; and then gives the mechanism of how to build the quantitative social network of semantic Web services, which includes building, storing, updating and querying on the social network; finally, summaries and compares with related work. The results of analysis show that our proposed strategy of social network-based Web services discovery is efficient and practical. Our methods could also lay a solid foundation for the semantic Web services discovery using Social networks Analysis.

Keywords: Semantic web service, Social network, Web service discovery, Quality of service

1. Introduction

Service oriented architecture (SOA) is one of the most widely accepted architectural approaches to building and integrating applications, it emphasizes the reuse of existing and distributed software services to suite their application needs. Web services are the most successful realization of the SOA. With the Web services increased dramatically, the mechanism is crucial to discovering and ranking web services automatically and efficiently.

Numerous efforts have been made to improve Web services discovery. The related approaches can be summarized into three categories: (1) the first methods extract textual information from the Web Service Description Language (WSDL) documents based on Information retrieval (IR) (e.g., [1][2][3]); (2) the second approaches propose ontology-based semantic Web services discovery, this approaches calculate the ontology concept semantic similarity from function capability and quality of services (QOS) (e.g.,[3][4][5]) or reason using the logic approaches [3]; (3) the last not only concern the syntax and the semantic but also try to focus on the interaction relationships, they propose a social network model to discover the expected Web services (e.g.,[6][7][8][9]).

In this paper, we systematically study the relationships between semantic Web services, e.g., Atomic services and Web services orchestration and choreography; based on the relationships, we give in detail how to build the quantitative social network of semantic Web services (See this description in Section 3), And then, Integrated the advantages of IR-based and ontology-based methods, we proposed method and mechanism based on the idea of TF-IDF [2] to improve query performance and to reduce the impact of social network aggregation.

The remainder of this paper is organized as follows. Section 2 provides the background and related work that was used in the rest papers. In section 3, we will describe how to construct the quantitative social network of semantic Web services and Web services consumers query on the social network, followed by the summary and analysis in section 4. Finally, the conclusion and future work are given in section 5.

2. Background and related work

Paper [7] combined Semantic Web and Social Networking technology to discovery Web service. The authors divided the Web services into the abstract Web services layer and actual
Web services layer (the actual Web service is an instance of the abstract Web services), they defined a formal Service Relation (SR) of Service Network (SN) as Eq. (2.1) [7], and then given the relationships between the three levels of Web services: the parameter level, the operation level, and the service level. Nevertheless, the authors did not explicitly give on how to use these relationships to build SN; on the other hand, the relationship between the services only for a qualitative study, and did not give quantitative research.

$$SN(V, E) = \text{Graphsn} <Vsn, \text{SRsn}>$$

Where, V represents the Web service Node, E represents the edge of Vi and Vj, if the Service Vi has relationship with Vj (Vi, Vj \in V). Vsn = \{Abstract Services\} \cup \{Actual Web Services\}, SRsn = Vsn \times Vsn.

In [8] the authors refined networks to highlight the two scenarios of Recommendation (RR) and Collaboration (RC) between Web services. Furthermore, they classify the Recommendation between Web services to Partnership (Rp) and robustness (Rr) [8]. They proposed a Social Network SN of a Web Service WS with a couple \(< WSi, t, w, WSh >\), where t is the name of the edge from WSi to WSh, and w is the weight of the edge. The authors calculated numerically the relationship between Web services. Nevertheless, they did not explain how to handle this scenario, for example, when a service WSi has a Partnership WSh and a collaborative WSh, and the weight \(< WSi, WSh >\) is equal to the weight \(< WSi, WSh >\). That is, we have \(< WSi, Rp, 0.2, WSh >\) and \(< WSi, RC, 0.2, WSh >\). When services consumers have selected WSi, which one of WSh and WSh should recommend to services consumers.

Paper [9] illustrated their algorithm from the aspect of Web services composition by an example of a travel to a conference. They measured the two Web services using Dijkstra’s Shortest Path based on the function and QOS. The authors given a preliminary investigation that can’t meet the complex, dynamic relationship of Web services on the internet.

3. Social network for semantic web services discovery

This section is the main of this paper, it consists of three parts: The first part gives the architecture of semantic web services social network. The second part discusses how to build the social network of semantic Web services. Finally, we propose an ontology-based query mechanism on the social network and how to maintain well-established social network.

3.1. Architecture of semantic Web services social network

We firstly define the social network of semantic Web service (SNws):

**Definition 1:** SN (V, E) = \text{Graphsn} <Vsn, \text{SRsn}>

Where the V and E are the set of Nodes and Edges, respectively.

Each Node n \in V is a tuple of the form \(<\text{ServiceID}, \text{type}, \text{hierarchy Degree}>\), where ServiceID is the unique identification of Web service; the type is the relation’s type of this node, it is a set \{Atomic, Competitive relation, Collaborative relation\}; the hierarchyDegree is degree of this service in hierarchy, the atomic Web service’ degree is 1, if two or more atomic composite a new service, the new service’s degree is 2, if the degree 2 of Web service composite with any service not include itself, the degree add 1, and so on.

Each edge e \in E is a tuple of the form \(<WSi, t, w, WSh>\), where t is the name of the edge from WSi to WSh, it is a set \{Competitive, Collaborative\}, and w is the weight of the edge.

$$w = \text{basicsimilarity} + \text{interaction relationship}$$

Where basic similarity include basic information, function similarity, no-function similarity and QOS similarity. Readers can refer to [4] for a thorough understanding of these calculations. In our work, we are concerned about building the social network using the similarity of Web services instead of calculating similarity.

Secondly, we define the architecture of semantic Web services social network in Fig.1. Social network-based Semantic web services discovery is to use the weight of the edge to find Web services. As depicted in Fig.1, the architecture consists of four modules:

1. The services provider module: The main functions include Web service publishing, Annotation, registering and updating.
2 Social network services module: this module provides Web services consumers with satisfactory services according to the weight edge of the social network services. It also builds and maintains the relationship of Web services on the social networking (See this description in Section 3.3.1 and 3.3.3).

3 The domain ontology set module provides semantic support for Web services.

4 The module of services consumers query mechanism will reply best results to the service consumers though the social network of semantic Web services (See this description in Section 3.3.2).

In this paper, we study the module 2 and the module 4. We focus on three aspects as following:

- How to build the social network of semantic Web services (In module 2).
- How to maintain the relationship of semantic Web services on the social networking (In module 2).
- How to reply the Web services request on the social networking (In module 4).

3.2. Building social network of semantic Web services

The social networks of Web services are different from the classical type of social networks of people; the latter is based on the absolute collaboration and mutual assistance between their members (i.e., no competition) [7][8][9][10]. To take advantage of classical social networking technologies, we propose two basic ideas, to create social network Web services.

- The first main idea of our work is to eliminate competition or to make the competition does not affect the social network analysis.
- The second main idea is to model the relationship of Web services from level of Atomic service.

Based on the two main ideas, we can translate the relationships to Atomic Web services competitive and collaborative relationship. The relationships of atomic services are shown in Table 1.

---

**Figure 1. Architecture of semantic web services social network**
Table 1. Relationships of atomic services

<table>
<thead>
<tr>
<th>Categories</th>
<th>Description (Atomic services S1 AND S2)</th>
<th>Relationship Types</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equal</td>
<td>S1 and S2 have the same basic information and operation</td>
<td>Competitive relationship</td>
</tr>
<tr>
<td>Exact</td>
<td>S1input ∈ S2output, or, S2input ∈ S1output</td>
<td>Competitive relationship</td>
</tr>
<tr>
<td>Plug-in</td>
<td>S1input ∈ S2output</td>
<td>Competitive relationship</td>
</tr>
<tr>
<td>Subsume</td>
<td>SRsubsume&lt; S1, S2&gt;=SRplug-in&lt; S1, S2&gt; Medium</td>
<td>Collaborative relationship</td>
</tr>
<tr>
<td>Invoke dependent</td>
<td>The parameter in S1 should be produced from S2</td>
<td>Collaborative relationship</td>
</tr>
<tr>
<td>orchestration</td>
<td>Sequence</td>
<td>Collaborative relationship</td>
</tr>
<tr>
<td></td>
<td>Choice</td>
<td>Collaborative relationship</td>
</tr>
<tr>
<td></td>
<td>Iterate</td>
<td>Collaborative relationship</td>
</tr>
<tr>
<td></td>
<td>Parallel execution (split joint and joint split)</td>
<td>Competitive relationship</td>
</tr>
<tr>
<td></td>
<td>Same as orchestration</td>
<td>Same as orchestration</td>
</tr>
</tbody>
</table>

As shown in table 1, the relationship types are the competitive relations and collaborative relations. We conclude the following transformation rules:

- Equal, exact, plug-in, subsume \( \rightarrow \) relationship about two atomic services
- Invoke dependent \( \rightarrow \) orchestration. Sequence
- Orchestration. Choice and iterate, parallel execution \( \rightarrow \) orchestration sequence

The following illustrate our approach through an example (modified example based on [8]), which section 3.2.1 describes the scenario; section 3.2.2 studies the competitive relationship; section 3.2.3 analyzes the relationships of collaborative; the section 3.2.4 shows the social network of this example.

3.2.1. Illustration about the example

Paper [8] gives a scenario purchase-order example, now we modify this example as fig. 2. The purchase process is, firstly, the customer order the product Web service (product-WS), then the product-WS order inventory-WS, if the number of Inventory is enough, then the inventory-WS invoke Shipper-WS; else should invoke the Supplier-WS firstly, then invoke the Shipper-WS.

![Figure 2. Modified purchase-order scenario representation](image)

To better illustrate our approach, we assume the product-WS and shipper-WS have two atomic Web services which can provide the similar functionality and QOS as in Table 2. The parameters of Web services are as follow:

Table 2. Parameters of atomic services

<table>
<thead>
<tr>
<th>Service Name</th>
<th>Input</th>
<th>Output</th>
<th>QOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product-WS1</td>
<td>productName, data, price</td>
<td>productID, orderNumber</td>
<td>High</td>
</tr>
<tr>
<td>Product-WS2</td>
<td>productName, color, price</td>
<td>productID, orderNumber</td>
<td>low</td>
</tr>
<tr>
<td>Inventory-WS</td>
<td>productID, existingNumber</td>
<td>productNo, number, price, time</td>
<td>High</td>
</tr>
<tr>
<td>Supplier-WS</td>
<td>productID, demandNumber</td>
<td>productNo, number, price, time</td>
<td>High</td>
</tr>
<tr>
<td>Shipper-WS1</td>
<td>productNo, number, price, time</td>
<td>satisfaction, not satisfaction</td>
<td>High</td>
</tr>
<tr>
<td>Shipper-WS2</td>
<td>productNo, number, price, time</td>
<td>satisfaction, not satisfaction</td>
<td>Medium</td>
</tr>
</tbody>
</table>

The weight about Web services as in Table 3.
Table 3. Weight of atomic Web services

<table>
<thead>
<tr>
<th>Two atomic Services</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Product-WS1, Product-WS2&gt;</td>
<td>-0.6</td>
</tr>
<tr>
<td>&lt;Product-WS1, Inventory-WS&gt;</td>
<td>0.6</td>
</tr>
<tr>
<td>&lt;Product-WS2, Inventory-WS&gt;</td>
<td>0.8</td>
</tr>
<tr>
<td>&lt;Inventory-WS, Supplier-WS&gt;</td>
<td>0.2</td>
</tr>
<tr>
<td>&lt;Inventory-WS, Shipper –WS1&gt;</td>
<td>0.8</td>
</tr>
<tr>
<td>&lt;Inventory-WS, Shipper –WS2&gt;</td>
<td>0.8</td>
</tr>
<tr>
<td>&lt;Supplier -WS, Shipper –WS1&gt;</td>
<td>0.8</td>
</tr>
<tr>
<td>&lt;Supplier -WS, Shipper –WS2&gt;</td>
<td>0.8</td>
</tr>
<tr>
<td>&lt;Shipper –WS1, Shipper –WS2&gt;</td>
<td>-0.9</td>
</tr>
</tbody>
</table>

3.2.2. Competitive Relations

Competitive relationship (CR) is when a Web service knows the peers that can substitute for it when it fails [8], such as the product-WS1 and product-WS2. The weight of competitive edge $W_{cr}$ is given by the following equation (A similar definition in paper [8], Nevertheless, this is negative.):

$$W_{cr}(WS_i, WS_j) = \frac{|WS_{selection}|}{|WS_{failure}|}$$  \hspace{1cm} Eq. (3.2)

Where $W_{cr}$ correspond to the number of times that $WS_i$ has failed in compositions and the number of times that $WS_j$ has substituted successfully for $WS_i$ in these compositions.

If product-WS1 has failed 5 times and there are three times that product-WS2 has substituted successfully, $W_{cr}(product – WS1, product – WS2) = -\frac{3}{5} = -0.6$.

The weight of assembled new Web service is the sum of its child-WS divides the number of child –WS.

$$W_{assembled}(WS_i) = \frac{sum(child – WS)}{Number(child – WS)}$$  \hspace{1cm} Eq. (3.3)

Now we give how to eliminate the competition, the definition is as following:

**Definition 2**: if a Web service (Inventory-WS1) has relationships with the other two services (Product-WS1 and Product-WS2) and the other two services have the negative relationship, we merge the two services into a large atomic service.

It can be shown in Fig.3. By above merging rules, the Fig.3 (a) converts to the Fig.3 (b). The Fig.3 (c) is the storage of social relations; the Fig.3 (d) is on behalf of the converted storage.

Figure 3. Merge the competitive Web services into a large atomic Web services
3.2.3. Collaborative relations

Collaborative relationship (COR) is the result of engaging Web services in the same or ongoing composition to answer complex users’ needs, such as the inventory-WS and supplier-WS. The weight of competitive edge \( W_{t,cor} \) is given by the following equation:

\[
W_{t,cor}(WS_i, WS_j) = \frac{WS_{participation_i}}{WS_{participation_j}}
\]

Where \( W_{t,cor} \) correspond to the number of times that WSi has participated and the number of times that WSj also has participated.

When inventory-WS occurs 5 times, the supplier-WS2 occurs 1 times successfully.

\[
W_{t,cor}(\text{inventory-WS}, \text{supplier-WS}) = \frac{5}{1} = 0.2.
\]

Now we give how to combine them into a large atomic Web service. The definition is as follows:

**Definition 3**: if a Web service (Inventory-WS) has relationships with the service (supplier-WS), and their relationship is collaborative, we merge the two services into a large atomic service.

It can be shown in Fig.4. By above merging rules, the Fig.4 (a) converts to the Fig.4 (b). The Fig.4 (c) is the storage of social relations; the Fig.4 (d) is on behalf of the converted storage, we only give the related nodes and edges.

![Figure 4. Merge the collaborative Web services into a large atomic Web services](image)

3.2.4. Results of the social network

Based on the above discussion, we established social network according to the example in section 3.2.1, which the gray circles represent assembled service node; the white circles are on behalf of atomic service node. The core social network is in the large oval, while dashed oval represents the local social network. As the relationship between the core network services, the social network provides a platform for study Web services using social network analysis.
In Fig.5, there are three new combination nodes, where the input and output for the combined input and output of atomic services, as shown in Table 4.

<table>
<thead>
<tr>
<th>Service Name</th>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product-WS</td>
<td>productName, date, color, price</td>
<td>ProductID, orderNumber</td>
</tr>
<tr>
<td>InventorySupplier-WS</td>
<td>productId, existingNumber, demandNumber</td>
<td>productNo, number, price, time</td>
</tr>
<tr>
<td>Shipper-WS</td>
<td>productNo, number, price, time</td>
<td>satisfaction, not satisfaction</td>
</tr>
</tbody>
</table>

3.3. ontology-based query and update mechanism

In order to query based on the social network of semantic Web services, we propose the TF-IDF of Web services. The rules are as following:

- The parameters (e.g., ontology of input, output) are term, the semantic Web service is document.
- The semantic Web service is term, the assembled semantic Web services.

We give their storage in Fig. 6.

3.3.1. Query mechanism

Here, we have an example of input about product-WS, to illustrate how to query, and other parameters and services are similar.

Firstly, finding the ontology of the customer’s request parameter; secondly, according the parameter ontology, finding the Web services that have this parameter; thirdly, carrying out XOR operation of these services to meet the query; and then comparing the weight; finally return the results. In the process of comparing the weight, the rules are following:

- If the type of Web service meted with the request is assembled Web service, we should find out which child-Web services involved in service. Then dynamically compute the weight according the child-Web services involved. For example, in request1, the product-WS can meet the request, in fact, the product-WS2 participated and product-WS1 did not participate, so the weight is 0.6/1=0.6 by the Eq.3.3.
- When atomic Web service of low degree and a composed large atomic Web service of high degree have the same weight, the low degree’s Web service rank first. For example, in request 1, the product-WS2 and the product-WS both can meet the request, but the product-WS’s degree is 2 greater than the degree of product-WS2 (the degree is 1), so the product-WS2 is front of product-WS.
• If only have one Web service can meet the request, we can’t comparing the weight. According the type of Web service, the program executes the process. For example, in request 2, only have product-WS can meet the request, and the type of product-WS is competitive, it include product-WS1 and product-WS2, so the competitive relationship convert to cooperation.

In request 1, the relationship between product-WS1 and product-WS2 is competition; while in request 2, due to meet the minimum atomic service is product-WS, so the relationship between product-WS1 and product-WS2 is collaboration. From the query process, we method can better handle this problem.

Figure 6. Query mechanism

3.3.2. Updating mechanism

The social network updating includes adding a new Web service node and a service withdrawing from the social network.

When adding a new Web service, we should calculate the similarity according the section 3.2, and then locate it in the appropriate node in the social network.

When a Web service withdraws from the social network, we should query the related assembled Web service like the 3.3.1, firstly; and then rebuild the social network relationship about the related assembled Web service. Finally, we delete this Web service node.

4. Summary and Analysis

This section compares our approach with the methods in [7], [8] and [9]. As summarized in Table 5, we can draw the conclusion that the main contributions of this work are the following:

• Based on the detailed analysis the relationships of semantic Web services, we systematically study how to build, storage, query and maintain the social network of Web services.

• Different from related social network of Web services, our approach eliminates the competition relationships, so bridges the gap between the study about social network of semantic Web services and the existing knowledge in social network analysis.
Our approach not only clearly gives the step of building the social network of Web services but also, considering the services consumers, describes the algorithm that how to find Web services on the social network.

Table 5. Comparison with related methods

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis of the relationship between Web services</td>
<td>Yes</td>
<td>Yes</td>
<td>A simple</td>
<td>Yes</td>
</tr>
<tr>
<td>Whether is the quantitative research</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>The steps of building the social network Services</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>The type of the social network Services</td>
<td>Competitive and collaborative</td>
<td>Competitive and collaborative</td>
<td>Considered only a simple</td>
<td>Integration of competition, to establish a collaborative relationship</td>
</tr>
<tr>
<td>Social network maintenance</td>
<td>No</td>
<td>Description</td>
<td>No</td>
<td>Formulation</td>
</tr>
<tr>
<td>Query mechanism of Web services customers</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Rank the Web services by the weight of Edges</td>
<td>No</td>
<td>Description</td>
<td>No</td>
<td>Formulation</td>
</tr>
</tbody>
</table>

Where “yes” present the paper give this aspect, “no” present not consider this aspect, “Description” is represent the paper describe but not formulate this aspect.

5. Conclusions and future work

This paper presents a novel semantic Web services discovering approach that is based on social network, which includes the building, maintaining the social network of semantic Web services and querying on the social network. Compare with existing methods, our approach can improve query performance, reduce storage space, simplify the query mechanism and provide a good platform for the classic social network analysis.

Our future work mainly includes two parts: on the one hand, we will continue to build the existing social network of Web services based on the actual Web services conditions; on the other hand, we will improve the semantic Web services discovery using the technology in social network analysis.

6. References