ISSN 1607-9264

JOURNAL of INTERNET TECHNOLOGY

網際網路技術學刊

No.4
Volume 15
2014

INVITED PAPER

Reducing Energy Consumption in Access Networks: Comparison and Analysis
Shijia Zhu, Deyun Gao, Yujing Zeng, Fei Song, Hongke Zhang

REGULAR PAPER

Impact of P2P-Assisted and Interaction-Oriented Content Distribution for Online Social Networks
Kai Shuang, Bin Du, Sen Su

Generic Utility Based Resource Allocation Optimization for Broadband Wireless Networks with Relays
Hongxian Tian, Bin Fan, Rongtao Xu, Jia You

The Design and Evaluation of an Affective Tutoring System
Kuo-Chun Hsu, Hao-Chiang Koong Lin, I-Long Lin, Jin-Wei Lin

An On-Demand Services Discovery Approach Based on Topic Clustering
Zheng Li, Keqing He, Jian Wang, Neng Zhang

An Efficient Multicast Strategy for On-Demand Data Dissemination in Wireless Networks
Shujuan Wang, Mangui Liang

Realization of Communications-as-a-Service: IPTV Distribution Platform and a Network-Based IPTV Audience Measurement Scheme
Chi-Yi Lin, Li-Chia Yeh, Ching-Sheu Wang, Jenn-Wei Lin

An Efficient Architecture for Parallel Skyline Computation over Large Distributed Datasets
He Li, Sumin Jang, Jaesoo Yoo

SPECIAL ISSUE ON ADVANCED CONVERGENCE TECHNOLOGIES: BIG DATA, IOT, CLOUD COMPUTING

Robot Service Framework Based on Big Data Technology
Yunsick Sung, Hwa-Young Jeong, Kyungeun Cho, Kyhyun Um

Organizing a User-Created Computing Environment by RESTful Web Service Open APIs in Desktop Grids
Joon-Min Gil, Jaehwa Chung, Young-Sik Jeong, Doo-Soon Park

A Comparative Study of Authentication Schemes with Security and Usability of IPAS
Sadiq Almuairfi, Prakash Veeraraghavan, Naveen Chilamkurti

Energy-Efficient Fragmentation Scheme for IEEE 802.11 DCF: Delay and Fairness Considerations
Prosper Mafole, Yuki Manabe, Teruaki Kitasuka, Masayoshi Aritsugi

Probability Modeling and Functional Validation of Dynamic Service Composition for Location Based Services with Uncertain Factors
Weimin Li, Zhengbo Ye, Xiaohua Zhao, Jiulei Jiang, Qun Jin

SPECIAL ISSUE ON SELECTED PAPERS IN ICS 2012

Heterogeneous Wireless Sensor Network with EPC Network Architecture for U-Life Environment
Yao-Chung Chang

A Data Lossless Message Hiding Scheme without Extra Information
Zhi-Hui Wang, Qian Mao, Chin-Chen Chang, Qiong Wu, Jianjun Li

The Association between College Students' Internet Usage and Interpersonal Relationships
Chih-Hung Lai, Chunn-Ying Lin, Cheng-Hung Chen, Sufen Chen, Hwei-Ling Gwung

Loopless Algorithms for Listing Zaks' Sequences in Gray-Code Order
Ro-Yu Wu, Cheng-Hsien Hsu, Jou-Ming Chang

New Strategy of Efficient SPA-Resistant Exponentiations
Wu-Chuan Yang, Shyh-Yih Wang

Message Broadcast Using Multiple Trees in DHT P2P Networks
Jeng-Wei Lin

LZ

Journal of Internet Technology
Volume 15, No. 4 July 2014

Contents

Invited Paper

1. Reducing Energy Consumption in Access Networks: Comparison and Analysis 493
Shijia Zhu, Deyun Gao, YuJing Zeng, Fei Song, Hongke Zhang

Regular Paper

1. Impact of P2P-Assisted and Interaction-Oriented Content Distribution for Online Social Networks 505
Kai Shuang, Bin Du, Sen Su
2. Generic Utility Based Resource Allocation Optimization for Broadband Wireless Networks with Relays 519
Hongxian Tian, Bin Fan, Rongtao Xu, Jia You
3. The Design and Evaluation of an Affective Tutoring System 533
Kuo-Chun Hsu, Hao-Chiang Koong Lin, I-Long Lin, Jin-Wei Lin
4. An On-Demand Services Discovery Approach Based on Topic Clustering 543
Zheng Li, Keqing He, Jian Wang, Neng Zhang
5. An Efficient Multicast Strategy for On-Demand Data Dissemination in Wireless Networks 557
Shujuan Wang, Mangui Liang
6. Realization of Communications-as-a-Service: IPTV Distribution Platform and a Network-Based IPTV Audience Measurement Scheme 565
Chi-Yi Lin, Li-Chia Yeh, Ching-Sheu Wang, Jenn-Wei Lin
7. An Efficient Architecture for Parallel Skyline Computation over Large Distributed Datasets 577
He Li, Sumin Jang, Jaesoo Yoo

Special Issue on Advanced Convergence Technologies: Big Data, IoT, Cloud Computing

1. Guest Editorial: Advanced Convergence Technologies: Big Data, IoT, Cloud Computing 589
Jong Hyuk Park, Jason C. Hung, Neil Y. Yen, Young-Sik Jeong
2. Robot Service Framework Based on Big Data Technology 593
Yunsick Sung, Hwa-Young Jeong, Kyungeun Cho, Kyhyun Um
3. Organizing a User-Created Computing Environment by RESTful Web Service Open APIs in Desktop Grids 605
Joon-Min Gil, Jaehwa Chung, Young-Sik Jeong, Doo-Soon Park
4. A Comparative Study of Authentication Schemes with Security and Usability of IPAS 615
Sadiq Almuairfi, Prakash Veeraraghavan, Naveen Chilamkurti
5. Energy-Efficient Fragmentation Scheme for IEEE 802.11 DCF: Delay and Fairness Considerations 625
Prosper Mafole, Yuki Manabe, Teruaki Kitasuka, Masayoshi Aritsugi
6. Probability Modeling and Functional Validation of Dynamic Service Composition for Location Based Services with Uncertain Factors 635
Weimin Li, Zhengbo Ye, Xiaohua Zhao, Jiulei Jiang, Qun Jin

An On-Demand Services Discovery Approach Based on Topic Clustering

Zheng Li[1], Keqing He[2], Jian Wang[2], Neng Zhang[2]
[1]School of Computer and Information Engineering, Henan University, China
[2]State Key Laboratory of Software Engineering, School of Computer, Wuhan University, China
{zhengli_hope, hekeqing, jianwang, nengzhang}@whu.edu.cn

Abstract

As more and more heterogeneous services are available in the cloud, how to discover services in an efficient and accurate way becomes a key issue. Services clustering is an effective way to facilitate services discovery. On the basis of domain-oriented services classification, this paper proposes a topic-oriented services clustering approach to group the classified services in a specific domain into topic clusters. This can greatly reduce the search space and enhance the efficiency of services discovery. Then the paper proposes a three-phase services discovery approach based on topic-oriented clustering to find more relevant services by "domain-topic cluster-service" matching. Finally, experimental results on real services demonstrate the feasibility and effectiveness of the proposed approach. The results show that our proposed services discovery approach can efficiently and accurately find users' desired services.

Keywords: Services discovery, Services clustering, Topic, Probability, Similarity.

1 Introduction

As a key services delivery platform in the field of services computing [1], cloud computing [2-3] provides environments to enable resource sharing by publishing various types of resoures as reusable services [4] (please note that the services mentioned in this paper are Web services). Leveraging services available in the cloud, users can compose new value-added business processes and further publish them as reusable services. However, as the number of services rapidly increases in the cloud, how to efficiently and accurately discover user desired services remains a big challenge.

One major technique is to establish service registries [1] to help users find relevant services. However, many public accessible UDDI registries have been closed. Currently, many non-UDDI service registries have emerged, such as Seekda (http://webservices.seekda.com/), WebServiceList (http://www.webservicelist.com/), ProgrammableWeb (PWeb, http://www.programmableweb.com). According to statistics in [5], there are 825,132 services collected from such major service registries. Moreover, these services have the obvious heterogeneous characteristics for the following two reasons. On the one hand, services follow diverse protocols such as SOAP (Simple Object Access Protocol), REST (Representational State Transfer), and XML-RPC (eXtensible Markup Language-Remote Procedure Call). On the other hand, there are various service description languages such as WSDL (Web Services Description Language), OWL-S (Ontology Web Language for Services), and WADL (Web Application Description Language); furthermore, a large number of services followed the REST protocol, namely, RESTful services, are described in natural language. What's more, the increase of network users leads to users' requirements showing a trend of personalization and diversity. During the process of services discovery, it needs to consider not only users' functional requirements (e.g., book tickets), but also their non-functional requirements (requirements to quality of service, such as response time and reliability). Therefore, although the above-mentioned service registries can help users in querying their desired services, there are still two major issues on services discovery.

First, the efficiency of services discovery is not high due to lack of effective services organization approach. As shown in Figure 1, take PWeb and Seekda as examples. Since APIs (Application Programming Interface) at PWeb represent reusable service components, we use the terms API and service interchangeably throughout this paper. As shown in Figure 1(a), although PWeb organizes services according to categories that each service belong to, users still need to examine each service in a specific category to determine which one is more proper. However, this process is labor-intensive and time-consuming, especially for those categories with hundreds of services, e.g., "Social". Moreover, PWeb presets a category named "Other" (the small rectangle in Figure 1(a)) to manage all uncategorized services from PWeb, which will further hinder the category-based services discovery. While Seekda organizes services according to service providers categorized by countries, as shown in Figure 1(b). Similarly, it is also more difficult for users to find their desired services.

The second major issue is that the accuracy of services discovery is not high because the current query capabilities of these service registries are limited. Still take PWeb and Seekda as examples, as shown in Figure 2. Figure 2(a) shows the search criteria such as keywords, category, or data format at PWeb. In order to find their desired services,

*Corresponding author: Jian Wang; E-mail: jianwang@whu.edu.cn
DOI: 10.6138/JIT.2014.15.4.05

Figure 1 (a) Browse APIs by Category at PWeb (b) Browse Services Providers by countries then Browse Services by Country Specific Providers at Seekda

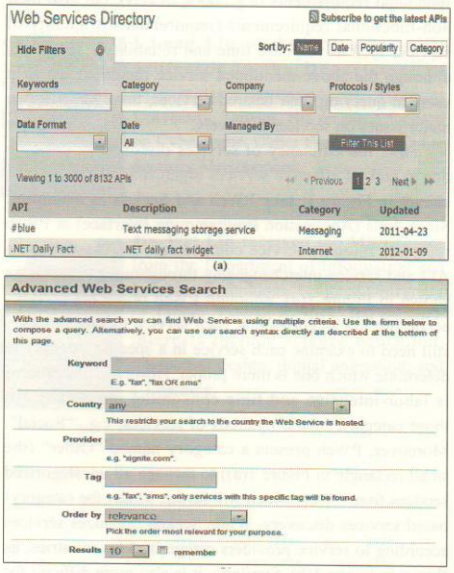

Figure 2 (a) Services Directory Search at PWeb (b) Advanced Services Search at Seekda

users can choose one or multiple search criteria to compose a query. Such search mechanism is very helpful to filter out services, but users can only find a few interested services from a long list of search results. This will lead to lower accuracy of services discovery. Similar to PWeb, users discover services by specifying one or multiple search criteria such as keyword, service provider or service tag from Seekda, as shown in Figure 2(b). Then users browse services in the result list but only find a few even no relevant services.

Therefore, as for these service registries, there exists a need for an effective organization method and the corresponding services discovery approach to enhance their query efficiency and accuracy.

Services clustering is an effective way to promote services discovery [6]. Clustering services based on functional similarity will greatly boost the ability of search engine to retrieve the most relevant services [7]. Instead of searching from all services with different functionalities in a registry, users locate their requests to a service cluster that contains functionally similar services and then choose interested services according to their preferences. This can effectively reduce the search space and greatly improve the query efficiency. There are several approaches [8-12] on services clustering. However, most of these approaches directly cluster service documents, without considering domains that each service belongs to. But using such clustering methods, services that contain similar terms or operations may be assigned to the same cluster even if they belong to different domains. Moreover, these clustering approaches are only for a single type of service documents, some for WSDL documents [8-9], some for OWL-S documents [10-11], some even for randomly generated services [12], and less emphasis on RESTful services described by the text.

On the basis of our domain-oriented services classification approach [13], we propose a topic-oriented services clustering approach to further organize domain-specific services into topic clusters. Then we propose a services discovery approach based on topic-oriented clustering. Experimental results show that our services discovery approach has higher efficiency and accuracy. The main contributions of this paper are summarized as follows:

(1) We propose a topic-oriented services clustering approach. On the basis of domain-oriented services classification, classified services in a specific domain are further grouped into functionally similar topic clusters. This is beneficial for the organization and management of services, consequently improving the efficiency of services discovery.

(2) We propose a three-phase services discovery approach based on "domain-topic cluster-service" matching. User requirements are gradually located from a specific domain to a specific topic cluster in the matched domain. Then the relevant services in the located topic cluster are discovered and returned. The benefit of such

discovery approach is to help users find their desired services and to improve the accuracy of services discovery.

(3) We conduct a series of experiments on real services to evaluate effectiveness of the proposed services clustering approach and discovery approach.

The rest of the paper is organized as follows. In Section 2, related work is briefly reviewed. In Sections 3 and 4, our topic-oriented services clustering approach and three-phase services discovery approach are described in detail, respectively. The experimental results and analysis are given in Section 5. Finally, conclusions are presented in Section 6.

2 Related Work

Different approaches are used to organize services, such as service community [14-15], service pool [16], service cluster [17]. In contrast to their work, we organize services by the domain and topic cluster that each service belongs to.

Services clustering is a technique for effectively facilitating services discovery [6]. Currently, using clustering [18] to promote services discovery has received considerable attention. Elgazzar et al. [7] propose an approach to improve services discovery by grouping services into functionality-based clusters. Dong et al. [19] propose the algorithms underlying a service search engine, Woogle, to support similarity search for services. Chen et al. [20] use both WSDL documents and service tags to improve the performance of clustering for more accurately discovering services. However, these researches use a weighting mechanism to combine multiple parts, such as service description, WSDL types, messages, for computing similarities between services. This will affect the accuracy of clustering, thereby affecting the accuracy of services discovery. In contrast, our topic-oriented services clustering method based on probability does not need to use such weighting mechanism.

Yu et al. [8] group services with similar functionalities into homogeneous communities by co-clustering of service operations and services, which greatly facilitates services discovery. Ma et al. [21] propose a clustering semantic algorithm for efficiently finding services. Wang et al. [22] adopt K-means [23] algorithm for clustering both requirements and candidate services, effectively reducing the search space. Dasgupta et al. [6] propose a self-organizing based clustering algorithm for efficient services discovery. Liu et al. [24] propose a two-layered P2P model for improving the efficiency of services discovery. However, an important limitation of these studies on services clustering is that researchers only use WSDL documents [8][21-22] or only use OWL-S documents [6] [24]. Furthermore, these studies directly cluster service documents, but do not consider the domain that each service belongs to. Therefore, services from different domains may be assigned to the same cluster, which will affect the accuracy of services discovery. In our research, service documents with different types are unified modeled as the corresponding vectors, so our services clustering approach based on domain-oriented services classification can cluster service documents such as textual description documents (RESTful services) and WSDL documents.

There have been many researches on semantic Web services discovery. OWLS-MX [25] and WSMO-MX [26] propose a hybrid semantic service matchmaker by combining logic-based reasoning and syntactic-based similarity computation. Sbodio et al. [27] propose SPARQL to describe pre-conditions, post-conditions of services, goals of agents, and then use SPARQL to discover semantic services. Kuang et al. [28] propose a composition-oriented discovery algorithm based on inverted indexing to facilitate semantic services discovery. Deng et al. [29] propose a method for semantic services discovery based on bipartite graph matching. In our research, we use the domain knowledge obtained by domain-oriented services classification (domain feature terms) and topic-oriented services clustering (topic feature terms), for incrementally building and enriching domain ontology and leverage it to facilitate semantic services discovery.

3 Topic-Oriented Services Clustering

In order to help users efficiently discover their desired services, it is essential to find an effective way to organize services. In this Section, we present our services organization approach based on service functionality. Firstly, we review our domain-oriented services classification approach based on ontology-empowered SVM (Support Vector Machine). Secondly, on the basis of services classification, we propose a topic-oriented services clustering approach based on probability. Classifying and then clustering services can greatly reduce the search space, thereby enhancing the efficiency of services discovery.

3.1 Domain-oriented Services Classification Based on Ontology-Empowered SVM

We have implemented an SVM engine using the LIBSVM (http://www.csie.ntu.edu.tw/~cjlin/libsvm/) for domain-oriented services classification [13]. The basic procedure is as follows. Firstly, we combine crawler and open Web API to get service documents such as textual description documents, WSDL documents from service registries such as PWeb, Seekda. For the collected service

documents with different types, we mainly extract the parts that can reflect their core functionalities. For example, for the three main service description documents, the exacted contents are as follows.

- Textual description documents (RESTful services): we mainly extract service name, service description, service tags, and so on.
- WSDL documents (SOAP services): we mainly extract service name, WSDL types, messages, ports, and so on.
- OWL-S documents (OWL-S services): we mainly extract service name, input, output, precondition and effects.

Secondly, we preprocess the extracted service documents including word segmentation, verbs and nouns extraction, stopword removal, stemming, and term frequency statistics. Thirdly, we identify domain-relevant and domain-irrelevant service sets from the preprocessed service documents. Fourthly, we randomly select the needed training set and testing set from the above specified service sets, and use keyword frequency-inverse document frequency -domain frequency (KF-IDF-DF, extension to TF-IDF [30]) to construct the vector space. Finally, we leverage SVM to classify services into either domain-relevant or domain-irrelevant.

Note that our services classification method adopts an iterative process to incrementally get the better services classification accuracy and to incrementally build and enrich domain ontology. We use keyword frequency-inverse repository frequency (KF-IRF) to calculate the ranking of terms in the relevant domain except the initial term ranking obtained by counting term frequency. Therefore, the termination criterion of iteration can be set when the ranking of top-ranked (e.g., top 60) terms remains unchanged in a new iteration. When the iteration is terminated, we can obtain the classified services and domain term ranklist (DTR, used to build domain ontology) ranked by KF-IRF in a specific domain. More related details can be found in [13].

3.2 Topic-oriented Services Clustering Based on Probability

LDA (Latent Dirichlet Allocation) is a generative probabilistic model to discover topics from a collection of documents. Its basic idea is that documents are represented as mixtures over latent topics, where each topic is characterized by a distribution over words [31]. For each document in the collection, the generative process in LDA is: each word w in a document d is generated by sampling a topic z from topic distribution, and then sampling a word from topic-word distribution. According to this process, the probability that each document contains different topics can be determinded. This is the distinguishing characteristic of LDA, namely, all documents in the collection contain the same set of topics, but each document exhibits these topics with different probability.

In this paper, we use LDA to find the corresponding latent topics for each domain-specific service document according to the corresponding probability distributions over the extracted terms. Then, according to the identified *topics mixed in each service document and the probabilities* that each service document contains the corresponding topics, we group the classified services in a specific domain into different topic clusters.

The general steps of topic-oriented services clustering approach are as follows.

Step 1: for the top k terms in the DTR generated by domain-oriented services classification, we calculate the domain representation degree (*DRD*) of each term using Equation (1). Then we remove those terms whose *DRD* are greater than a given threshold (e.g., 0.9), to build the domain feature term set (DFTS).

$$DRD(t_i, d) = \frac{\left| \left\{ S_j \middle| t_i \in S_j \wedge S_j \in d \right\} \right|}{|d|} \quad (1)$$

Where S_j denotes the service that contains the term t_i in domain d, $|\{S_j \mid t_i \in S_j \wedge S_j \in d\}|$ denotes the number of this kind of services, and |d| denotes the total number of services in domain d. Equation (1) shows that if $DRD(t_i, d)$ is greater, then the term ti appears in more services of domain d. This indicates that the term t_i can highly represent domain d. Such terms are very important for domain-oriented services classification in Section 3.1. However, these terms have little significance to topic-oriented services clustering because such terms nearly appear in each service of the domain. Therefore, we remove the terms which can highly represent a given domain when clustering.

Step 2: we remove those terms in each service document but not in the DFTS, to reduce dimensionality of all classified service documents in a given domain.

Step 3: we leverage JGibbLDA (http://jgibblda.sourceforge.net/), a Java implementation of LDA by using Gibbs sampling (http://en.wikipedia.org/wiki /Gibbs_sampling) for parameter estimation and inference, to get the probability distributions of each service document and its containing topics as well as the probability distributions of each topic and its containing terms.

Step 4: according to the obtained probability distributions, we cluster a domain-specific service document into the corresponding topic cluster if the probability that the service document contains the corresponding topic is greatest (we consider that

if the probability that a service document contains a topic is greater, the likelihood that the service document belongs to the corresponding topic cluster is also greater).

For example, assume that service document S_i from "Shopping" domain contains 5 topics, and the probability that it contains topic 1 is greatest, then S_i is assigned to topic cluster 1.

Step 5: repeat step 4 until clustering all classified service documents from a given domain into T topic clusters.

According to the clustering process described above, we can get a set of topic clusters in a specific domain, probability distributions of each service document and its containing topics, and probability distributions of each topic and its containing terms. Moreover, we can select those terms with higher probability from each topic in a specific domain to enrich the corresponding domain ontology, thereby providing better support for semantic services discovery.

4 Services Discovery Based on Topic Clustering

On the basis of domain-oriented services classification and topic-oriented services clustering, we propose a three-phase approach to discover services by "domain-topic cluster-service" matching. As shown in Figure 3, different from traditional query approach which finds relevant services from the whole service registry, in phases 1 and 2, we intend to identify the matched domain and the matched topic cluster in the matched domain by the corresponding functional similarity computation; In phase 3, from the functional and non-functional aspects, we compute the similarity between user requirements and candidate services in the matched topic cluster to discover user desired services. In other words, phase 3 intends to quantify the relevance of candidate services to user requirements. In the following Sections, the proposed three-phase services discovery approach based on "domain-topic cluster-service" matching will be discussed in detail.

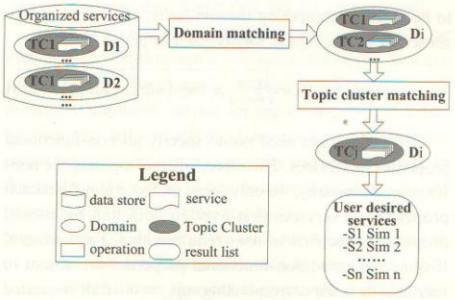

Figure 3 The Three-Phase Services Discovery Approach

4.1 Matching between User Requirements and Candidate Domains

We use cosine similarity to compute the similarity between user requirements and candidate domains, and then locate user requirements to the domain with the maximum similarity. The general steps are as follows.

Step 1: parse user requirements. As mentioned in Section 3.1, we use the similar preprocessing technology to parse user requirements description. Furthermore, the parsed functional and non-functional requirements are stored separately in the form of text files to facilitate similarity computation.

Step 2: construct the weighted term vectors for a candidate domain and the parsed user functional requirements.

As described in Section 3.1, after domain-oriented services classification, we can get the DTR ranked by KF-IRF which indicates the relevance between a term and the domain. Here, we choose the top-ranked (e.g., top 10) terms in the DTR which highly represent a domain as domain keyword set (DKS). Note that terms in the DKS are corresponding to concepts in the domain ontology.

When constructing the weighted term vectors, we consider both term frequency and the corresponding values of KF-IRF. More specifically, let f_{t_i} be the frequency of a term t_i in the parsed functional requirements in step 1, and $kf-irf_{t_i}$ denotes the KF-IRF value of the term t_i in the DKS, then the weight w_{t_i} of the term t_i in the requirements term vector ($WT(RTV)$) can be defined as

$$w_{t_i} = \begin{cases} f_{t_i} \times kf - irf_{t_i}, t_i \in DKS \\ 0, \quad otherwise \end{cases} \tag{2}$$

So the weighted term vector for user functional requirements is defined as

$$WT(RTV) = \left\{ \left(t_i, w_{t_i} \right) : t_i \in DKS \right\} \tag{3}$$

Similarly, the weight w_{k_j} of keyword k_j in the domain keyword vector $WT(DKV)$ can be defined as

$$w_{k_j} = \begin{cases} w_{t_i} = f_{t_i} \times kf - irf_{t_i}, k_j = t_i \\ kf - irf_{k_j}, \quad otherwise \end{cases} \tag{4}$$

Then the weighted keyword vector for the domain is defined as

$$WT(DKV) = \left\{ \left(k_j, w_{k_j} \right) : k_j \in DKS \right\} \quad (5)$$

Step 3: compute the cosine similarity. According to the weighted term vectors for user functional requirements and the domain obtained by step 2, the cosine similarity between them can be calculated by Equation (6).

$$Sim(RTV, DKV) = \frac{WT(RTV) \cdot WT(DKV)}{|WT(RTV)||WT(DKV)|} = \sum_{i=j=1}^{n} \frac{w_{t_i} \cdot w_{k_j}}{\sqrt{\sum_{i=1}^{n} (w_{t_i})^2} \cdot \sqrt{\sum_{j=1}^{n} (w_{k_j})^2}} \quad (6)$$

Step 4: repeat steps 2 and 3 to get similarities between user functional requirements and each candidate domain.

Step 5: discover the matched domain. According to the similarities obtained in step 4, user requirements are matched to the domain with the maximum similarity.

4.2 Matching between User Requirements and Candidate Topic Clusters

As mentioned in Section 3.2, when the clustering process finishes, we can get the probability distributions of each topic and its containing terms, and terms in each topic are ranked by the probability that each term appears in the corresponding topic. Therefore, we choose the top-ranked (e.g., top 10) terms with higher probability to form the topic feature term set (TFTS). Here, terms from the TFTS also correspond to concepts in the domain ontology.

When constructing the weighted term vectors for a candidate topic cluster in the matched domain and user functional requirements, we consider both term frequency and the corresponding probability that the term appears in a specific topic. Then similar to Section 4.1, we also use cosine similarity to compute similarities between user functional requirements and each candidate topic cluster in the matched domain. Finally, user requirements are matched to the topic cluster with the maximum similarity.

4.3 Matching between User Requirements and Candidate Services

In this Section, from the functional and non-functional perspectives (Quality of Service [32-33]), we compute similarities between user requirements and services in the matched topic cluster, then rank services according to the obtained similarities, and return services selected by a predefined similarity threshold (e.g., 0.4). The general steps are as follows.

Step 1: Non-functional similarity computation.

When computing non-functional similarities between user requirements and services, we use the modeling and normalization methods of non-functional properties in [34]. Firstly, all non-functional properties of service s are modeled as a vector $NFP_s = \{nfp_{s,1}, nfp_{s,2}, ..., nfp_{s,m}\}$. Similarly, the parsed non-functional requirements r in Section 4.1 are also modeled as a vector $NFP_r = \{(nfp_{r,1}, w_1), (nfp_{r,2}, w_2), ..., (nfp_{r,n}, w_n),\}$, and $\sum_{i=1}^{n} w_i = 1$, $w_i \in [0,1]$. Where w_i, optional, are user-specified weights for reflecting users' preference to different non-functional properties. For example, suppose that user rating is one of non-functional properties, and one user may consider that the service is better if its user rating is higher. Thus, the user may set the weight of user rating greater than the weights of other non-functional properties. By default, if the number of non-functional properties in user requirements is n, then the weight of each non-functional property is $1/n$. Then, values of different non-functional properties with different measurement units need to be normalized. As described in [34], some non-functional properties (e.g., response time) need to be normalized by minimization, while other non-functional properties such as availability, reliability, need to be normalized by maximization.

Next, we use Equation (7) to calculate non-functional similarities between user requirements R and services S. Where μ_i is a coefficient whose value is 1 or -1. For a specific non-functional property in user requirements, if its normalized value is smaller than the normalized value of non-functional property of services, then μ_i is set to be 1 (we consider that a service with greater normalized value of non-functional property can satisfy the user better). Otherwise, μ_i is set to be -1. For this case, if the normalized value of non-functional property of a service is more close to the normalized value of non-functional property in user requirements, then the service can satisfy the user better.

For example, a user specifies the values of user rating and availability as 4 and 0.9, respectively. For two services $S1$ (5, 0.85), $S2$ (4.5, 0.75), the coefficient μ_1 of user rating is set to be 1 while the coefficient μ_2 of availability is set to be -1 when computing the similarity. Moreover, results show that service $S1$ can satisfy the user better.

$$Sim_{nf}(R,S) = \sqrt{\sum_{i=1}^{n} \mu_i \cdot w_i \cdot \left(nfp_{r,i} - nfp_{s,i} \right)^2} \quad (7)$$

Note that users need not to specify all non-functional properties of services. Therefore, when computing the non-functional similarity, we only focus on those non-functional properties of services that overlap with non-functional properties specified in user requirements. Furthermore, if some requested non-functional properties are absent in services, then the corresponding $nfp_{s,i} = 0$. If all requested non-functional properties are absent in services, then

directly set the non-functional similarity $Sim_{nf}(R,S) = 0$. If the radicand in Equation (7) is negative, then we take the negative square root of absolute value of the radicand as the non-functional similarity. For example, if the radicand is -0.25, then the non-functional similarity is the negative square root (-0.5) of 0.25.

Finally, for all services in the matched topic cluster, we filter out those services with negative non-functional similarities, to obtain the initial service set. Note that if all non-functional similarities are negative, we select the services that exceed a predefined similarity threshold (e.g., -0.5) as the initial service set.

Step 2: Functional similarity computation. According to the initial service set obtained by step 1, we compute the functional similarity between user requirements and each service in the initial service set to reduce the computational load. Similar to Sections 4.1 and 4.2, we still use cosine similarity to compute the functional similarity between two services. The difference is that when constructing the weighted term vectors, terms in each service should overlap with terms in the TFTS of the matched topic cluster.

Step 3: Similarity integration. According to the functional similarity and non-functional similarity obtained from steps 1 and 2, the overall similarity $Sim(R,S)$ between user requirements R and service S is computed using Equation (8).

$$Sim(R,S) = (1-\lambda)Sim_f(R,S) + \lambda Sim_{nf}(R,S) \quad (8)$$

Where $\lambda \in [0,1]$, is the weight of non-functional similarity. More specifically, when $\lambda = 0$, $Sim(R,S)$ equals functional similarity, and when $\lambda = 1$, $Sim(R,S)$ equals non-functional similarity. When $\lambda \in (0,1)$, $Sim(R,S)$ is calculated as the weighted sum of functional and non-functional similarity. For this case, if the non-functional similarity from step 1 is negative, which further leads to the negative overall similarity, then $Sim(R,S)$ is directly set to be zero.

In general, the weight of functional similarity is set greater than non-functional similarity. However, in some special context, for example, assume that one user wants to request "payment" service for online shopping, and then the user may have strong requirements for non-functional properties of services such as security and response time. In this context, the weight of non-functional similarity should be set greater than functional similarity.

Step 4: services delivery. We rank services according to overall similarities obtained by step 3, and then return those services that exceed a predefined similarity threshold (e.g., 0.4).

5 Experimental Results and Analysis

We conduct a series of experiments to evaluate our proposed approach. All experiments are implemented in Java, and conducted on PCs with Intel Core(TM) i5 CPU 760, @ 2.80 GHz and 4 GB main memory, running the Windows 7 operating system.

5.1 Experimental Data Set

Our data set for experiments is from PWeb, a popular service and mashup registry. PWeb provides programmable APIs for users to get descriptive data about the registered services, such as API name, tags, description. We combine PWeb open API and crawler to collect such descriptive data of all registered services from PWeb. For example, as shown in Figure 4, we mainly extract the profile information shown in the rectangles including functional aspects of "PayPal Invoicing" API (http://www.programmableweb.com/api/paypal-invoicing) such as API name, description, and non-functional aspects, e.g., user rating. Moreover, the extracted two kinds of data are stored separately in the form of text files to facilitate similarity computation. For services followed SOAP protocol, we also collect WSDL documents corresponding to such services. We totally collect 7,190 service documents from 63 domains at PWeb (data gathered on Sep. 7, 2012). Then these service documents are preprocessed by using the preprocessing in Section 3.1, to prepare for the subsequent operations.

5.2 Analysis of Clustering Accuracy and Time

Before performing topic-oriented services clustering, we leverage ontology-empowered SVM to classify the preprocessed service documents. Currently, we mainly focus on 24 domains with more than 100 APIs except "Other" domain due to the following two reasons. On the one hand, most domains at PWeb contain different

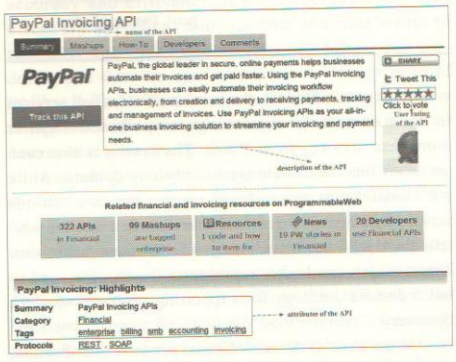

Figure 4 A Snapshot of "PayPal Invoicing" API Profile from Pweb

number of APIs, such as "Internet" domain containing 428 services, "Dating" domain only containing 2 services. The unbalanced distribution of APIs in different domains will lead to difficult to unified specify the scale of training set for all domains when using SVM to classify services. On the other hand, if the scale of training set is too small, then it is difficult to obtain higher classification accuracy by using training model generated from the small scale training set. For the chosen 24 domains, we perform domain-oriented services classification according to the procedure described in Section 3.1. Table 1 shows the classification results from randomly selected 5 domains.

Table 1 Classification Results from 5 Domains

Domain	Number of classified services		Domain term ranklist (Top 10, ranked by KF-IRF)
	PWeb	Ontology-empowered SVM	
Financial	217	255	finance, stock, trade, invoice, currency, tax, exchange, money, quote, payment
Internet	412	532	internet, url, cloud, ip, host, monitor, link, utility, website, server
Mapping	229	309	map, location, address, place, direction, gp, latitude, longitude, gi, route
Shopping	210	243	shopping, deal, product, coupon, affiliate, merchant, payment, cart, order, price
Social	342	472	social, twitter, network, friend, share, medium, post, tweet, photo, platform

As shown in Table 1, the number of classified services in each domain from ontology-empowered SVM is more than services from PWeb. The reason is that each service from PWeb belongs to only one domain, while our classification approach can support the case that one service belongs to two or multiple domains. Table 1 also lists the top 10 terms ranked by KF-IRF from 5 domains. Terms which can highly represent a domain are used to build domain ontology for supporting semantic services discovery.

On the basis of domain-oriented services classification, we perform topic-oriented services clustering according to the steps in Section 3.2. For the DTR generated from services classification, we choose the top 200 terms in the DTR and set the threshold of *DRD* to be 90%, and then get the DFTS to reduce dimensionality of classified services in a given domain. Next, we leverage JGibbLDA to obtain the corresponding probability distributions, and cluster each service into the corresponding topic cluster according to the obtained probability distributions.

We design a set of experiments to compare the clustering accuracy and clustering time among our topic-oriented clustering approach (TOCA), directly applying LDA to services clustering (D-LDA) and K-means.

We adopt the purity to evaluate the clustering results. In general, the higher the purity is, the better the clustering result is. Purity is the number of services in a specific cluster that overlap with services in the standard cluster (obtained by manually clustering) divided by the number of all services in a specific cluster. The purity of one cluster [35] can be defined as

$$PC(TC_j) = \frac{1}{|TC_j|} \max_i (n_j^i) \tag{9}$$

Where TC_j represents the set of services that are clustered into the jth topic cluster; $|TC_j|$ denotes the number of services in topic cluster TC_j; n_j^i indicates the number of services in the ith standard cluster are assigned to the jth topic cluster. On this basis, the overall purity of services clustering in a specific domain is defined as

$$PC(TC) = \sum_{j=1}^{n} \frac{|TC_j|}{|DS|} PC(TC_j) \tag{10}$$

Where $|DS|$ denotes the total number of all classified services in a specific domain.

However, identification of the standard clusters in a specific domain is a non-trivial task. Here, we compute the similarity between each service and all topics in a specific domain using the similar approach in Section 4.2, and each service is divided into the corresponding topic cluster with the maximum similarity. Then we manually determine the final standard clusters.

Next, we evaluate the purity and clustering time of using K-means, D-LDA and our approach TOCA for the selected 5 domains, the results are shown in Figure 5 and Figure 6 respectively.

As shown in Figure 5, for any of the selected 5 domains, our TOCA outperforms both K-means and D-LDA in purity. More specifically, purity of each domain obtained by our TOCA exceeds 70%, while the maximum purity obtained by D-LDA reaches 64.4% ("Internet" domain), and the maximum purity by K-means only reaches 41.6%

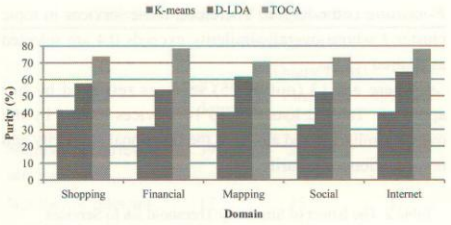

Figure 5 Comparisons of Purity

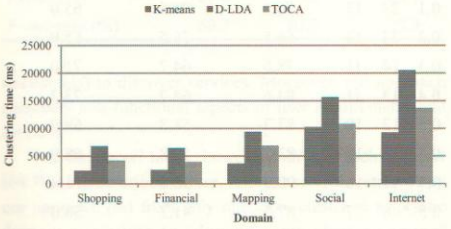

Figure 6 Comparisons of Clustering Time

("Shopping" domain). Such results indicate that our TOCA can effectively improve the clustering results.

The reasons for such results are analyzed as follows. Compared to TOCA, the purity obtained by D-LDA is lower due to without adopting the dimensionality reduction strategy, while the purity of K-means is worst may be due to the following two reasons: (1) we just use term frequency to represent service documents as vectors when computing the similarity; (2) the number of services in each cluster generated from K-means is greatly different. More specifically, there are one or two clusters containing the vast majority of services in each domain, while the purity of such clusters is lower which leads to lower overall purity according to the definition in Equation (10).

Figure 6 shows that the clustering time becomes longer with the increase in the number of services. However, for the same number of services, e.g., services in a specific domain, the clustering time of three approaches are greatly different. More specifically, the clustering time using K-means is the least, while D-LDA spends the longest time. For example, the time for clustering the classified services in "Shopping" domain by using three approaches are respectively 2,365 ms (K-means), 6,789 ms (D-LDA), and 4,199 ms (TOCA). For other 4 domains, we can get similar results.

The reasons for such results are: the clustering time of K-means is the least because it needs less iteration due to without Gibbs sampling; while TOCA takes less time than D-LDA because it uses the DFTS to reduce dimensionality of service documents, consequently accelerating the process of Gibbs sampling.

According to the above analysis, we can conclude that the clustering time of K-means is the least, but its purity is the lowest. The purity of D-LDA is higher than K-means but its clustering time is longest. The clustering time of our TOCA is slightly longer than K-means, but this overhead is acceptable because services clustering only needs to be performed periodically before services discovery. More importantly, the purity of TOCA is the highest.

5.3 Accuracy and Efficiency Analysis of Services Discovery

In this Section, we evaluate the accuracy and efficiency of our discovery approach based on "domain-topic cluster-service" matching through a scenario. We adopt three indexes, *Precision*, *Recall*, and *F-measure*, to evaluate the accuracy of services discovery, and use query time to evaluate the efficiency of services discovery.

Precision is the number of retrieved services that are relevant to user requirements divided by the number of all retrieved services; *Recall* is the number of relevant services divided by the number of services that should have been retrieved; *F-measure* is a weighted average of *Precision* and *Recall*. *Precision*, *Recall*, and *F-measure* are defined in Equation (11).

$$Precision = \frac{\left|\{relevant\ services\} \cap \{retrieved\ services\}\right|}{\left|\{retrieved\ services\}\right|},$$

$$Recall = \frac{\left|\{relevant\ services\} \cap \{retrieved\ services\}\right|}{\left|\{relevant\ services\}\right|}, \quad (11)$$

$$F-measure = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

The following scenario is provided to evaluate the accuracy and efficiency of our three-phase services discovery approach. Suppose that one user wants to accurately query the exchange rates of RMB against U.S. dollar, and then the user may query "currency rates of RMB to dollar" as functional requirements, and expect user ratings of candidate services is greater than "4.0" (non-functional requirements).

According to "domain-topic cluster-service" matching approach, in phase 1, we find the matched domain from 24 candidate domains. As described in Section 4.1, user's requirements description is parsed into two parts: functional requirements and non-functional requirements. Then we choose the top 10 terms in the DTR to get the DKS and compute the similarity between user's functional requirements and each candidate domain. According to the calculation results, user's requirements are located to

"Financial" domain with the similarity of 0.524.

In phase 2, we find the matched topic cluster in "Financial" domain. We choose the top 10 terms in each topic from "Financial" domain to get the TFTS, and then compute the similarity between user's functional requirements and each topic cluster. Then user's query is matched to topic cluster 1 in "Financial" domain with the similarity of 0.646.

In phase 3, we find user desired services by computing both functional and non-functional similarities between user's query and services in topic cluster 1 using the steps in Section 4.3.

There are totally 34 services in topic cluster 1 in "Financial" domain, but we need not to compute similarities between user's query and all of 34 services. As mentioned in step 1 in Section 4.3, to obtain the initial service set, we can filter out those services with negative non-functional similarities. There are 9 services to be filtered out, which reduce the computational overhead to 73.5%. Then, we set the weight of non-functional similarity to be 0.2 to compute the overall similarity between user's query and each of 25 services in the initial service set.

According to keyword-based query approach, a total of 39 services are obtained from all domains and 32 services are retrieved from "Financial" domain at PWeb by finding "currency rates." In order to evaluate the accuracy of query results, twenty graduate students and twenty undergraduate students are asked to identify how many services retrieved by the keyword-based method are closely related to user's requirements. The judgment of individual student is very personal and subjective, thus the number of relevant services retrieved by each method is the average of results from forty students.

Next, we analyze the effect of the overall similarity threshold *sth* to services discovery results. The threshold *sth* is set to be 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, and 0.9, respectively. Then we consider the 17 relevant services retrieved from all domains at PWeb as the relevant services set, to calculate the values of *Precision*, *Recall*, and *F-measure* by the corresponding definition in Equation (11). The results are shown in Table 2. *#rt* denotes the number of retrieved services that exceed the specified similarity threshold, *#rv* denotes the number of retrieved services that are relevant to user requirements (obtained by the average of judgment results from forty students).

As shown in Table 2, *Precision* becomes higher as the threshold *sth* increases, while *Recall* becomes lower. *F-measure* fluctuates as the threshold *sth* becomes larger. Since *F-measure* comprehensively considers *Precision* and *Recall*, we determine the value of the threshold *sth* according to *F-measure* in this scenario. Table 2 shows that when the threshold *sth* is set to be 0.4, the value of *F-measure* is the largest. Therefore, those services in topic cluster 1 whose overall similarity exceeds 0.4 are selected as desired results.

There are 13 (out of 25) services retrieved by our approach. Table 3 lists the top 10 services ranked by the overall similarity, and also lists the functional similarity and non-functional similarity.

Table 2 The Effect of Similarity Threshold *sth* to Services Discovery Results

sth	*#rt*	*#rv*	*Precision* (%)	*Recall* (%)	*F-measure* (%)
0.1	23	13	56.5	76.5	65.0
0.2	23	13	56.5	76.5	65.0
0.3	14	11	78.6	64.7	71.0
0.4	**13**	**11**	**84.6**	**64.7**	**73.3**
0.5	12	10	83.3	58.8	68.9
0.6	12	10	83.3	58.8	68.9
0.7	11	10	90.9	58.8	71.4
0.8	8	7	87.5	41.2	56.0
0.9	4	4	100.0	23.5	38.1

Table 3 Top 10 Services Retrieved from Topic Cluster 1 in "Financial" Domain

API name	$Sim_f(R, S)$	$Sim_{nf}(R, S)$	$Sim(R, S)$ ($\lambda = 0.2$)
XigniteCurrencies	0.958	1.000	0.966
Mondor Currency Exchange XML	0.947	1.000	0.958
Xurrency	0.898	1.000	0.918
Open Source Currency	0.940	0.800	0.912
XigniteMoneyMarkets	0.847	1.000	0.878
Foxrate	0.954	0.500	0.863
The Currency Cloud	0.941	0.500	0.853
Xavier Finance Exchange Rates	0.758	1.000	0.807
Exchange Rate	0.937	0.000	0.749
VirWoX	0.921	0.000	0.737

Table 4 summarizes the results of two methods. As shown in Table 4, our approach outperforms keyword-based query approach in both *Precision* and *F-measure* indexes. For the *Precision* index, our approach reaches 84.6%, while the keyword-based query approach is very low (43.6% and 46.9% respectively). The reasons are analyzed as follows. The *Precision* of keyword-based approach is lower due to the ambiguity of keywords. While our approach adopts "domain-topic cluster-service" matching, and use domain ontology (domain knowledge extracted from service

Table 4 Comparisons of Query Results

	Keyword-based		Our approach
	Query from all domains	Query from Financial domain	
Number of retrieved services	39	32	13
Number of relevant services	17	15	11
Precision (%)	43.6	46.9	84.6
Recall (%)	100.0	88.2	64.7
F-measure (%)	60.7	61.2	73.3

documents) to discover services. Moreover, our approach considers non-functional aspects of user requirements and services.

For the *Recall* index, however, our approach is lower (64.7%). As described before, relevant services retrieved by our approach just from only one topic cluster in a specific domain according to "domain-topic cluster-service" matching. That is why the *Recall* of our approach is lower. Although our approach has lower *Recall*, it can retrieve relevant services from other domain. For this scenario, the service named "The Currency Cloud" (row 8 in Table 3) from "Payment" domain can be discovered. Moreover, in the near future, we can get more relevant services by recommending those highly related services from other topic clusters in a specific domain using the corresponding services recommendation approach, to improve the *Recall* index.

We further conduct experiments to evaluate the efficiency of our approach. Table 5 lists the query time of two approaches. The results indicate that the query time of our approach is apparently less than keyword-based approach. The reason is that we do not search user desired services from all domains or "Financial" domain, but gradually locate user requirements from the matched domain to the matched topic cluster, and filter out those services with negative non-functional similarity. All these operations greatly reduce the query time, thereby improving the efficiency of services discovery.

Table 5 Comparisons of Query Time

Approaches		Query time (milliseconds)
Keyword-based	Query from all domains	9,730
	Query from Financial domain	506
Our approach		242

The above analysis shows that our approach has higher services discovery accuracy and greatly improves the efficiency of services discovery. The proposed approach is particularly valuable in building search engine for small scale service registries, and it is also benefical for services composition and recommendation. Furthermore, the proposed approach has been applied to the modules of services organization and services discovery in CloudCRM (http://202.114.107.230:8080/CloudCrm/login.jsp). CloudCRM is a platform for SaaS service management and customization. A large number of Web services and Web APIs are registered in this platform. The approach proposed in this paper can support on-demand service customization for users from small and medium enterprises.

6 Conclusion

It is a challenging task to efficiently and accurately discover services that satisfy user's personalized requirements. Services clustering can effectively promote services discovery. In this paper, on the basis of domain-oriented services classification, we propose a topic-oriented services clustering approach. Then we propose a three-phase services discovery approach based on "domain-topic cluster-service" matching. Two major benefits are that: (1) clustering services into functional similar topic clusters can greatly reduce the search space; (2) services discovery based on "domain-topic cluster-service" matching can improve the accuracy of services discovery. To sum up, the proposed services discovery approach based on topic-oriented clustering can efficiently find user desired services and has higher accuracy of discovery.

In the future, we plan to continue our research from the following aspects. Firstly, we will obtain more non-functional properties of services such as response time and reliability. Secondly, according to the invocation/composition relationships (e.g., API invocation relationships among mashups) and usage patterns among services, we will establish the relationships among topic clusters, to provide support for services composition and recommendation. Thirdly, we will apply our approach to enhance existing service registry-oriented search engine.

Acknowledgements

The work described in this paper is partially supported by the National Basic Research Program of China under Grant No. 2014CB340404, the National Science & Technology Pillar Program of China under grant No. 2012BAH07B01, the National Natural Science Foundation of China under Grant No. 61202031 and 61373037, Education Department Program of Henan Province No.

14A520008, and Research Program of Henan University No. 2013YBZR015.

References

[1] Liang-Jie Zhang, Jia Zhang and Hong Cai, *Services Computing*, Springer and Tsinghua University Press, Beijing, China, 2007.

[2] Wen-Lung Shiau, Patrick Y. K. Chau and Han-Chieh Chao, *Guest Editors' Introduction to Special Issue on Cloud Computing*, *Journal of Internet Technology*, Vol.15 No.1, 2014, pp.i-ii.

[3] Qi Qi, Jianxin Liao and Yufei Cao, *Cloud Service-Aware Location Update in Mobile Cloud Computing*, *IET Communication*, Vol.8, No.8, 2014, pp.1417-1424.

[4] Liang-Jie Zhang and Qun Zhou, *CCOA: Cloud Computing Open Architecture*, *Proc. IEEE International Conference on Web Services*, Los Angeles, CA, July, 2009, pp.607-616.

[5] Wei Jiang, Dongwon Lee and Song-Lin Hu, *Large-Scale Longitudinal Analysis of SOAP-Based and RESTful Web Services*, *Proc. IEEE International Conference on Web Services*, Honolulu, HI, June, 2012, pp.218-225.

[6] Sourish Dasgupta, Satish Bhat and Yugyung Lee, *Taxonomic Clustering and Query Matching for Efficient Service Discovery*, *Proc. IEEE International Conference on Web Services*, Washington, DC, July, 2011, pp.363-370.

[7] Khalid Elgazzar, Ahmed E. Hassan and Patrick Martin, *Clustering WSDL Documents to Bootstrap the Discovery of Web Services*, *Proc. IEEE International Conference on Web Services*, Miami, FL, July, 2010, pp.147-154.

[8] Qi Yu and Manjeet Rege, *On Service Community Learning: A Co-clustering Approach*, *Proc. IEEE International Conference on Web Services,* Miami, FL, July, 2010, pp.283-290.

[9] Christian Platzer, Florian Rosenberg and Schahram Dustdar, *Web Service Clustering Using Multidimensional Angles as Proximity Measures*, *ACM Transactions on Internet Technology*, Vol.9, No.3, 2009, pp.1-26.

[10] Jian-Xiao Liu, Ke-Qing He and Da Ning, *Web Service Aggregation Using Semantic Interoperability Oriented Method*, *Journal of Information Science and Engineering*, Vol.28, No.3, 2012, pp.437-452.

[11] Gilbert Cassar, Payam Barnaghi and Klaus Moessner, *Probabilistic Methods for Service Clustering*, *Proc. the 4th International Workshop on Semantic Web Service Matchmaking and Resource Retrieval, Organised in Conjunction with the International Semantic Web Conference*, Shanghai, China, November, 2010, pp.1-17.

[12] Ping Sun and Chang-Jun Jiang, *Using Service Clustering to Facilitate Process-Oriented Semantic Web Service Discovery*, *Chinese Journal of Computers*, Vol.31, No.8, 2008, pp.1340-1353.

[13] Jia Zhang, Jian Wang, Patrick C. K. Hung, Zheng Li, Neng Zhang and Ke-Qing He, *Leveraging Incrementally Enriched Domain Knowledge to Enhance Service Categorization*, *International Journal of Web Services Research*, Vol.9, No.3, 2012, pp.43-66.

[14] Wei Liu and Wilson Wong, *Discovering Homogenous Service Communities through Web Service Clustering*, *Proc. AAMAS International Workshop on Service-Oriented Computing: Agents, Semantics, and Engineering*, Estoril, Portugal, May, 2008, pp.69-82.

[15] Xuan-Zhe Liu, Gang Huang and Hong Mei, *Discovering Homogeneous Web Service Community in the User-Centric Web Environment*, *IEEE Transactions on Services Computing*, Vol.2, No.2, 2009, pp.167-181.

[16] Gang Huang, Li Zhou, Xuan-Zhe Liu, Hong Mei and Shing-Chi Cheung, *Performance Aware Service Pool in Dependable Service Oriented Architecture*, *Journal of Computer Science and Technology*, Vol.21, No.4, 2006, pp.565-573.

[17] Liang-Jie Zhang and Bing Li, *Requirements Driven Dynamic Services Composition for Web Services and Grid Solutions*, *Journal of Grid Computing*, Vol.2, No.2, 2004, pp.121-140.

[18] Huseyin Okcu and Mujdat Soyturk, *Distributed Clustering Approach for UAV Integrated Wireless Sensor Networks*, *International Journal of Ad Hoc and Ubiquitous Computing*, Vol.15, No.1/2/3, 2014, pp.106-120.

[19] Xin Dong, Alon Halevy, Jayant Madhavan, Ema Nemes and Jun Zhang, *Similarity Search for Web Services*, *Proc. the 13th International Conference on Very Large Data Bases*, Toronto, Canada, September, 2004, pp.372-383.

[20] Liang Chen, Liu-Kai Hu, Zi-Bin Zheng, Jian Wu, Jian-Wei Yin, Ying Li and Shui-Guang Deng, *WTCluster: Utilizing Tags for Web Services Clustering*, *Proc. International Conference on Service-Oriented Computing*, Paphos, Cyprus, December, 2011, pp.204-218.

[21] Jian-Gang Ma, Yan-Chun Zhang and Jing He, *Efficiently Finding Web Services Using A Clustering Semantic Approach*, *Proc. International Workshop on Context Enabled Source and Service Selection, Integration and Adaptation*, Beijing, China, April, 2008, pp.1-8.

[22] Xian-Zhi Wang, Zhong-Jie Wang and Xiao-Fei Xu, *Semi-empirical Service Composition: A Clustering Based Approach*, *Proc. IEEE International Conference on Web Services*, Washington, DC, July, 2011, pp.219-226.

[23] John MacQueen, *Some Methods for Classification and Analysis of Multivariate Observations*, *Proc. the 5th Berkeley Symposium on Mathematical Statistics and Probability*, Berkeley, CA, June, 1967, pp.281-297.

[24] Zhi-Zhong Liu, Yu-Lan Liu and Yi-Hui He, *A Two-Layered P2P Model for Semantic Service Discovery*, *Proc. the 4th International Conference on New Trends in Information Science and Service Science*, Gyeongju, Korea, May, 2010, pp.41-46.

[25] Matthias Klusch, Benedikt Fries and Katia Sycara, *OWLS-MX: A Hybrid Semantic Web Service Matchmaker for OWL-S Services*, *Web Semantics*, Vol.7, No.2, 2009, pp.121-133.

[26] Matthias Klusch and Frank Kaufer, *WSMO-MX: A Hybrid Semantic Web Service Matchmaker*, *Web Intelligence and Agent Systems*, Vol.7, No.1, 2009, pp.23-42.

[27] Marco-Luca Sbodio, David Martin and Claude Moulin, *Discovering Semantic Web Services Using SPARQL and Intelligent Agents*, *Web Semantics*, Vol.8, No.4, 2010, pp.310-328.

[28] Li Kuang, Shui-Guang Deng, Ying Li, Jian Wu and Zhao-Hui Wu, *Using Inverted Indexing to Facilitate Composition-Oriented Semantic Service Discovery*, *Chinese J. Software*, Vol.18, No.8, 2007, pp.1911-1921.

[29] Shui-Guang Deng, Jian-Wei Yin, Ying Li, Jian Wu and Zhao-Hui Wu, *A Method of Semantic Web Service Discovery Based on Bipartite Graph Matching*, *Chinese J. Computers*, Vol.31, No.8, 2008, pp.1364-1375.

[30] Karen Spärck Jones, *A Statistical Interpretation of Term Specificity and Its Application in Retrieval*, *Journal of Documentation*, Vol.28, No.1, 1972, pp.11-21.

[31] David Blei, Andrew Ng and Michael Jordan, *Latent Dirichlet Allocation*, *Journal of Machine Learning Research*, Vol.3, 2003, pp.993-1022.

[32] Hao-Li Wang, Rong-Guei Tsai and Ray Yueh-Min Huang, *Enhanced Quality of Service Control Scheme with High Availability for Self-optimising Wireless Sensor Network*, *International Journal of Internet Protocol Technology*, Vol.8, No.1, 2014, pp.25-32.

[33] Vankadara Saritha and Vankadara Madhu Viswanatham, *Approach for Channel Reservation and Allocation to Improve Quality of Service in Vehicular Communications*, *IET Networks*, Vol.3, No.2, 2014, pp.150-159.

[34] Jun Yan and Jing-Tai Piao, *Towards QoS-Based Web Services Discovery*, *Proc. International Conference on Service-Oriented Computing*, Sydney, Australia, December, 2008, pp.200-210.

[35] Ying Zhao and George Karypis, *Criterion functions for document clustering-experiments and analysis*, February, 2002. Department of Computer Science/ Army HPC Research Center, University of Minnesota, Technical Report RC #01-40.

Biographies

Zheng Li received the PhD degree in State Key Lab of Software Engineering, School of Computer, Wuhan University, China. She is a lecturer in School of Computer and Information Engineering, Henan University, China. Her current research interests include software engineering and services computing.

Keqing He received the PhD degree in Computer Science from Hokkaido University in 1995. He is a professor in State Key Lab of Software Engineering, School of Computer, Wuhan University, China. He has published over 100 papers in international conferences and journals. His research interests include software engineering and services computing.

Jian Wang received the PhD degree in State Key Lab of Software Engineering, School of Computer, Wuhan University, China in 2008. He is a lecturer in State Key Lab of Software Engineering, School of Computer, Wuhan University, China. His current research interests include software engineering and services computing.

Neng Zhang received the BE in Software Engineering from Wuhan University in 2012. He is pursuing the ME degree in State Key Lab of Software Engineering, School of Computer, Wuhan University, China. His current research interests include software engineering and services computing.