2019 Master's Thesis

Service Similarity Based User Centric IoT Service Management System for Service Search

A Thesis Submitted to the Department of Computer Science and Communications Engineering, the Graduate School of Fundamental Science and Engineering of Waseda University in Partial Fulfillment of the Requirements for the Degree of Master of Engineering

Quan Huilan

5118F057-2

Supervisor: Prof. Yoshiaki Fukazawa

Research Guidance: Research on Software Development Engineering

Fukazawa Laboratory

Submission Date: January 29, 2020

Abstract

The Internet is extending to the physical world, forming the premises of the Internet of Things (IoT). IoT environment has some characteristics like IoT service's potential huge number, the heterogeneity of the objects that host them, their distribution and their mobility which results in objects roaming from one smart space to another. So, how does a user search and select IoT services suiting his requirements in such an environment? All contribute to making the search of relevant IoT services for a given situation challenging.

In the process of service search, we consider the dynamic nature of the IoT environment and also believe that since the differences of the context of each user, each user has different requirements for services. To address these issues, we examine two major research problems, how to design the architecture of service management system to manage the IoT services and how to use service similarity information for searching the services. In this paper, we present a user centric social service network (USSN) system for service management and search.

Content

1. Introduction 1

1.1 IoT Service 1

1.2 Service matching and matching elements 1

1.3 IoT Service Management and Search 2

2. Problem Analysis and Definition 4

2.1 Characteristics of the IoT environment 4

2.2 Solution of Service Search in IoT 5

2.2.1 Classification and Clustering 5

2.2.2 Social Network 5

3. Related Work 7

3.1 Service Matching 7

3.2 Service Search 8

3.2.1 Service Search Using the Clustering 9

3.2.2 Service Search Using the Social Network 9

4. User centric Service Management System in the IoT 11

4.1 Proposed System Model (Social Service Network) 11

4.1.1 Social Link 12

4.1.2 Community 14

4.2 Construction of Social Service Network 14

4.2.1 Find Service using Social Link 14

4.2.2 Constructing a Social Service Network 16

4.3 Service Search and management 17

5. Evaluation 18

5.1 Simulation Setting 18

5.2 Result and analysis 19

5.2.1 Service Search result 19

5.2.2 Comparison of Matching Cost 21

5.2.3 Analysis 23

6. Conclusion ***24***

Acknowledgments ***25***

References ***26***

Chapter 1

Introduction

1.1 IoT Service

The Internet of Things (IoT) based on SOA (service-oriented architecture), and it is a sophisticated platform that interconnect trillions of geographical distributed small physical objects or things [1]. The functionality of the smart physical objects can be abstracted as a software service and we define this service as IoT service. And an IoT application can be built by combining the IoT services which are provided by smart devices.

1.2 Service matching and matching elements

Each user has different requirements for the service, and the service elements considered are also different. In order to search a service that meets user requirements, we need to perform a matching calculation of query and service in service search process. If a service meets the user's requirements for these elements, we can judge that the service is the service requested by the user and return it to the user.

Service matching is an important part of service search, which mainly involves two aspects: matching elements and matching strategies. Next, we will briefly introduce the matching elements in the current service search approach.

In IoT, a user needs to select desirable high-quality IoT services that satisfy both user's functional and non-functional requirements. In this section, we introduce the service elements about functional and non-functional requirements. Each object provides its functionality through standard services that can be directly accessed on the Web. Service's functionality is a set of functional properties that represent the description of the service tasks in terms of operation signatures. Functional property is the functional semantics of a service that describes what a service actually does. such as operation names, service description, and input/output schema.

QoS refers to the quality of a service, it can respond to a query, perform related tasks with a certain quality of service, and provide quality of service to meet

expectations. the most commonly used non-functional attribute parameters include cost, response time, availability, and reliability [9]. Cost represents the cost that the consumer must pay to invoke the service. Response time represents total time about execution time of the service and the communication time between the service consumer and the service providers. Service availability is defined as T/t, where t is a time interval and T is the execution time of a service in the time interval. Service reliability is defined as N/n, where N is the number of successful services executions. n is the total number of invoked services.

In IoT, we also need to consider a non-functional element and that is Context. Context is defined in a variety of ways in the literature [10,11]. Here we use one of the most referenced definitions for context: "any information that can be used to characterize the situation of an entity," where entity is defined as "a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and the application themselves" [12].

1.3 IoT Service Management and Search

There are new challenges of revisiting and extending existing techniques of service-oriented computing to be more scalable and efficient for dynamic IoT environments [22], such as service search and selection. And we need an IoT service management system model for efficient service search. Most existing computing models are centralized and fail to scale with respect to the number of advertised services [3]. Therefore, recent studies have suggested distributed computing models based on the Mobile Ad-hoc Network (MANET) [27] to address the scalability issue [29].

Then, a distributed IoT service management system, as can be seen in Fig.1. In this framework, a *task* is a description of a user's requirement, which entails the services required to provide necessary functionalities. A *service* defines a set of functionalities that are necessary to support an activity for a user task. And within IoT service management system a service that exist at the physical Layer and a service in the management system are considered as one-to-one mapping.

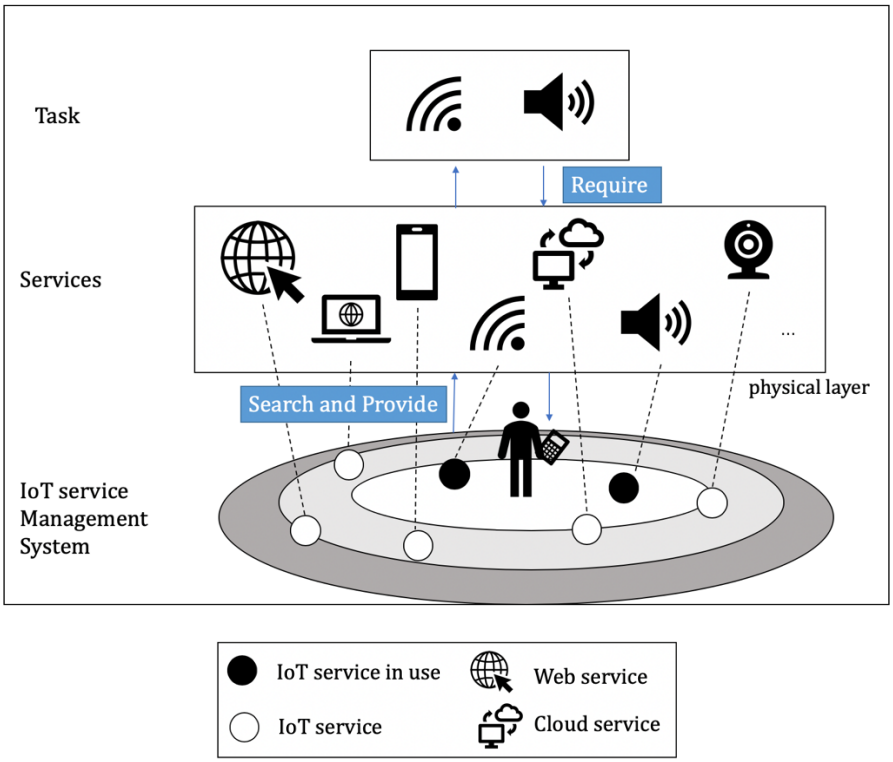

Fig.1Distributed IoT Service Management System

But in the process of service management and search, we need to consider the characteristics of the IoT environment. Firstly, the dynamics of the IoT environment are reflected in users and services. Secondly, Differences in each user context in the IoT environment. Users in different contexts have different requirements for services.

Based on the above characteristics, we need to consider what structure to use to manage services to facilitate service search. In the earlier works, the service search issue in IoT was addressed through the use of semantic knowledge to find similarities among the services [5], [6], social networks [7], [8] etc. In paper[7], it prove if service network tend to bring similar users together as neighbors, the homophily is effective in facilitating service search with shorter paths. Hence, we decided to apply the concept of social network to user centric service management system to establish social relationships between services.

The rest of the paper is organized as follows. In the chapter 2 and 3, we provide a review of problem analysis and definition and related works. Chapter 4 presents the proposed user centric service management system model for the service search. In chapter 5, we provide an experimental evaluation about the proposed system and search approach. Finally, concluding remarks are given in chapter 6.

Chapter 2

Problem Analysis and Definition

2.1 Characteristics of the IoT environment

In the process of service management and search, we need to consider the characteristics of the IoT environment.

Firstly, the dynamics of the IoT environment are reflected in users and services. And because the IoT services run directly on resource constrained smart objects. Services' QoS values are dynamic and may change significantly during use, since smart objects can join, leave, fail, or new services with better quality can appear. In order to effectively search for services, the service management system must reflect the changes of services each time with the user environment changes. In other word, user needs to manage services in real time and need to quickly add the service or quickly search the service to delete it in the management system.

Secondly, different from traditional web services, users' requirements and service performance are affected by their context. Each user is in a different context and has different requirements for the service. For example, The user is in a high-speed mobile state and prefers a service with short response time, high throughput, or in same geographical environment. When the user is busy, he wants to choose a service that can be used for a long time and does not want to switch services every time. So, he prefers a service with a large amount of device battery and a stable one. Differences in each user context in the IoT environment. Hence, if a large number of services are managed by one centralized service management system. For each user's request, it is impossible to quickly and efficiently search for services. We need to consider how to design a service management system that can satisfy each user with a different context.

In addition to the above characteristics, due to the massive, highly resource-constrained natures of devices, the unreliability of wireless network in IoT, services provided by devices have different characteristics with traditional Web services, and existing Web service search approaches can't satisfy the requirements of service search in IoT.

2.2 Solution of Service Search in IoT

In order to effectively manage and search IoT services, we need to think about the following points:

2.2.1 Classification and Clustering

To overcome scalability issues, some researches use classification and clustering approaches to manage services, this approach needs to be based on certain user-customized criteria. If a user has multiple criteria, it needs to construct multiple system models, as shown in Fig.2.

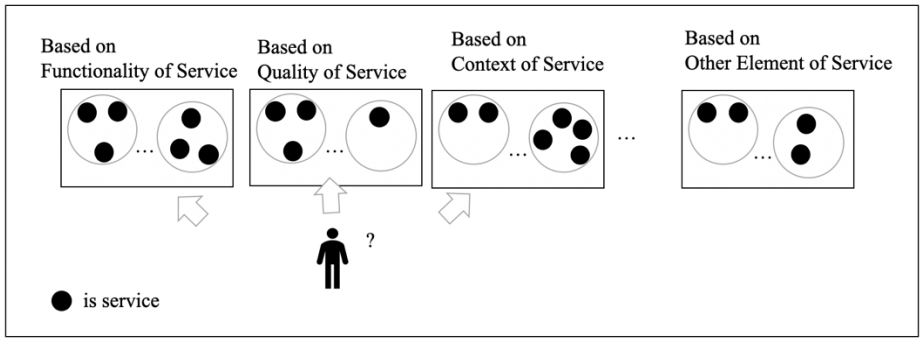

Fig.2 Manage Services by Classification and Clustering approaches

Here, we need to consider how to satisfy multiple service elements and reduce search space through a management system.

2.2.2 Social Network

In addition, in many social network-based approaches [7][17][31], it has been verified that the user or object's social relationship or social profile can effectively search for services. Social network consists of a set of participants and the pairwise relationships between them. And the social relationships considered can be diverse, such as common hobbies, the same school or company, and so on. From the social network generated it is possible to manage the scalability and navigability of the network, allowing efficient search of services and objects. In order for each user to easily find the service they want; the established relationship can be established by its owners. Accordingly, in the service management process, we can consider whether a user can create so-called "social relationships" based on similar relationships between services, and effectively search services based on these "relationships".

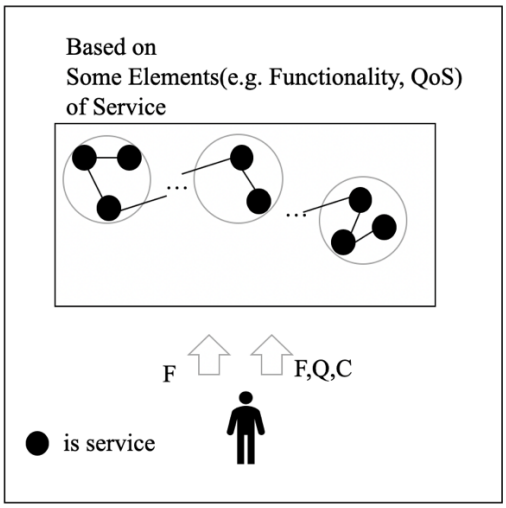

Fig.3 Manage Services Using the Social Network

Based on the above analysis, we propose a user centric service management system which can constructed by user. As shown in Fig. 3. In this system, we create social links between services based on the similarity between services elements. Compared to classification and clustering approach which manage multiple target system, social network can form a network through the links between services and services, and only need to manage one target system.

Chapter 3

Related Work

3.1 Service Matching

In aspect of service matching, Bianchini et al. [14] consider the limited resources of mobile devices. So, they proposed a lightweight service search method that only matches service operation names and outputs. While Zhang et al. [15] only match the service input and output and QoS. Mokhtar et al. [16] believe that the time cost of semantic reasoning is large, and it reduces the number of semantic matches by classifying services.

In Chen's [17] research, they consider the input and output of a service and have arbitrary data type to calculate the service functionality. And when several services have similar values of functionality, their QoS properties such as price, availability, reliability and reputation become important to ensure the quality of the services. The specific calculation method is not proposed, and the numerical value is simply used to express the QoS of the service. Sara et al. [23] proposed a new context-based solution based on QoS exploiting both functional and non-functional user's requirements and providing the user ability to control and proceed the search of web services. However, this solution needs to know the maximum value and minimum value of QoS factors in the entire services. Instead, Ahmed et al. [24] proposed a method of similarity calculation, Proposed measure is applicable to all situations, including the ones where some characteristics are unknown or ignored. This constitutes a generic measure with a wide range of utilizations.

Liu et al. [25] believes take into consideration the context of customers is believed to produce better recommendations and proposed a novel context similarity metric to guide the aggregation process and show how it can be extended across multiple context dimensions. From above related work of service matching, we can know when designing an IoT service search algorithm, we should choose different matching strategies and matching elements according to different user scenarios.

3.2 Service Search

We will briefly review prior work related to service search.

Service search phase identifies services with similar functionality. The service search method based on UDDI (universal description search and integration) is a typical service search method using a centralized architecture. Fig. 4 illustrates a centralized architecture of Web service search. While this is acceptable when dealing with a small set of described entities, it is inefficient to use a centralized approach to semantically process, store and search amongst thousands of IoT service descriptions [3].

Distributed architecture, namely peer-to-peer (P2P) architecture, is mainly used for service search in mobile network environments. Fig. 5 illustrates a distributed architecture of Web service search. Moreover, each peer can be either a service provider, a service requester, or both.

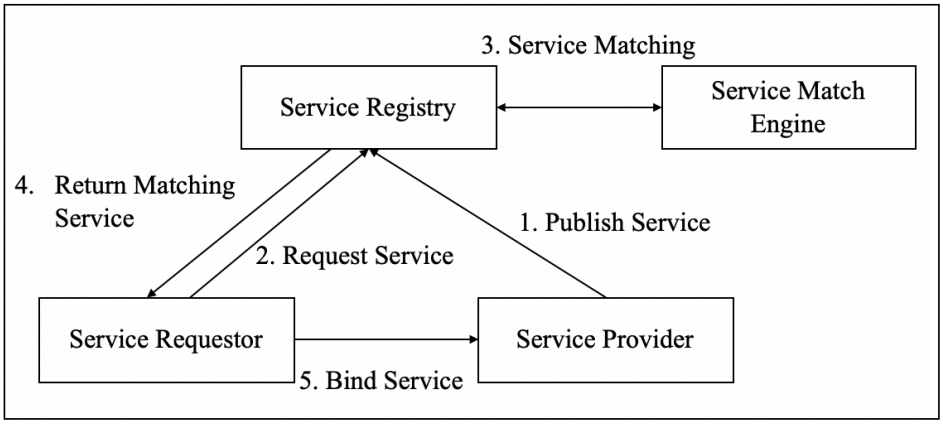

Fig.4 The centralized architecture of service search [32]

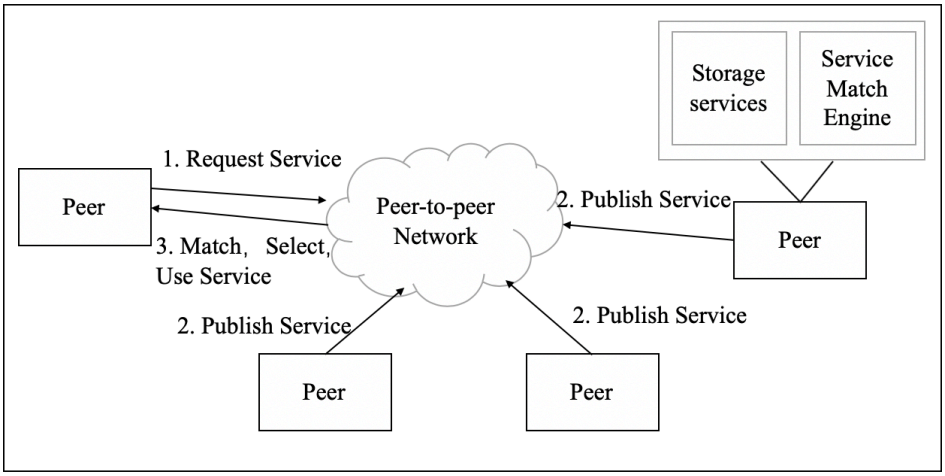

Fig.5 The distributed architecture of service search [32]

3.2.1 Service Search Using the Clustering

To overcome scalability issues, some research works propose to classify, cluster, or filter services based on certain criteria in advance to reduce the search space. Zhang et al. [13] believes that context-based search mechanism can be used to reduce search space, thereby reducing search response time and saving energy on resource-constrained devices. Sameh Ben et al. [3] proposed search system embeds clustering and information aggregation mechanisms based on a criterion that they defined. Sameh Ben et al. [26] also presented Statistical clustering (e.g., K-means, Hierarchical clustering) has been investigated in clustering Web services based on similarity metrics. But they do not adapt well to dynamic contexts.

From above related work about clustering, we can know that use clustering approach to manage similar services in a collection, reduce the matching cost. But, if the user's context changes, the criteria for clustering need to be redefined. But from the above research, we can know that managing similar services in a set can reduce the search space and also can reduces service search time.

3.2.2 Service Search Using the Social Network

Other research suggests using social relationships for efficient service search. In Chen's [17] research, they construct a global social service network for better quality of web service search, and they indicate isolated service islands mean that service search is confronted with some issues. but their research is about Web service, the considerations for constructing a global social service network are not applicable to the IoT environment. As for an IoT service, various requirements besides quality have to be considered, such as the correlation between two services and the location of the services. Besides, it is not quite appropriate to use quality only to judge whether a service is satisfactory or not. Iury et al. [18] proposed a solution for service search in a Social IoT network. This solution uses the relationships between objects to improve the search scalability and considers their social profiles to meet the requisitions in a more satisfactorily way. Guo et al. [19] proposed a distributed approach for IoT device management and service composition from a social network point of view. According to the relationships between IoT devices, social network theory is applied to model IoT services in different dimensions. And they classified IoT services into three dimensions which are location, type and correlation and manage them. In Michele's

[20] research, a Social IoT-based object search mechanism is proposed. The novelty of this algorithm is that the next hop to query is chosen based on two properties: one that is intrinsic to the network and is based on object friendships, and an external one that takes into account the similarity between the object and the query. From above research works, we also can realize the Social IoT paradigm, where objects establish social-like relationships, has become popular as it presents several interesting features to improve network navigability and implement efficient search methods.

Chapter 4

User centric Service Management System in the IoT

4.1 Proposed System Model (Social Service Network)

The proposed service management system is a user-centric social service network (USSN). In IoT environment, the functionality of the smart devices can be abstracted as a software service [4]. We treat these services as virtual services and manage them with a social network structure. As shown in Fig.6, in the service search process, when a user requests a service, we need to search for the service in the service storage and return it to the user. In the service management process, when the sensor senses changes in the service (such as joining, leaving), we need to search in the service storage based on the service information and update the service information.

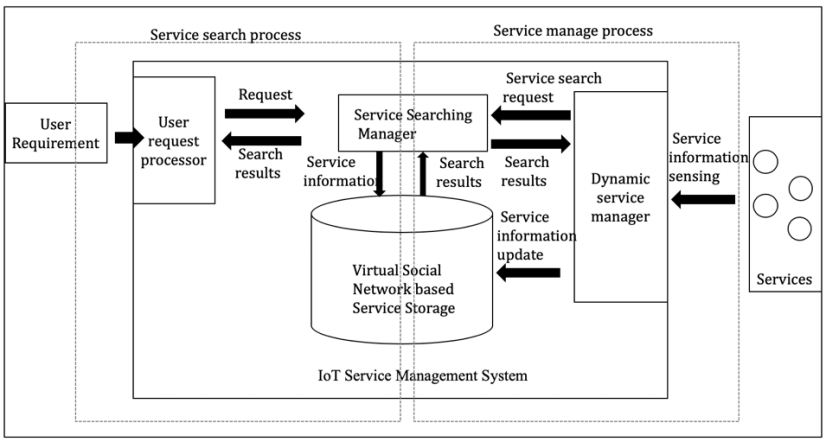

Fig.6 IoT service management system Architecture

Fig.7 depicts the scenario of using USSN system model, each user establishes his own social network to manage these services. Because the user's context and requirements are different, the social network established by each user is different. Social network updates frequency and the connection properties between nodes and nodes are different.

Definition1 (User centric social service network)

USSN system structure is an undirected graph G = <*V, E*> on the IoT. To avoid confusion, within this paper a service and a node are considered as one-to-one mapping; therefore, the terms ''service'' and ''node'' are totally interchangeable.

where:

- *V* represents a set of services; and
- *E* represents a set of undirected edges, with each edge corresponding to social link

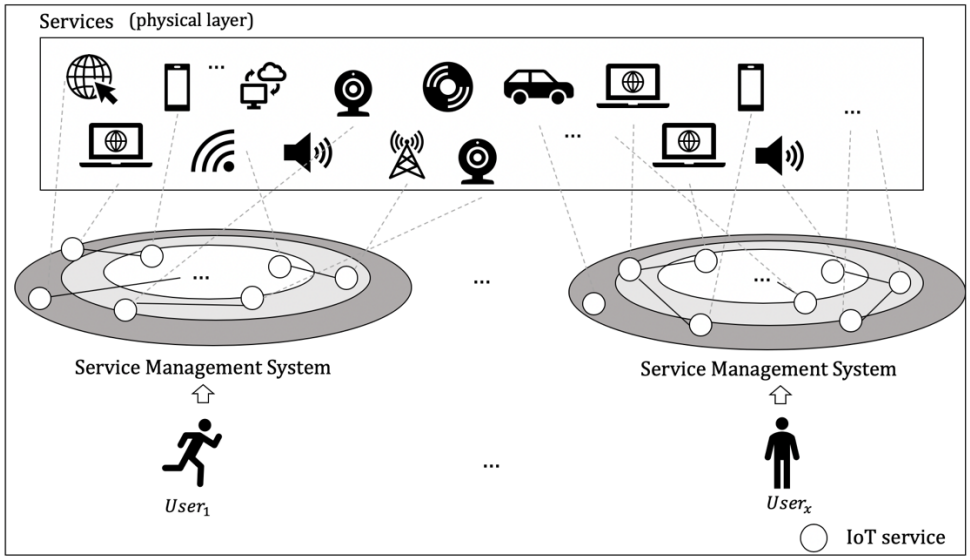

Fig.7 Scenario of Using Service Management System

Definition2 (Service Elements)

- Node: V = $\{v_1, v_2, v_3, \ldots v_n\}$, n represents the number of nodes in social service network
- Edge: E = $\{e_{(1,2)}, e_{(1,3)}, e_{(2,1)}, \ldots e_{(i,j)}\}$, i, j represents the id of the service.
- Service elements: R = $\{r_1, r_2, r_3, \ldots r_m\}$, Users use service elements to search and select the service. So, these elements are decided by a user. For instance, Functionality, QoS, Context

4.1.1 Social Link

To connect services into a social service network, social links are formed between isolated services. Because our social link is based on the similarity of service elements, in order to get the similarity value, we need to perform service matching. Here we define the type of social link that represents the similarity between services.

Definition3 (Service Similarity and Type of Social Link)

- Similarity of service element: $Sim_{r_m}(v_i, v_j)$ represents the similarity of

service elements r_m between v_i and v_j. The higher the similarity value, the more similar the two services are. When compare the same services, the similarity will be 1. In order to get the type of social link between two services, we need to perform service matching on the service elements. If the value of the similarity of a certain service element meets the user's threshold, we will define this element as the type of social link between two services.

- Every edge stores the type of social link: $e_{(i,j)}$= {r_1, r_2,...}.
- A node can have multiple edges, indicating different relationships with the neighbor nodes.
- When we calculate the similarity between two nodes, and the similarity of each element of the service satisfies the user defined threshold *th*, we call the two nodes are fully connected.

To facilitate understanding we take the social service network shown in Fig. 8 as an example: when a user considers the service elements R = {r_1, r_2, r_3},if a user defined threshold is *th*, and $Sim_{r_1}(v_1, v_5) > th$ and $Sim_{r_2}(v_1, v_5) > th$, $Sim_{r_3}(v_1, v_5) < th$, we define the type of social link between $v_1$and v_5 is {r_1, r_2}. Accordingly, node v_1 is connected to v_5 with edge $e_{(1,5)}$,and $e_{(1,5)}$= {r_1, r_2}. And node v_1 has multiple edges, node v_1 is connected to v_2 with edge $e_{(1,2)}$,and $e_{(1,2)}$= {r_1, r_2, r_3},and is connected to v_3 with edge $e_{(1,3)}$,and $e_{(1,3)}$= {r_1, r_2, r_3}.We also can know v_1 *and* v_2, v_1 *and* v_3, v_1 *and* v_4 are fully connected in this social service network. Instead, v_1 *and* v_6, v_1 *and* v_5 are non-fully connected in this social service network.

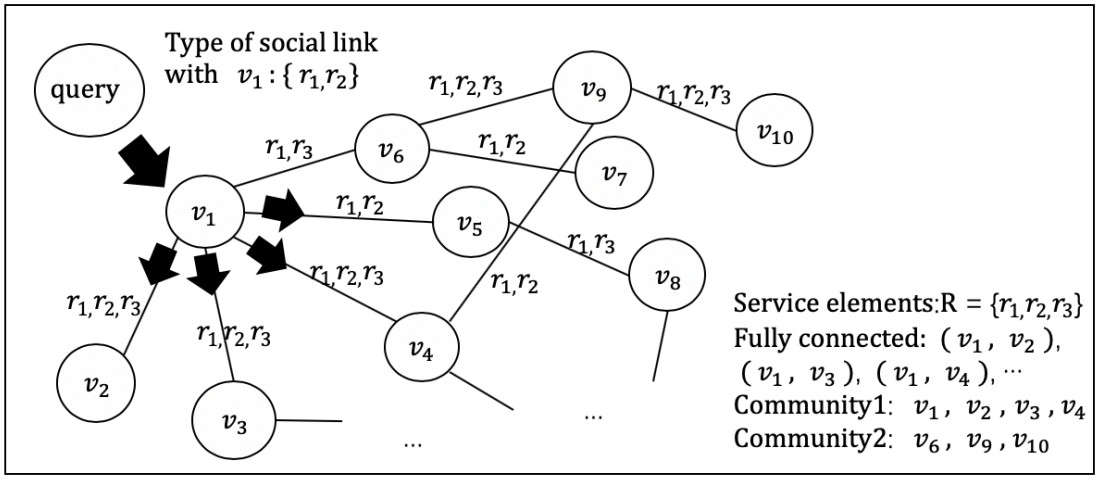

Fig.8 An example search service in USSN.

4.1.2 Community

In section 2.2.1, we analyze that manage services by classification, which is helpful for service search. In other words, we need to consider building a service group in the social network, and the services of this service group can be replaced with each other with a high probability. We analyze advantage to construct a community with similar services：

- The services in the community can be replaced by each other with high probability. If we find a similar service, we can quickly get a series of services similar to this service from same community. So, when the service needs to be replaced, service candidates can be quickly found.
- The number of comparisons in a community is limited. A Community can be virtualized into a large object for comparison. If you do not find the service after comparing it several times, then come out of this community and select a new community (or lead to a new community in the process of comparison). It can reduce the number of comparisons.

Definition4 (Community)

So, in social service network, the collection formed by the fully connected services we call the community. In Fig.8, v_1, v_2, v_3, v_4 consist a community and v_6, v_9, v_{10} consist a community.

4.2 Construction of Social Service Network

4.2.1 Find Service using Social Link

To add services as linked service in social service network, it is important find service to link together based on type of social link. we developed the service search algorithm to find similar services using type of social link as defined above.

In this process, we should reduce the overhead in the search process by determining search direction and reducing the number of visited services. Therefore, we use the community to reduce the number of comparisons. We show the specific search process as Algorithm1. In this paper, we consider three typical service elements, which are functionality, QoS, and Context. So, we define R = {*fun, qos, context*} and the leader service represents the first service generated in a community. The idea of the Algorithm1 is searching a service based on type of social link. When

we select a service that exists in the USSN to perform service matching between two services, if there is no element meets the threshold required by the user, we can judge that we don't need to compare with other services which connected to this service in the same community. And when there are elements meets the threshold and get the type of social link between two services, we can select the next service to compare based on this type of social link and finally find a service that can be fully connected. For easy understanding, we give an example in Fig. 8. when we add a node in USSN, we send a query to find a node which can be fully connected. We assume compare this node compare with v_1 firstly, and similarity result(type of social link) is $\{r_1, r_2\}$.Based on this result select the v_2, v_3, v_4, v_5 to compare in next step.

Algorithm 1: Search the similar service

Input: G<V, E>, threshold *th*;

Output: id of service which can fully connected

Variables: service queue *Sq*; service has Type of Social Link with other services; elements value of each services, Target service S_i, Resource service S_j;

Randomly select the leader service in a community from G<V, E> and put this node in *Sq*;

While (*Sq* is not empty) **do**

S_j= *Sq*.pop;

Calculate every elements similarity

$Sim_{fun}, Sim_{qos}, Sim_{context}$between S_i and S_j

If ($Sim_{fun}(S_i, S_j) > th$ and $Sim_{qos}(S_i, S_j) > th$ and

$Sim_{context}(S_i, S_j) > th$) **then**

return id of S_j;

end

else

Select services that are connected to S_j with the same type of social link $e_{(i,j)}$ and put these services in *Sq*;

End else

end

4.2.2 Constructing a Social Service Network

Considering the issues discussed above, we use an algorithm to build our network. We introduce the parameters used in the construction process in Table 1.

parameters	Function
n	The number of nodes added at every step of the algorithm
p	Number of nodes that can be fully connected to a node
q	Number of nodes that can be non-fully connected to a node
th	Threshold of similarity value
Leader node	The first node generated in a community
Target node	The node to be added
Resource node	The service existing in the USSN

Table 1. Parameters and their function

The procedure starts with empty social network and performs one of the following three actions at every step.

1. New target nodes are added. Target node iterate through all the leader nodes in the community for similarity comparison to find a node that can be fully connected. The first target node joins the USSN directly.
2. Using the Algorithm1, target node searches the fully connected resource node to connect. We need to note that each node can only connect *p* fully connected nodes to avoid connecting too many nodes at one node. If there is no element meets the threshold, this target node constitutes one community in USSN. If found, target node is fully connected to this resource node.
3. The target node also gets some type of social links with non-fully connected resource nodes when it finished step2. From previous calculation record, it selects $^{q}/_{2}$ non-fully connected resource nodes with high similarity value of other communities. This step ensures connectivity between communities and a target node with multiple type of social link with different resource nodes. We also need to note that each node can only connect *q* non-fully connected nodes

4.3 Service Search and management

We describe how to construct a USSN in 4.2.2, and then a user uses this USSN to manage services. In the previous research [2], we proposed an approach to select the best service based on feedback from service candidates in real time. Therefore, in the process of service search, the user needs to find a certain amount of services to be provided for selection each time. The service search process is showed in Fig. 9. When a user sends a query and he want to search the *m* service candidates for selecting. we will first search from the node which is selected randomly to find fully matched nodes considering service elements similarity. If the service is fully matched with service query, we put this node in service candidates set and increase (k)the number of service candidates in the set. If not, the query needs to be further forwarded, the node will be based on the previous service matching result (type of social link) to find associated nodes. And if user cannot find the next associated nodes, he needs to forward query to a new leader node in another community. Through above procedure, our system provides the *m* service candidates which all satisfies user's requirements.

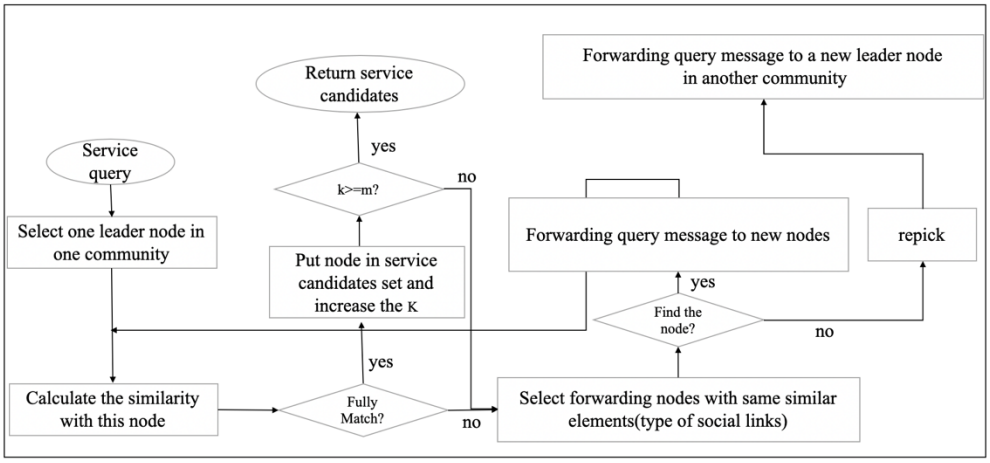

Fig.9 Service search process flow chart

When user manages the services, due to the dynamic nature of the IoT environment, new services will need to be added to the USSN and services will need to be deleted. The process of adding new services is discussed in the construction of the USSN, When the service leaves, we need to delete the service from the USSN, we have to find a specific service based on the information of the service elements. At this time, we only need to set the threshold of similarity to 1 for each element.

Chapter 5

Evaluation

5.1 Simulation Setting

In this section we verify our user centric service management system and service search approach. To validate our proposed approach, we set up two research questions：

RQ1: Can we successfully find a service that meets user requirements for multiple elements every time?

RQ2： Do we reduce the comparison space during the search service?

In this paper, we set that a user considers the following three elements of the service; the functionality, QoS and context of the service. We set different numbers of services and randomly give values for each element of each service. The similarity calculation approach can apply the following research approaches [17][23][24][25]. Because of the dynamic nature of the IoT environment, when the service being used by the user becomes unavailable or the user experience is degraded due to low performance, other high-quality services need to be selected to replace from the service candidates. A user needs to make sure he can search a certain number of desired services each time among a large number of services. Then a user selects the best quality service from the searched services.

A user needs to construct a USSN in advance and provide a threshold for similarity when establishing a USSN, and a threshold for similarity when searching for a service, and the number of services to searched. And in our USSN, all fully connected services in a community are services with a high probability of similarity. As the number of services connected between two services increases, the similarity between the two services is likely to decrease. In order that the similarity between any two services in a community should not be too low, we set the threshold when constructing the USSN to 0.9.

For state-of-the-art approaches, we compare with a service management approach which is clustering service using some criteria. There are many clustering approaches. Here, we select to use the approach named k-means since it is the most common clustering algorithm. When we use k-means approach we need to consider the

number k that needs to be classified. After service clustering, when a user needs to search the services, his query is compared with the representative nodes in each service group and return all the services in the most similar group. In the experiment, k-means cluster services based on 3 elements to get the three groups：Serviceset_F (functionality-based clustering), Serviceset_Q (QoS-based clustering)， Serviceset_C (context-based clustering) that are most similar to the user's requirements. The services in each group meet the user's threshold requirements for this element. To meet user requirements for all elements, we return to the user services that exist in all three groups: Serviceset_F∩Serviceset_Q ∩Serviceset_C.

The simulations were performed on a macOS 10.14.3 machine within an Intel i5-7500U 2.30 GHz CPU, 8 GB RAM, and Intel Iris Plus Graphics 640 GPU.

5.2 Result and analysis

5.2.1 Service Search result

We first verify the RQ1. We explain the simulation parameters and their default values in Table 2. We perform 10 experiments on one setting, and with different parameters value we can get the different service search results. From Table3 and Table 4 show the different results with two different settings. From Table 3, we can know our proposed approach can search a certain number of services from 10000 services every time. In contrast, when *k* is 20, k-means approach can search services based one element. But if a user wants to search services that meet user's requirements for three elements， the user can only search single-digit services or cannot search services. From Table 4, we can know if *k* is 100, a user cannot search services that meet user's requirements for three elements each time.

Setting:

Parameters	Value	Function
n	10000	Number of services
th1	0.9	Threshold of similarity when constructing USSN
th2	0.8	Threshold of similarity when searching services
m	30	Number of services to be searched (it can be decided by a user)
k	20/100	Number of clustered sets

Table 2. Simulation parameters and their default values.

Test ID	1	2	3	4	5	6	7	8	9	10
Number of services meet a user needs	396	676	522	637	504	668	390	359	322	270
Proposed approach	30	30	30	30	30	30	30	30	30	30
k-means approach	1	2	**0**	3	**0**	1	1	4	1	1
Serviceset_F	548	396	656	396	395	491	299	667	612	447
Serviceset_Q	679	412	573	618	412	412	618	573	742	746
Serviceset_C	459	428	453	428	453	428	680	972	428	866

Table 3. Service search result when k is 20

Test ID	1	2	3	4	5	6	7	8	9	10
Number of services meet a user needs	632	622	453	404	626	672	561	425	556	389
Proposed approach	30	30	30	30	30	30	30	30	30	30
k-means approach	0	0	0	0	0	0	0	0	0	0
Serviceset_F	218	159	242	224	141	35	70	64	141	119
Serviceset_Q	160	152	81	64	112	223	59	223	142	160
Serviceset_C	187	156	131	131	85	152	179	140	61	45

Table 4. Service search result when k is 100

Our approach and k-means approach both manage services with similarity between services, the purpose is to search the service that user needs every time and reduce unnecessary comparisons. From above results, proposed approach can consider multiple service elements at the same time and every time we can successfully search services that meets the user's requirements for multiple elements. However, if user uses k-means clustering approach to manage services, they will have the following difficulties. First of all, user should consider how to give the value of *k*, because different values of *k* will directly affect the success rate of service search results. And the results are not optimistic when considering multiple elements at the same time.

Even if Clustering approach can always search the services based on one element, an additional step is still required to search which services satisfy the other elements' requirement.

5.2.2 Comparison of Matching Cost

Because the clustering approach cannot find all the services that meet the requirements of a user every time. In order to verify RQ2, we also compared the prototype performance to a non-optimized centralized approach, simulated by a single agent where all contained services are matched against a received request. We will compare matching costs on the basis of ensuring that the corresponding service can be searched every time. We use the following metrics to measure the performance of our system:

- Matching cost: The total number of query-service matching operations until finding all suitable services

We assume a user search all services that meets his requirements among a large number of services, we compare average matching cost. We calculate the average matching cost over 10 requests. The results are shown in Fig.10. We compare the evolution of the matching cost of our approach with the centralized one. The results show that proposed approach about half less matching cost than the centralized approach.

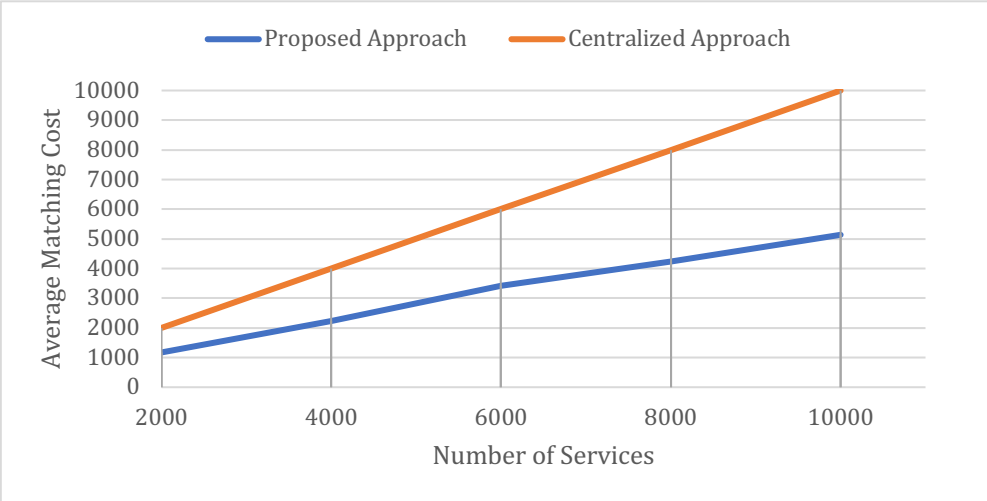

Fig.10. Matching cost evolution per number of services.

Because the dynamics of the IoT environment are reflected in users and services. To manage services dynamically, adding and removing services is also an

important step in the process of managing services. So, we compare the matching cost when a user adds a service in USSN, and a user deletes a service from USSN. We calculate the average matching cost over 10 times. The results are shown in Fig .11 and Fig.12. As shown in Fig.11, when adding a service to USSN, we need some comparison process to find a suitable place for this service. However, the centralized approach does not require any comparison process at this time. As shown in Fig.12, we compare the evolution of the matching cost when deleting a service. As the total number of services increases, average matching cost increase rate in proposed approach is smaller than the centralized approach. The process of deleting a service is also equivalent to the user searching a specific service, that is, the user puts forward specific requirements for each element and proposed approach searches that service in the USSN. As shown in Fig.13, we compare the evolution of the matching cost when adding and deleting a service. We can see that our overall matching cost is less than the centralized approach.

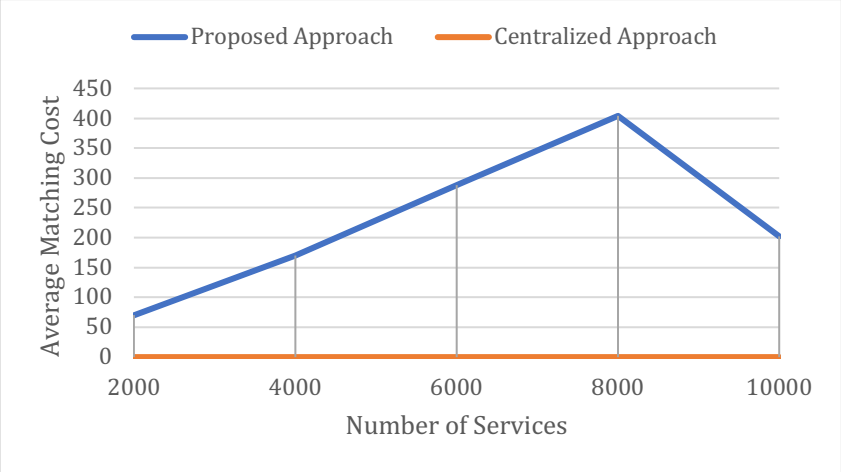

Fig.11. Matching cost evolution when add a service

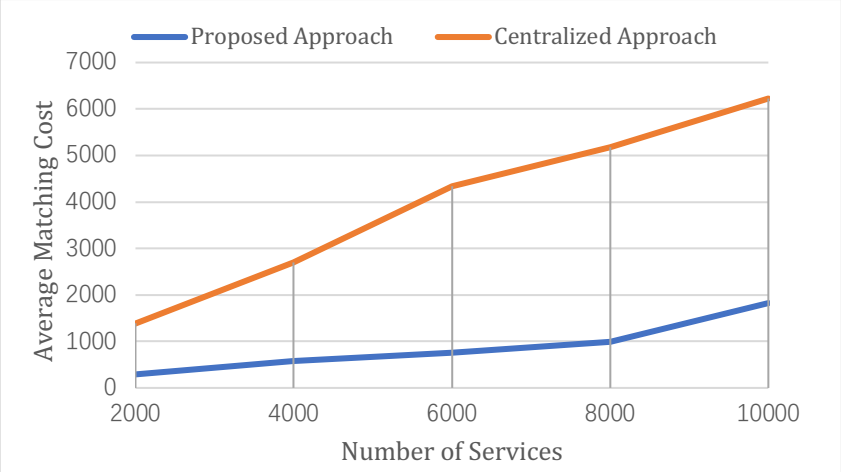

Fig.12. Matching cost evolution when delete a service

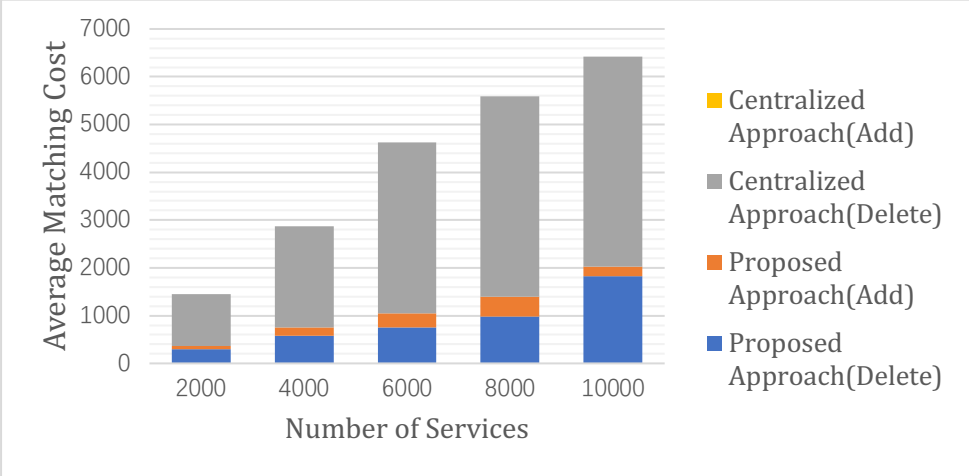

Fig.13. Matching cost evolution when add and delete a service

5.2.3 Analysis

It can be seen from the above experiment results that both our proposed approach and the clustering approach have a common idea. That is managing similar services together to reduce unnecessary comparison processes. Clustering approach is based on creating various service collections, and proposed approach uses the concept of community and uses service social links to connect services. But when a user considers multiple elements of the service, our approach can successfully provide the service that the user requested. And the user can decide the number of services the user needs, the service elements to be considered. If a user wants to search some similar services, the threshold for the similarity of each element can be set to a higher value of 0 to 1. If a user wants to get a specific service, user only need to set the threshold of similarity to 1. However, the clustering approach needs to consider each element of the service separately. Moreover, the number of cluster k must be determined by the user, but in reality, the user does not know how many services he will receive and does not know how many categories there are.

And in order to ensure that a user can search services that meet the requirements of multiple elements every time, we choose to compare with the non-optimized centralized approach, and proposed approach can also find that our approach can greatly reduce the matching cost and it can also manage services dynamically.

Chapter 6

Conclusion

Herein we propose a user centric social service network (USSN) system for service management and search. And an intelligence search approach is also proposed to effective service search. In this process we analyze some of the characteristics of social network and the advantages of community and make full use of these points in our system to create the type of social links based on the similarity between various elements between services. Determine search paths based on relationship intelligence in these service social relationships. And fully consider the characteristics of the IoT environment, such as the dynamic nature of the environment, the user context is different. Finally, the evaluation demonstrates the advantages of our proposed system in IoT service search.

Acknowledgments

I will always be grateful and indebted to my supervisor, Professor Yoshiaki Fukazawa and Teaching assistant Ryuichi Takahashi for their wise and generous comments, encouragement, support and guidance. They made it possible for me to finish this research.

Obviously, I will never forget the kind support of my whole family, friends, and lab mates.

References

[1] I. Mashal, O. Alsaryrah, T.-Y. Chung, C.-Z. Yang, W.-H. Kuo, and D. P. Agrawal. Choices for interaction with things on the Internet and underlying issues. In Ad Hoc Networks, vol. 28, pp. 68-90, 2015/05/01/,2015. 10.1016/j.adhoc.2014.

[2] Huilan Quan, Ryuichi Takahashi, Fukazawa Yoshiaki. Dynamic Service Selection based on User Feedback in the IoT Environment, " pp. 1-5, 2019/08/29/,2019. In International Conference on Computer, Information and Telecommunication Systems 2019.

[3] Sameh Ben Fredj, Mathieu Boussard, Daniel Kofman, Ludovic Noirie. A scalable IoT service search based on clustering and aggregation. In IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing.2013.

[4] Santosh Pattar, Rajkumar Buyya, Kuppanna Rajuk Venugopal, S. S. Iyengar, L. M. Patnaik. Searching for the IoT Resources: Fundamentals, Requirements, Comprehensive Review and Future Directions. In IEEE Communications Surveys and Tutorials 20(3): 2101-2132. 2018.

[5] S. Zhao, L. Yu, B. Cheng, and J. Chen. IoT Service Clustering for Dynamic Service Matchmaking. In Sensors, vol. 17, no. 8, p. 1727, 2017.

[6] S. N. Han and N. Crespi. Semantic Service Provisioning for Smart Objects: Integrating IoT Applications into the Web. In Future Generation Computer Systems, vol. 76, pp. 180–197, 2017.

[7] B. Yuan, L. Liu, and N. Antonopoulos, "Efficient Service Search in Decentralized Online Social Networks," Future Generation Computer Systems, pp. 73–78, 2017.

[8] M. S. Roopa, S. Pattar, R. Buyya, K. R. Venugopal, S. S. Iyengar, and L. M. Patnaik. Social Internet of Things (SIoT): Foundations, Thrust Areas, Systematic Review and Future Directions. In Computer Communications, vol. 139, pp. 32–57, 2019.

[9] Tarik Fissaa, Hatim Guermah, Mahmoud El Hamlaoui, Hatim Hafiddi, Mahmoud Nassar. An Intelligent Approach for Context-Aware Service Selection using Machine Learning. In the International Conference on Learning and Optimization Algorithms: Theory and Applications 2018.

[10] Baltrunas, L.; Ludwig, B.; Peer, S.; and Ricci, F. Context relevance assessment and exploitation in mobile recommender systems. In Personal and Ubiquitous Computing, 16, 5, 507–526.2012

[11] Lieberman, H., and Selker, T. Out of context: Computer systems that adapt to, and learn from, context. In IBM Systems Journal, 39, 3/4, 617–632.2000.

[12] Dey, A.K. Understanding and using context. In Personal and Ubiquitous Computing, 5, 1, 4–7.2001.

[13] Zhang Daqiang, Laurence T Y, Huang Hongyu. Searching in Internet of things: vision and challenges. In Proceedings of the 9th IEEE International Symposium on Parallel and Distributed Processing with Applications, Busan, South Korea, May 2011. Washington, DC, USA: IEEE Computer Society, 2011: 201-206.

[14] Bianchini D, Antonellis V D, Melchiori M, et al. Lightweight ontology-based service search in mobile environments. In Proceedings of the 17th International Conference on Database and Expert Systems Applications (DEXA 06), Krakow, Poland, Sep 2006. Washington, DC, USA: IEEE Computer Society: 359-364. 2006.

[15] Zhang Ying, Qu Youli, Huang Houkuan, et al. An ontology and peer-to-peer based data and service unified search system. In Expert Systems with Applications, 36(3): 5436-5444. 2009.

[16] Mokhtar S B, Kaul A, Georgantas N, et al. Efficient semantic service search in pervasive computing environments.In Proceedings of the ACM/IFIP/USENIX 2006 International Conference on Middleware (Middleware 2006), Melbourne, Australia, Nov 2006. New York, NY, USA: Springer-Verlag, 240-259. 2006.

[17] Wuhui Chen, Incheon Paik, and Patrick C.K Hung. Constructing a Global Social Service Network for Better Quality of Web Service discovery. In IEEE Trans. Services Computing 8(2): 284-298 .2015.

[18] Iury Arau´jo(B), Mikaelly F. Pedrosa, Jessica Castro, Eudisley G. dos Anjos, and Fernando Matos. Service Search Based on Social Profiles of Objects in a Social IoT Network. In 19th International Conference, Saint Petersburg, Russia, July 1-4, 2019.

[19] Guo Chen, Jiwei Huang∗, Bo Cheng, Junliang Chen. A Social Network based Approach for IoT Device Management and Service Composition. In IEEE World Congress on Services, New York City, NY, USA, June 27 - July 2, 2015.

[20] Michele Nitti, Virginia Pilloni and Daniele D. Giusto. Searching the Social Internet of Things by Exploiting Object Similarity. In Multi-Path Relationship Preserved Social Network Embedding IEEE Access 2019

[21] Osama Alsaryrah , Ibrahim Mashal , and *Tein-Yaw Chung .Energy-Aware Services Composition for Internet of Things. In 4th IEEE World Forum on Internet of Things, WF-IoT, Singapore, February 5-8, 2018.

[22] Issarny, V., Bouloukakis, G., Georgantas, N., Billet, B. Revisiting service-oriented architecture for the IoT: a middleware perspective. In ICSOC 2016. LNCS, vol. 9936, pp. 3–17. Springer, Cham .2016.

[23] Sara Samir, Amany M. Sarhan, Alsayed Algergawy. Context-Based Web Service discovery framework with QoS Considerations. In 11th International Conference on Research Challenges in Information Science (RCIS).2017.

[24] Ahmed Belkhirat, Abdelkader Belkhir, Abdelghani Bouras. A New Similarity Measure for the Profiles Management. In 13th International Conference on Modelling and Simulation 2011.

[25] Liwei Liu, Nikolay Mehandjiev, Dong-Ling Xu. Context Similarity Metric for Multidimensional Service Recommendation. In International Journal of Electronic Commerce.2015.

[26] Sameh Ben Fredj, Mathieu Boussard, Daniel Kofman, Ludovic Noirie . Efficient semantic-based IoT service search mechanism for dynamic

environments. In IEEE 25th International Symposium on Personal, Indoor and Mobile Radio Communications.2014.

[27] Corson, S., Macker, J. Mobile ad hoc networking (MANET): routing protocol performance issues and evaluation considerations. In No. RFC 2501 .1998.

[28] Kyeong-Deok Baek, In-Young Ko. Spatially Cohesive Service Discovery and Dynamic Service Handover for Distributed IoT Environments. In Web Engineering - 17th International Conference, ICWE 2017, June 5-8, 2017.

[29] Groba, C., Clarke, S. Opportunistic service composition in dynamic ad hoc environments. In IEEE Trans. Serv. Comput. 7(4), 642–653 .2014.

[30] Abdelmuttlib Ibrahim Abdallaahmed, Abdullah Gani, Siti Hafizah Ab Hamid, Abdelzahir Abdelmaboud, Hassan Jamil Syed, Riyaz Ahamed Ariyaluran Habeeb, Ihsan Ali. Service Management for IoT: Requirements, Taxonomy, Recent Advances and Open Research Challenges. In IEEE Access 7: 155472-155488 .2019.

[31] Iury Araújo, Mikaelly F. Pedrosa, Jessica Castro, Eudisley G. dos Anjos, Fernando Matos. Service Discovery Based on Social Profiles of Objects in a Social IoT Network. In 19th International Conference, Saint Petersburg, Russia, July 1-4, 2019.

[32] WEI Qiang, JIN Zhi, LI Ge, LI Lixing. Preliminary Study of Service Discovery in Internet of Things: Feasibility and Lim- itation of SOA. In Journal of Frontiers of Computer Science and Technology 1673-9418.2013.