IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON INTELLIGENT SYSTEMS FOR THE INTERNET OF THINGS

Received August 16, 2017, accepted September 10, 2017, date of publication September 26, 2017,
date of current version October 12, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2756446

A Decentralized Trustworthy Context and QoS-Aware Service Discovery Framework for the Internet of Things

JUAN LI[1], (Member, IEEE), YAN BAI[2], NAZIA ZAMAN[1], AND VICTOR C. M. LEUNG[3], (Fellow, IEEE)

[1]Department of Computer Science, North Dakota State University, Fargo, ND 58108, USA
[2]Institute of Technology, University of Washington Tacoma, Tacoma, WA 98402, USA
[3]Department of Electrical and Computer Engineering, The University of British Columbia, Vancouver, BC V6T 1Z4, Canada

Corresponding author: Juan Li (j.li@ndsu.edu)

ABSTRACT The Internet of Things envisions a multitude of heterogeneous objects and interactions with physical environments. It has provided a promising opportunity to build powerful services and applications by leveraging the growing ubiquity of smart objects connected to the Internet. In this environment, it is challenging to locate desirable services due to the considerable diversity, large scale, dynamic behavior, and geographical distribution of the services provided by physical objects. In this paper, we propose a decentralized semantics-based service discovery framework, which can effectively locate trustworthy services based on requester's quality of service demands and changing context requirements. The proposed service discovery system is built upon fully distributed peer-to-peer architectures and includes two discovery approaches which are scalable, robust, and trust assured. Moreover, an efficient trust propagation mechanism is proposed, which seamlessly integrates with the proposed service discovery system without adding additional infrastructure overhead. Comprehensive simulation studies have demonstrated the effectiveness of the proposed framework.

INDEX TERMS Internet of Things, discovery, context-aware, semantics, QoS, trust.

I. INTRODUCTION

Internet of Things (IoT) connect uniquely identifiable embedded computing devices within the existing Internet infrastructure to offer advanced connectivity of devices, systems, and services. IoT goes beyond machine-to-machine communications and covers a variety of protocols, domains, and applications [1]. The interconnection of these embedded devices is expected to usher in automation in nearly all fields, covering wide range of applications such as healthcare, utilities, and transportation [2].

In order to effectively use services and data generated in the IoT, search and discovery mechanisms are crucial. These mechanisms should support locating resources and services related to an entity of interest in the physical world. IoT services are enabled by devices located in widely distributed, and heterogeneous information systems. Designing an effective and efficient discovery mechanism faces many new challenges and requirements. In this paper, we focus on five major performance requirements that aim to address the challenges standing out for service discovery in the IoT.

A. SCALABILITY

Real-world services are provided by IoT devices in the physical world. It is anticipated that the number of embedded devices connected to the Internet will be orders of magnitude larger than the number of computers connected to the Internet today [3]. Consequently, data of real world objects and events are available globally and in vast amounts. This challenge requires the discovery mechanism to be scalable with respect to large number of objects.

B. SEMANTICS-AWARENESS

The IoT involves a large number of different business, applications and heterogeneous devices. IoT services may reside in different layers of the enterprise, e.g., different operational units, different kinds of information networks. Representations of IoT-related services are much more heterogeneous than those for traditional services in the Internet. Therefore, another major problem in the IoT is inherent heterogeneity with respect to the nature of components, standards, data formats, protocols, etc. This challenge requires the discovery

mechanism to enable interoperability among the various IoT service providers and requesters.

C. CONTEXT AND QoS-AWARENESS

Due to the mobility of physical entities and devices and other dynamics (e.g., users enter in and exist from smart environment, objects change their positions) the relations between IoT services, IoT devices, and the physical environments may also change over time. Most IoT discovery services should provide near real-time states of things in the physical world, which is quite different from traditional web services that are entirely virtual entities encapsulating enterprises business logics, and relatively static. In addition, as there are multiple service providers offer similar services, it is important for a service requester to select the best one among many of the providers based on the service's QoS.

D. LIGHTWEIGHT

Real-world services are deployed in resource-constrained devices, e.g., with limited power, computing, communication and storage capabilities. This requires significant simplification, optimization, and adaptation of existing tools and standards [4].

E. TRUST AND SECURITY

In an open IoT network, any IoT device can publish services. Without knowing the trustworthiness of a service provider, a consumer may select a poor quality, time-consuming, expensive, or even harmful service. Therefore, trust and reputation mechanisms are needed to help service consumers to distinguish good services from bad ones. Moreover, IoT services are deployed on devices of their respective providers, located in different administrative domains with various policies, separated by firewalls and virtual private networks. The interaction of things and the use of resources from different providers are subject to new security challenges [5].

To the best of our knowledge, there is no existing work that meets all of the aforementioned requirements. To support effective service discovery in IoTs, we propose a new discovery framework that addresses all of the five challenging requirements. The main contributions of this paper include the followings:

1. We developed a light-weight semantic model that seamlessly integrates with the discovery framework to support context and QoS-awareness.
2. We proposed a fully decentralized discovery scheme that utilizes social trust to enable scalable trust-based service discovery.
3. We proposed another decentralized discovery scheme that utilizes a structured P2P network to enable trust-assured locality-preserving discovery scheme.
   Working together or separately, the two discovery schemes (2 and 3) can provide trust-assured, scalable and robust discovery service.
4. We designed an efficient trust propagation mechanism which seamlessly integrates with our service discovery system without adding additional infrastructure overhead.

The rest of the paper is organized as follows. Section II surveys the related work. Section III describes the ontology model used to present service descriptions including context and QoS. Section IV details the decentralized discovery mechanisms. Section V presents the evaluation of the proposed framework. Concluding remarks are provided in VI.

II. RELATED WORK

A. SERVICE DISCOVERY IN IoT

The development of the Electronic Product Code (EPC) to support the spread use of RFID in supply chain management (SCM) by Auto-ID [32] and EPCglobal [8] is one of the most important efforts in the IoT. Several research projects have implemented the discovery service for the EPC network. The first implementation was a Directory Look-up approach [34]. When an EPC event is first stored in a company's EPC information service (EPCIS), a discovery service is notified, and related information is stored in the repository. Users can query the discovery service with an EPC. Related EPCIS URLs will be supplied to the user.

The BRIDGE project [35] provides high-level descriptions and analysis of a number of approaches for implementing discovery services. In this project [35], eight discovery service approaches were evaluated and four were identified as promising candidates for large-scale discovery services. The first approach, Directory of Resources, is similar to the aforementioned Directory Look-up approach [34]. The second approach is called Notification of Resource, in which clients subscribe to the discovery service for their interested EPC numbers. When receiving notifications of EPCIS events from an EPCIS, the discovery service will search for a match of the event with its subscriptions. Once a match is found, the subscriber will be informed. These two approaches are also called as Directory Service (DS) approaches. The other two approaches, Nonfiction of Clients and Query Propagation, are referred to Query Relay (QR) approach. In the Nonfiction of Clients approach, EPCIS servers notify the discovery service for every new EPC about which they have information. Once a client shows interest in a particular EPC the discovery server will notify all relevant EPCIS servers. The servers will then send an availability notification to the interested clients. The Query Propagation approach is similar to the Nonfiction of Clients approach except the information is sent to a client directly by the EPCIS server. The BRIDGE project leaves the choice of the implementation of the proposed approaches open. LDAP (Lightweight Directory Access Protocol), DNS, P2P networks (e.g., distributed hash table) or search engines can be potential implementation candidates. A prototype based on LDAP was implemented as a reference model.

An Aggregating Discovery Service (ADS) was proposed by Lorenz *et al.* [33]. In contrast to the separation of discovery and data access originally envisioned by EPCglobal, this approach integrates actual data and their aggregation into the discovery service layer. It forwards client queries

to relevant EPCIS servers, and aggregates their responses as the result to reply to the client. This approach combines the advantage of the Directory Service and the Query Relay approaches. It shifts the complexity of query parallelization and the aggregation of EPCIS responses from the client to the discovery service.

Recently, the use of P2P systems to implement scalable and robust distributed service discovery in IoT has been investigated [36]–[38]. To reduce the cost of broadcasting in the P2P network, various mechanisms have been proposed. For example, one project employed multicasting to replace broadcasting [39]; another project performed selective forwarding to limit the number of search hops [40]; in another work, probabilistic forwarding of queries was proposed to replace flooding [41]. The aforementioned P2P networks are unstructured networks, which do not impose a particular structure on the overlay network by design, but rather are formed by nodes that randomly connecting to each other. On the other hand, there are IoT discovery frameworks based on structured P2P networks. For example, structured P2P architecture has been used for information exchange and resource discovery among participants of a supply chain [7]. Lorenz *et al.* proposed a structured P2P discovery service for the EPCglobal network [42]. It offers item-level track and trace capabilities along the whole supply chain even when items are not directly visible. Most of the structured P2P systems use distributed hash tables (DHTs) as their underlining communication structure.

As pointed out by Paganelli *et al.*, most of the aforementioned DHT-based discovery frameworks support queries by providing an object identifier (typically the EPC code) as input, but they do not support more flexible query schemes based on object attributes, though these types of information queries could become very important in the future IoT [43]. To support complex queries, in particular, multi-attribute and range queries, Paganelliet *et al*. proposed a layered functional architecture over DHTs [43]. The proposed architecture used a Space Filling Curves (SFC) linearization technique for mapping a multidimensional domain into a one-dimensional one. It used a Prefix Hash Tree (PHT) search structure leveraging on a generic DHT interface. Another system, ERQ [44], used the similar approach with Balanced Kautz (BK) tree to map the m-dimensional data space onto DHT nodes, and then adopted a distributed algorithm to process range queries. However, existing DHT approaches cannot control where the service metadata is placed, which is a disadvantage for system security.

B. CONTEXT-AWARE SERVICE DISCOVERY

Modeling context and selecting services based on the right context is vital for IoT service discovery. As defined by Saadon *et al.* [45], context-aware discovery is to make use of the context information to discover the most appropriate services to the client against the service offered. Many systems have applied context-aware techniques to discover Web services. Yang *et al.* [46] proposed an event-driven rule based context-aware services discovery approach. They consider two types of context, namely the service requesters' context and the Web services' context. They provide an ontology-based context model to represent context and utilize the context to assist service discovery. In another study [47], Al-Masri *et al*. consider device capabilities and user preferences as the context. By matching with the device profile information, relevant services are ranked based on the degree of relevance between the service advertisement and clients' device. Some other approaches extend the service discovery by exploiting the user profile, device profile and service profile properties in the discovery process [13], [48]. A RESTful Web services-based Service Discovery protocol to IoT applications was proposed [49]. This protocol provides a service selection technique based on the context information of user and services. Furthermore, it employs a demand-based adaptive timer and caching mechanism to reduce the communication overhead.

Generally, context modeling can be classified as six categories: key-value, markup, graphical, object-based, logic-based, and ontology-based [17]. Our context modeling belongs to the ontology-based category, we generalize and reuse some of the existing ontologies [15], [18], [25] and model the most fundamental context. In Section III, when we present our ontological modeling and reasoning schemes, we also provide a detailed description of the most related existing research in these fields. Therefore, we do not repeatedly explain the details of existing work in this section.

C. QoS-AWARE SERVICE DISCOVERY

QoS is non-functional properties of services. It refers to how well a service performs its behavior to the customer. QoS-aware service discovery assists users to discover the most appropriate services based on QoS properties.

To ensure discovering quality IoT services, it is necessary to model and measure their QoS properties. W3C provides guidelines to model QoS requirements of Web services, and defines 13 possible generic QoS metrics, including Performance, Reliability, Scalability, Capacity, Robustness, and so on [22]. Other works [19]–[21], also study QoS assessment models and evaluation indicators.

Yu *et al.* adapt the concept of QoS in web services and introduce the concept of Quality of Web Service (QoWS) [50]. They divide the QoWS into two categories, namely, runtime quality and business quality. Runtime quality represents the response time, reliability, availability, accessibility and integrity. Business quality refers to the cost, reputation, and regulatory. Al-Masri introduces Web Service Relevancy Function (WsRF) for measuring the relevancy of requested service with QoWS properties [51]. They adapted quality parameters such as response time, throughput, availability, accessibility, interoperability analysis and cost. Canfora *et al.* modeled QoS-aware service composition as an optimization problem [52]. They adopt genetic algorithms to this aim. They claim that genetic algorithms are scalable, and suitable to handle generic QoS attributes.

Zhang and Tongguang propose a fast QoS-aware web service selection approach [53]. It adopts a particle swarm optimization algorithm to select the most appropriate service based on user's QoS requirement. Another approach [54] proposes the Evolutionary Computing using Particle Swarm Optimization (PSO) as an optimization approach to discover the most relevant services with QoWS properties. The considered QoWS are execution cost, execution time, availability, successful rate, reputation and frequency.

Many research works have used semantic web-based approach to match QoS properties. For example, one work develops an ontology language that complements OWL-S and uses vector space model to calculate semantic distance followed by an ontology hierarchical matching [55]. Another similar approach by Zhu *et al.* [56] utilizes OWL-S as a service description language and designs a prototype of service discovery which adopts Three-layer Pervasive Semantic Service Matching Algorithm.

D. CONTEXT AND QoS-AWARE SERVICE DISCOVERY

Mokhtar *et al.* [25] present a system – EASY to support Semantics-based context- and QoS-aware service discovery for pervasive computing. EASY provides a service description language Easy-L for describing both functional and nonfunctional service properties and additionally providing a formalism for conformance between service capabilities. This has been used to develop a QoS-enabled and context-aware semantic discovery system. ConQo is another Web Service discovery architecture that supports QoS-aware and context-aware at the same time [57]. ConQo utilizes Web Service Modeling Ontology (WSMO) and introduces WSMO-QoS and WSMO-Context. Niazi and Mahmoud present an ontology-based system for discovering mobile services with respect to user preferences and device capabilities [24]. They provide a semantically enriched profile-based service description and integrate it with the Delivery Context Ontology.

E. TRUST MANAGEMENT IN IoT

Research on trust management in IoT are gaining an increasing momentum. Bao and Chen focused on human-carried or human-related IoT object [9], [10]. They proposed a trust management protocol taking into account social trust and QoS trust metrics. Direct observations and indirect recommendations are applied to update trust values. Three trust parameters including honesty, cooperativeness, and community-interest are used in trust evaluation. A similar work to evaluate the trustworthiness of things is proposed [58]. It is based on a social Internet of Things (SIoT) paradigm, which integrates social networking concepts with IoT. It is assumed that things are capable of establishing social relationships in an autonomous way with respect to their owners. In the SIoT network, each node computes the trust of its friends based on its own experience and the opinion of common friends. A feedback system is employed. The credibility and centrality of the IoT nodes are applied to evaluate their trust level. Chen, *et al.* [11] proposed a trust management model based on fuzzy reputation for IoT with a focus on specific IoT with wireless sensors. They use QoS properties such as packet forwarding/delivery ratio and energy consumption as trust metrics. To handle the trust issue in decentralized and dynamic scenarios in IoTs, Mahalle *et al.* [59] developed a Fuzzy approach to the Trust Based Access Control (FTBAC). The approach applies the linguistic information of devices to address access control in the IoT. They calculated trust values based on experience, knowledge and recommendation by capturing their corresponding vague values. Different from the aforementioned work on service selection and trust management, we aim to look into how trust information is collected and distributed in IoT. To the best of our knowledge, existing work mainly focuses on trust management, rather than trust information collection and distribution.

III. SEMANTIC SERVICE MODEL

The open and dynamic IoT environments call for system's awareness of context and QoS. However, it is hardly to achieve a single common syntactic agreement between various IoT service providers and requesters. Therefore, it is important for the IoT service discovery system to express and process the semantics of service descriptions and requests including context and QoS description. Semantic Web technologies, in particular ontology, can represent resources in machine-understandable format. Ontology standards support inference mechanisms that can be used to enhance rule and policy reasoning in service discovery. A large number of efforts (e.g., [13]–[15]) have been conducted in service description using semantic web technologies in various pervasive and mobile computing environments. However, applying semantic web technologies for service discovery in IoT faces a major problem: the underlying semantic reasoning is computational costly for IoT entities. Most existing research in semantic service discovery in IoT [13], [14], [16] did not address this issue. Therefore, in this project, we need to simplify the costly semantic reasoning to make it applicable in resource limited IoT.

Our objective is not to propose yet another semantic web model, but to apply, customize, simplify and optimize existing semantic models and methods to support context- and QoS-aware service discovery in IoT environments.

A. ONTOLOGY REPRESENTATION

Different ontologies for context (e.g., [13], [15], [17], [18]) and QoS (e.g., [19]–[22]) have been investigated in the past. Instead of reinventing the wheel, we have reused, integrated and revised existing related context and QoS ontologies, and make them customizable based on specific IoT application requirements. In particular, for context ontology, we follow our previously defined ontologies [15]; for QoS ontology, we follow the W3C QoS requirements guidelines for web services [22]. We model the most fundamental context and QoS concepts, i.e., a set of upper-level entities, and provide

flexible extensibility to add specific concepts in different application domains (e.g., home, office or vehicle).

Inspired by the SIoT paradigm proposed by Atzori *et al.*, [23], we introduce one special context to model important social relationships between IoT devices. As shown in Fig. 1, objects belonging to the same owner may establish a 'co-ownership' relationship; objects located in the same place may create a 'co-location' relationship; objects collaborating to provide a common IoT task may establish a 'co-work' relationship; an 'encounter' relationship may be created between objects come into contact when their owners come in touch with each other. Through these social relationships, the discovery system can crawl and discover IoT services following different levels of trustworthiness between IoT entities.

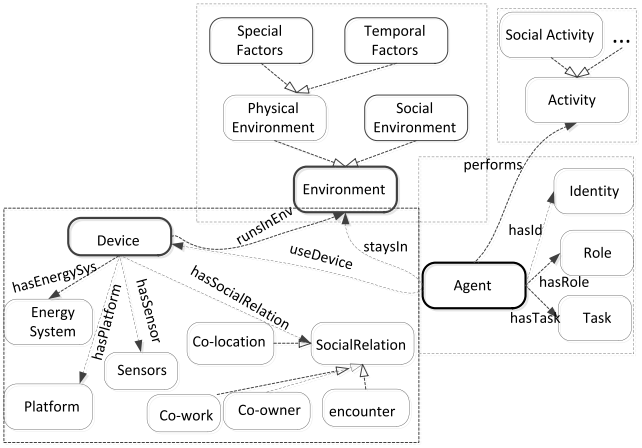

FIGURE 1. A snippet of high-level context ontology.

Fig. 1 shows the high-level context ontology used in our framework. It is categorized into three distinctive but related themes: (a) ontologies about physical device, (b) ontologies about agents (both human and software agents), (c) ontologies about the environment context of the agents. We emphasize on the social relationships between devices. Fig. 2 shows part of the defined high-level QoS ontology. It consists of QoS metric (e.g., reliability and availability), QoS description (e.g. from points of contact or certificate) and QoS role (e.g., providers vs. consumers). With the context and QoS ontology defined, we can effectively model the context and QoS information of the services and requests. In Section III.B, we present how to use the ontology model to filter appropriate services which satisfy the context and QoS requirements.

B. ONTOLOGICAL SERVICE MATCHING

Based on the service ontology defined in Section III.A, we can effectively match services in terms of their functional and non-functional properties.

Lots of research work (e.g., [14], [18], [24]) has been conducted in matching semantic Web services. However, most proposed technologies and tools for semantic reasoning

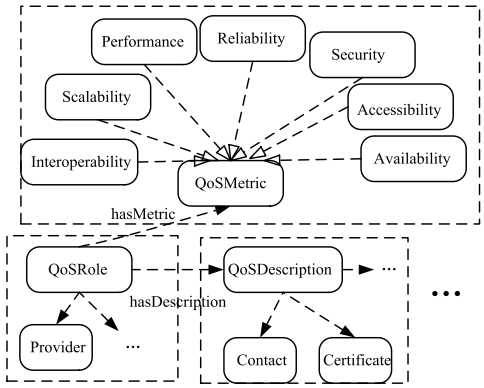

FIGURE 2. A snippet of high-level QoS ontology.

and matchmaking are computationally expensive. Therefore, we cannot use them in IoT environments in which devices have only limited computational resources. We follow the idea of signature matching [26], which is a simple method to identify the *subsumption* relationships between concepts describing web services. To further reduce matching cost, we numerically encode ontologies, thus transforming the costly semantic reasoning into a simple numeric computation [25]. We describe our approach as follows:

Both service requests and service advertisements can be presented using inputs and outputs parameters, which are semantically defined using ontology concepts. There are four levels of matching between ontology concepts of service request and service advertisement: (1) *exact*: if the concepts are equivalent, (2) *plug in*: if the advertised concept subsumes the requested one, (3) *subsume*: if the requested concept subsumes the advertised one, and (4) *fail*: if there is no subsumption relation between the two concepts.

The aforementioned four levels of matching can be converted to numerical values to represent various matching degrees between two concepts $c1$, $c2$ of an ontology O. For example, if $c1$ and $c2$ have an *exact* matching relation, their matching degree value would be *1*. If $c1$ and $c2$ have *plug in* or *subsume* matching relation, their matching degree can be set as *1* divided by the number of levels between $c1$ and $c2$. Using this idea, we can measure how well the advertised services match the requested services. Specifically, all the required properties *Req* specified in the requested service description are compared with provided properties of the advertised services *Adv*. Quantitative properties (e.g., price less-than *10*) are numerically compared, whereas qualitative properties (e.g., Operating-System *is-a* Linux) are checked using the aforementioned concept matching. If two or more advertised services equally satisfy the functional properties, we select the service that best satisfy the non-functional properties (e.g., context and QoS).

C. OPTIMIZATION WITH ONTOLOGICAL ENCODING

To reduce the cost of semantic reasoning on ontologies, we adopt the numerical encoding optimization mechanism

proposed by Mokhtar *et al.* [25]. Specifically, ontologies are classified into hierarchies offline. A prime-based encoding technique is then applied to allow for *subsumption* testing of classes in ontologies. This is done by assigning a unique prime number *Pi* to each class *Ci* in the ontology hierarchy. Assigning unique prime numbers to class will ensure the encoding to be conflict-free. Encoding of a class *Ci*, *E(Ci)* is defined as $E(Ci) = \prod_j P_j$, where P_j is the assigned prime number of class *Ci* and its ancestors.

Fig. 3 illustrates the encoding of a multiple inheritance hierarchy including five nodes: *A*, *B*, *C*, *D*, and *E*. In this figure, class *C* inherits prime numbers *2*, *3* from its ancestors, and is assigned *5* as its own number, therefore its encoding is $2*3*5 = 30$. Similarly, *E*'s encoding is the production of *2*, *7*, and *11* obtained from its ancestors and itself. As classes *A* and *D* have no ancestors, their encoding is their assigned number. Based on the encoding, a class *X* is subsumed by a class *Y* if the prime number of class *Y* divides the encoding of class *X*. For example, as shown in Fig. 3, class *C*'s encoding *30* can be divided by *A*'s prime number *2*, and then we can infer that *C* is subsumed by *A*. The next available prime number can be used to incrementally encode a new class, and the least common multiple function of the encoding of its parents is the encoding of the new class.

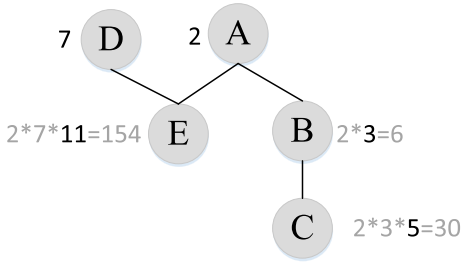

FIGURE 3. Hierarchy encoding using prime numbers.

As can be seen from the example, this encoding can efficiently identify whether two concepts in an ontology are related without performing costly semantic reasoning online.

IV. DECENTRALIZED DISCOVERY INFRASTRUCTURE

In order to efficiently locate trustworthy services, we propose a fully decentralized discovery mechanism based on a P2P communication model. We consider trust in the service discovery process and seamlessly integrate service discovery with trust propagation. The discovery mechanism consists of two models: a social recommendation-based model and a decentralized locality-preserving model. The two models complement each other offering an effective trust-based service discovery solution.

A. SOCIAL RECOMMENDATION-BASED MODEL

Scientific evidences have shown that a large number of individuals tied in a social network can provide far more accurate answers to complex questions than a single individual, or a small group of even knowledgeable individuals [23]. The exploitation of such a principle has been widely investigated in the Internet-related research. Several schemes have been proposed in the current literature, which use social networks to search Internet resources, to route traffic, or to select effective policies for content distribution, e.g., [27], [28]. The intuition behind our discovery model is based on this social networking paradigm in which IoT entities give more trust towards services recommended by their trusted "friends".

Our system utilizes social links between IoT entities to forward service advertisements/recommendations in the IoT network. As presented in Section III.A, our ontological model enables the system to present the trustworthy social relationships (such as 'co-ownership', 'co-location', and 'co-work' relationship). Following the social links, service recommendations can be propagated and stored in an IoT entity based on the "usefulness" of the advertisement to this entity. Intuitively, if a recommendation is from a trusted contact, it is more useful; if a recommendation is semantically more similar to the IoT entity's interest, it is more useful; if the recommender of a recommendation is more confident to the recommendation, it is more useful; finally, if a recommendation is more recent, it is more useful. Based on this intuition, we define a metric, *usefulness score*, to measure the usefulness of an IoT service recommendation.

Let us assume that node *m* is node *n*'s trusted social contact. *m* recommends a service *R* to *n*. The usefulness score of a recommendation *R* with respect to node *n* is determined based on four factors including: (1) the recommended service's relevance to node *n*'s service interest, (2) the trust level that node *n* gives to the recommender *m*, (3) the confidence of the recommender *m* towards this recommendation, and (4) the time stamp of the transaction based on which the recommendation is made.

Definition (Usefulness): The usefulness score of a recommendation *R* (recommended by node *m*) towards an IoT node *n* is defined as:

$$\text{usefulness (R|n)} = \frac{\alpha \text{sim (R, n)} + \beta \text{trust (n, m)} + \gamma \mathit{conf}\ (m, R)}{e^{t_{now} - t_R}} \quad (1)$$

In this definition:

- α, β, and γ are weights for each of the three usefulness factors, where $\alpha + \beta + \gamma = 1$.
- *sim(R, n)* is a function that computes the semantic similarity between the recommendation *R* and node *n*'s interest, which is based on the similarity between ontological concepts of *R* and *n*. The similarity between two concepts C_a and C_b in a given ontology has been defined as:

$$\mathit{ConceptSim}\,(C_a, C_b) = \begin{cases} \dfrac{1}{numoflevelsbetweenC_aandC_b}, & Ca \overset{p}{\Longleftrightarrow} Cb \\ 1, & Ca \overset{E}{\Longleftrightarrow} Cb \end{cases} \quad (2)$$

in which, $Ca \overset{p}{\Longleftrightarrow} Cb$ means that C_a and C_b have plug-in or subsume relation; $Ca \overset{E}{\Longleftrightarrow} Cb$ means that C_a and C_b exactly match. Based on the definition of concept similarity, the semantic similarity between a recommendation R and a node n's interest is defined as:

$$sim(R, n) = \begin{cases} \frac{\sum_{i=1}^{p} ConceptSim(R_i, n_i)}{p}, & p > 0 \\ 0, & p = 0 \end{cases} \quad (3)$$

where p is the number of concept matching pairs between R and n.

- *trust* (n, m) is the trust level (between 0 to 1) n gives to its social contact m who recommends R to n. The trust level is identified based on the social relationship between n and m as defined in Fig. 1 in Section III.A. In our experiment, based on the social closeness, we assign the following trust values to the major social relationships: *1* to *ownership* relationship, *0.9* to *co-work* relationship, *0.8* to *co-location* relationship, and *0.7* to *social* relationship. Applications can vary the trust values to fit for their environments and needs.
- *conf* (m, R) is the confidence value that the recommender m gives to this recommendation R. It is based on trust relationship between node m to the recommendation sources p, who directly consumed the recommended service R. Once node n decides to accept recommendation R, n will update its confidence to R, *conf (n, R)*, which is recursively computed as:

$$\text{conf (n, R)} = \text{conf (n, R)} + \text{conf (m, R)} \times \text{trust (n, m)} \quad (4)$$

  Initially, *conf (n, R)* is 0 when node n does not know R. According to Jøsang *et al.* [29], trust and confidence is not strictly transitive. Therefore, during the recommendation forwarding process, the confidence of the recommendation will be updated at each virtual hop. Intuitively, the closer the distance between the direct consumer and node n is, the more useful the recommendation is to n.
- t_{now} is the current time and t_R is the time when the recommended service was consumed. The more recent the recommendation is, the more useful it is.

Each IoT node will maintain a set of service recommendations obtained from their trusted social contacts. For scalability, a node only keeps a limited set of services which is most useful from its own perspective.

A service recommendation R has a Uniform Resource Identifier (URI) as its identification, it includes information about the recommended service *R.s*, the service consumption/transaction timestamp, and the recommender's confidence to the recommended service. Service *R.s* contains the metadata of the service, such as service provider and QoS description. Upon receiving a recommendation R, a node n needs to decide whether or not to store and forward R, and how to store R based on how ''useful'' the recommendation is.

Algorithm 1 illustrates the procedure of recommendation store and forward using the usefulness score. At first, the algorithm tries to figure out if the recommendation R has been received before. If so, the node can simply ignore the recommendation (line 1 to 3). Node n retrieves the recommended service *R.s* from recommendation R (line 4). Then n searches the recommended service *R.s* in its recommendation storage S. If *R.s* is found in its recommendation storage, which means that the service has been recommended to the node before (from a different sources and/or different transactions), node n will store the recommendation $R's$ ID and update its confidence to service *R.s*. (line 5, 6). The confidence updating is based on equation (4) defined earlier in this section. If the service is not found in S (*R.s* is a new service), node n needs to determine whether there is enough storage space to save *R.s*: if there's space, n saves *R.s* into S and update the minimum of usefulness score of S if necessary (line 9 -13). Otherwise, n will find a replacement for the recommendation based on the usefulness score. A recommendation with a low usefulness score is always replaced by a recommendation with a higher score (line 15 to 18). Next, node n sends R to its trusted friends until the Time to Live (TTL) is reached (line 21 to 26).

An IoT node only stores service recommendations that are most useful to itself. Recommendations will be propagated in the IoT ''social communities'', and nodes will exchange and accumulate recommendations. As can be seen from the experimental results in Section V, service discovery can be effectively performed using recommendations stored locally. This social-recommendation-based discovery adds social trust to control the propagation of service data. Therefore, it is more secure compared with other P2P-based discovery approaches.

B. LOCALITY-PRESERVING MODEL

In this section, we present another discovery approach that provides an alternative discovery mechanism to users. We adopt our previously proposed SkipNet-based overlay structure [15] to enable scalable and secure distributed discovery services. As mentioned, P2P architecture, especially distributed hash tables (DHTs), has been used for service discovery in many IoT applications. However, DHTs have an inherent shortcoming: they cannot control where data or metadata is stored. This causes security risks. Our SkipNet-based discovery system can effectively address this problem.

SkipNet [30] is a scalable overlay network that provides controlled data placement and guarantees routing locality by organizing data primarily by string names. It follows the idea of Skip List that maintains a list of nodes and pointers that ''skip'' over varying number of nodes. In a SkipNet overlay of n nodes, every node maintains connections to *O(log n)* other nodes, and any node can send a message to any other node while traversing only *O(log n)* intermediate hops.

We utilize SkipNet's locality-preserving property to implement a communication overlay for service discovery. In particular, IoT nodes (both service providers and service consumers) form a SkipNet overlay topology, in which nodes

Algorithm 1 Recommendation Store and Forward Algorithm

/∗After receiving a service recommendation *R* from a neighbor *m*, a node *n* uses this algorithm to determine whether or not and how to store and forward the recommendation ∗/

```
if R has been received before then
    return
end if
extract the recommended service R.s from R
if R.s is in n's recommendation storage S then
    n adds R.id into S and updates the confidence of R.s
else
    compute the usefulness of R: usefulness(R|n)
    if there's room in Sthen
        store R.s in S (including the confidence of R.s)
        if usefulness(R|n)< minUsefulness of S then
            minUsefulness= usefulness(R|n)
        endif
    else
        if usefulness(R|n)> minUsefulness of S then
            replace the record of minUsefulness with R.s
            update minUsefulness
        end if
    end if
end if
if R.TTL >0 then
    R.TTL–
    for    eachneighbor i of n (except m) do
        forward R to i
    end for
end if
```

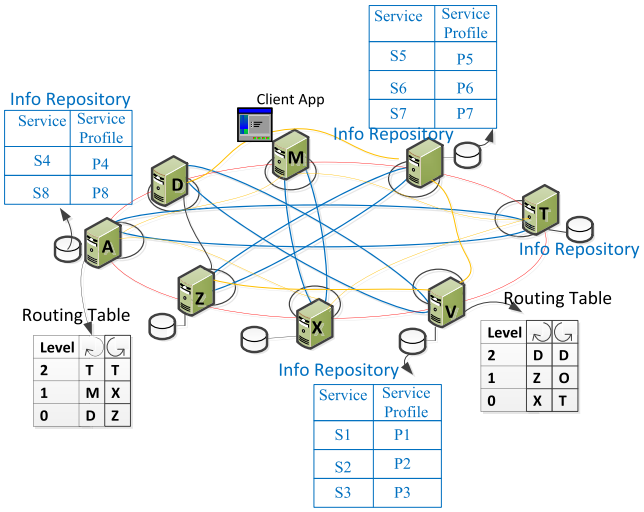

FIGURE 4. Locality-preserving discovery overlay structure.

are organized into several doubly-linked lists. Each node is a member of several of these lists, but not all of them. Each of these lists is sorted using the nodes' string identifiers. Every pair of nodes that are adjacent in these lists forms a bidirectional network connection. As shown in the example in Fig. 4, eight peers are simultaneously interconnected into four levels. The red lines connect neighboring peers. The yellow lines connect peers that are two hops away. As a result, the overlay peers are divided into two disjointed rings. Similarly, nodes connected by blue lines form four disjointed rings, and so forth.

Every overlay node acts as a rendezvous site for other nodes to publish their services' metadata. For this purpose, as shown in Fig. 4, every node in the discovery overlay provides an information repository. Each peer exposes a set of discovery APIs that can be invoked by client applications to search for desirable services. At the same time, each node also maintains a routing table to enable efficient query routing. The routing table records links to neighbors at different levels. For example, in Node *A*'s routing table, the bottom row records node *A*'s level *0* neighbor (i.e., neighbors in the red line circle); the next row includes *A*'s level *1* neighbor (i.e., neighbors in the yellow line circle); the top row contains *A*'s level *2* neighbors (i.e., blue line neighbors). Using these neighboring links at different level, routing can be efficiently achieved by "skipping" various number of nodes and traversing only *O(log n)* intermediate nodes.

Nodes in the overlay network together constitute a repository that provides discovery services. To publish a service record, the service provider will register the service metadata to a corresponding information repository. The service metadata includes the service description, the context and QoS description represented with ontology, the identification of the related objects, and the URI of the service.

This overlay network supports two message routing algorithms: string-based name ID routing and numeric ID routing. We incorporate service provider's Domain Name System (DNS) records (i.e., domain name) into the service registration code. For example, to register a service with identity *service0001* in the organization *sample.domain.org*, we name the service as *sample.domain.org/service0001*, which indicates that it is associated with and published by the provider "*sample.domain.org*". Then we can explicitly place data on specific overlay nodes or across nodes within an organization. Sub-organizations can be presented by adding their sub-organization names. For example, branch *subOrg1* in organization *sample.domain.org* can be represented as *sample.domain.org/subOrg1*. This yields path locality for organizations in which all services share a single DNS suffix (and hence share a single name ID prefix). To support SkipNet's Constraint Load Balancing (CLB), we divide a data object's name into two parts, namely the domain part over which load balancing is performed and the name part that is used as input to the DHT's hash function.

Peers are ordered by their name IDs along each ring of SkipNet and a routing message will not be forwarded past its destination, therefore all peers encountered during the routing process have name IDs between the source and the destination. If the source peer shares a common prefix with the destination peer, all peers traversed by the routing message

will have name IDs that share the same prefix as well. This guarantees that message traffic routed between two overlay nodes within the same organization stays within it, i.e., routing path locality.

The discovery system always tries to locate local services first. The property of content and path localities of the discovery mechanism described in the above is very important for IoT. Content locality can explicitly specify where the metadata is placed; while the path locality guarantees that a routing message between any two nodes within a single administrative domain always stays within the administrative domain's boundary. This isolation will effectively avoid malicious attacks by foreign entities outside the administrative domain, for example denial of service attacks, Sybil attack, eavesdropping, and network traffic analysis.

V. EVALUATION

We have conducted a set of simulation experiments to measure the effectiveness of our service discovery mechanisms. In order to evaluate the performance of the discovery framework, we implement a message-level, event-driven, discrete-time simulator. In these experiments, we generate the network topology using a popular topology generator GT-ITM [31]. We use GT-ITM to generate the Transit-Stub (TS) network model, which reproduces the hierarchical structure of the topology of the Internet. In this topology, a stub domain corresponds to an Autonomous System (AS) which carries only traffic that originates or terminates in the domain. While transit domains represent wide or metropolitan-area networks (WANs or MANs). To simulate the dynamics of an IoT network, in any simulation time unit, a node may die or leave with a probability of 5%. In the experiments, each node maintains certain number of trusted social contacts. All of the nodes can be a service provider and a service consumer. There is a global hierarchical ontology defined for IoT services, in which there are 10 service categories, and each category has 1 to 5 sub-categories. Each service belongs to a service category. At any simulation time unit, each node has a possibility of 10% to generate a service query, which is most possibly falls into the node's interests (with a possibility of 90%).

We simulate two types of nodes in the network: good nodes and bad nodes. Good nodes provide the services with QoS which they advertised. The bad nodes advertise bogus QoS. Bad nodes are not always bad. They have a small amount of time of being "honest" in their life time. In the experiment, we simply assume that a node consumes the service after it gets results from a query. Good nodes recommend good services which they have consumed; a bad node recommends bad services. A good node's close social contacts (such as nodes with *co-ownership* relationship or nodes in the same AS organizations) are also good nodes. Each experiment is performed ten times with different random initialization and the average results are presented. The various simulation parameters and their default values are listed in Table 1.

TABLE 1. Parameters used in the simulations.

parameter	range (default)
network size	128-8192 (1000)
maximum no. of trusted contacts	3-18 (8)
node churn possibility per time slice	5%
node query possibility per time slice	10%
service recommendation TTL	5
no. of semantic entities per service profile	3-15
no. of semantic entities per service request	1-4 (2)
recommendation storage size in no. of entities	128-512 (256)
type of service domains	10
services provided per node	1-3
semantic similarity threshold	0.6
% of bad nodes	0-50% (10%)
% of "good" behavior of bad nodes	10%-30% (10%)
α in usefulness computation	0-1 (0.4)
β in usefulness computation	0-1 (0.3)
γ in usefulness computation	0-1 (0.3)

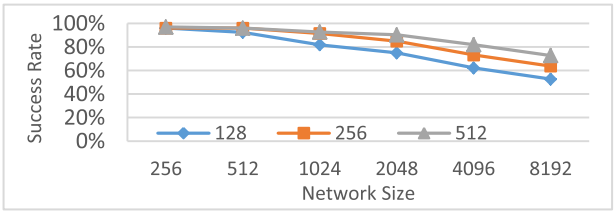

FIGURE 5. Success rate vs. network size for social recommendation-based service discovery (with 3 different recommendation storage size).

In the first part of the experiment, we evaluate the performance of the trust-based social recommendation discovery mechanism. The system performance is evaluated in terms of the query success rate, which is defined as:

$$SuccessRate = \frac{|SatisfiedRequests|}{|TotalRequests|} \tag{5}$$

A service request is satisfied if the service profile matches the request in functional and non-functional properties; more importantly, the service is provided by a "good" service provider.

In the discovery mechanism, each node maintains a fix-sized recommendation storage. Periodically nodes propagate and exchange their trusted services info stored in the recommendation storage with their trusted social contacts. Note that in the following figures, the X-axis is in exponential growth.

Fig. 5 plots query success rates under different network sizes. The three lines represent three different recommendation storage sizes: 128 entities, 256 entities, and 512 entities. As shown in the figure, social recommendation-based discovery mechanism achieves high success rate even in a very large network (i.e., 8192 number of nodes). The results also indicate that it only requires a limited-size storage size to achieve a reasonable success rate.

In order to evaluate the performance of the "usefulness" function in the social recommendation-based discovery mechanism, experiments with different recommendation store and forward strategies were conducted.

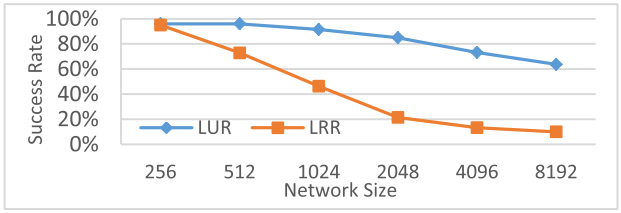

FIGURE 6. Success rate vs. network size for social recommendation-based service discovery with different store-and-forward strategy.

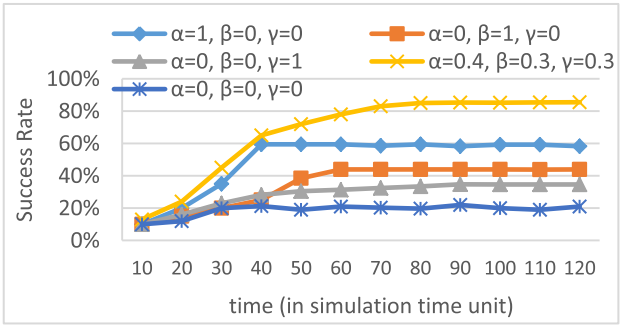

FIGURE 7. Success rate vs. simulation time for social recommendation-based service discovery (with 5 different usefulness parameter combinations).

Fig. 6 displays the results. In Fig. 6, the first store and forward strategy "Least Useful Replacement" (LUR) uses the "usefulness" function defined in Definition 1 to decide which items to discard to make room for the new recommendations; the second one "Least Recently Recommended" (LRR) simply stores and forwards every new coming recommendation, and substitutes a newly arrived recommendation with the oldest record in the recommendation list when space is used up. Each node can store up to 256 recommendation records. As shown in Fig. 6, compared with LRR-based store and forward approach which treats service recommendations equally without distinction, searching based on the LUR dramatically improves the system's success rates as the recommendations stored locally are "useful". Therefore, most of the queries can find answers from the local trust recommendation storage of a node.

To further study the properties of the usefulness function, we varied the value of its parameters α, β, and γ to test the impact of the three usefulness factors: (a) semantic similarity, (b) trust to recommender, and (c) the recommender's confidence to the recommendation. We keep the total number of nodes to 2048 and there are 10 percent of bad nodes. As shown in Fig. 7, compared with simple LRR-based store and forward strategy ($\alpha = 0$, $\beta = 0$, $\gamma = 0$), the proposed usefulness function and their individual factors dramatically improve the system performance by increasing the percentage of successful transaction. When $\alpha = 1$, $\beta = 0$, $\gamma = 0$, the usefulness function degrades to a similarity function which only measures the similarity between the service discovery query and the service metadata. Similarity can help the discovery system to locate relevant services. However due to

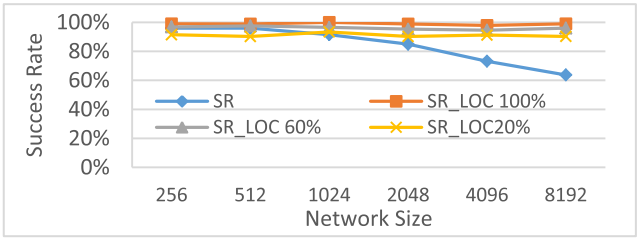

FIGURE 8. Success rate vs. network size.

the existence of bad service providers and the dynamics of the systems, not all of the relevant services are valid. As shown in the figure, this function can make the discovery performance converge quickly, however the performance does not change much over time. When $\alpha = 0$, $\beta = 1$, $\gamma = 0$, the usefulness function only considers the trust value of the recommender. This one-hop trust would help the service requester excludes some bad services recommended by untrusted parties. However, the stored service information may be irrelevant to the requester's interest. Moreover, it is still possible to get bad services due to the unknown intermediate recommenders. Therefore, as shown in the figure (the orange line), its impact is limited. When $\alpha = 0$, $\beta = 0$, $\gamma = 1$, the usefulness function focuses on the recommender's confidence of recommendation, which is recursively determined by all the intermediate nodes between the recommender and the node who initiates the recommendation, i.e., who actually consumes the service. As illustrated by the grey line, as time going, the discovery performance increases. Gradually the system has a better understanding (confidence) for each of the services. As expected, combing all three factors can achieve the best result. The ratio of these three factors is *4:3:3* in this figure, i.e., $\alpha = 0.4$, $\beta = 0.3$, $\gamma = 0.3$. The discovery system can build a big picture of trusted services from participating nodes. Therefore, the performance of usefulness function improves as time increases. An application should set the values of α, β, and γ according to its own properties, such as the typical life time of the network, the similarity between the service providers' profile and their services, etc.

In the next part of the experiments, we added the SkipNet-based locality preserving overlay into the discovery framework. Fig. 8 demonstrates the performance of the integrated discovery system (SR-LOC) which combines social recommendation (SR) with the locality preserving (LOC) discovery. In this experiment, we tested three different service distributions. In particular, we deployed 100%, 60%, and 20% of the requested services in requester's local ASes. We assume that there are no malicious service providers in local AS. From the figure, we can see that adding locality preserving-based discovery dramatically improves the query success rate. When nodes cannot find satisfying results from their local recommendation repository, they can always turn to the locality-preserving overlay to find the results. Also, as shown in the figure, the system performs better when the percentage of local services increases. This is because the locality-preserving based discovery can control the discovery routing

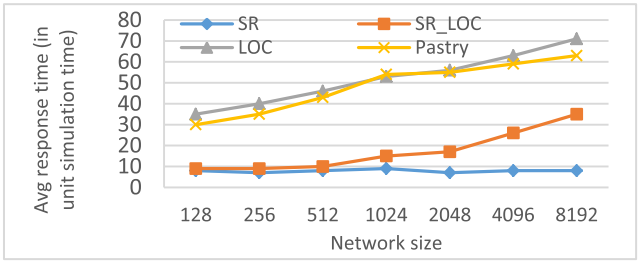

FIGURE 9. Average query response time vs. network size.

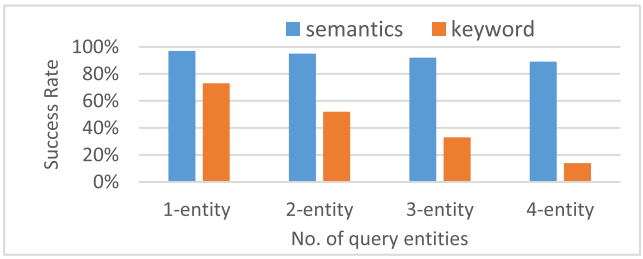

FIGURE 10. Success rate vs. no. of query entities.

path and always locate satisfying local services, which are more secure and trustworthy compared with services provided by other providers.

Fig. 9 illustrates the response time as a function of the network size. In this figure, the time is measured in terms of simulation time units. For comparison, we also implemented a well-known DHT overlay Pastry [28] based on FreePastry [29]. As can be seen from the figure, using either our proposed locality-preserving overlay (LOC) or Pastry, the response time increases logarithmically as network size grows. This demonstrates the good scalability of the LOC overlay structure, as it scales well to a large network like a DHT network. Integrating LOC and the social recommendation-based structure (SR) can further reduce the response time, as most of the queries can find results from the recommendation repository. Based on the results from both Fig. 8 and Fig. 9, we can see that integrating SR and LOC can improve the system performance in both success rate and response time.

In the last part of the experiment, we try to verify the effectiveness of our semantics-based context and QoS filtering technique. In this experiment, we used our predefined ontologies containing 273 entities (51 classes, 13 properties and 209 instances). Based on this ontology, we created services and their corresponding context and QoS instances by selecting and instantiating the ontology. The distribution of services follows the Zipf distribution. Service query is also specified using the same ontology. All service description is instantiated to the instance level, and the query can be in either the class-level or the instance-level. As we try to focus on the matching mechanism, we did not use any discovery schemes; instead all the queries are matched in a centralized service-set. We compare our semantics-based matching with keyword-based matching, which matches only in the vocabulary level. Fig. 10 shows the comparison of the success rate of semantics-based matching and keyword-based matching. We vary the number of query entity/criteria from 1-entity to 4-entity. We can see that semantics-based matching can dramatically improve the matching success rate, by finding more entities which are semantically related although literally may be irrelevant.

From the experimental results presented in this section, we can see that the proposed discovery framework successfully addressed the five requirements listed at the beginning of this paper. It can provide scalable, trustworthy, and context-, QoS-, and semantics-aware discovery services to IoT networks.

VI. CONCLUSION

In this paper, we propose a trust-based context and QoS-aware service discovery framework. To locate services based on context and QoS requirements, we develop an ontological model to present and match service with context and QoS information. Feedbacks from services consumers are collected and propagated, and used as QoS-based trust information to help users select trustworthy services. Two decentralized trust propagation and service discovery mechanisms ae designed to enable effective and efficient service discovery. Like DHT-based discovery systems, our discovery is fully decentralized and robust. Different from DHTs, our social recommendation based overlay can quickly locate trustworthy services; while our SkipNet-based overlay can control where service and service metadata are placed. This largely improves the security and integrity of the discovery framework. The advantages of the proposed mechanisms have been demonstrated via comprehensive simulations experiments.

REFERENCES

[1] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Comput. Netw.*, vol. 54, no. 15, pp. 2787–2805, Oct. 2010.
[2] T. Teixeira, "Service oriented middleware for the Internet of Things: A perspective," in *European Conference on a Service-Based Internet*. Berlin, Germany: Springer, 2011.
[3] Y.-K. Chen, "Challenges and opportunities of Internet of Things," in *Proc. 17th Asia South Pacific Design Autom. Conf. (ASP-DAC)*, Feb. 2012, pp. 383–388.
[4] Z. Sheng, S. Yang, Y. Yu, A. Vasilakos, J. McCann, and K. Leung, "A survey on the IETF protocol suite for the Internet of Things: Standards, challenges, and opportunities," *IEEE Wireless Commun.*, vol. 20, no. 6, pp. 91–98, Dec. 2013.
[5] T. Xu, J. B. Wendt, and M. Potkonjak, "Security of IoT systems: Design challenges and opportunities," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, Nov. 2014, pp. 417–423.
[6] H. Balakrishnan, "Looking up data in P2P systems," *Commun. ACM*, vol. 46, no. 2, pp. 43–48, 2003.
[7] Y.-H. Zhong and Z. Ying-Chun, "Key impact factors of collaborative knowledge management in IoT-related low carbon supply chains," *Int. J. Digit. Content Technol. Appl.*, vol. 7, no. 1, p. 126, 2013.
[8] R. Rantzau, "Discovery services—Enabling RFID traceability in EPCglobal networks," in *Proc. 13th Int. Conf. Manag. Data (COMAD)*, Dec. 2006, pp. 214–217.
[9] F. Bao and C. Ing-Ray, "Dynamic trust management for internet of things applications," in *Proc. Int. Workshop Self-Aware Internet Things*, 2012, pp. 1–6.

[10] F. Bao, I.-R. Chen, and J. Guo, "Scalable, adaptive and survivable trust management for community of interest based Internet of Things systems," in *Proc. IEEE 11th Int. Symp. Auton. Decentralized Syst. (ISADS)*, Mar. 2013, pp. 417–423.

[11] D. Chen, "TRM-IoT: A trust management model based on fuzzy reputation for Internet of Things," *Comput. Sci. Inf. Syst.*, vol. 8, no. 4, pp. 1207–1228, 2011.

[12] W. Leister and T. Schulz, "Ideas for a trust indicator in the Internet of Things," in *Proc. SMART*, 2012, p. 12.

[13] A. Toninelli, A. Corradi, and R. Montanari, "Semantic-based discovery to support mobile context-aware service access," *Comput. Commun.*, vol. 31, no. 5, pp. 935–949, 2008.

[14] K.-L. Skillen *et al.*, "Ontological user modelling and semantic rule-based reasoning for personalisation of help-on-demand services in pervasive environments," *Future Generat. Comput. Syst.*, vol. 34, pp. 97–109, May 2014.

[15] J. Li, N. Zaman, and H. Li, "A decentralized locality-preserving context-aware service discovery framework for Internet of Things," in *Proc. IEEE Int. Conf. Services Comput. (SCC)*, Jun. 2015, pp. 317–323.

[16] R. Mietz, "Semantic models for scalable search in the internet of things," *J. Sens. Actuator Netw.*, vol. 2, no. 2, pp. 172–195, 2013.

[17] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context aware computing for the Internet of Things: A survey," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 414–454, First quarter 2014.

[18] T. Strang, C. Linnhoff-Popien, and K. Frank, "CoOL: A context ontology language to enable contextual interoperability," in *IFIP International Conference on Distributed Applications and Interoperable Systems*. Berlin, Germany: Springer, 2003.

[19] I. V. Papaioannou, "A QoS ontology language for web-services," in *Proc. 20th Int. Conf. Adv. Inf. Netw. Appl. (AINA)*, vol. 1. 2006, pp. 1–6.

[20] H. M. Kim and A. S. J. Evermann, "MOQ: Web services ontologies for QoS and general quality evaluations," *Int. J. Metadata, Semantics Ontologies*, vol. 2, no. 3, pp. 195–200, 2007.

[21] S. Chhun, N. Moalla, and Y. Ouzrout, "QoS ontology for service selection and reuse," *J. Intell. Manuf.*, vol. 27, no. 1, pp. 187–199, 2016.

[22] *W3C QoS*. Accessed: Sep. 28, 2017. [Online]. Available: http://www.w3c.or.kr/kr-office/TR/2003/ws-qos/

[23] L. Atzori, A. Iera, and G. Morabito, "From 'smart objects' to 'social objects': The next evolutionary step of the Internet of Things," *IEEE Commun. Mag.*, vol. 52, no. 1, pp. 97–105, Jan. 2014.

[24] R. Niazi and Q. H. Mahmoud, "An ontology-based framework for discovering mobile services," in *Proc. 7th Annu. IEEE Commun. Netw. Services Res. Conf. (CNSR)*, May 2009, pp. 178–184.

[25] S. B. Mokhtar, "EASY: Efficient semAntic Service discoverY in pervasive computing environments with QoS and context support," *J. Syst. Softw.*, vol. 81, no. 5, pp. 785–808, 2008.

[26] S. B. Mokhtar, D. Preuveneers, N. Georgantas, V. Issarny, and Y. Berbers, "Towards efficient matching of semantic web service capabilities," in *Proc. Int. Workshop Web Services-Modeling Test. (WS-MaTe)*, 2006, pp. 137–152.

[27] S. Marti, P. Ganesan, and H. Garcia-Molina, "SPROUT: P2P routing with social networks," in *International Conference on Extending Database Technology*. Berlin, Germany: Springer, 2004.

[28] P. Costa, C. Mascolo, M. Musolesi, and G. P. Picco, "Socially-aware routing for publish-subscribe in delay-tolerant mobile ad hoc networks," *IEEE J. Sel. Areas Commun.*, vol. 26, no. 5, pp. 748–760, Jun. 2008.

[29] A. Jøsang and S. Pope, "Semantic constraints for trust transitivity," in *Proc. 2nd Asia–Pacific Conf. Conceptual Modelling*, 2005, pp. 59–68.

[30] N. J. A. Harvey *et al.*, "SkipNet: A scalable overlay network with practical locality properties," in *Proc. 4th Conf. USENIX Symp. Internet Technol. Syst.*, 2003, pp. 113–126.

[31] *GT-ITM*. Accessed: Sep. 28, 2017. [Online]. Available: http://www.cc.gatech.edu/projects/gtitm/

[32] *Auto-IDLabs*. Accessed: Sep. 28, 2017. [Online]. Available: https://autoidlabs.org/

[33] M. Lorenz, "Discovery services in the EPC network," in *Designing and Deploying RFID Applications*. Rijeka, Croatia: InTech, 2011.

[34] S. Beier, T. Grandison, K. Kailing, and R. Rantzau, "Discovery services-enabling RFID traceability in EPC global networks," in *Proc. COMAD*, vol. 6. 2006, pp. 214–217.

[35] *BRIDGE*. Accessed: Sep. 28, 2017. [Online]. Available: http://www.bridge-project.eu/

[36] P. Urien, "LLCPS: A new security framework based on TLS for NFC P2P applications in the Internet of Things," in *Proc. Consumer Commun. Netw. Conf. (CCNC)*, Jan. 2013, pp. 845–846.

[37] K. Chung and R. C. Park, "P2P cloud network services for IoT based disaster situations information," *Peer-to-Peer Netw. Appl.*, vol. 9, no. 3, pp. 566–577, 2016.

[38] R. Mietz, "A P2P semantic query framework for the Internet of Things," *Praxis der Informationsverarbeitung und Kommunikation*, vol. 36, no. 2, pp. 73–79, 2013.

[39] B. Polaczyk, P. Chołda, and A. Jajszczyk, "Peer-to-peer multicasting inspired by Huffman coding," *J. Comput. Netw. Commun.*, vol. 2013, May 2013, Art. no. 312376.

[40] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker, "Search and replication in unstructured peer-to-peer networks," in *Proc. 16th Int. Conf. Supercomput.*, 2002, pp. 84–95.

[41] D. Tsoumakos and N. Roussopoulos, "Adaptive probabilistic search for peer-to-peer networks," in *Proc. 3rd Int. Conf. (P2P)*, Sep. 2003, pp. 102–109.

[42] M. Lorenz, J. Mueller, M.-P. Schapranow, A. Zeier, and H. Plattner, "A distributed EPC discovery service based on peer-to-peer technology," in *Proc. 7th Eur. Workshop Smart Objects, Syst., Technol. Appl. RFID SysTech*, 2011, pp. 1–7.

[43] F. Paganelli and D. Giuli, "An ontology-based system for context-aware and configurable services to support home-based continuous care," *IEEE Trans. Inf. Technol. Biomed.*, vol. 15, no. 2, pp. 324–333, Feb. 2011.

[44] Y. Zhang, L. Liu, D. Li, F. Liu, and X. Lu, "DHT-based range query processing for Web service discovery," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, Jul. 2009, pp. 477–484.

[45] N. A. Saadon and R. Mohamad, "A comparative evaluation of web service discovery approaches for mobile computing," in *Informatics Engineering and Information Science*. 2011, pp. 238–252.

[46] S. J. H. Yang, J. Zhang, and I. Y. L. Chen, "A JESS-enabled context elicitation system for providing context-aware Web services," *Expert Syst. Appl.*, vol. 34, no. 4, pp. 2254–2266, 2008.

[47] E. Al-Masri and Q. H. Mahmoud, "MobiEureka: An approach for enhancing the discovery of mobile Web services," *Pers. Ubiquitous Comput.*, vol. 14, no. 7, pp. 609–620, 2010.

[48] R. Peng, Z. Mi, and L. Wang, "An OWL-S based adaptive service discovery algorithm for mobile users," in *Proc. 4th Int. Conf. IEEE Wireless Commun. Netw. Mobile Comput. (WiCOM)*, Oct. 2008, pp. 1–5.

[49] T. A. Butt, "Adaptive and context-aware service discovery for the internet of things," *Internet of Things, Smart Spaces, and Next Generation Networking*. Berlin, Geramany: Springer, 2013, pp. 36–47.

[50] Q. Yu, "Deploying and managing Web services: Issues, solutions, and directions," *VLDB J. Int. J. Very Large Data Bases*, vol. 17, no. 3, pp. 537–572, 2008.

[51] E. Al-Masri and Q. H. Mahmoud, "Qos-based discovery and ranking of web services," in *Proc. 16th Int. Conf. Comput. Commun. Netw. (ICCCN)*, Aug. 2007, pp. 529–534.

[52] G. Canfora, M. Di Penta, R. Esposito, and M. L. Villani, "An approach for QoS-aware service composition based on genetic algorithms," in *Proc. 7th Annu. Conf. Genetic Evol. Comput. ACM*, 2005, pp. 1069–1075.

[53] T. Zhang, "QoS-aware Web service selection based on particle swarm optimization," *J. Netw.*, vol. 9, no. 3, pp. 565–570, 2014.

[54] A. Bhuvaneswari and G. R. Karpagam, "Reengineering semantic Web service composition in a mobile environment," in *Proc. IEEE Int. Conf. Recent Trends Inf., Telecommun. Comput. (ITC)*, Mar. 2010, pp. 227–230.

[55] L. Zhou, "An approach of semantic Web service discovery," in *Proc. Int. Conf. Commun. Mobile Comput. (CMC)*, vol. 1. Apr. 2010, pp. 1–4.

[56] Y.-A. Zhu, and X.-H. Meng, "A framework for service discovery in pervasive computing," in *Proc. 2nd Int. Conf. IEEE Inf. Eng. Comput. Sci. (ICIECS)*, 2010, pp. 1–7.

[57] I. Braun, A. Strunk, G. Stoyanova, and B. Buder, "ConQo—A context- and QoS-aware service discovery," in *Proc. IADIS Int. Conf.*, 2008, p. 2.

[58] I.-R. Chen, F. Bao, and J. Guo, "Trust-based service management for social internet of things systems," *IEEE Trans. Depend. Sec. Comput.*, vol. 13, no. 6, pp. 684–696, Dec. 2016.

[59] P. N. Mahalle, P. A. Thakre, N. R. Prasad, and R. Prasad, "A fuzzy approach to trust based access control in Internet of Things," in *Proc. 3rd Int. Conf. IEEE Wireless Commun., Veh. Technol., Inf. Theory. Aerospace Electron. Syst. (VITAE)*, Jun. 2013, pp. 1–5.

JUAN LI received the B.S. degree in computer science from Beijing Jiaotong University, Beijing, China, in 1997, the M.S. degree in computer science from the Chinese Academy of Sciences, Beijing, in 2001, and the Ph.D. degree in computer science from The University of British Columbia, Vancouver, BC, Canada, in 2008. She is currently an Associate Professor with the Department of Computer Science, North Dakota State University, Fargo, ND, USA. She is the author of one book and over 70 articles. Her major research interest lies in distributed systems, intelligent systems, social networking, and semantic web technologies.

YAN BAI received the Ph.D. degree in electrical and computer engineering from The University of British Columbia, Canada, in 2003. She is an Associate Professor with the Institute of Technology, University of Washington Tacoma, and the Co-Director of the Master of Cybersecurity and Leadership. She has authored or co-authored over 60 refereed papers in his research areas. Her research interests include computer networking, multimedia communications, cybersecurity, eHealth, Internet of Things, and cloud computing. She currently serves on the Editorial Board for IGI Global *International Journal of E-Health and Medical Communications*, and the *EAI Transactions on Industrial Networks and Intelligent Systems*.

NAZIA ZAMAN received the B.S. and M.S. degrees from the University of Dhaka, Dhaka, Bangladesh, in 2007 and 2009, respectively. She is currently pursuing the Ph.D. degree with the Department of Computer Science, North Dakota State University, Fargo, ND, USA. Her current research focuses on intelligent systems, social networking, and natural language processing.

VICTOR C. M. LEUNG (S'75–M'89–SM'97–F'03) is a Professor of electrical and computer engineering and holder of the TELUS Mobility Research Chair with The University of British Columbia. He has co-authored over 1000 journal/conference papers and book chapters, and co-edited 12 book titles. His research is in the areas of wireless networks and mobile systems. He is a fellow of the Royal Society of Canada, the Canadian Academy of Engineering, and the Engineering Institute of Canada. Several of his papers had won best-paper awards. He has provided leadership to the technical program committees and organizing committees of numerous international conferences. He was a recipient of the 1977 APEBC Gold Medal, the NSERC Postgraduate Scholarships from 1977 to 1981, the 2012 UBC Killam Research Prize, the IEEE Vancouver Section Centennial Award, and the 2017 Canadian Award for Telecommunications Research. He is serving on the editorial boards of the IEEE TRANSACTIONS GREEN COMMUNICATIONS AND NETWORKING, the IEEE WIRELESS COMMUNICATIONS LETTERS, the IEEE ACCESS, and several other journals.