Future Generation Computer Systems 86 (2018) 775–791

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Efficient service discovery in decentralized online social networks

Bo Yuan [a,b], Lu Liu [b,*], Nick Antonopoulos [b]

[a] The Key Laboratory of Embedded System and Service Computing, Ministry of Education, Tongji University, Shanghai, China
[b] College of Engineering and Technology, University of Derby, Derby DE22 1GB, United Kingdom

HIGHLIGHTS

- A homophily-based user model based on social relationship and semantic content.
- An adaptive olfactory-sensitive search algorithm.
- Improved service discovery performance and reduced network traffic.

ARTICLE INFO

Article history:
Received 15 November 2016
Received in revised form 1 April 2017
Accepted 13 April 2017
Available online 19 May 2017

Keywords:
Service discovery
Decentralized online social networks
Peer-to-Peer
Swarm intelligence

ABSTRACT

Online social networks (OSN) have attracted millions of users worldwide over the last decade. There are a series of urgent issues faced by existing OSN such as information overload, single-point of failure and privacy concerns. The booming Internet of Things (IoT) and Cloud Computing provide paradigms for the development of decentralized OSN. In this paper, we build a self-organized decentralized OSN (SDOSN) on the overlay network of an IoT infrastructure resembling real life social graph. A user model based on homophily features is proposed considering social relationships and user interests and focuses on the key OSN functionality of efficient information dissemination. A swarm intelligence search method is also proposed to facilitate adaptive forwarding and effective service discovery. Our evaluation, performed in simulation using real-world datasets, shows that our approach achieves better performance when compared with the state-of-the-art methods in a dynamic network environment.

1. Introduction

Online social networks (OSN) connect millions of Internet users and have played a principal role in changing the world into a new era of globalization, i.e. the online society. OSN websites, such as Facebook, Myspace, Twitter, Google+, and Flickr have become increasingly popular online applications for people to communicate information, share moments, and distribute ideas. As of 2016, the number of social network users reached 2.3 billion, which represents 68.3 percent of Internet users, and these figures are expected to increase in future years [1]. Most OSNs sites provide free storage for users to share their social content. In 2014, the data warehouse of Facebook, the largest social website, stored upwards of 300 PB of Hive data using a cloud system, with an incoming daily rate of about 600 TB user-generated content [2]. Social data is being produced faster than any organization has had to deal with before. To add to the challenge, the data is heterogeneous and produced in many formats, including plain text, document, image, video and so on. This flood of data is being generated from any number of connected devices—from PCs, smart phones, tablets to streaming set-top boxes, gaming consoles, digital cameras and even fitness sensors. It is evident that we are entering into a social big data world that is urging the development of novel computing models and new types of architecture to cope with the sheer volume of data. Cloud Computing is a model for enabling ubiquitous, on-demand access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort [3]. After nearly ten years' development, Cloud Computing is maturing and addressing barriers for big data problems. Therefore, OSN can be hosted on Cloud platforms to utilize scalable computing and storage services to process their data in either privately owned, or third-party data centers with lower infrastructure costs [4].

However, all the mainstream OSN providers have a cloud based infrastructure that is designed with a logically centralized architecture controlled by a single authority, i.e. the social network service provider. Though large websites use content distribution networks and distributed computing for performance reasons, there is still a central repository for user data. In this sense, all sensitive personal data is stored in the providers' repository and so users are susceptible to loss of control over ownership of their privacy.

* Corresponding author.
E-mail address: L.Liu@derby.ac.uk (L. Liu).

http://dx.doi.org/10.1016/j.future.2017.04.022
0167-739X/

Furthermore, along with the tremendous amount of user requests and social data, the performance bottleneck has become a critical problem in real-time systems to ensure a smooth continuation and acceptable performance. Besides, the centralized nature has other drawbacks including single points of failure, a need to be online for every transaction and a lack of locality. These issues have provided momentum to the research and development of open, decentralized alternatives; this provides the motivation and context for our work. The architecture of decentralized OSN (DOSN) is created through participation by a set of autonomous OSN users collaborating with each other without a centralized repository. The social data will be stored in local devices and controlled by end users instead of OSN providers. In this way, the huge amount of social data will be distributed in potentially hundreds of millions of end devices rather than a single monolithic system. Therefore, DOSN can alleviate the rigid privacy-control issues, as well as provide adequate flexibility and capability to deal with big data problems

In the meanwhile, the mushroom growth of the IoT and the rapid development of associated technologies provide paradigms for the development of new DOSN. The IoT is a technical revolution that leads the development of next-generation of computing and communications. The IoT can utilize high-end computing processing power, provide interoperable networks and communication protocols and achieve greater flexibility and availability. Moreover, the Social IoT (SIoT) has become an emerging trend of augmenting physical devices and objects with social capability in order to make full use of the collective resources of connected "things". In any SIoT scenario, entities connect to form social groups, enabling collaboration to achieve specific collective goals. It is estimated that the IoT will consist of almost 50 billion objects by 2020 [5]. This prediction suggests that it is reasonable to anticipate that "things" will also form enormous social networks: thing-to-thing social networks resembling human society in the near future.

The infrastructure of the thing-to-thing social networks can be generalized as Peer-to-Peer (P2P) social networks. In recent years, P2P networks [6–11] have been proposed to support the decentralized infrastructure of OSN. The similarity between P2P networks and social networks, where peers can be considered as Internet users and connections reflect social relationships, leads us to believe the principles of P2P networks are suitable to guide the research on the design of DOSN. The nature of P2P networks resembles a real-life social graph built from bottom to top rather than the monolithic, top-down centralized structure that has existed to date, providing people with more control over what they share. Moreover, it possesses many favorable topological features such as self-organization, strong adaptability and fine scalability for large-scale networking applications. Furthermore, P2P networks can gather and harness tremendous computation and storage across the Internet. With SIoT technology, intelligent devices, vehicles, buildings and other objects embedded with sensing, computing, and communication capabilities can participate in an OSN with a dual role of computational processing as well as the content provision.

Until recently, P2P networks have been principally used for file sharing applications. Research into the mechanisms of DOSN, which is still at an early stage, has identified many problems and challenges, which are yet to be solved [9,10]. We examine two major research problems, namely how to design the architecture of DOSN and how to support efficient social information dissemination and service discovery. It should be noted that the service mentioned in the paper is a general concept that is not restricted to well-defined web services, but many types of data including text, hashtags, pictures or video clips in social networking systems. The service discovery should be considered to be a capability to locate varied, multisource social resources.

This paper proposes a self-organized decentralized social network (SDOSN) featured by P2P infrastructure with a swarm intelligence [12] search strategy. Specifically, a self-organized P2P social overlay network is introduced to form a local knowledge index that can facilitate accurate and personalized service discovery. The key challenges of this stage are how to define similarity measurements for social users to establish acquaintance shortcuts and how to utilize limited knowledge to make predictions for neighbor selection. A homophily-based user model is proposed to select promising neighbors in each routing step considering social relationships as well as content semantic features. Furthermore, a novel olfactory sensitive search (OSS) algorithm is proposed by exploring the swarm intelligence. The free and uncertain foraging behavior of swarm collaboration is abstracted and modeled to optimize the service discovery process in the DOSN. The OSS is well suited to this environment because it integrates knowledge of collective intelligence and the highly self-organized social features of users. In this way, a user can progressively gain experience during the discovery process and make future discovery more focused and accurate.

The performance of the proposed approach is evaluated using simulation experiments. A P2P social network platform is developed that is able to simulate network structure dynamisms (topology construction, churns, and routing) and service discovery processes (indexing, bootstrapping, and searching). The experiments use real-world datasets in three different network structures. The results show that the performance of the proposed service discovery algorithm in SDOSN achieves less average visited nodes, higher success rate and a higher recall when compared against existing state-of-the-art methods.

The main contributions of this paper are summarized as follows:

- A self-organized architecture for social overlay network of DOSN is proposed. This architecture fills the research gap between the traditional P2P infrastructure and the decentralized architecture of OSN.
- A homophily-based user model is introduced to capture the homophily similarity that integrates social relationship and user interest. This model is able to identify promising neighbors those are similar to the service provider and have the high number of connections.
- A novel olfactory sensitive search algorithm is proposed to guide the service discovery with an adaptive forwarding degree. The algorithm utilizes the collective swarm intelligence to discover the shortest paths with maximum desired services.
- A software simulation platform is designed and developed. It can simulate dynamic unstructured P2P networks with configurable routing protocols to support social overlay networks, decentralized service discovery, and evaluation of search models.

The remainder of the paper is structured as follows. Related works are identified in Section 2. Discussions on the design of self-organized architecture and supporting service discovery mechanisms are presented in Section 3. The homophily-based user model and the matchmaking method are presented in Section 4. Section 5 introduces the swarm intelligence algorithm necessary to achieve adaptive forwarding. The simulation platform, experiments and results are discussed in Section 6, and finally the conclusion is presented in Section 7.

2. Related works

In this section, the related works of service discovery in decentralized architectures are compared and contrasted. In decentralized architectures, all nodes are considered to be equal and an

arbitrary topology exists. This type of architecture provides greater flexibility and adaptability. Nodes only have a partial view of the network structure or service organization and need collaboration with the rest of the system in order to succeed in the service discovery process. In general, existing approaches employ blind or informed search methods to locate target services.

When using blind search methods, nodes do not keep information about service locations. In order to find target services, search methods such as Gnutella [13] use a flooding method, employing the Breadth First Search (BFS) on the overlay network graph with a depth limit. However, this generates a large number of duplicate messages and as a result, does not scale well. Many alternative schemes have been proposed to address the limitations of Gnutella, such as random walks [14], iterative deepening [15], and attenuated bloom filter based search [16]. These schemes apply bandwidth-efficient techniques instead of flooding the query, reducing the overhead in the discovery process by shortening the average routing length and reducing the number of visited nodes. However, blind search methods are unreliable, and the success of queries is no guarantee as the nodes are continually connecting and disconnecting. In addition, the bandwidth cost of searching grows exponentially in relation to the number of connected users, often saturating connections and rendering slower nodes useless. As a result, most search queries may only reach a very small part of the network, often being dropped before successfully locating the target information.

In order to prevent the generation of excess traffic, informed search methods that maintain local information have been proposed. In these methods, nodes store some metadata or knowledge to establish shortcuts among nodes, which facilitate the query routing to reduce the traffic overhead in the service discovery process. Routing Indices [17] allow nodes to forward queries to the neighbor that is most likely to have the required resources. Each node maintains a routing index with information about the number of documents along the path and the number of documents on each topic of interest. If a node cannot answer the query, it forwards the query to a subset of its neighbors based on its local routing index rather than randomly selecting or flooding the network. The problem with this proposal is that it is necessary to keep the large amount of information updated. The number of messages required to propagate changes in the system could overload the network. If the information update process is delayed, a node can have information about routes that are invalid. Moreover, the precision of this method depends on the number of categories that are considered in the search process. Adaptive probabilistic search (APS) [18] is an algorithm that is based on a combination of *k*-random walk algorithm and probabilistic forwarding. Each peer has a local index that keeps one entry for each neighbor. The value of each entry is a tuple that contains the identifier of a neighbor and the probability that the neighbor will be selected the next time based on previous searches. Analogous research [19] proposes an intelligent search mechanism that allows peers to identify links that are likely to have relevant information. A drawback of these algorithms is that a period of time is required to collect the information to improve the search. Moreover, if the links between peers change frequently, the statistical information stored in the local indexes could become inoperative. A further limitation is that some of the heuristics that are used to guide the search process could overload some peers and leave other potential peers without traffic. Freenet [20] employs a heuristic key-based routing method where each file is associated with a key, and files with similar keys tend to cluster on a similar set of nodes. Freenet uses a steepest ascent depth first and queries are likely to be routed to the node that is associated with the most similar key without needing to visit many peers. However, Freenet searches in a greedy fashion, which does not guarantee the global optimum. In addition, the routing based on cluster similarity cannot resolve unfamiliar queries (i.e. area beyond the similar cluster) and often reaches only a very small range of the network. NeuroGrid . [21] is a decentralized adaptive search system. Unlike previous methods, NeuroGrid adopts the historical information of previous searches to guide peer nodes to make routing decisions. However, NeuroGrid is effective for previously queried keywords only and is not suitable for networks where peer nodes join and leave rapidly. In ESLP [22], the connections of peer nodes are adaptive with cached knowledge and only a number of associated connections are stored in each node. ESLP utilizes an adaptive forwarding degree approach to guarantee service quality without having exorbitant overhead. However, ESLP does not provide bandwidth-efficient routing due to redundant queries and messages. The adaptive forwarding degree achieves promising performance but it lacks probabilistic explanation. Literature [23] proposes a decentralized service discovery in open service-oriented multi-agent systems. The authors propose the homophily in service-oriented multi-agent systems to create efficient decentralized and self-organized structures where agents have a greater probability of establishing links with similar agents than with dissimilar ones. A second contribution is an algorithm for service discovery that considers the local information that is related to the homophily between agents. However, this model employs a greedy search method, following every hop only the most promising nodes are selected to forward the query, which could require multiple unnecessary hops to locate the required service. In addition, the recall of this approach is low.

Other approaches use biologically inspired techniques to locate and organize resources. For instance, ant algorithms are suitable for unstructured network because they do not rely on global knowledge about the network. The algorithm proposed by Michlmayr et al. [24] uses ant algorithms to guide the search in P2P networks. Every peer maintains a repository of documents, each of which has a keyword, the neighbor provides the document and the quality of pheromone. There are two types of ants in the system: forward ants and backward ants. The forward ants navigate the network until the document is found or the TTL finishes. The main problem of this research is that the pheromone information is only based on keywords of the document. Therefore, if a peer is searching for a document using a keyword that is not specifically used in the document, the peer will not be able to locate the information in the network even when similar documents exist. IAPS [25] is a fully distributed and bandwidth-efficient algorithm, which uses ant-colony optimization and takes file types into consideration in order to search for file container nodes with high probability of success. The search performance is affected by the K walker deployed in the system. The results show that IAPS reduces duplicated messages and increased success rate when compared to random walk and APS. However, IAPS does not consider semantic features of the node content, and it is only evaluated in a static P2P network without churn.

3. System architecture

3.1. Social overlay network

In SDOSN, users run P2P clients (peer nodes) on their hosts to post content on the timeline and to browse the profiles of friends. The social network is modeled as an undirected graph $G = (V, E)$, where V contains users and E is the friendship relations among them. To avoid confusion, within this paper a user and a node are considered as one-to-one mapping; therefore, the terms "user" and "node" are totally interchangeable.

Due to the nature of fully unstructured P2P network, peer nodes resemble social users, and every node is considered to be equal in the network. This paper adopts the friend-to-friend approach [26]

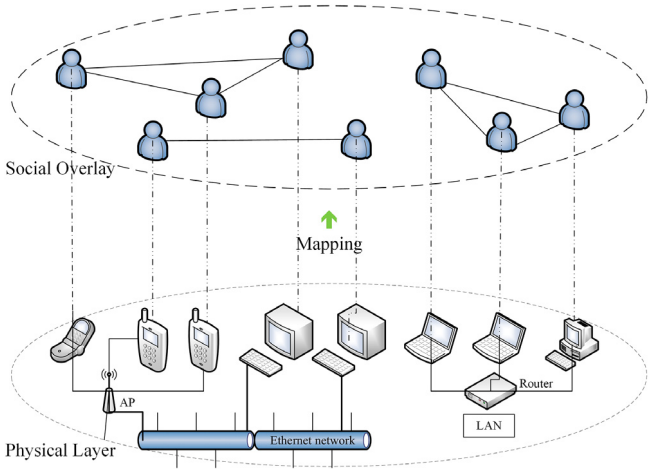

Fig. 1. Social overlay network.

to build a social overlay network, where communication among nodes is enabled only if their owners know each other. In this design, peers are arranged in a social overlay where one-to-one mapping mirrors the underlying network as showed in Fig. 1. The maintenance of the overlay network is in a self-organized manner, which means node connections are established or detached in a full-unstructured P2P network based on social knowledge during user interactions. The social knowledge is historical information stored locally to maintain shortcut relationships between nodes. Some of the main benefits of choosing the unstructured friend-to-friend overlay are summarized as follows: Firstly, unlike links in a DHT, links in a social overlay represent friendship. The hypothesis here is that friends are more likely to interact and cooperate with other friends than with random strangers. A recent study [27] showed that approximately 78% of user interactions take part within one-hop neighborhoods. Secondly, because people tend to talk to people they know, a social overlay enables shortcuts and constrains traffic to small network sections at a time when handling frequent interactions between friends. Thirdly this overlay improves locality of P2P social network as people tend to connect to people that are alike, with geographical co-location playing a key role. Therefore, paths over social overlays are likely to be short and localized. Moreover, privacy issues are mitigated since communication is restricted to friends; only the identity of the original user needs to be checked [28]. With the utilization of social overlay network, SDOSN discovers the identity of users who likely hold the desired service, and then optimizes the search routes to it by using OSS algorithm.

Since SDOSN operates the service discovery on an unstructured P2P infrastructure, the network dynamism is one of the main challenges for maintaining the social overlay network and managing the services. Dynamism features are studied from two aspects: friendship network and overlay network. In SDOSN, new friendship relationships are continually being formed or old ones are being severed. However, in real online social networking systems, such changes are infrequent to the extent that they are insignificant within the time scale of our problem of information dissemination and service discovery. Therefore, for the purpose of this research the friendship network is considered immutable. The dynamism of social overlays cannot be disregarded in the same way. Since users may disconnect or stop their devices at any time they choose, any given user may be offline and not available for some periods. This phenomenon, in P2P terminology, is known as the churn. The social overlay network undergoes frequent reconfiguration, which affects the performance of information propagation and service discovery.

Node0	NeighborList	InterestList	Pheromone		
			Topic1	Topic2	Topic3
Node1	0|2	Photography	0	1	0
Node2	0|1|5	Algorithms|Graphics	1	0	2
Node3	0|5|6	Math|Technology	1	0	0

Fig. 2. Example of knowledge index structure.

In SDOSN, nodes update two distinct indexes: a local service index and a knowledge index. The local service index is an inverted index that organizes various types of local services, such as documents, music, video, etc. The knowledge index stores the overlay shortcuts information obtained during social interactions, i.e. the knowledge index contains associations between a node and its immediate neighbors based on historical searches. The structure of knowledge index is a relational table: each row represents an immediate neighbor, and the columns include the neighbor's neighbor list, the neighbor's interests list, and pheromone information (i.e. topic hit). In the knowledge index, a user knows not only the set of neighbors (friends), but also the neighbor list of neighbors (friend's list of friends). This assumption is reasonable because in modern OSN e.g. Facebook or LinkedIn, the set of friends is already part of the profile and incremental updates (in the form of new friendship events involving your friends) are showed in the newsfeed [29]. For instance, if A and B are friends in a social network, A is able to see B's friends list as this is a part of B's profile and vice versa. If B and C have formed a new friendship, A will receive the update that B has a new friend C in the newsfeed.

An example of knowledge index for Node0 is showed in Fig. 2. In this example, Node0 has three neighbors Node1, Node2, and Node3. Node0 maintains knowledge of the neighbor list, interest list and pheromone table of its three neighbors. The real-time pheromone is calculated according to specific queries and used to determine the adaptive query forwarding in the service discovery process. The details of how to utilize the knowledge index will be introduced in Sections 4 and 5.

Initially, a newly joined node has no knowledge of the network. It will send a limited number of queries to a set of randomly selected peer nodes. If a search is successful, the requesting node updates its knowledge index to associate the peer nodes those have responded data successfully with the requested topic and connects to these nodes. Social interests and friends list for a user are public information between friends in most mainstream OSN. During the connection process, this publicized information is obtained and stored in the knowledge index. In a parallel process, the requesting node removes invalid cached knowledge according to the results of searches. For example, if a neighbor of node A has removed a certain shared service corresponding with a query topic q, then node A will no longer be able to locate the service directly. Any future query q directed to node A will be unsuccessful and so node A removes the query topic q from its knowledge index. In addition, we also need to consider the churn in the network. If one neighbor disconnects from the network, the current node detects it and updates the knowledge by setting the neighbor's list as -1. When the neighbor re-joins the network, the neighbor's degree needs to be updated accordingly. Therefore, peer nodes can learn from the results of previous searches, which make future searches more focused. As more searches are performed, more knowledge can be collected from the search results. As the process continues, each node will cache a great deal of useful knowledge that is effective in making future decisions to re-visit peer nodes with access to the required services.

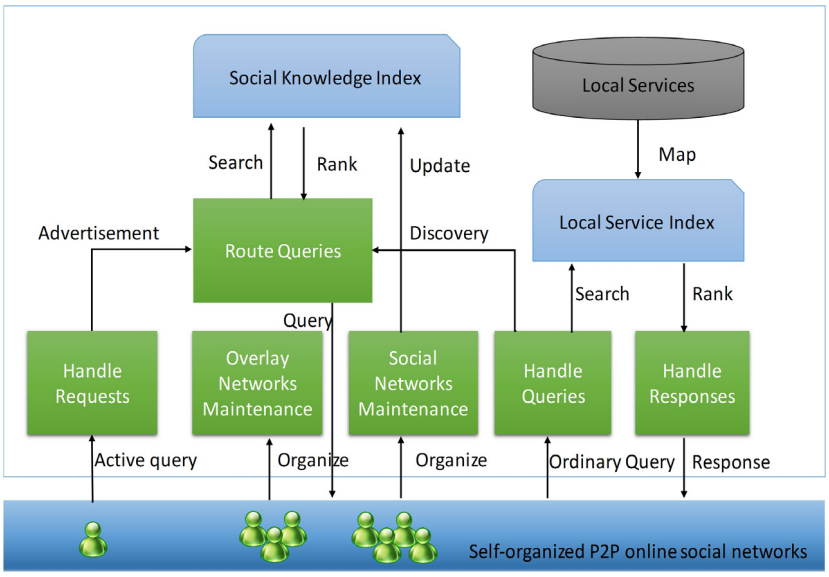

Fig. 3. Frame diagram of service discovery.

In existing methods of informed search, nodes often utilize queries to gather knowledge about the network; the target nodes sharing the desired services will respond with the information related to the requested topic only. Using this principle, a node will issue a large number of queries to gather information about the network. This may lead to a huge number of messages and result in significant overhead. Therefore, this paper proposes a fast knowledge acquisition approach to fetch a batch of information with Active Query during the advertisement stage, as introduced in our previous work [22]. The Active Query is an extension of a traditional query, which can absorb a significant amount of information about the network but with fewer queries. The target nodes will not only respond to the requested topic but also inform the originator of other associated topics shared within the same interest area. An interest area in SDOSN is a semantic area with a set of topics. The corresponding interest area of a specific topic and other topics in this interest area can be found from the open directory DMOZ [30] which is a widely distributed database of Web content using a hierarchy topic structure. The DMOZ has been widely used in popular Internet services, such as Google, Netscape, Lycos, Hotbot, Dogpile, Thunderstone, Linux, Mars Society, etc.

3.2. Service discovery in social overlay network

In social networks, people remember and update potentially useful knowledge from social interactions; as a consequence random and diffuse behaviors gradually become highly organized [31]. In SDOSN, service discovery is also based on knowledge; a user knows not only the friend list of friends, but also the interests of friends. The service discovery is intended to locate the required service based on the historical interactions of users.Furthermore, in a social network, people use tags by creating user-defined terms or selecting system-provided terms to declare their social interests. The social interests are significant indicators to classify user groups and recommend new friends in social networks, which can improve the accuracy of service discovery.

Fig. 3 depicts the system structural frame and module functional frame of service discovery. The underlying infrastructure is an unstructured P2P network. The overlay network is a one-to-one mapping mirroring the underlying network. Above the overlay network, users connect to others to build a social network and the communication among users is enabled only if users know each other, i.e. they are connected by the social network. Service discovery, where users interact with others through query and response messages, is one of the most important interactions in a social network. Each user builds a local service index and a knowledge index. The local service index maintains an inverted index table of all of the social network services a peer node possesses in local storage. The knowledge index stores network shortcuts i.e. the information about immediate neighbors. When a new user joins the social network, the social relationship and the initial knowledge index will be built rapidly through advertisement of the user's interests and active queries. Details of the advertisement will be introduced in the next section.

The query process is showed in Fig. 4. When a user receives a query, he will first search from the local service index to find matched services considering homophily similarity. If the query needs to be further forwarded, the user will consult the knowledge index to find associated users using the OSS algorithm and multicast the query to these selected users. The OSS algorithm is a swarm intelligence approach, which employs pheromone information and olfactory sensibility to support adaptive forwarding. During query forwarding, the number of duplicated query messages will be reduced without losing the probability of finding requested services.

The query process will be terminated if the query has been successfully resolved or the number of query hops has exceeded a user-defined limit.

4. Service discovery module

4.1. Overview

The problem of service discovery in a P2P network is known as Semantic Query Routing Problem (SQRP). The general goal of SQRP is to determine the shortest path from a node that issues a query to nodes that can appropriately resolve the query by providing the requested service. Furthermore, the number of services located is maximized and the number of steps to locate the services is minimized. The query traverses the network moving from the initiating node to neighboring nodes and then to neighbors of neighbors and so forth until it locates the requested service or exceeds a user defined limit.

The service discovery comprises two stages: the advertisement stage and the discovery stage. The advertisement stage is the

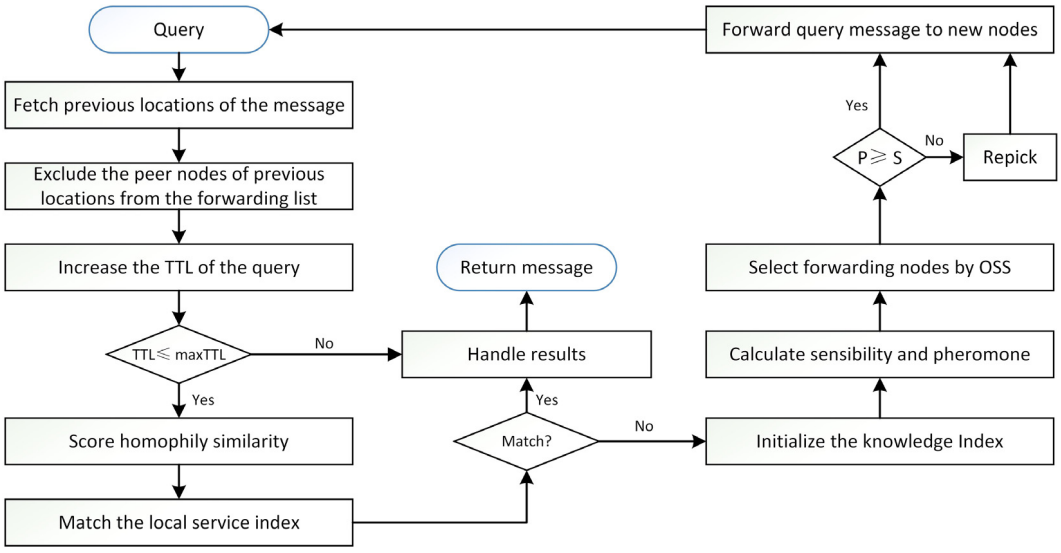

Fig. 4. Query forwarding process flow chart.

initial bootstrapping phase when each new joined user periodically advertises an active query message containing digest information about services to be shared along with their declared interests. The digest information is pre-processed based on the Bloom Filter [16] techniques that are able to compress both query requests and responses. The Bloom Filter is a hash-based, space-efficient data structure that compresses the query message thereby significantly reducing the size of messages transferring in the network layer. Though there is a small probability of false positives in Bloom Filter, this is beyond the topic of this paper. In the discovery process, when a node searches for a specific service, it deploys an ordinary query message to find the node holding the desired service. The query message is forwarded by intermediate peer nodes based on their pheromone information that will be discussed in the next section. If the target node is located, a response message will be returned to the requester and pheromone information is laid along the routing path. In the future, if other query messages are seeking similar services, the query messages could follow previously laid pheromone information and not traverse the network in a blind way.

4.2. Advertisement stage

The aim of the advertisement stage is to increase the probability that nodes find a peer node containing the requested service. To achieve this goal, each node advertises its shared services and personal interest information using active query messages. For example, if the originator generates a query with the topic "Paintball", a target node that shares the desired services will answer the query about "Paintball" as well as the associated topics in this interest area such as "High Impact Paintball", "Lemmings Paintball", and "Ultimate Paintball" in this interest area. The obtained new information will be added to the local knowledge index by the originator for future queries. With active query messages, the originator can gather more pieces of knowledge from each successful query; however, additional traffic will be generated when transferring the additional knowledge. The extra traffic could be significant; if every node generates all queries in this manner, the traffic may be excessive for bandwidth-limited networks. In contrast, if search is undertaken with only ordinary queries, each new node accumulates knowledge slowly by gathering one piece of knowledge from each successful query, especially for those peer nodes that are seldom present or rarely query the network. To address these issues, a trade-off scheme is designed for bandwidth-limited networks. The recently joined nodes will utilize active query messages to gather a large amount of useful information quickly regarding their interests. After the amount of cached knowledge reaches a certain threshold ratio with respect to the maximum size of the knowledge index, the peer nodes will utilize ordinary queries to discover the required files with low traffic cost. The threshold ratio was presented and evaluated in our previous work [22]. This process is not only applicable for recently joined nodes during the advertisement stage, but also enables peer nodes to recover from unpredictable knowledge loss during the discovery stage.

4.3. Discovery stage

In the discovery stage, nodes utilize query messages to search the network in a decentralized manner. However, as service environments have become significantly complex and chaotic lacking hierarchical organization or centralized control, the vast majority of services exist without explicit associated semantic descriptions. On the other hand, the current service discovery methods in P2P networks are mainly supported by keyword-based solutions that cannot provide a context-aware and personalized search for end users. As a result, many services that are relevant to a specific user's request may not be considered during the service discovery process. In order to deal with these problems, an interest-based model is proposed to exploit service content and users' social interest.

4.3.1. Service representation and user interest profiling

In order to achieve accurate and efficient service discovery in SDOSN, semantic features and social features are both considered in our work. The topic model Latent Dirichlet Allocation (LDA) [32] is extended and improved to extract latent topics from the user's service repository to form users' profiles LDA is a very popular technique for discovering the main themes or topics from a large collection of unstructured documents and has been widely applied in various fields, including text mining, computer vision, finance, bioinformatics, cognitive science, music and social sciences. Using LDA a document can be represented as a random mixture of latent topics, where each topic can be characterized by a probability distribution over a vocabulary of words. However, the social network service contents are usually short documents and directly applying the conventional LDA topic model on such short texts may not

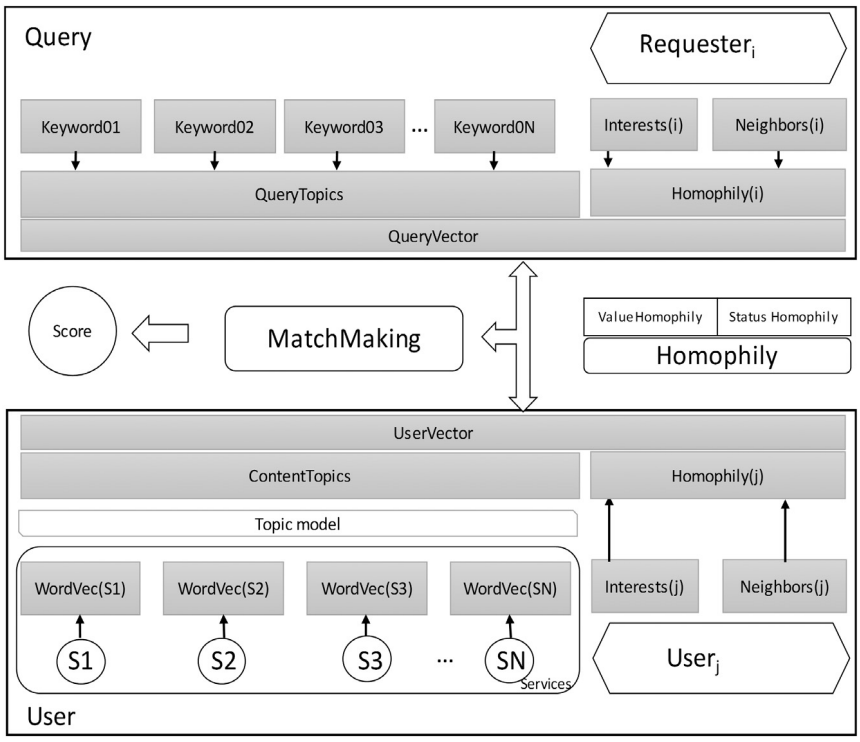

Fig. 5. Service representation and match making.

work well. The reason lies for this being that LDA implicitly captures the document-level word co-occurrence patterns to reveal topics, and thus suffers from the severe data sparsity in short texts. Therefore, we extended the conventional LDA using principles proposed in BTM [33] The implementation of the extended LDA model is outside of the scope of this research. Using the topic model, heterogeneous service contents (such as documents, tags, and photo/video descriptions) are converted to a topic space which has fewer dimensions than the traditional term-document space.

Moreover, this paper also integrates user's interests to deal with personalized social information. Social interests are the terms that describe the personal concerns or involvement of a user in a social network. Interest terms are user-defined or system-provided terms (e.g. social tags) that do not necessarily appear in user's local repository. Interests are more semantically recapitulative concepts than topics extracted from local services and are strong indicators of a users' preference or expertise.

For instance, Bob is a botanist, and he issues a query "apple production in the US" to find out the apple fruit production in the US. His interests include "botany, biology, plant, flower". Kate is a sales manager working with Apple Inc. and lists interests as "Apple, products, sales, marketing, business". When Kate receives Bob's query, traditional matchmaking will match the local service repository of Kate and "resolve" the query However, this result is a false positive and the service provided by Ann does not match Bob's intention. By applying the interest-based matchmaking, when Bob issues the query, the method can filter out the unrelated information. Interests between Bob and Kate are very different, hence lower ranking of Kate regarding this query.

When a user is being queried, SDOSN not only considers the similarity based on queries and services, but also takes users' interests into account. Furthermore, according to recent study [34–36], individuals tend to interact and establish links with individuals who have similar interests. This phenomenon is called homophily in social network research. Homophily influences diffusion patterns over a social network in two ways: it affects the way a social network develops and individuals are more likely to successfully influence others when they are similar to them [37].

The matchmaking process is tailored specifically to an individual's social feature by incorporating information about the individual beyond specific query provided, which is showed in Fig. 5. The main purpose is to achieve a comprehensive similarity between a query request and a node that is receiving the query. A query message is augmented with the query requester's interests and neighbor information to represent an estimated target profile vector. In this way, the problem of the matchmaking between a query and a user has transformed to the matchmaking between two users' profile vectors. As a result, heterogeneous information, such as queries and service contents, as well as user interests and neighbors are converted to the same profile vector space, then represented, discovered and compared under a homogeneous data structure. Besides, it can effectively alleviate the synonymy problem (i.e. different concepts referring to the same meaning) and reduce the term dimensions in service discovery thus making the discovery process more accurate and efficient.

The matchmaking module measures the similarity between the query vector and user vector, which considers both homophily similarity and semantic similarity. Homophily similarity is a linear combination of status homophily and value homophily. Status homophily counts the common neighbors and value homophily gauges the common interests. The semantic similarity is based on the likeness of meaning between the query keywords and service topics. Matchmaking applies a cosine similarity to calculate the overall similarity between the query vector and user vector. Cosine similarity is a commonly used approach in modern information retrieval [38,39]. A threshold of cosine similarity that determines a user can provide a "similar enough" service to the query requester is set to 0.25 in our system. This threshold is evaluated in the experiment's section.

4.3.2. User modeling and neighbor scoring

Homophily is one of the most salient properties in complex networks [40–42]. The idea behind homophily is that individuals

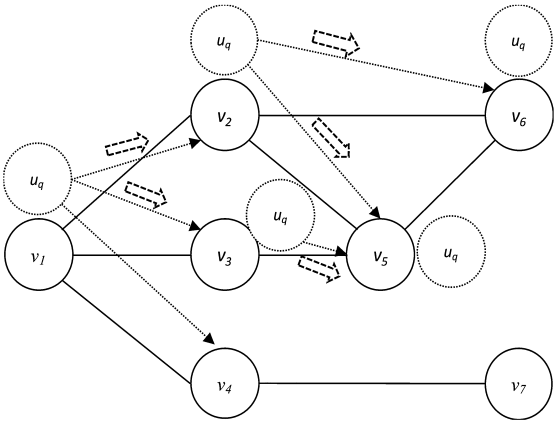

Fig. 6. User query model.

tend to interact and establish links with similar individuals. In their original formulation of homophily, Lazarsfeld and Merton [34] distinguished the concept between status homophily and value homophily. This subsection introduces the formal definition of status homophily and value homophily to model the similarity of users.

In SDOSN, the homophily is defined as the linear combination of status homophily (SH) and value homophily (VH):

$$H(u, v) = \delta SH(u, v) + (1 - \delta)VH(u, v). \tag{1}$$

The δ parameter regulates the importance of the influence of status homophily and value homophily in the overall homophily of two nodes. $H(u, v)$ is a measurement of interpersonal similarity that influences the service matchmaking and forwarding degree in our service discovery process. Users are more likely to successfully provide the services or guide the discovery when they are similar to each other [39].

The status homophily defines the social familiarity, which calculates the common friends (neighbors) of two nodes. If user u and v are familiar, it tends to be they share many common friends. The status homophily between user u and v is denoted as $SH(u, v)$:

$$SH(u, v) = \frac{|N_u \cap N_v|}{|N_u \cup N_v|}. \tag{2}$$

Where N_u, N_v are the neighbor set of user u and v respectively. However, those users who have high SH may have very different interests. A query is likely to be resolved by a user whose interests cover topical areas of the query. So in the next step, the personal interests are also considered. The value homophily calculates the degree of matching between two sets of content topics including interests in the user profile vector, which is defined as:

$$VH(u, v) = \frac{|T_u \cap T_v|}{|T_u \cup T_v|}. \tag{3}$$

Where $T_u T_v$ are the interest set of user u and v, respectively. Equation (2) and (3) apply union and intersection operation in the neighbor set and the interest set to achieve fast calculation of homophily score.

In SDOSN, the service discovery is powered by query routing of immediate neighbors. When a node searches for an unknown target service, it sends a query to the users in its neighbors list. Likewise, when a node receives a query about a service that cannot be provided, it will forward the query to its neighbors Each forwarding process between a node and its immediate neighbors is counted as one hop of a query. This process will carry on until the number of hops exceeds a pre-defined Time-to-Live (TTL) value. As stated earlier, the goal of the query routing is to determine the shortest path from a node that issues a query to nodes that can appropriately resolve it. However, the query requester only has a very limited knowledge of the decentralized social network, and often it does not know the path and distance to the target nodes that can provide the requested service. During the discovery process, traditional approaches match the query and services in a node to determine whether it is a target or not, which is nearly a brute force search with very little information about the targets. In our work, a heuristic method is proposed by modeling a virtual target node with potential services that meet the query requests and homophily features that are similar to the requester. This method is beneficial for the service discovery for two reasons. Firstly, the virtual target node has more information to describe the query request, which makes the matchmaking more accurate and unbiased. Secondly, homophily helps the discovery process to access social information that is useful for promising neighbor selection during the query routing.

As showed in Fig. 6 the query q issued by node u is modeled as a dotted node of $u_q.u_q$ a temporary shadow of u,contains the information on the profile of node u (such as interests, neighbor list) as well as the service requirements and maxTTL. The shadow node u_q can be considered to be an estimate of the profile of possible target nodes. The estimated target node should have the required services as well be likely to share some common features with the service requester according to the homophily study [23]. The lifecycle of $u_q$starts from the query being issued and ends with multiple successful service discovery replies until the maxTTL expiry. The u_q information is processed based on the hash techniques mentioned in the previous subsection.

This paper utilizes homophily-based factor and degree-based factor (number of neighbors) in an exponential function to measure the probability of a neighboring node to be capable of resolving the query. This probability will be used to determine whether or not to forward a query to a neighbor and so the probability is called choice probability. The most promising neighbor is the most similar neighbor to the estimated target node and has the highest number of connections. Therefore, when node v_y forwards query $u_q$to its neighbor set $N_{v_y}$the following equation calculates choice probability of $\forall v_x \in N_{v_y}$

$$CP(v_x, u_q) = 1 - \left(1 - \left(\frac{H(v_x, u_q)}{\sum_{v \in N_{v_y}} H(v, u_q)}\right)\right)^{|N_{v_x}|}. \tag{4}$$

The choice probability CP considers the homophily between the node v_x and the estimated target node u_q. Moreover, it also considers the degree of the neighbors. The degree is an important factor in query routing. Those nodes with a higher degree have more influence on the information dissemination. The reason is that they can forward information to more nodes in one hop and also make more impact on the network traffic. The exponential function used in Eq. (4) creates a marginal improvement of the ranking score by applying a power exponent of the degree. This function is not only able to capture the homophily between the neighboring node and the estimated target node to encourage interest-based discovery, but also integrates the degree factor to accelerate the service discovery process. As a result, the search range in each hop can be more accurate and overall query messages can be resolved with fewer hops on average.

We now present an example calculation of choice probability In Fig. 6, before forwarding the query to the neighbors, node v_1 needs to determine the choice probability of v_2, v_3 and v_4 in the knowledge index. Using example values for the homophily score between q and v_2, q and v_3, and q and v_4:

$$H\left(v_2, u_q\right) = 0.4, H\left(v_3, u_q\right) = 0.5, H\left(v_4, u_q\right) = 0.6.$$

Then the choice probabilities of v_2, v_3 and v_4 are given by

$$CP(v_2, u_q) = 1 - \left(1 - \left(\frac{0.4}{0.4 + 0.5 + 0.6}\right)\right)^3 = 0.61,$$

$$CP(v_3, u_q) = 1 - \left(1 - \left(\frac{0.5}{0.4 + 0.5 + 0.6}\right)\right)^2 = 0.55,$$

$$CP(v_4, u_q) = 1 - \left(1 - \left(\frac{0.6}{0.4 + 0.5 + 0.6}\right)\right)^2 = 0.64.$$

In this example case, the choice probability is calculated by using homophily score and connection degree. It can be seen that the choice probability is high when both homophily score and connection degree are high. v_4 has the highest choice probability is this case. In the meanwhile, v_2 has the lowest homophily score, but the highest connection degree The choice probability of v_2 is higher than v_3 is because the connection degree plays a more import role in Eq. (4).

The use of Eq. (4) has generated a probability of resolving the query q for each of the neighboring nodes of v_y. Node v_y can forward the query to its neighborhood based on the calculated probabilities $CP(v_x, u_q)$. The subsequent requirement is to determine the number of nodes to be forwarded to in the current circumstance. In existing query routing methods [21,23,25,43] the number of nodes to be forwarded in each hop is a static value or a simple threshold. Methods with a fixed number of forwarding degree have limitations and cannot balance the recall and traffic overhead in a bandwidth-limited network In each hop, if nodes only select the most promising node, the number of visited nodes and message traffic will be reducing to the lowest level. However, many less promising nodes also may have the requested service. As a result, the recall of this greedy neighbor selection is low. On the contrary, if nodes utilize a large fixed forwarding degree in each hop to achieve high recall (finding a greater number of relevant services), the traffic in the network grows exponentially in relation to the forwarding degree. In bandwidth-limited networks, the performance is relatively low as there are many duplicated query messages.

Fig. 7 shows examples of how the forwarding degree affects the performance. There are two routing strategies with a static number of receivers per hop ($d = 3$) in Fig. 7(a) and an adaptive number of receivers per hop ($d = 2 \sim 4$) in Fig. 7(b). The black dots represent the nodes that share requested services, while the gray dots show the nodes that are highly correlated with the query and know who has the requested services. As showed in Fig. 7(a) and (b), the adaptive routing strategy achieves better search performance by finding more target nodes with fewer visited nodes and fewer query messages.

In order to improve the trade-off situation between network traffic and recall and conduct a more efficient service discovery in SDOSN, an adaptive forwarding degree approach is proposed to adjust the number d (i.e. forwarding degree) of nodes to be forwarded according to the olfactory sensibility and pheromone information. The goal of query routing is to determine the shortest reply paths while the number of found services is maximized and the steps to locate the services are minimized.

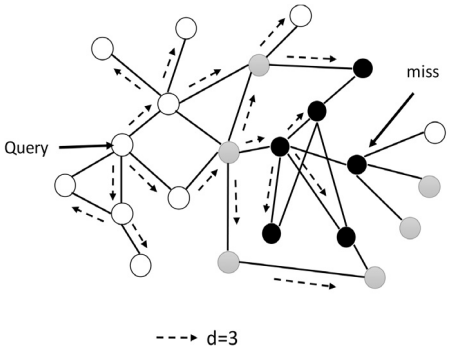

Fig. 7a. Fixed forwarding degree.

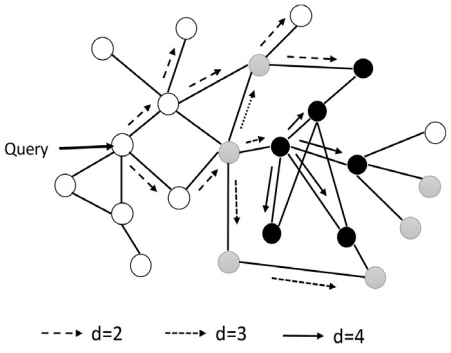

Fig. 7b. Adaptive forwarding degree.

5. Adaptive forwarding algorithm framework

5.1. Overview

This section introduces our proposed swarm intelligence approach called olfactory sensitive search (OSS) algorithm inspired by the food seeking strategies of biological swarms. For instance, ants use their olfactory senses to find food and communicate with others due to their short vision range. Ants migrate between the nest and the food source by leaving trails of pheromones to others, succeeding ants usually move preferentially in the direction of higher pheromone intensity. Pheromones are a special chemical secretion of species commonly known to lead other members of its own species towards the point of interest, while imposing a territorial boundary in the form of an allomone to organisms outside of their species. As an ant proceeds to the food location from its nest, the trail pheromone aids a narrow and precise pathway route for other members of the same colony to follow [44]. During the foraging process, ants visit the shorter paths more frequently, whereby leaves more trails of pheromones in the respective paths. Ants usually identify the shortest path to the food source by following the highest pheromone intensity. Most ants will be attracted to follow the shortest path to the food source again which further increases the pheromone intensity.

From our observation, service discovery process in SDOSN shares a close similarity to this pheromone phenomenon. Users generally have a very limited knowledge about the network and conduct service discovery by exchanging information with their neighbors. The behavior of user discovering target services resembles the phenomenon of biological swarms locating food resources. The likelihood of finding the target services relies on the users' knowledge about the search path, similar to the trial pheromone mechanism. Through the learning process of the knowledge index, users gradually gain useful information about the targets during discovery and form paths to the targets, similar to pheromone diffusion.

The proposed OSS algorithm is designed by adopting the service discovery behavior of swarm intelligence. This algorithm maximizes the phase of pheromone exploitation by utilizing the olfactory sensibility of swarms (i.e. nodes). The level of olfactory sensibility is adaptive and depends on the environment (i.e. pheromone) and individual factors (i.e. choice probability based on homophily).

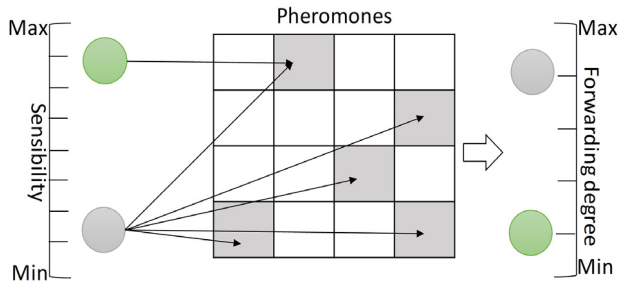

Fig. 8. The relationships of forwarding degree with the olfactory sensibility and pheromone.

Individuals with the lowest level of sensibility can select any locations marked with pheromones in each hop for maximizing the forwarding degree. When the level of sensibility increases, individuals ignore the locations marked with lower level of pheromones and select higher level of pheromone locations, whereby reducing the forwarding degree. In an olfactory-based discovery, the olfactory sensibility is usually large for distantly located targets. In this way, individuals ignore the nearby pheromones and try to explore further in the network with a small forwarding degree. When individuals move closer to the target area, their olfactory sensibility reduces accordingly to launch a search in nearby space and further to spread the pheromone information. With more pheromone information, individuals effectively utilize a large forwarding degree to exploit the current location and eventually locate all possible targets. The relationships of the olfactory sensibility, pheromone and forwarding degree are demonstrated in Fig. 8.

An individual member of the swarm can move step by step through multi-dimensional search space. During the search process, each one takes discovery walks. The aim of walks is to find a flavor using their olfactory sensibility. During the exploration, each one gets some flavors and distributes a pheromone in an amount proportional to the amount of the found flavor. The flavors can be seen as the historical query topics of a node in the social network. The pheromone can be seen as the topic hits, so it will be enhanced after each success walk. The pheromone information is stored in the knowledge index; A scenario of the discovery process is illustrated in Fig. 9. In this scenario, Node v_1 receives a query q. Upon receiving the query, v_1 first searches its local index for identifying the requested resource. When v_1 cannot resolve the query, v_1 utilizes it knowledge index to calculate the choice probability of its neighbors v_2, v_3 and v_4 based on the "Neighborlist" and "Interestlist" columns. Having obtained the values of choice probability in the neighborhood, v_1 learns the pheromone regarding the query q using "Pheromone" column. Using the OSS algorithm, v_1 determines the overall scores of its neighbors and forwards the query to the selected neighbors v_2 and v_3. Then the query is handled by v_2 and v_3 and still remains unsolved. v_2 and v_3 utilize their knowledge index to forward the query to selected neighbors those have not received the query before. We can see that v_5 receives the query from v_2 and v_3. However v_5 only deals with the first received message and drops the duplicated message. When the query reaches v_6 the requested service is found in v_6's local service index and a successful response message is sent back to the query requester. Upon successfully resolving the query, the corresponding pheromone information along the path will be updated i.e. the pheromone table in the knowledge index of v_1 and v_2 will be updated to guide future discovery of the same query TTL is used to count query message hops and a MaxTTL is defined as the upper bound of TTL. The query routing process is terminated either the requested service has been found or when TTL meets the upper bound.

5.2. Algorithm

OSS algorithm is an adaptive forwarding algorithm, which selects a subset of promising neighbors to resolve a query. The aim of OSS algorithm is to determine the shortest paths from a node when it issues a query to other nodes those can appropriately resolve the query with more relevant services. The number of forwarding nodes depends on the level of sensibility and pheromone in each hop.

Now we introduce the notations used in the remainder of this paper. Swarm member j is the current query handling node; m is the total number of nodes in the network; n is the number of neighbors of j; and k is the number of flavors (i.e. topics) of pheromone in a neighboring node of j The search space of j $(1 \leq j \leq m)$ is defined as $X_{ij} = (x_{0ij}, x_{1ij}, \ldots, x_{kij})$ *where* i $(1 \leq i \leq n)$ is the $n-$ dimensional neighbor space, and $t(1 \leq t \leq k)$ is the k-dimensional topic space in each dimension of the neighbor space. $x_{tij}(1 \leq t \leq k, 1 \leq i \leq n)$ is the location information of a topic t marked by swarm member $i$for the receiving member j. $Q_j = \{q_1, q_2, \ldots, q_c\}$ is a queue of queries to be resolved by j.

The structure of the OSS algorithm consists of three phases as initialization, discovery, and termination.

In the initialization phase, the initial location of j is defined as X_{0j}. j loads the received queries along with the knowledge index into its local memory to establish the search space. Then j selects a query from Q_j, and initializes the pheromone P_{0j} and the olfactory sensibility S_{0j}. Define $P_{0j} = S_{0j} = \max CP(X_{ij}, u_q)$ *where* X_{ij} *is* j*'s neighbor* $(1 \leq i \leq n)$; $CP(X_{ij}, u_q)$ is calculated by using Eq. (4), and the max choice probability of j' neighbors is defined as the initial sensibility of j. The initial sensibility S_{0j} is calculated by the choice probability of utilizing the homophily score and connection degree of neighborhood to estimate the max likelihood of resolving a query before exploring its pheromone table.

During the discovery phase, the walk behavior of swarm member j is described as $f_{ti} = f(x_{tij})$, where $t(1 \leq t \leq k)$ is the flavor of pheromone, i.e., the topic of a neighbor in the knowledge index of j. Each walk of j has k small steps to get the flavors of i. The f function is a match function between a query q and t based on the distance between the semantic concepts in DMOZ. The function presented by [45] is used to calculate the distance between the semantic concepts. The f is defined as following:

$$f_{ti}(q, t)_{1 \leq t \leq k} = e^{-\alpha l} \frac{e^{\beta h} - e^{-\beta h}}{e^{\beta h} + e^{-\beta h}} \tag{5}$$

where $q \in Q_j$ is the query concept and t is a topic concept. h is the shortest path length between concept q and t in an organizational ontology, in our case, DMOZ. l is the depth distance of the two concepts in the ontology. $\alpha \geq 0$ and $\beta \geq 0$ are parameters scaling the contribution of shortest path length and depth, respectively.

$$P_{i(i \in R_j)} = max_{1 \leq t \leq k} f_{ti} \tag{6}$$

where P_i is the location marked with pheromone from i, which is the best value of k flavors (i.e. topics) explored by j. $P_{\min}$ and $P_{\max}$ are the minimal and maximal possible values of the pheromone trails of $P_i(1 \leq i \leq n)$. Here the upper bound and lower bound of the olfactory sensibility are defined as:

$$S_{\max} = P_{\max}; S_{\min} = P_{\min} \tag{7}$$

$S_{\min}$ and $S_{\max}$ are the minimal and maximal possible values of the sensibility of the node j. As showed in Fig. 8, using $S_{\max}$ in every hop j ignores the locations marked with a low level of pheromone and only sense the highest level of pheromone. This will lead to a minimal number forwarding degree. While using $S_{\min}$, j will select all the locations marked with pheromone. As a result, the forwarding degree will be maximized.

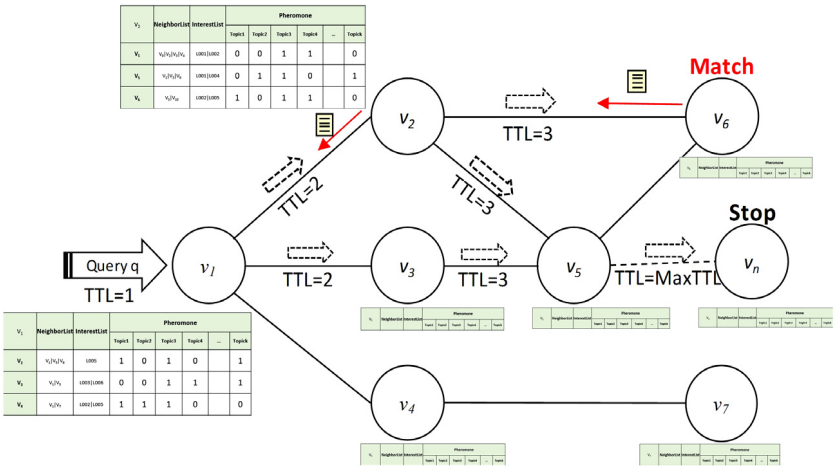

Fig. 9. Query routing based on the knowledge index.

After the exploration walks in the neighbor space, j has learned the pheromone regarding the current query q. So j's initial sensibility needs to be adjusted to adapt to the new achievements. The adaptive olfactory sensibility of j is generated by:

$$S_{ij} = S_{\max} - (S_{\max} - S_{\min}) \times S_{0j} \tag{8}$$

S_{ij} is the new sensibility of j after exploring its pheromone table in the knowledge index, and S_{0j} is the initial sensibility which is calculated using Eq. (4). For instance, a higher S_{0j} for j reflects its larger probability to resolve the query. Therefore, the olfactory sensibility is reduced to pick up more locations marked with pheromone.

Based on this adaptive sensibility, only the areas marked with higher pheromones than j's sensibility are sensed and considered by j in the next forwarding process. By comparing the adaptive sensibility and marked pheromones in the neighborhood, new locations for next hop are generated as:

$$X'_{0j} = \begin{cases} X_{ij} & if (P_i \geq S_{ij}) 1 \leq i \leq n \\ X_{0j} & otherwise. \end{cases} \tag{9}$$

This is a decision function to determine whether to explore a new location or not. Here X_{ij} is the best locations marked with pheromone where individual j has found in its neighbor space. If the level of pheromone $P_i (1 \leq i \leq n)$ is not less than the j's sensibility S_{ij} new locations X'_{0j} will be selected by j in the next hop. In other words, the algorithm will only select the neighbors whose pheromone is greater than or equal to j's sensibility for forwarding the query. Neighbors with less pheromone value than j's sensibility will not receive the query from j.

In the termination phase, the algorithm will be terminated either when the query q has been resolved or when the TTL of q exceeds the pre-defined maximal TTL. Then response messages will be sent back to the query requester. Finally, the memory resource of real-time computation will be released.

The mechanism of sensibility and pheromones determining the forwarding nodes for a query q between v_1 and its neighbor v_2, v_3 and v_4 is illustrated as follows. We use the same example in Section 4.3.2 showed in Fig. 6. In the initialization phase, by using Eq. (4), the choice probabilities of v_2, v_3 and v_4 were calculated as $CP(v_2, u_q) = 0.61$, $CP(v_3, u_q) = 0.55$, $CP(v_4, u_q) = 0.64$. The initial sensibility of v_1 is assigned as $S_{0v1} = \max[0.61, 0.55, 0.64] = 0.64$. In the discovery phase, assuming the pheromone values from neighbor space are $Pv_2 = 0.7, Pv_3 = 0.5, Pv_4 = 0.2$ based on Eq. (5) and (6) considering semantic similarity, we have the v_1's sensibility range $S_{\max} = 0.7; S_{\min} = 0.2$ after learning pheromone information. By adjusting the sensibility range, the adaptive sensibility of v_1 can be obtained as $S_{v1} = S_{\max} - (S_{\max} - S_{\min}) \times S_{0v1} = 0.7 - (0.7 - 0.2) \times 0.64 = 0.38$. Next comparing the learned pheromone values and the adaptive sensibility, we obtain $Pv_2 > S_{v1}; Pv_3 > S_{v1}; Pv_4 < S_{v1}$. Using the decision function Eq. (9), eventually v_1 forwards the query to v_2 and v_3. Note in this example, though v_4 has the highest choice probability based on the homophily feature and connection degree, OSS excludes v_4 from the forwarding list based on its adaptive sensibility, since the pheromone information Pv_4 for query q is below the level of sensibility of v_1.

To conclude, when a node receives a query message, it first checks whether the query has been already received during the past. Redundant query will be discarded without further processing. Then the node utilizes the local index to score the homophily between its user vector and the query vector to determine whether the requested service can be provided. If the query needs to be further forwarded, the query-handling node initializes the knowledge index to find promising associated nodes using OSS algorithm and multicast the query to the chosen nodes. In OSS algorithm, each node maintains a pheromone table in the knowledge index that records the information of its immediate neighbors. Intuitively, the pheromone of a neighbor is a shortcut link representing the historical success in query routings via that neighbor. In order to achieve high recall and low messages overhead, an adaptive forwarding degree is proposed. The number of nodes to be forwarded in each hop is adjustable based on both the olfactory sensibility and the level of pheromone in the neighborhood. The olfactory sensibility is an adaptive value depending on the level of pheromone and the choice probability. With higher level of pheromone information and large choice probability, the node utilizes a large forwarding degree to exploit the current location to locate more possible targets. On the contrary, lower level of pheromone information and small choice probability will lead to a small forwarding degree for bandwidth-efficient discovery. A detailed description of this algorithm is showed in Fig. 10.

6. Experiment

6.1. Simulation methodology

IoT offers advanced and complex connectivity of devices, systems, and services comprising heterogeneous components, and generates large-scale unstructured data and covers a variety of protocols, domains, and applications. In order to focus on the core value of service discovery in a decentralized IoT environment, simulation is a cost-effective method than implementing physical systems to evaluate the performance of our research. A software

Algorithm 1: Olfactory Sensitive Search (OSS)

Input: Query processing node j, Neighbor list of j : $R_j = \{r_1, r_2, \ldots, r_n\}$, A finite queue $Q = \{q_1, q_2, \ldots, q_c\}$ of Queries, Max Time-to-Live value $MaxTTL$

Output: The forwarding list of selected neighbors of node j

Initialization(SeachSpace X_{tij} QueryQueue Q)
$X_{tij} \leftarrow loadKnowledgeIndex(j, R_j)$
// Q contains the queries that need to be forwarded by j
$Q \leftarrow getForwardingQueryMessages()$
for $l \leftarrow 1$ ***to*** c **do**
  if $getQuery().Next() \neq \emptyset$ **then**
    $q_l \leftarrow Q.Next()$
    $q_l.ttl \leftarrow q_l.getTTL()$
  $U_{q_l} \leftarrow q_l.getQueryRequestNode()$
  $P_{0j} = S_{0j} \leftarrow HomophilyScore(j, U_{q_l})$
  if $ttl \leq MaxTTL$ **then**
    for $i \leftarrow 1$ ***to*** n **do**
      // Iterate over all neighbors of j
      if $getNeighbors().Next() \neq \emptyset$ **then**
        $r_i \leftarrow getNeighbors(j).Next()$
      for $t \leftarrow 1$ ***to*** k **do**
        // Iterate over all topics in the i-th neighbor
        if $getTopics().Next() \neq \emptyset$ **then**
          $x_{tij} \leftarrow getTopics(r_i).Next()$
        $f_{ti} \leftarrow f(x_{tij})$
        // Semantic distance between query q_h and topic t
        $f_{ti}(q_k, t) = e^{-\alpha l} \frac{e^{\beta h} - e^{-\beta h}}{e^{\beta h} + e^{-\beta h}}$
      // Learn Pheromone from i
      $P_i = max\ f_{ti}$
      $PheromoneList.Add(P_i)$
    // Update sensibility of j
    $PheromoneList.Sort()$
    $P_{\max} = PheromoneList().Max\ P_{\min} = PheromoneList().Min$
    $S_{\max} = P_{\max}\ S_{\min} = P_{\min}$
    $S_j = S_{\max} - (S_{\max} - S_{\min}) \times S_{0j}$
  for $i \leftarrow 1$ ***to*** n **do**
    if $P_i \geq S_j$ **then**
      $ForwardingList(j).Add(r_i)$
      return X_{ij}
    else
      return X_{0j}
  $forwardQueryto(q_h, ForwardingList(j))$
  $q_l.setTTL() = q_l.ttl + 1$
return $ForwardingList(j)$

Fig. 10. Olfactory sensitive search algorithm.

simulation platform is designed and implemented to simulate a real world unstructured P2P network with configurable routing protocols and basic social network functions. The platform can conduct simulations of service discovery on different network structures with self-organized social overlay network. The main components of the simulation platform are illustrated in Fig. 11.

Basically, the architecture of the simulation platform comprises two layers. The bottom layer offers a decentralized P2P network infrastructure including network elements initialization, content generation and distribution, topology initialization. The upper layer encompasses the service discovery components including dynamic network churns generation, query generation and forwarding, social knowledge updating and multi-metrics evaluation.

To be specific, each peer nodes in the social network has a set of local services, a set of friends and a set of interest tags. At the network initialization stage, the simulation creates a node instance by allocating a range of memory spaces and establishing the social attributes in an object-oriented design. Then the attributes are assigned with values of real-world social services. Furthermore, every node instance generates a local service index and a knowledge index to utilize the assigned features. The local index is an inverted index for topic-service of the local service repository, while the knowledge index is a routing table that stores social information about immediate neighbors. The node instances, social connections and services are generated in a batch mode based on different network configurations. At the simulation stage, when the social network has been successfully established, every node exhibits an absent probability insisting their probability for disconnection from the social network and being unable to provide services in a simulation instance. This phenomenon is called as the churn in a decentralized social network and it is very common in real IoT environments. The simulation platform encompasses a special module to impose churn in the network to mimic a more realistic scenario. Then large numbers of randomly selected queries are generated by the nodes and the communication between two nodes is enabled if they are connected in the social network and both physically present during the simulation. All query messages have a globally unique identifier (GUID) and a max TTL. If a node receives the GUID, it will disregard the query to avoid loops. The max TTL is the upper bound hops of a query. A query is forwarded through neighbors to conduct service discovery by utilizing the knowledge index and the process continues until either the query has been resolved or its TTL exceeds the max TTL. After each discovery, the successful hits and the number of returned services are considered to evaluate the performance of the proposed algorithm against the state-of-the-art methods.

6.2. Simulation design

Dataset preparation: this study uses DMOZ dataset [30], and the Social-ODP-2k9 dataset [46]. DMOZ dataset spans from 1998 to 2016, which is the most widely distributed database of Web content using a hierarchy topic structure. *682* topics from subcategories are extracted from DMOZ. These topics are treated as user interests and the topic distribution to nodes is achieved using a power law stating few nodes contain many topics in their repositories and the rest of the nodes contain few topics. Previous studies [47,48] observed that the distribution of keywords in a large repository can be approximated by Zip's law in the form of $y \sim \frac{1}{x^{\gamma}}$ where y is the frequency, x the rank and γ is a constant. Our simulation uses the estimated distribution described in the works of [49]. Social-ODP-2k9 is a dataset spanning from December 2008 to January 2009 with data retrieved from the social bookmarking sites Delicious, StumbleUpon, and Open Directory Project (DMOZ). The Social-ODP-2k9 includes the annotation data and document data. This dataset comprises *12,616* unique URLs, all representing their corresponding social annotations. The content of each document including title, keywords, and descriptions is extracted to construct a structured document-topic vector where each document is correlated with top 10 topics achieved by applying the topic model [33]. Meanwhile, the user annotation data is used as the benchmark for precision and recall. The document-topic vectors are used as service content data, which is distributed among the nodes based on the power law. Within each user's repository, the user interests and the service content are correlated with each other.

Each network structure of our experiment is an undirected graph among *1000* nodes. The social network connections and the service repositories are created statically, whereas the queries are launched randomly during the simulation. Each node is assigned with a list of possible query topics to search. This list is limited by the total amount of topics extracted from the DMOZ. During each step of the experiment, each node evenly selects random topics from the list of possible topics, whereby all the nodes have the same probability of generating service queries. A query message consists of two features that characterize the estimated target provider and the target service description. Each query message will be forwarded by the nodes until either the query has been successfully resolved or the query reaching the maxTTL. A query is successfully solved when a node offers a "similar enough" service

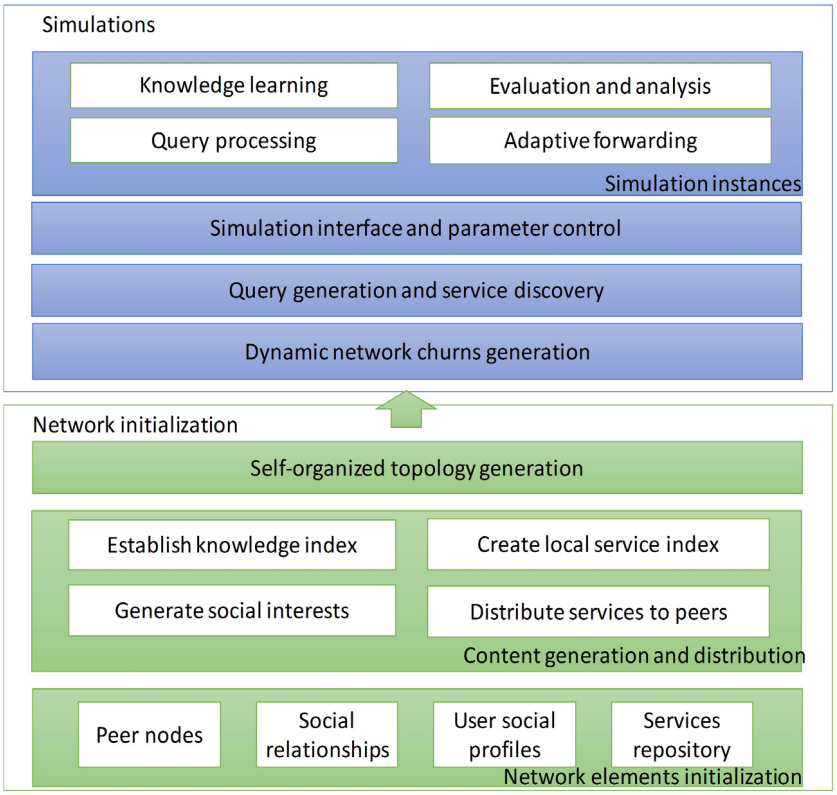

Fig. 11. Simulation platform components.

to the query message. A threshold θ is defined to determine this similarity. We run each simulation for *10,000* time cycles (queries) to observe the average performance for every *100* units of time.

Programming environment: Lenovo Thinkstation with Intel Core i5-6400 Processor, 16GB RAM, Windows 10 Pro X64 The simulation platform is implemented using Java SDK 1.80 and Eclipse J2EE IDE.

6.3. Performance evaluation

To evaluate the service discovery algorithm, extensive experiments are conducted in the simulation platform. The experiments compare the number of visited nodes, success rate, and average recall under three commonly used network structures in complex networks. The considered network structures include the following:

Random Networks (RN): where links between peers are established randomly.

Scale-Free Networks (SFN): where links between peers are established based on the degree of connection. Peer nodes with a high degree of connectivity have a greater probability of receiving a new link than agents with a low degree of connection [50].

Homophily Networks (HN): Peers in these networks are linked based on the homophily value and the degree of connection [23].

The OSS search strategy for decentralized service discovery has also been compared with various search strategies used in complex networks, differing by the way of neighbor selection in each step. These strategies are:

Random: a search process utilizes random walks with a fixed forwarding degree [13,51];

Degree: a search process utilizes only the degree of connection information with a fixed forwarding degree [15,52,53];

FreeNet: a search model utilizes similarity with deep first search with a fixed forwarding degree [20];

Neurogrid: a search model utilizes similarity with broad first search with a fixed forwarding degree [21].

ESLP: a search model utilizes similarity with broad first search with an adaptive forwarding degree [22].

EDSD: a search model utilizes similarity integrated with random search in greedy search mode [23].

IAPS: a search model utilizes resource type score with a fixed forwarding degree [25].

Recall, precision, and F-1 measure are the most common metrics to evaluate the effectiveness of a decentralized system. These metrics represent the number of positive results returned by a search model, which is defined as

$$Recall = \frac{|\{relevant\ results\} \cap \{retrieved\ results\}|}{|\{relevant\ results\}|}$$

$$Precision = \frac{|\{relevant\ results\} \cap \{retrieved\ results\}|}{|\{retrieved\ results\}|}$$

$$F1 = \frac{2 \times Recall \times Precision}{Recall + Precision}.$$

Recall on Time-to-Live (RTTL) quantifies the performance of the search node based on the *TTL* used to find all relevant services. The RTTL measures are given in terms of the average recall and the maximum steps allowed in each node.

Number of visited nodes counts the average number of nodes visited by a query to locate the required services. Lesser the number of visited nodes, shorter is the discovery path with reduced traffic.

Parameter setting: In the simulation, TTL is evaluated over the range 2∼8 In a moderately connected Gnutella network more than 70% of the generated messages are redundant resulting in a flooding with a TTL of 7 [54]. Also in our experiments, with the TTL of 8, query messages can reach almost all the nodes in the

Table 1
Simulation parameters and their default values.

Parameters	Value	Function
maxTTL	2–8	Max hops of a query
m	1000	Number of nodes
k	100	Max number of topics in pheromone table
δ	0.5	Homophily regulating parameter
α	0.3	Scaling parameter the contribution of depth
β	1	Scaling parameter the contribution of path length
θ	0.25	Cosine similarity threshold: FreeNet, Neurogrid, ESLP, EDSD,OSS
d	1–5	Forwarding degree Fixed models: Random= 2, Degree= 2, FreeNet= 2, Neurogrid= 2, EDSD= 2,IAPS= 2 Adaptive models: ESLP=1∼5, OSS=1∼5
γ	0.9	Zipf law for distribution of services
Churn	0.2	Nodes leaving and joining the network
Services	12616	Total number of services
Interests	682	Total number of interests
P2P model	unstructured	Pure unstructured P2P

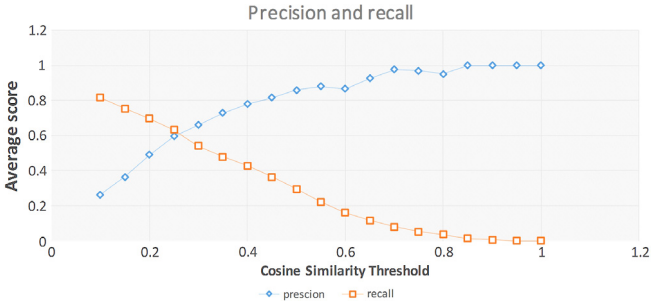

Fig. 12. Similarity threshold evaluation of precision and recall.

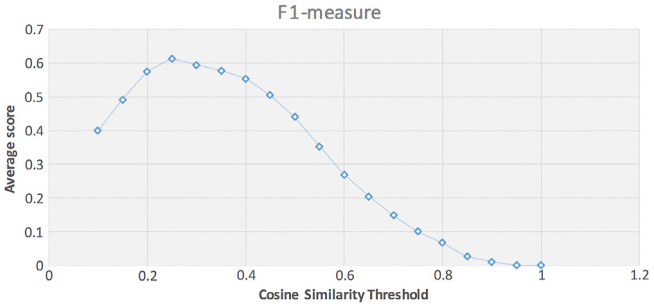

Fig. 13. F1-measure.

overlay networks. The total number of nodes in the experiment is set as 1000 for all experiments. The max number of topics in the pheromone table is 100. The max size of the knowledge index is 100. The Cosine similarity threshold θ is set as 0.25 for all the similarity based search models including Neurogrid, ESLP, EDSD and OSS. The homophily parameter δ is set to 0.5. The adaptive forwarding degree in our experiment is evaluated in the range 1∼5. The average forwarding degree of all the models in the simulations is approximated to 2 based on the experimental observations. Note: the original model of FreeNet, EDSD forwards queries only to a single neighbor in each hop to reduce the query messages. In order to conduct a fair comparison of the forwarding degree, we have altered the forwarding degree of FreeNet, EDSD similar to other models. Table 1 shows the default values of simulation parameters and the function description.

Network churn: In a dynamic and unpredictable internet environment, network churn is usually caused for two primary reasons: firstly when peer nodes frequently go online and offline and secondly during frequent sharing and removal of contents. In order to simulate realistic P2P systems, our simulations include a churn rate, where approximately 20% of peer nodes are presented less than 30% of the time. The content of nodes in our simulations is considered to be static.

In the scenario showed in Fig. 12, the cosine similarity threshold is evaluated through a service discovery task, which is to return a set of relevant enough services for a given query. Precision is the probability that retrieved services are relevant. Recall is the probability of retrieving a relevant service in a search. The annotation dataset provides the ground truth of relevancy between a query and a service. Recall and Precision are inversely proportional; the precision usually increases at the cost of reducing the recall whenever the cosine similarity threshold increases. It is can be observed that the two curves intersect with each other for the defined threshold value of 0.25.

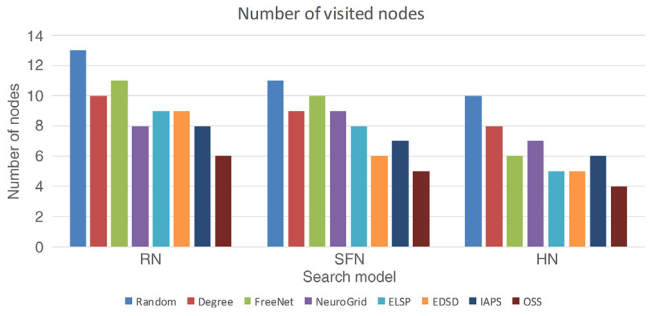

Fig. 14. Number of visited nodes under different network structures.

Typically, precision and recall are not discussed in isolation. Instead, either one of the two measures is evaluated against the other measure (e.g. precision at a recall level of 0.6) or both are combined into a single measure. Fig. 13 shows a combined measure of precision and recall which is known as F1-measure. The F1-measure is the weighted harmonic mean of precision and recall. From this graph, it is can be observed that the F1-measure reaches its peak value when the cosine similarity threshold approaches 0.25. Based on the observation in Figs. 12 and 13 the parameter of cosine similarity threshold θ is set as 0.25 in order to determine a "similar enough" service for all the similarity based models (Neurogrid, ESLP, EDSD and OSS) in our experiments.

Fig. 14 illustrates the average number of visited nodes in all the studied models for each network structure. For each network structure, the shortest path with a minimal number of visited nodes is obtained based on their adopted strategy of building the network, insisting that the search strategy and the network structure are closely associated. Random network structure (RN) and scale free network structure (SFN) connects the nodes without considering the interests and semantic content of the nodes.

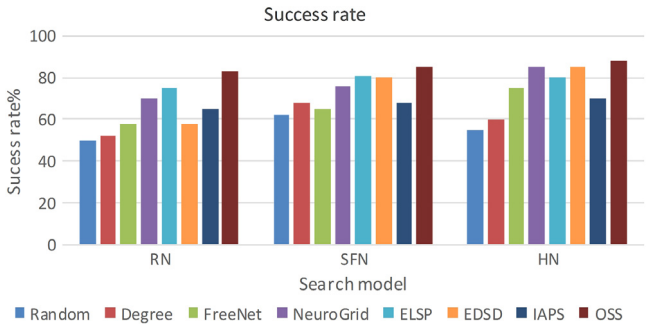

Fig. 15. Success rates under different network structures.

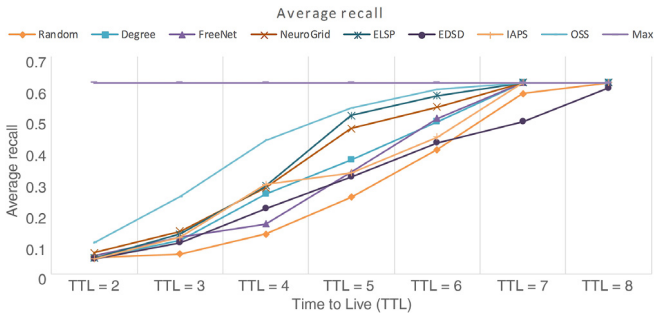

Fig. 16. Average recalls on time-to-live(TTL).

Considering the differences between the content of nodes and their neighbors, the homophily network structure (HN) establishes user connections with similar interest and content. It can be observed from Fig. 14 that the average number of visited nodes is quite low for most of the search models in HN in comparison to those in RN and SFN. This insists that the homophily is effective in facilitating service discovery with shorter paths, since users tend to interact with similar users in social networks. HN tends to bring similar users together as neighbors, thus the number of visited nodes to resolve queries are lower than RN and SFN. Furthermore, it is clearly evident that our proposed OSS algorithm outperforms all the compared models achieving the minimal number of visited nodes, with the best performance achieved in HN. This is because in each query forwarding hop, OSS accurately measures the distance to the target service provider. OSS utilizes a homophily enhanced query message as an estimated target provider. The query message usually encloses the service description of the target provider along with the homophily features of the target provider. OSS improves the similarity measurement by considering both semantic distance and homophily distance to the estimated target provider. Besides, OSS uses an adaptive forwarding degree, whereby nodes with lower similarity are ignored during the forwarding process, thus OSS search model can successfully resolve queries with a minimal number of visited nodes in all the three network structures.

Fig. 15 depicts the percentage of queries resolved by the studied models before the TTL expires for all the three network structures. In all the three networks, the success rates of search models are affected by forwarding degree and routing strategy. It can be observed that the knowledge-based models of FreeNet, NeuroGrid, ELSP, EDSD, IAPS and OSS are achieving better results than the uninformed models such as Random and Degree. Among the knowledge-based models, BFS based search models including NeuroGrid, ELSP, and OSS are exhibiting a higher average success rate than FreeNet which uses DFS based search. Comparing ELSP and OSS based on the adaptive forwarding degree, OSS has a more comprehensive knowledge index for gaining more social information about the neighbors including interests, social relationships. ELSP only has a topic information of immediate neighbors in its knowledge index, but the neighbor's degree and the homophily are not considered in ELSP. During the service discovery process OSS obtains homophily information in social network and utilizes a more accurate neighbor selection than ELSP, thus OSS exhibits a better success rate than ELSP. The network structure RN is exhibiting a lower success rate for all the models and the average success rate is above 60% for all the search models in SF. Nodes with more candidate neighbors for the query routing process usually affects the success rate in SF. In HN, the success rate is above 75% for the knowledge-based search models. Once again, these results further validate that the homophily network can improve the performance of service discovery. Overall, in the three network structures, OSS achieves a much better success rate above 80%, outperforming all the compared models. This is because of the fact that OSS not only utilizes semantic similarity but also considers neighbors' connection degree as an important factor for query forwarding, which combines the advantages of similarity based search models and degree based search models. As a result, OSS achieves a higher average success rate than other models and further exhibits a better tolerance for different network structures.

Fig. 16 illustrates average recalls achieved by different models for various TTL values under the HN network structure. The optimal recall (represented by the max curve in the figure) is used as the max recall in the decentralized environment. It can be observed that the TTL determining the average recall in all the search models. Longer the TTL, higher will be the recall. The random search model is exhibiting the worst recall efficiency, which increases very slowly with increasing TTL. Our proposed OSS model is very sensitive to the TTL value; an increase in the TTL is having a more positive effect on recall than any other models. This feature is very important in DOSN, since a smaller TTL with a relatively high recall can drastically reduce the messages. In HN, users are connecting with similar users. OSS is sensitive to homophily features and is able to utilize a relatively larger forwarding degree for many queries when these queries are close to service providers. Other models either use a static forwarding degree or not sensitive to homophily features. Thus even with a small TTL, OSS can find more relevant services than any other models.

6.4. Summary

This paper utilizes an informed search method to conduct service discovery in a fully decentralized network. The performance of service discovery depends on structures of social overlay network and how to utilize local knowledge to adapt to different structures. The results of experiments allow us to conclude that the homophily network structure possesses the most desirable characteristics for providing a preferable social similarity-based network to support efficient service discovery. Search models running on the homophily network exhibit a higher percentage of success with a low number of visited nodes in the service discovery process. The proposed OSS model outperforms the state-of-the-art search models in terms of the average number of visited nodes, success rate and recall in the three different network structures. Moreover, OSS is able to achieve better performance of service discovery than the compared models despite the type of the network structures.

The main overhead of our approach is the maintenance of a knowledge index in users' local storage. The knowledge index contains associations between a node and its immediate neighbors based on historical search results, which includes the neighbor's neighbor list, the neighbor's interests list, and the pheromone table (i.e. topic hit table). On one hand, the neighbor list and interest

list do not need frequent updates. In this research when the social overlay network has been established, the neighbor list and interest list are considered immutable, since such changes are infrequent to the extent that they are insignificant within the time scale of our problem of information dissemination and service discovery. It is a one-time operation to record them in the knowledge index when friendship relationships are being formed. During the service discovery process, the neighbor list and interest information will not be updated. On the other hand, only the pheromone table in the knowledge index needs frequent updates. The pheromone table is created based on historical searches, updated upon success requests. The maximum size of pheromone table is a user-defined variable. In our simulation, the size of pheromone table is defined as 100 topic items. When the number of topic records reaches the maximum size, the old records will be replaced with new records using the least recently used strategy. It is can be observed from the experimental results that we achieved more than 30% higher performance than uninformed search models by utilizing a small-sized knowledge index. Besides, our proposed model has fewer visited nodes and higher success rate than other models. With fewer visited nodes, the number of forwarding messages over the network decreases correspondingly. Therefore, the network traffic overhead is lower than the compared models. Therefore, the simulation results allow us to conclude that with the significant improvement of service discovery performance, the maintenance overhead of the small-sized knowledge index in our model is reasonable and cost-effective.

7. Conclusions

Cloud Computing and IoT are revolutionary technologies that are powering ubiquitous wireless communication and real-time computing to expand the Internet-connected automation into new application areas. This paper examines the problems of current OSN and proposes a self-organized decentralized architecture for future online social networks to enable efficient service discovery in dynamic social IoT environments.

This decentralized architecture fills the research gap between the traditional P2P infrastructure and the decentralized architecture of OSN, which can alleviate the rigid privacy-control problems, as well as provide adequate flexibility and capability for service discovery. Then a homophily-based user model is introduced to integrate social relationship and user interest. This model is able to identify promising neighbors having considerable similarity to the potential service provider as well as having the high number of connections. Further to achieve a bandwidth-efficient search, a novel OSS algorithm has been proposed to provide an adaptive forwarding degree in each routing hop. This algorithm utilizes the olfactory sensibility and pheromone information of swarms to discover the shortest paths with maximum desired service. Moreover, a Java based simulation platform has been designed to simulate a dynamic unstructured P2P network with knowledge-based routing protocols for service discovery in DOSN. The proposed user model and OSS algorithm have been simulated under different network scenarios to evaluate its efficiencies against a range of state-of-the-art search models in unstructured P2P networks. Experimental results demonstrate that our approach achieves better performance under three dynamic network structures than the compared models.

Although this paper has addressed several issues of service discovery in DOSN, there are still other research problems to be taken into account in the future. At present, this paper only considers user interests and social relationships to determine the homophily similarity. As a future work, we plan to consider other social features such as authority, centrality, and time/location attributes, which might further improve the overall performance of service discovery in social networks. Moreover, unlike a closed simulation system, the topology structure of an IoT-based DOSN is ad-hoc in practice. With objects joining and leaving the system frequently and contents being generated and updated quickly, service unavailability, resilience under high network churn are a few commonly prevailing issues in service discovery. In addition, testing the system performance in a real open system is also one of our future research directions.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China under Grant Nos. 61502209 and 61502207 and UK–China Knowledge Economy Education Partnership under grant No. 90812.

References

[1] www.statista.com, Number of social media users worldwide from 2010 to 2020, 2016.
[2] K.W. Pamela Vagata, Scaling the Facebook data warehouse to 300 PB, vol. 2017, 2014 http://code.facebook.com/.
[3] P. Mell, T. Grance, The NIST definition of cloud computing, 2011.
[4] M. Haghighat, S. Zonouz, M. Abdel-Mottaleb, CloudID: trustworthy cloud-based and cross-enterprise biometric identification, Expert Syst. Appl. 42 (2015) 7905–7916.
[5] D. Evans, The internet of things: How the next evolution of the internet is changing everything, CISCO White Paper, vol. 1, 2011, pp. 1–11.
[6] S. Buchegger, Schi, D. Berg, L.H. Vu, A. Datta, PeerSoN: P2P social networking: early experiences and insights, in: Proc Acm Workshop on Social Network Systems, 2009, pp. 46-52.
[7] Z. Li, H. Shen, Social-P2P: Social network-based P2P file sharing system, in: 2012 20th IEEE International Conference on Network Protocols, ICNP, 2012, pp. 1-10.
[8] L.A. Cutillo, R. Molva, T. Strufe, Safebook: a privacy-preserving online social network leveraging on real-life trust, IEEE Commun. Mag. 47 (2009).
[9] S. Buchegger, A. Datta, A case for P2P infrastructure for social networks-opportunities & challenges, in: Wireless On-Demand Network Systems and Services, 2009, WONS 2009, Sixth International Conference on, 2009, pp. 161-168.
[10] C.-m.A. Yeung, I. Liccardi, K. Lu, O. Seneviratne, T. Berners-Lee, Decentralization: The future of online social networking, in: W3C Workshop on the Future of Social Networking Position Papers, 2009, pp. 2-7.
[11] S. Jahid, S. Nilizadeh, P. Mittal, N. Borisov, A. Kapadia, DECENT: A decentralized architecture for enforcing privacy in online social networks, in: Pervasive Computing and Communications Workshops, PERCOM Workshops, 2012 IEEE International Conference on, 2012, pp. 326-332.
[12] X.-S. Yang, Z. Cui, R. Xiao, A.H. Gandomi, M. Karamanoglu, Swarm Intelligence and Bio-Inspired Computation: Theory and Applications, Newnes, 2013.
[13] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, S. Shenker, Making gnutella-like p2p systems scalable, in: Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, 2003, pp. 407-418.
[14] C. Gkantsidis, M. Mihail, A. Saberi, Random walks in peer-to-peer networks, in: INFOCOM 2004 Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies, 2004.
[15] B. Yang, H. Garcia-Molina, Improving search in peer-to-peer networks, in: Distributed Computing Systems, 2002 Proceedings, 22nd International Conference on, 2002, pp. 5-14.
[16] A. Broder, M. Mitzenmacher, Network applications of bloom filters: a survey, Internet Math. 1 (2004) 485–509.
[17] A. Crespo, H. Garcia-Molina, Routing indices for peer-to-peer systems. in: Distributed Computing Systems, 2002 Proceedings, 22nd International Conference on, 2002, pp. 23-32.
[18] D. Tsoumakos, N. Roussopoulos, Adaptive probabilistic search for peer-to-peer networks, in: Peer-to-Peer Computing, 2003, P2P 2003, Proceedings, Third International Conference on, 2003, pp. 102-109.
[19] V. Kalogeraki, D. Gunopulos, D. Zeinalipour-Yazti, A local search mechanism for peer-to-peer networks, in: Proceedings of the Eleventh International Conference on Information and Knowledge Management, 2002, pp. 300-307.
[20] I. Clarke, O. Sandberg, B. Wiley, T.W. Hong, Freenet: A distributed anonymous information storage and retrieval system, in: Designing Privacy Enhancing Technologies, 2001, pp. 46-66.
[21] S. Joseph, NeuroGrid: Semantically Routing Queries in Peer-To-Peer Networks, Springer, Berlin, Heidelberg, 2002.

[22] L. Liu, N. Antonopoulos, S. Mackin, J. Xu, D. Russell, Efficient resource discovery in self-organized unstructured peer-to-peer networks, Concurrency Comput. Pract. Exp. 21 (2009) 159–183.
[23] E.D. Val, M. Rebollo, V. Botti, Enhancing decentralized service discovery in open service-oriented multi-agent systems, Auton. Agents Multi Agent Syst. 28 (2014) 1–30.
[24] E. Michlmayr, Ant algorithms for search in unstructured peer-to-peer networks, in: 22nd International Conference on Data Engineering Workshops, ICDEW'06, 2006, pp. x142–x142.
[25] M. Shojafar, J.H. Abawajy, Z. Delkhah, A. Ahmadi, Z. Pooranian, A. Abraham, An efficient and distributed file search in unstructured peer-to-peer networks, Peer-to-Peer Netw. Appl. 8 (2015) 120–136.
[26] W. Galuba, Friend-to-Friend computing: Building the social web at the internet edges, 2008.
[27] F. Benevenuto, T. Rodrigues, M. Cha, V. Almeida, Characterizing user navigation and interactions in online social networks, Inform. Sci. 195 (2012) 1–24.
[28] K. Graffi, P. Mukherjee, B. Menges, D. Hartung, A. Kovacevic, R. Steinmetz, Practical security in p2p-based social networks, in: 2009 IEEE 34th Conference on Local Computer Networks, 2009, pp. 269–272.
[29] G. Mega, A. Montresor, G.P. Picco, Efficient dissemination in decentralized social networks, in: Peer-to-Peer Computing, P2P, 2011 IEEE International Conference on, 2011, pp. 338–347.
[30] https://www.dmoz.org/, DMOZ - The Directory of the Web, 2016.
[31] H. Tajfel, Human groups and social categories: Studies in social psychology: CUP Archive, 1981.
[32] D. Blei, A. Ng, M. Jordan, Latent dirichlet allocation, J. Mach. Learn. Res. 3 (2003) 993–1022.
[33] X. Yan, J. Guo, Y. Lan, X. Cheng, A biterm topic model for short texts, in: Proceedings of the 22nd International Conference on World Wide Web, 2013, pp. 1445–1456.
[34] M. McPherson, L. Smith-Lovin, J.M. Cook, Birds of a feather: homophily in social networks, Annu. Rev. Sociol. (2001) 415–444.
[35] D.J. Watts, P.S. Dodds, M.E. Newman, Identity and search in social networks, Science 296 (2002) 1302–1305.
[36] E. Bakshy, I. Rosenn, C. Marlow, L. Adamic, The role of social networks in information diffusion, in: Proceedings of the 21st International Conference on World Wide Web, 2012, pp. 519– 528.
[37] M. Yavaş, G. Yücel, Impact of homophily on diffusion dynamics over social networks, Soc. Sci. Comput. Rev. 32 (2014) 354–372.
[38] R. Mihalcea, C. Corley, C. Strapparava, Corpus-based and knowledge-based measures of text semantic similarity, in: AAAI, 2006, pp. 775–780.
[39] M. Steinbach, G. Karypis, V. Kumar, A comparison of document clustering techniques, in: KDD Workshop on Text Mining, 2000, pp. 525–526.
[40] D.B. Kandel, Homophily, selection, and socialization in adolescent friendships, Am. J. Sociol. (1978) 427–436.
[41] M. Ruef, H.E. Aldrich, N.M. Carter, The structure of founding teams: Homophily, strong ties, and isolation among US entrepreneurs, Am. Sociol. Rev. (2003) 195–222.
[42] S. Aral, L. Muchnik, A. Sundararajan, Distinguishing influence-based contagion from homophily-driven diffusion in dynamic networks, Proc. Natl. Acad. Sci. 106 (2009) 21544–21549.
[43] S. Joseph, P2P metadata search layers, in: International Workshop on Agents and P2P Computing, 2003, pp. 101–112.
[44] E.O. Wilson, M. Pavan, Glandular sources and specificity of some chemical releasers of social behavior in dolichoderine ants, Psyche 66 (1959) 70–76.
[45] P. Fu, S. Liu, H. Yang, L. Gu, Matching algorithm of web services based on semantic distance, in: Proceedings of 2009 International Workshop on Information Security and Application, 2009.
[46] A. Zubiaga, R. Martínez, V. Fresno, Getting the most out of social annotations for web page classification, in: Proceedings of the 9th ACM symposium on Document engineering, 2009, pp. 74–83.
[47] R.L. Axtell, Zipf distribution of US firm sizes, Science 293 (2001) 1818–1820.
[48] L.A. Adamic, B.A. Huberman, Zipf's law and the internet, Glottometrics 3 (2002) 143–150.
[49] P. Makosiej, G. Sakaryan, H. Unger, Measurement study of shared content and user request structure in peer-to-peer Gnutella network, in: Proc. of the International Conference on Design, Analysis, and Simulation of Distributed Computing System, 2004.
[50] R. Pastor-Satorras, A. Vespignani, Epidemic spreading in scale-free networks, Phys. Rev. Lett. 86 (2001) 3200.
[51] M. Ripeanu, Peer-to-peer architecture case study: Gnutella network, in: Peer-to-Peer Computing, 2001 Proceedings, First International Conference on, 2001, pp. 99–100.
[52] L. Liu, J. Xu, D. Russell, P. Townend, D. Webster, Efficient and scalable search on scale-free P2P networks, Peer-to-Peer Netw. Appl. 2 (2009) 98–108.
[53] L.A. Adamic, R.M. Lukose, A.R. Puniyani, B.A. Huberman, Search in power-law networks, Phys. Rev. E (3) 64 (2001) 046135.
[54] S. Jiang, L. Guo, X. Zhang, H. Wang, Lightflood: minimizing redundant messages and maximizing scope of peer-to-peer search, IEEE Trans. Parallel Distrib. Syst. 19 (2008) 601–614.

Bo Yuan received the BSc. degree in computer science and technology from the Tongji University, Shanghai, China in 2011. He is currently a Joint Ph.D. student with Tongji University and University of Derby. His research interests include service discovery, online social networks, mobile computing, and peer-to-peer systems.

Lu Liu is the Professor of Distributed Computing at the School of Computing and Mathematics in the University of Derby and adjunct professor at the School of Computer Science and Communication Engineering in Jiangsu University. Prof. Liu received his Ph.D. degree from University of Surrey. He is the Fellow of British Computer Society. Prof. Liuś research interests are in the areas of Cloud Computing, Social Computing, Service-oriented Computing and Peer-to-Peer Computing.

Nick Antonopoulos is currently Dean of the College of Engineering & Technology, University of Derby and the University of Derby Technical Coordinator of the framework collaboration with CERN as well as the ALICE experiment. Nick holds a Ph.D. in Computer Science from the University of Surrey. His research interests include Cloud Computing, P2P Computing, software agent architectures and security. Nick has over 18 years of academic experience and has published more than 150 articles in fully refereed journals and international conferences.